RÉPUBLIQUE FRANÇAISE
Liberté
Égalité
Fraternité

GUIDE D'IMPLÉMENTATION D'IPSEC INTÉGRANT LA CRYPTOGRAPHIE POST-QUANTIQUE POUR LA PROTECTION DE DONNÉES DIFFUSION RESTREINTE

ANSSI-PA-120
17/11/2026

GUIDE ANSSI

PUBLIC VISÉ :

Développeur Administrateur RSSI DSI Utilisateur

Informations



Attention

Ce document rédigé par l'ANSSI s'intitule **« Guide d'implémentation d'IPsec intégrant la cryptographie post-quantique pour la protection de données diffusion restreinte »**. Il est téléchargeable sur le site `cyber.gouv.fr`.

Il constitue une production originale de l'ANSSI placée sous le régime de la « Licence Ouverte v2.0 » publiée par la mission Etalab.

Conformément à la Licence Ouverte v2.0, le document peut être réutilisé librement, sous réserve de mentionner sa paternité (source et date de la dernière mise à jour). La réutilisation s'entend du droit de communiquer, diffuser, redistribuer, publier, transmettre, reproduire, copier, adapter, modifier, extraire, transformer et exploiter, y compris à des fins commerciales.

Sauf disposition réglementaire contraire, les recommandations n'ont pas de caractère normatif ; elles sont livrées en l'état et adaptées aux menaces au jour de leur publication. Au regard de la diversité des systèmes d'information, l'ANSSI ne peut garantir que ces informations puissent être reprises sans adaptation sur les systèmes d'information cibles. Dans tous les cas, la pertinence de l'implémentation des éléments proposés par l'ANSSI doit être soumise, au préalable, à la validation de l'administrateur du système et/ou des personnes en charge de la sécurité des systèmes d'information.

Évolutions du document :

VERSION	DATE	NATURE DES MODIFICATIONS
1.0	17/11/2026	Version initiale

Table des matières

1 Introduction **3**

2 Présentation du protocole IPsec **4**
2.1 Fonctionnement d'IPsec 4
2.1.1 Les modes transport et tunnel d'IPsec 4
2.1.2 Le protocole IKEv2 et les associations de sécurité 5
2.1.3 La fragmentation des messages dans IKEv2 6
2.2 Transition post-quantique d'IPsec 7

3 Règles et Recommandations **9**
3.1 Implémentation d'IPsec 10
3.1.1 Moyens d'authentification dans IPsec 12
3.2 Implémentation d'IKEv2 13
3.2.1 Implémentation de la cryptographie post-quantique dans IKEv2 15
3.3 Mécanismes cryptographiques dans IPsec PQC DR 17
3.3.1 Chiffrement des messages 17
3.3.2 Intégrité des messages 18
3.3.3 Fonctions pseudo-aléatoires pour la génération des clés 19
3.3.4 Échange de clé 19
3.3.5 Authentification 20

Bibliographie **22**

Annexe A Configuration strongSwan **24**

1
Introduction

Ce document succède au corpus documentaire IPsec DR [1] afin de prendre en compte l'utilisation de la cryptographie post-quantique dans le protocole IPsec (*Internet Protocol Security*) pour la protection des données. Il présente les règles et les recommandations relatives au choix d'implémentation du protocole IPsec pour la protection de données sur les réseaux de niveau diffusion restreinte (DR). Le fait d'imposer l'utilisation de mécanismes d'authentification post-quantiques n'est pas traité dans ce document, car la rédaction des standards l'intégrant dans le protocole n'est pas terminée. Ce document sera mis à jour après la finalisation de ces standards. La prise en compte de la cryptographie post-quantique dans ce document se base sur des standards afin de faciliter l'implémentation par les développeurs.

Ce document est principalement destiné aux développeurs des produits implémentant le protocole IPsec visant une qualification pour la protection de données sur les réseaux de niveau DR. Il vise à favoriser la sécurité et l'interopérabilité des chiffreurs et des clients VPN IPsec visant la qualification.

IPsec [15] est un protocole de communication sécurisé qui permet de protéger un flux de la couche réseau. Il vise à assurer la confidentialité, l'intégrité et la sécurité anti-rejeu des paquets échangés, ainsi que l'authentification des participants. Ce document rappelle d'abord le fonctionnement général du protocole IPsec. Ensuite, il présente les règles et recommandations pour l'implémentation du protocole pour la protection de données sur des réseaux de niveau DR. Enfin, il présente des exemples de configuration de la version 6.1.0 [1] de l'implémentation de strongSwan [2] permettant de respecter ces règles et recommandations. Cette implémentation a été retenue en raison de son usage répandu dans la mise en œuvre d'IPsec. Ce choix répond à un objectif pédagogique et ne constitue pas une recommandation spécifique d'utiliser strongSwan.

1. Version la plus récente lors de la rédaction de ce document
2. https://strongswan.org/

2
Présentation du protocole IPsec

Ce chapitre présente le protocole IPsec ainsi que l'impact de la transition post-quantique sur le protocole. Un autre document publié par l'ANSSI [2] présente un état des lieux plus détaillé de la transition post-quantique du protocole.

2.1 Fonctionnement d'IPsec

IPsec est un protocole de communication sécurisé qui permet de protéger les paquets IP en confidentialité, intégrité et anti-rejeu. Le protocole s'exécute entre un *initiateur* et un *répondeur*, et assure leur authentification mutuelle. L'architecture globale du protocole IPsec est définie dans la RFC 4301 [15]. IPsec est composé de trois protocoles (figure 1) :

- Le protocole IKE exécute la phase d'échange de clés et d'authentification mutuelle entre l'initiateur et le répondeur. La version la plus récente du protocole est IKEv2, définie dans la RFC 7296 [12].
- Les protocoles AH (*Authentication Header*) défini dans la RFC 4302 [13], et ESP (*Encapsulating Security Payload*) défini dans la RFC 4303 [14], protègent les paquets IP. Les deux protocoles assurent une protection en intégrité et anti-rejeu, le protocole ESP assure en plus la confidentialité.

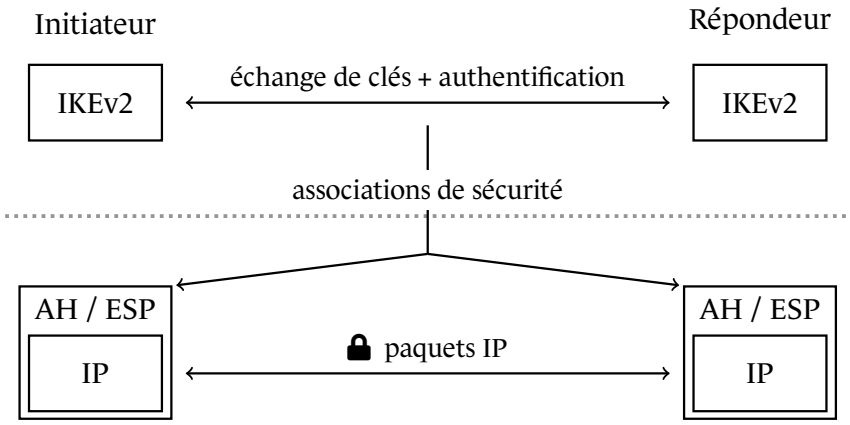


FIGURE 1 – Protocole IPsec

2.1.1 Les modes transport et tunnel d'IPsec

La protection des paquets IP peut se faire avec deux modes : transport ou tunnel (figure 2). Avec le mode transport, la protection est effectuée sur le paquet IP d'origine, alors qu'avec le mode

tunnel, le paquet IP d'origine est encapsulé dans un nouveau paquet IP avec un nouvel entête IP. Le protocole ESP en mode tunnel garantit notamment la confidentialité de l'entête IP d'origine, en particulier des adresses IP source et destination.

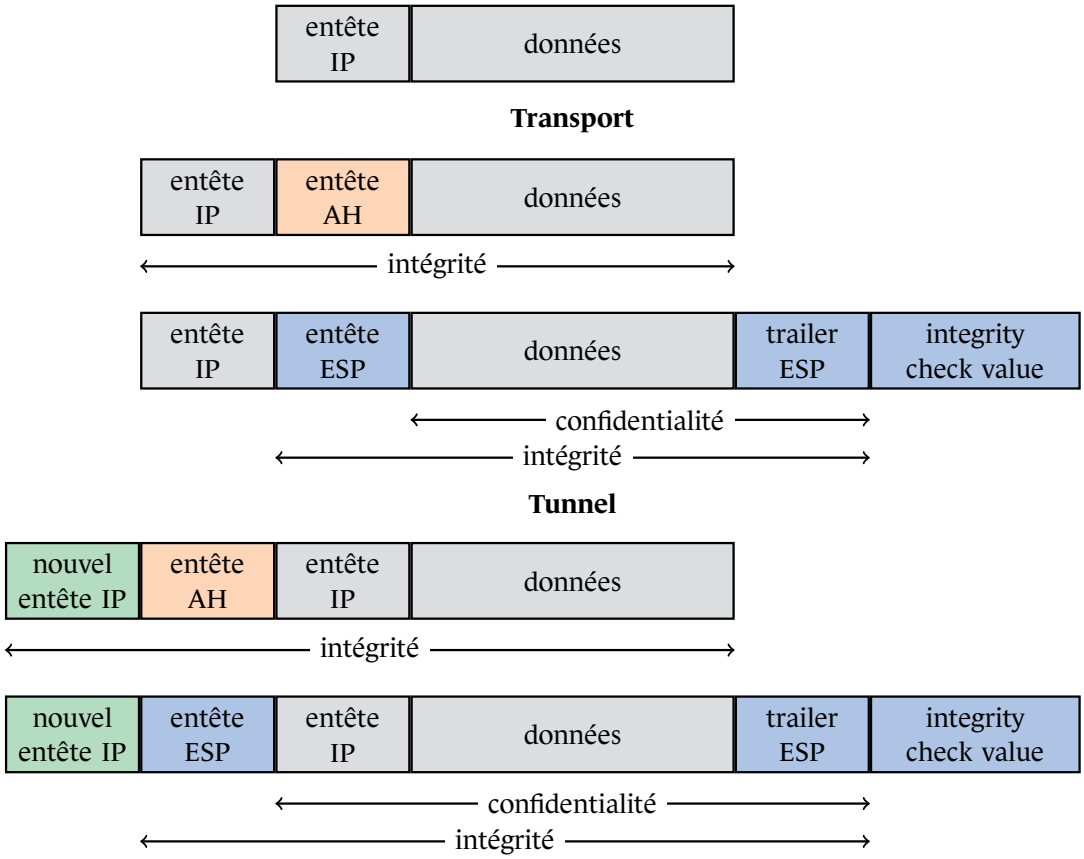


FIGURE 2 – Les modes transport et tunnel d'IPsec

2.1.2 Le protocole IKEv2 et les associations de sécurité

Le protocole IKEv2 permet à un initiateur et à un répondeur de s'authentifier mutuellement, de négocier une suite cryptographique et d'établir des secrets partagés. Un échange IKEv2 se compose d'une paire de messages, constituée d'une requête émise par l'initiateur et d'une réponse envoyée par le répondeur. Afin d'assurer la protection des flux de données, IKEv2 établit des structures appelées associations de sécurité (SA), qui regroupent l'ensemble des paramètres de sécurité, par exemple les suites cryptographiques, les clés symétriques, la taille de la fenêtre anti-rejeu, etc. Deux types d'associations de sécurité sont établies avec IKEv2 (Figure 3) : IKE SA et Child SA.

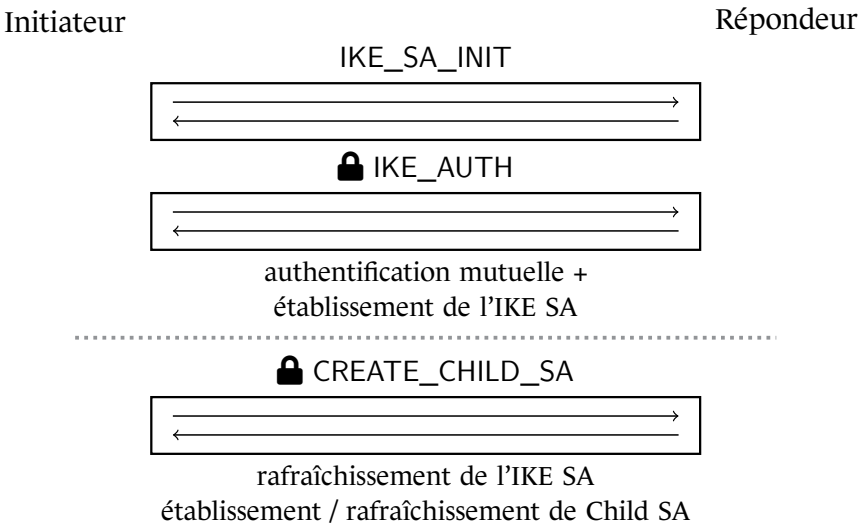


FIGURE 3 – Le protocole IKEv2

- Les premiers échanges entre l'initiateur et le répondeur servent à établir l'association de sécurité IKE SA. Notamment, les messages IKE_SA_INIT servent à négocier les suites cryptographiques et établir un secret partagé, et les messages IKE_AUTH servent à l'authentification mutuelle de l'initiateur et du répondeur. L'association IKE SA créée est bidirectionnelle et définit les paramètres de sécurité pour protéger les échanges dans le tunnel de service IKEv2. Les échanges IKE_AUTH sont notamment chiffrés avec les clés de l'association de sécurité IKE SA, dont l'établissement est finalisé après la fin de l'authentification mutuelle. Cette dernière peut être réalisée par signature et certificat, ou en utilisant une clé pré-partagée.
- Les échanges suivants CREATE_CHILD_SA servent à rafraîchir (*i.e.*, mettre à jour les clés et les paramètres cryptographiques) l'IKE SA, ou à établir et rafraîchir des associations de sécurité Child SA. Ces dernières sont unidirectionnelles et définissent les paramètres de sécurité pour protéger les paquets IP, notamment le protocole (AH ou ESP), le mode (tunnel ou transport) et le flux à protéger (adresse(s) IP source et destination, port(s) source et destination, etc). Ainsi, ces associations de sécurité permettent de contrôler le niveau de granularité pour la protection des paquets IP. Par exemple, une seule paire d'associations de sécurité Child SA peut être établie entre l'initiateur et le répondeur pour protéger tous les paquets IP du tunnel IPsec, ou une paire d'associations de sécurité par connexion TCP, etc.

2.1.3 La fragmentation des messages dans IKEv2

Le protocole IKEv2 ne gère pas la fragmentation de ses messages. De plus, il fonctionne sur le protocole de transport UDP, qui ne gère pas non plus la fragmentation des messages. Ainsi, un message d'IKEv2 encapsulé dans un datagramme UDP, et puis dans un paquet IP, peut être fragmenté par le protocole IP si sa taille dépasse l'unité de transmission maximale (MTU) configurée par l'interface réseau transmettant le paquet. [3] Néanmoins, la fragmentation au niveau IP est considérée comme peu fiable, car la perte d'un seul fragment entraîne le rejet de l'ensemble du paquet. En outre, certains pare-feux décident de rejeter les paquets IP fragmentés pour éviter des attaques par déni de service (*e.g.*, [3, 5, 9]).

3. Pour IPv6, la MTU minimale à implémenter est égale à 1280 octets. En pratique, il existe des implémentations qui fixent la MTU à une taille supérieure (*e.g.*, 1500 octets).

Dans le protocole IKEv2, le risque de la fragmentation a lieu notamment pour les messages d'authentification IKE_AUTH, où les certificats et chaînes de certification peuvent avoir des tailles conséquentes. Pour cela, la RFC 7383 [27] a été publiée en 2014, proposant un mécanisme de gestion de la fragmentation, pour que la bonne réception des fragments soit contrôlée au niveau protocolaire d'IKEv2. Ce mécanisme permet de fragmenter tous les messages IKEv2 chiffrés. Ceci inclut tous les messages, à l'exception de l'échange IKE_SA_INIT (figure 3).

2.2 Transition post-quantique d'IPsec

Les mécanismes de cryptographie asymétrique utilisés aujourd'hui sont vulnérables à la menace quantique. Les protocoles AH et ESP mettent en œuvre des mécanismes symétriques pour protéger les paquets IP, alors que le protocole IKEv2 met en œuvre des mécanismes asymétriques pour les échanges de clés et l'authentification par signature et certificat. Ainsi, seul le protocole IKEv2 est vulnérable à la menace quantique.

En 2023, la RFC 9370 [31] a été publiée, permettant d'effectuer plusieurs échanges de clés successifs dans IKEv2. La solution proposée est résumée dans la Figure 4. Dans cette solution, des messages intermédiaires [28] IKE_INTERMEDIATE et IKE_FOLLOWUP_KE servent à effectuer des échanges de clés supplémentaires. La négociation des algorithmes correspondants se fait toujours dans les messages IKE_SA_INIT et CREATE_CHILD_SA respectivement.

Cette solution permet d'effectuer un échange de clés hybride, avec un échange de clés basé sur le schéma Diffie-Hellman, suivi par un échange de clés basé sur un mécanisme d'encapsulation de clés post-quantique. De plus, les messages intermédiaires sont chiffrés, donc le traitement de leur fragmentation est automatiquement pris en compte grâce au mécanisme dans la RFC 7383 [27]. Ainsi, il est possible d'effectuer des échanges de clés dans ces messages intermédiaires en utilisant des mécanismes d'encapsulation de clés post-quantiques, ayant éventuellement des tailles de clés et de chiffrés conséquentes.

Concernant l'authentification par signature dans IKEv2, l'utilisation de mécanismes de signature post-quantique ou hybride ne nécessite pas de modification des échanges : les messages IKE_AUTH peuvent contenir des certificats et des signatures post-quantiques ou hybrides, leur fragmentation est gérée grâce à la RFC 7383 [27].

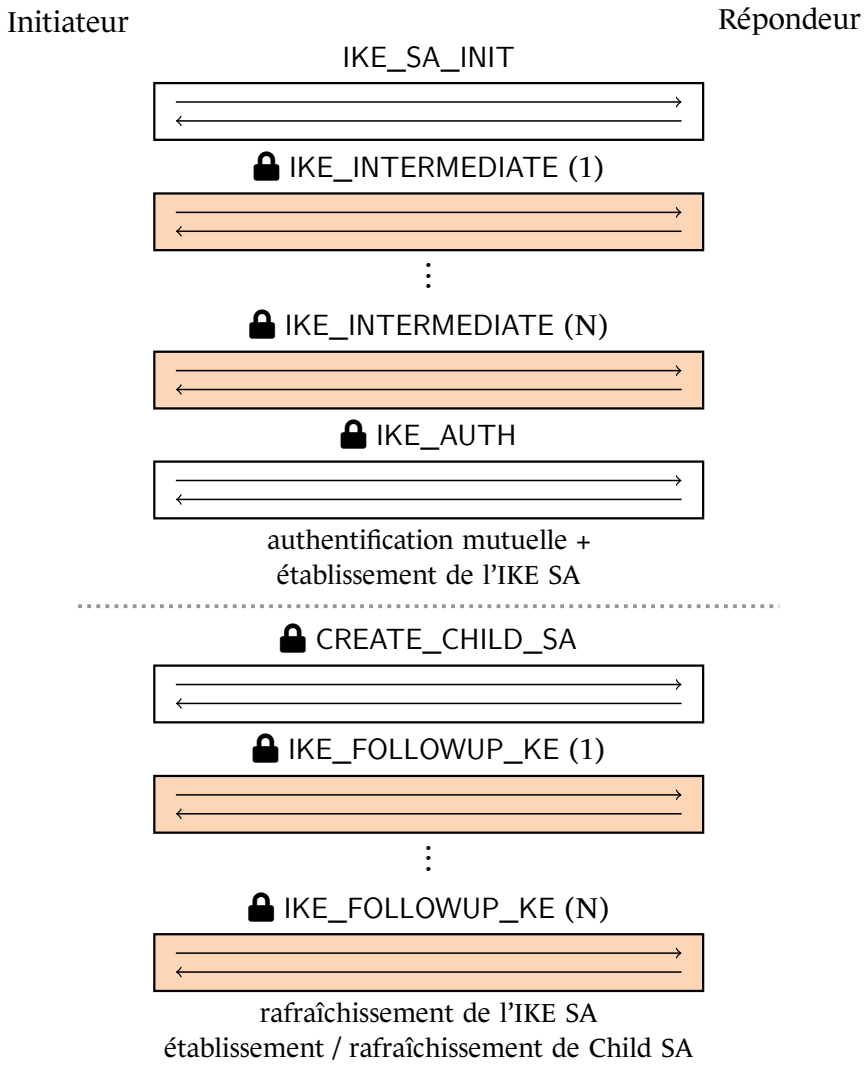


FIGURE 4 – Le protocole IKEv2 avec $N + 1$ échanges de clés tel que proposé dans la RFC 9370 [31]

3
Règles et Recommandations

Ce chapitre décrit les règles et les recommandations pour l'implémentation du protocole IPsec, permettant d'atteindre un niveau de sécurité compatible avec la protection de données DR. Une telle implémentation sera référencée par la suite par « IPsec PQC DR ». Une implémentation d'IPsec respectant toutes les règles de ce chapitre est dite conforme IPsec PQC DR. L'application des recommandations n'est pas formellement requise pour la conformité au référentiel. Cependant, leur implémentation est fortement conseillée.

Règle 1

Implémenter les RFC obligatoires dans IPsec PQC DR

IPsec PQC DR doit implémenter les RFC suivantes :

- RFC 4301 [15] : *Security Architecture for the Internet Protocol*,
- RFC 4303 [14] : *IP Encapsulating Security Payload (ESP)*,
- RFC 6023 [10] : *A Childless Initiation of the Internet Key Exchange Version 2 (IKEv2) Security Association (SA)*,
- RFC 7296 [12] : *Internet Key Exchange Protocol Version 2 (IKEv2)*,
- RFC 7383 [27] : *Internet Key Exchange Protocol Version 2 (IKEv2) Message Fragmentation*,
- RFC 7427 [16] : *Signature Authentication in the Internet Key Exchange Version 2 (IKEv2)*,
- RFC 9242 [28] : *Intermediate Exchange in the Internet Key Exchange Protocol Version 2 (IKEv2)*,
- RFC 9370 [31] : *Multiple Key Exchanges in the Internet Key Exchange Protocol Version 2 (IKEv2)*.

Règle 2

Garantir la sécurité et l'interopérabilité en cas d'implémentation d'autres extensions

IPsec PQC DR peut intégrer d'autres extensions non obligatoires et non interdites dans ce document, à condition de ne pas dégrader la sécurité de l'implémentation, ni nuire à son interopérabilité avec les autres implémentations d'IPsec PQC DR.

Règle 3

Rendre activable un état de configuration conforme à IPsec PQC DR, via un élément unique de configuration

Si le produit implémentant IPsec propose plusieurs choix de configurations du protocole, un élément de configuration simple à utiliser (*e.g.*, case à cocher, commande unique spécifique, etc.) doit permettre l'activation d'un état de configuration conforme à IPsec PQC DR (activation des fonctionnalités requises, désactivation des fonctionnalités interdites, etc.).



Note 3.0.a : Évaluation

L'intégration d'autres extensions (Règle 2) et les choix de configuration (Règle 3) devront être évalués afin de s'assurer qu'ils ne dégradent pas la sécurité et l'interopérabilité de l'implémentation d'IPsec PQC DR.

3.1 Implémentation d'IPsec

Le protocole IPsec permet de contrôler le niveau de granularité du flux de données à protéger. Par exemple, un seul tunnel peut être établi entre l'initiateur et le répondeur pour protéger tous les paquets IP, ou un tunnel par connexion TCP. Les règles de protection sont définies par des sélecteurs de trafic : des plages d'adresses IP source et destination, des plages de ports source et destination, des noms de protocoles de transport, etc. Ainsi, des intersections peuvent exister entre les différentes règles. Par exemple, dans la RFC 4301 [15], la base de données SPD contient les règles de la politique de sécurité. Cette base est répartie en trois parties : SPD-S pour le trafic à protéger, SPD-O pour le trafic sortant à ne pas protéger et SPD-I pour le trafic entrant sans protection. Dans le cas où une intersection existe entre une règle dans SPD-S et une règle dans SPD-O, la règle la plus contraignante qui est indiquée dans SPD-S doit être privilégiée.

Règle 4

Ordonner la consultation des règles de la politique de sécurité

Lors de la consultation des règles de protection de la politique de sécurité dans IPsec PQC DR, la règle la plus contraignante en termes de sécurité doit être appliquée pour le trafic sortant. Par exemple, en cas d'utilisation d'une base de données SPD comme définie dans la RFC 4301 [15], la recherche dans la table SPD-S doit être réalisée avant la table SPD-O.

IPsec PQC DR doit assurer la confidentialité et l'intégrité des paquets IP, via le protocole ESP.

Règle 5

Utiliser obligatoirement le protocole ESP

Il est obligatoire d'utiliser le protocole ESP pour la protection des paquets IP, et d'assurer la confidentialité et l'intégrité de ces derniers. L'utilisation du protocole AH est interdite.

Le protocole ESP propose également un mécanisme de protection contre le rejeu de paquets. Dans IPsec PQC DR, ce mécanisme doit être activé. De plus, la taille de la fenêtre anti-rejeu doit être

définie de manière à limiter le rejet de paquets en cas de déséquencement de paquets ou de débit faible.

Règle 6

Supporter et utiliser l'anti-rejeu dans le protocole ESP

Il est obligatoire de supporter la protection contre le rejeu, et la fenêtre anti-rejeu doit être de taille au moins 1024 paquets.

Dans IPsec PQC DR, le protocole ESP doit être utilisé en mode tunnel (figure 2). En effet, le mode transport ne protège les données de l'entête IP ni en confidentialité ni en intégrité.

Règle 7

Utiliser uniquement le mode tunnel

Il est obligatoire d'utiliser le mode tunnel avec le protocole ESP. L'utilisation du mode transport est interdite.

Dans IPsec PQC DR, la gestion des clés doit être faite en utilisant le protocole IKEv2. En effet, la gestion manuelle des clés est complexe et peut vite introduire des problèmes de sécurité. Le protocole IKE est le protocole de gestion de clés automatique défini et privilégié pour être utilisé avec IPsec. De plus, sa dernière version (IKEv2) est la plus robuste.

Règle 8

Utiliser le protocole IKEv2

Il est obligatoire d'utiliser le protocole IKEv2, définie dans la RFC 7296 [12], pour la gestion des clés et des associations de sécurité dans IPsec PQC DR.

Le protocole IKEv2 permet d'établir et de gérer les associations de sécurité utilisées pour protéger les données. Il existe des associations de sécurité « imbriquées » (*nested SAs*) qui permettent certaines utilisations comme la séparation de flux. Néanmoins, leur gestion complexe peut apporter une incertitude sur le comportement et engendre ainsi des risques de sécurité. L'intérêt de leur utilisation est moindre par rapport aux risques associés pour la protection des données DR.

Règle 9

Proscrire les nested SA

Les associations de sécurité imbriquées ou *nested SAs* sont interdites dans IPsec PQC DR.

Le protocole IKEv2 permet de rafraîchir les associations de sécurité déjà établies, en particulier les clés cryptographiques. Ce rafraîchissement est nécessaire pour renforcer la propriété de confidentialité persistante (ou *Perfect Forward Secrecy* (PFS)) des échanges IKEv2 et des paquets IP protégés.

Règle 10

Supporter le rafraîchissement des associations de sécurité

Il est obligatoire de mettre en œuvre le rafraîchissement des associations de sécurité dans le protocole IKEv2, sans devoir réinitialiser l'association IKE SA en cours.

L'utilisation d'un système de traduction d'adresses (NAT) avec le protocole IPsec peut causer des problèmes. Par exemple, un NAT peut modifier le numéro de port d'un paquet et mettre à jour

la somme de contrôle (*checksum*). Cette action n'est pas compatible avec le chiffrement effectué par le protocole ESP et peut provoquer le rejet de paquets par le destinataire. La mise en place du mécanisme NAT-*Traversal* permet d'éviter ces problèmes.

Reco 1

Utiliser le mécanisme NAT-traversal

Il est recommandé de mettre en place le mécanisme NAT-*Traversal*. De plus, il est recommandé d'utiliser uniquement le port UDP 4500 pour IKEv2. Enfin, il est recommandé que l'ensemble des paquets ESP soit transporté encapsulé sur le port UDP 4500.

La compression des paquets IP peut être mise en place dans le protocole IPsec et négociée dans le protocole IKEv2. Cette compression présente quelques avantages comme le gain de bande passante, intéressant notamment dans le cas de liaison coûteuse (*e.g.*, satellite). Néanmoins, la compression IP peut introduire des problèmes comme la complexité de gestion de l'implémentation. Ainsi, il est recommandé de l'utiliser uniquement si c'est nécessaire dans IPsec PQC DR.

Reco 2

Éviter la compression IP

Il est recommandé d'implémenter la compression IP dans IPsec PQC DR uniquement en cas de besoin clair. En cas de réseaux à large bande passante, il est recommandé de ne pas l'utiliser.

3.1.1 Moyens d'authentification dans IPsec

Pendant le protocole IKEv2, l'initiateur et le répondeur s'authentifient mutuellement. Plusieurs moyens d'authentification sont possibles. Dans IPsec PQC DR, seules l'authentification par signature et l'authentification par clé pré-partagée sont autorisées.

Règle 11

Limiter les moyens d'authentification autorisés dans IKEv2

Les seuls moyens d'authentification autorisés dans IKEv2 sont l'authentification par signature et l'authentification par clé pré-partagée. Il est interdit d'utiliser les méthodes d'authentification EAP (*Extensible Authentication Protocol*).

En cas d'authentification par signature, un certificat validant l'identité du signataire doit être utilisé. Un certificat auto-signé, n'étant pas délivré par une autorité de confiance, ne prouve pas l'identité et ne doit pas être utilisé. De plus, le format X.509 de certificat, ainsi que les mécanismes de révocation CRL et OCSP doivent être supportés pour assurer la sécurité et l'interopérabilité.

Règle 12

Proscrire l'authentification avec des certificats auto-signés

En cas d'authentification par signature, cette dernière doit être accompagnée d'un certificat délivré par une autorité de confiance. L'authentification par certificat auto-signé est interdite.

Règle 13

Supporter le format de certificat X.509 et proscrire Hash and URL

Pour l'authentification par signature, le format de certificat X.509 doit être supporté. Le format *Hash and URL* est interdit.

Règle 14

Implémenter les mécanismes de révocation CRL et OCSP

Les mécanismes de révocation CRL et OCSP doivent être implémentés.

Note 3.1.a : Réauthentification dans IKEv2

Après l'authentification pendant les échanges IKE_AUTH, l'association de sécurité IKE SA est établie et sa durée de vie peut être définie par l'implémentation. Cette durée de vie est à définir de manière cohérente avec les cycles de vie des certificats utilisés et les politiques de révocation. Une réauthentification mutuelle est nécessaire pour s'assurer toujours de la validité des certificats et des identités de l'initiateur et du répondeur.

3.2 Implémentation d'IKEv2

Dans IKEv2, un initiateur qui envoie une requête doit attendre la réponse à sa requête avant d'en envoyer une deuxième, sauf s'il avait reçu du répondeur la notification SET_WINDOW_SIZE. En effet, cette dernière permet de notifier que l'implémentation émettrice de la notification est capable de traiter plusieurs requêtes à la fois. Toutefois, la gestion du chevauchement de requêtes est complexe et peut générer des problèmes de sécurité et de performance.

Règle 15

Proscrire le chevauchement de requêtes IKEv2

Il est interdit de négocier la taille de fenêtre pour le chevauchement de requêtes (interdiction d'utiliser SET_WINDOW_SIZE). Cette fenêtre doit toujours être de taille 1.

Pendant la première négociation du protocole IKEv2, l'association de sécurité IKE SA est établie avec l'authentification mutuelle de l'initiateur et du répondeur (figure 3). Pendant ces échanges, la première paire d'associations de sécurité Child SA peut également être établie, avec des paramètres négociés dans les échanges IKE_AUTH. En revanche, les clés générées pour ces associations de sécurité Child SA sont dérivées à partir des mêmes secrets partagés que l'association de sécurité IKE SA. En d'autres termes, il n'y a pas de second échange de clés effectué pour ces associations. Ainsi, dans IPsec PQC DR, il est interdit de créer des associations de sécurité Child SA à ce stade. La RFC 6023 [10] répond à cette interdiction. La création des associations de sécurité Child SA doit se faire uniquement grâce à des échanges CREATE_CHILD_SA, pour assurer la propriété de confidentialité persistante (ou *Perfect Forward Secrecy* (PFS)).

Règle 16

Utiliser la négociation « childless initiation » dans IKEv2

Il est obligatoire d'utiliser la négociation *childless initiation* définie dans la RFC 6023 [10].

Après les premiers échanges du protocole IKEv2, l'initiateur et le répondeur sont mutuellement authentifiés et leurs échanges IKEv2 sont protégés en confidentialité et intégrité. De nouvelles paires d'associations de sécurité Child SA peuvent être créées, avec les échanges CREATE_CHILD_SA. Ces derniers sont protégés avec l'association de sécurité IKE SA en cours. L'échec de l'établissement des associations de sécurité Child SA n'invalide pas l'association de sécurité IKE SA active, qui ne doit pas être supprimée d'emblée. Si les tentatives infructueuses se multiplient, d'autres mécanismes peuvent être utilisés, pouvant ensuite supprimer l'association IKE SA, comme par exemple la détection de la perte de connectivité [12].

Règle 17

Proscrire la suppression de l'association de sécurité IKE SA directement après un échec de création de Child SA

L'échec de l'établissement des associations de sécurité Child SA ne doit pas directement entraîner la suppression de l'association de sécurité IKE SA déjà établie. La répétition d'échecs successifs doit déclencher d'autres mécanismes pouvant conduire à la suppression de l'association IKE SA.



Note 3.2.a : Évaluation

La pertinence des mécanismes déclenchés après des échecs successifs de l'établissement des associations de sécurité Child SA (Règle 17) devra être évaluée afin de s'assurer qu'ils ne dégradent pas la sécurité et l'interopérabilité de l'implémentation d'IPsec PQC DR.

Lors d'une négociation de suites cryptographiques, l'initiateur peut être amené à faire une nouvelle tentative en cas d'échec de la première. Par exemple, un échec peut avoir lieu si l'initiateur envoie une clé publique Diffie-Hellman pour son groupe de préférence, différent du groupe choisi par le répondeur. Pendant une nouvelle tentative, l'initiateur doit proposer la même liste de suites cryptographiques, pour éviter la manipulation de la sélection de la suite cryptographique.

Règle 18

Négocier la même liste de mécanismes cryptographiques lors d'une nouvelle tentative d'envoi de suites cryptographiques

En cas de nouvelle tentative d'envoi de suites cryptographiques par l'initiateur, ce dernier doit toujours inclure la même liste des mécanismes cryptographiques pour éviter la manipulation de la sélection de la suite cryptographique.

Le protocole IKEv2 propose l'utilisation d'un mécanisme de *cookies* pour protéger une implémentation IKEv2 contre les attaques par déni de service. Lorsqu'un répondeur est surchargé et reçoit des messages d'initiation IKE_SA_INIT, il peut répondre avec un message contenant une valeur appelée *cookie*, invitant l'initiateur à retenter la connexion plus tard. Pendant une nouvelle tentative, l'initiateur renvoie le *cookie* qu'il a reçu, pour permettre au répondeur d'identifier la connexion. Pour que ce mécanisme de *cookies* protège réellement contre les attaques par déni de service, il faut que la valeur du *cookie* soit générée de manière sécurisée et imprédictible pour un attaquant. Par exemple, la valeur du *cookie* peut contenir la valeur hachée d'un secret généré aléatoirement et connu uniquement par le répondeur.

Règle 19

Utiliser un mécanisme de cookie robuste dans IKEv2

Pour se protéger contre les attaques par déni de service, il est obligatoire de supporter et d'utiliser systématiquement le mécanisme de *cookie* dans IKEv2. De plus, la valeur du *cookie* doit être générée de manière sécurisée et imprédictible.

La RFC 7296 [12] propose un mécanisme de *cookie* dont la génération est robuste et satisfait les propriétés de sécurité nécessaires.

Reco 3

Implémenter le mécanisme de cookie proposé dans la RFC 7296

Il est recommandé d'utiliser le mécanisme de *cookie* défini dans la RFC 7296 [12].

La RFC 7296 [12] liste plusieurs erreurs qui peuvent avoir lieu pendant les échanges du protocole IKEv2. De manière générale, répondre à des requêtes IKEv2 avec des messages d'erreur précis peut causer d'éventuelle fuite d'information ou des risques de sécurité. En fonction des différents cas d'erreur, il peut être prudent de privilégier le silence, comme indiqué dans la RFC.

Reco 4

Privilégier le silence dans les échanges IKEv2

Il est recommandé que l'implémentation du protocole IKEv2 soit silencieuse dans le traitement des erreurs pendant les échanges IKEv2, comme précisé dans la RFC 7296 [12].

3.2.1 Implémentation de la cryptographie post-quantique dans IKEv2

Comme expliqué dans les sections 2.1.3 et 2.2, le protocole IKEv2 ne gère pas nativement la fragmentation des messages échangés. Dans le cadre de la transition post-quantique, l'utilisation de mécanismes d'encapsulation de clés post-quantiques pour l'échange de clés résulte en des messages de taille qui provoque la fragmentation IP. Ainsi, il est obligatoire de supporter le mécanisme de la fragmentation des messages introduit dans la RFC 7383 [27] pour pouvoir effectuer des échanges de clés hybrides sans risque de perte de messages.

Règle 20

Implémenter la fragmentation dans IKEv2

Il est obligatoire d'implémenter la fragmentation des messages dans le protocole IKEv2, définie dans la RFC 7383 [27].

La RFC 9370 [31] étend le protocole IKEv2 pour effectuer des échanges de clés multiples, comme décrit brièvement dans la section 2.2. Cette extension permet notamment d'effectuer des échanges de clés hybrides, pour assurer la confidentialité hybride des messages du protocole IKEv2.

Règle 21

Implémenter uniquement des échanges de clés hybrides dans IKEv2

Il est obligatoire d'implémenter des échanges de clés hybrides dans le protocole IKEv2, basés sur la RFC 9370 [31] et le mécanisme des messages intermédiaires de la RFC 9242 [28].

Pendant le premier échange de clés pour l'établissement de l'association de sécurité IKE SA, il est obligatoire de commencer par un échange de clés basé sur le schéma Diffie-Hellman dans les messages IKE_SA_INIT, suivi par un échange de clés basé sur un KEM (Key Encapsulation Mechanism) post-quantique dans des messages IKE_INTERMEDIATE. En effet, les messages IKE_SA_INIT ne peuvent pas être fragmentés par le mécanisme de la RFC 7383 [27], qui ne concerne que les messages chiffrés.

Règle 22

Implémenter l'échange de clés hybride dans IKEv2 (IKE_SA_INIT)

Dans un échange de clés hybride pendant l'établissement de l'association de sécurité IKE SA, un premier échange de clés basé sur le schéma Diffie-Hellman doit être effectué dans les messages IKE_SA_INIT, suivi par un second échange de clés basé sur un KEM post-quantique dans les messages intermédiaires IKE_INTERMEDIATE. Ces échanges doivent être effectués avant les messages IKE_AUTH.

Les messages CREATE_CHILD_SA permettent de rafraîchir l'association de sécurité IKE SA en cours, ou de créer ou rafraîchir une paire d'associations de sécurité Child SA. Dans IPsec PQC DR, il est obligatoire d'effectuer un échange de clés hybride dans tous les échanges CREATE_CHILD_SA. En particulier, un échange de clés basé sur le schéma Diffie-Hellman doit être effectué dans les messages CREATE_CHILD_SA, suivi par un échange de clés basé sur un KEM post-quantique dans les messages IKE_FOLLOWUP_KE.

Règle 23

Implémenter l'échange de clés hybride dans IKEv2 (CREATE_CHILD_SA)

Il est obligatoire d'effectuer un échange de clés hybride dans tous les échanges CREATE_CHILD_SA (que ce soit pour le rafraîchissement de l'association de sécurité IKE SA, ou pour la création ou le rafraîchissement d'une paire d'associations de sécurité Child SA). Pour cela, un premier échange de clés basé sur le schéma Diffie-Hellman sera effectué dans les messages CREATE_CHILD_SA, suivi par un second échange de clés basé sur un KEM post-quantique dans les messages intermédiaires IKE_FOLLOWUP_KE.

Pour assurer la propriété de confidentialité persistante (ou *Perfect Forward Secrecy* (PFS)), chaque échange de clés doit être effectué avec des clés fraîchement générées.

Règle 24

Utiliser des clés éphémères à chaque nouvel échange de clés

Pour chaque échange de clés dans le protocole IKEv2, il est obligatoire d'utiliser des clés fraîchement générées pour le schéma Diffie-Hellman et le KEM post-quantique.

La RFC 8784 [6] propose une extension du protocole IKEv2 permettant d'utiliser une clé prépartagée pour garantir la confidentialité, en combinaison avec l'échange de clés basé sur le schéma Diffie-Hellman. La clé utilisée est différente de celle utilisée pour l'authentification par clé prépartagée. Néanmoins, la solution proposée permet de protéger uniquement les paquets IP avec les

futures associations de sécurité (Child SA). Ainsi, la solution n'offre pas de protection en confidentialité pour les messages du protocole IKEv2 (IKE SA). Par la suite, la RFC 9867 [29] a été publiée. Cette dernière propose une solution qui étend l'utilisation de la clé pré-partagée pour dériver aussi les clés qui protègent les échanges du protocole IKEv2. D'autre part, ces solutions n'assurent pas la propriété de confidentialité persistante en cas de compromission des clés pré-partagées. Ainsi, dans IPsec PQC DR, la confidentialité des échanges dans le protocole IKEv2 doit être assurée en utilisant des échanges de clés hybrides (Règle 21, Règle 22, Règle 23).

Règle 25

Proscrire l'utilisation de clé pré-partagée pour assurer la confidentialité des échanges dans le protocole IKEv2

Il est interdit d'utiliser une clé pré-partagée pour assurer la confidentialité des échanges IKEv2, telles que les solutions définies dans les RFC 8784 [6] et 9867 [29].

3.3 Mécanismes cryptographiques dans IPsec PQC DR

L'objectif de cette section est de présenter des règles concernant le choix des mécanismes cryptographiques dans les protocoles ESP et IKEv2. Il s'agit de limiter, via un fichier de configuration, les mécanismes cryptographiques utilisables dans ces protocoles. À des fins d'interopérabilité des « **mécanismes obligatoires** » sont distingués : ils doivent obligatoirement être proposés pour être conforme IPsec PQC DR. En outre, des « **mécanismes facultatifs** » sont également autorisés et peuvent être proposés : ils offrent des alternatives acceptables aux mécanismes obligatoires. Ces règles sont déclinées pour les mécanismes cryptographiques utilisés pour chiffrer les messages (section 3.3.1), en protéger l'intégrité (section 3.3.2), générer ou dériver des clés (section 3.3.3), échanger des clés (section 3.3.4) et s'authentifier (section 3.3.5).

Les tableaux des sections 3.3.4 et 3.3.5 disposent d'une colonne T/PQ. Un mécanisme cryptographique T (pour « traditionnel ») fournit un niveau de sécurité satisfaisant contre un adversaire classique mais, utilisé seul, n'offre pas de sécurité face à un adversaire quantique. En revanche, un mécanisme cryptographique PQ (pour « post-quantique ») fournit une résistance aux attaques quantiques. Les mécanismes PQ étant récents, il est nécessaire de les hybrider avec au moins un mécanisme T pour éviter une régression de sécurité. Un mécanisme T+PQ offre un niveau de sécurité suffisant vis-à-vis des adversaires classiques et quantiques, qu'il s'agisse d'un mécanisme unique ou d'une combinaison hybride.

3.3.1 Chiffrement des messages

Règle 26

Mécanismes de chiffrement des messages obligatoires

Les mécanismes de chiffrement des messages suivants doivent obligatoirement être proposés.

Mécanisme	Taille de clé	IANA	Spécification	Notes
ENCR_AES_CTR	256 bits	13	[18]	3.3.a, 3.3.b
ENCR_AES_GCM_16	256 bits	20	[4, 23]	3.3.a

Règle 27

Mécanismes de chiffrement des messages facultatifs autorisés

En complément des mécanismes obligatoires prévus par la Règle 26, seuls les mécanismes de chiffrement des messages suivants peuvent être proposés à titre facultatif.

Mécanisme	Taille de clé	IANA	Spécification	Notes
ENCR_AES_CBC	192 bits	12	[12]	3.3.a, 3.3.b
	256 bits			
ENCR_AES_CTR	192 bits	13	[18]	3.3.a, 3.3.b
ENCR_AES_CCM_16	192 bits	16	[4]	3.3.a
	256 bits			
ENCR_AES_GCM_16	192 bits	20	[4, 23]	3.3.a



Note 3.3.a : Génération de clés

Les clés utilisées par ces algorithmes cryptographiques doivent être générées conformément au protocole à l'aide d'une des fonctions pseudo-aléatoires présentées dans la section 3.3.3.



Note 3.3.b : Intégrité

Les mécanismes ENCR_AES_CBC et ENCR_AES_CTR ne fournissent pas, seuls, de protection en intégrité. Ils doivent donc nécessairement être combinés avec un des mécanismes protégeant l'intégrité des messages présentés dans la section 3.3.2.

3.3.2 Intégrité des messages

Règle 28

Mécanisme d'intégrité des messages obligatoire

Le mécanisme d'intégrité des messages AUTH_HMAC_SHA2_256_128 doit obligatoirement être proposé.

Mécanisme	Taille de clé	IANA	Spécification
AUTH_HMAC_SHA2_256_128	256 bits	12	[7]

Règle 29

Mécanismes d'intégrité des messages facultatifs autorisés

En complément du mécanisme obligatoire prévu par la Règle 28, seuls les mécanismes d'intégrité des messages suivants peuvent être proposés à titre facultatif.

Mécanisme	Taille de clé	IANA	Spécification
AUTH_HMAC_SHA2_384_192	384 bits	13	[7]
AUTH_HMAC_SHA2_512_256	512 bits	14	[7]

3.3.3 Fonctions pseudo-aléatoires pour la génération des clés

Règle 30

Fonction pseudo-aléatoire obligatoire

La fonction pseudo-aléatoire `PRF_HMAC_SHA2_256` doit obligatoirement être proposée.

Mécanisme	IANA	Spécification
`PRF_HMAC_SHA2_256`	5	[7]

Règle 31

Fonctions pseudo-aléatoires facultatives

En complément du mécanisme obligatoire prévu par la Règle 30, seules les fonctions pseudo-aléatoires suivantes peuvent être proposées à titre facultatif.

Mécanisme	IANA	Spécification
`PRF_HMAC_SHA2_384`	6	[7]
`PRF_HMAC_SHA2_512`	7	[7]

3.3.4 Échange de clé

Règle 32

Mécanismes d'échange de clé obligatoires

Les mécanismes d'échange de clé suivants doivent obligatoirement être proposés. En outre, il est obligatoire de combiner, via le mécanisme d'hybridation d'IKEv2, au moins un mécanisme T avec au moins un mécanisme PQ.

Mécanisme	IANA	Spécification	T/PQ	Notes
`256-bit random ECP group`	19	[30]	T	
`brainpoolP256r1`	28	[17]	T	
`ml-kem-768`	36	[11]	PQ	
`ml-kem-1024`	37	[11]	PQ	

Règle 33

Mécanismes d'échange de clé facultatifs autorisés

En complément des mécanismes obligatoires prévus par la Règle 32, seuls les mécanismes d'échange de clé suivants peuvent être proposés à titre facultatif. En outre, l'exigence d'hybridation de la Règle 32 s'applique également.

Mécanisme	IANA	Spécification	T/PQ	Notes
`384-bit random ECP group`	20	[30]	T	
`521-bit random ECP group`	21	[30]	T	
`brainpoolP384r1`	29	[17]	T	
`brainpoolP512r1`	30	[17]	T	
`Curve25519`	31	[22]	T	
`Curve448`	32	[22]	T	

3.3.5 Authentification

Règle 34

Mécanismes d'authentification obligatoires

Les mécanismes d'authentification suivants doivent obligatoirement être proposés.

Mécanisme	IANA	Spécification	T/PQ
ECDSA_SHA-256_secp256r1	9	[8, 30]	T
ECDSA_SHA-256_brainpoolP256r1	14	[16]	T

Règle 35

Mécanismes d'authentification facultatifs autorisés

En complément des mécanismes obligatoires prévus par la Règle 34, seuls les mécanismes d'authentification suivants peuvent être proposés à titre facultatif. Si un mécanisme PQ est utilisé, il est obligatoire de le combiner, via un mécanisme d'hybridation de signature, avec au moins un mécanisme T. Les mécanismes T+PQ peuvent être utilisés seuls.

Mécanisme	IANA	Spécification	T/PQ	Notes
ECDSA_SHA-384_secp384r1	10	[8, 30]	T	
ECDSA_SHA-512_secp521r1	11	[8, 30]	T	
ECDSA_SHA-384_brainpoolP384r1	14	[16]	T	
ECDSA_SHA-512_brainpoolP512r1	14	[16]	T	
Ed25519	14	[16, 21]	T	
Ed448	14	[16, 21]	T	
ML-DSA-65	14	[16, 19, 25, 26]	PQ	3.3.e
ML-DSA-87	14	[16, 19, 25, 26]	PQ	3.3.e
SLH-DSA-*-192*	14	[16, 20, 26]	T+PQ	3.3.e
SLH-DSA-*-256*	14	[16, 20, 26]	T+PQ	3.3.e
Preshared Key	2	[12]	T+PQ	3.3.d

Note 3.3.c : Authentification post-quantique

Lorsque des mécanismes d'authentification post-quantiques tels que ML-DSA [19, 25, 26] et SLH-DSA [20, 26] seront intégrés définitivement dans la norme IKEv2, **le présent guide sera mis à jour pour imposer leur utilisation**. L'utilisation de ML-DSA sera nécessairement combinée à celle d'un mécanisme T via un mécanisme d'hybridation. On renvoie à [24] pour la prise en compte de cette hybridation dans les certificats X.509, et à [25] pour un draft de RFC sur la prise en compte dans le protocole IKE.
L'absence de prise en charge de ces mécanismes d'authentification post-quantiques pourra donc entraîner une non-conformité à la prochaine version de ce guide.



Note 3.3.d : Clé pré-partagée

L'authentification par clé pré-partagée doit être réalisée conformément au protocole à l'aide d'une des fonctions pseudo-aléatoires présentées dans la section 3.3.3. La clé pré-partagée utilisée pour l'authentification doit être générée et stockée de manière sécurisée : elle doit être de longueur 256 bits, d'entropie maximale et être protégée en confidentialité et en intégrité lors de son stockage.



Note 3.3.e : Variantes de ML-DSA et SLH-DSA

Il est recommandé d'utiliser les variantes dites « hedged » de ces algorithmes [19, 20].

Bibliographie

[1] ANSSI. Corpus documentaire IPsec DR à destination des industriels – Version 1.0. https://cyber.gouv.fr/nous-connaitre/publications/publications-scientifiques/corpus-documentaire-ipsec-dr-a-destination-des-industriels.

[2] ANSSI. Transition post-quantique d'IPsec. https://messervices.cyber.gouv.fr/guides/Transition-post-quantique-protocole-IPsec.

[3] Antonios Atlasis. Attacking ipv6 implementation using fragmentation. *Blackhat europe*, pages 14–16, 2012.

[4] David L. Black and David McGrew. Using Authenticated Encryption Algorithms with the Encrypted Payload of the Internet Key Exchange version 2 (IKEv2) Protocol. RFC 5282, August 2008.

[5] Jeffrey Bosma, Benno Overeinder, and Willem Toorop. Discovering Path MTU black holes on the Internet using RIPE Atlas, 2012.

[6] Scott Fluhrer, Panos Kampanakis, David McGrew, and Valery Smyslov. Mixing Preshared Keys in the Internet Key Exchange Protocol Version 2 (IKEv2) for Post-quantum Security. RFC 8784, June 2020.

[7] Sheila Frankel and Scott G. Kelly. Using HMAC-SHA-256, HMAC-SHA-384, and HMAC-SHA-512 with IPsec. RFC 4868, May 2007.

[8] David E. Fu and Jerome Solinas. IKE and IKEv2 Authentication Using the Elliptic Curve Digital Signature Algorithm (ECDSA). RFC 4754, January 2007.

[9] Justin Iurman and Benoit Donnet. The Razor's Edge : IPv6 Extension Headers Survivability. In *International Conference on Passive and Active Network Measurement*, pages 3–29. Springer, 2025.

[10] Rajeshwar Jenwar, DENG Hui, Hannes Tschofenig, and Yoav Nir. A Childless Initiation of the Internet Key Exchange Version 2 (IKEv2) Security Association (SA). RFC 6023, October 2010.

[11] Panos Kampanakis. Post-quantum Key Exchange with ML-KEM in the Internet Key Exchange Protocol Version 2 (IKEv2). Internet-Draft draft-ietf-ipsecme-ikev2-mlkem-09, Internet Engineering Task Force, July 2026. Work in Progress.

[12] Charlie Kaufman, Paul E. Hoffman, Yoav Nir, Pasi Eronen, and Tero Kivinen. Internet Key Exchange Protocol Version 2 (IKEv2). RFC 7296, October 2014.

[13] Stephen Kent. RFC 4302 : IP authentication header, 2005.

[14] Stephen Kent. RFC 4303 : IP encapsulating security payload (ESP), 2005.

[15] Stephen Kent and Karen Seo. RFC 4301 : Security architecture for the Internet protocol, 2005.

[16] Tero Kivinen and Joel Snyder. Signature Authentication in the Internet Key Exchange Version 2 (IKEv2). RFC 7427, January 2015.

[17] Johannes Merkle and Manfred Lochter. Using the Elliptic Curve Cryptography (ECC) Brainpool Curves for the Internet Key Exchange Protocol Version 2 (IKEv2). RFC 6954, July 2013.

[18] S murthy, Sean Shen, and Yu Mao. Using Advanced Encryption Standard Counter Mode (AES-CTR) with the Internet Key Exchange version 02 (IKEv2) Protocol. RFC 5930, July 2010.

[19] National Institute of Standards and Technology. FIPS PUB 204 : Module-Lattice-Based Digital Signature Standard, 2024.

[20] National Institute of Standards and Technology. FIPS PUB 205 : Stateless Hash-Based Digital Signature Standard, 2024.

[21] Yoav Nir. Using the Edwards-Curve Digital Signature Algorithm (EdDSA) in the Internet Key Exchange Protocol Version 2 (IKEv2). RFC 8420, August 2018.

[22] Yoav Nir and Simon Josefsson. Curve25519 and Curve448 for the Internet Key Exchange Protocol Version 2 (IKEv2) Key Agreement. RFC 8031, December 2016.

[23] Yoav Nir, Tero Kivinen, Paul Wouters, and Daniel Migault. Algorithm Implementation Requirements and Usage Guidance for the Internet Key Exchange Protocol Version 2 (IKEv2). RFC 8247, September 2017.

[24] Mike Ounsworth, John Gray, Massimiliano Pala, Jan Klaußner, and Scott Fluhrer. Composite Module-Lattice-Based Digital Signature Algorithm (ML-DSA) for use in X.509 Public Key Infrastructure. RFC 10073, November 2026.

[25] Tirumaleswar Reddy.K, Scott Fluhrer, Jun Hu, Yasufumi Morioka, and Guilin WANG. Use of Composite Signatures in IKEv2. Internet-Draft draft-ietf-ipsecme-pq-hybrid-auth-01, Internet Engineering Task Force, November 2026. Work in Progress.

[26] Tirumaleswar Reddy.K, Valery Smyslov, and Scott Fluhrer. Signature Authentication in the Internet Key Exchange Version 2 (IKEv2) using PQC. Internet-Draft draft-ietf-ipsecme-ikev2-pqc-auth-12, Internet Engineering Task Force, August 2026. Work in Progress.

[27] Valery Smyslov. Internet Key Exchange Protocol Version 2 (IKEv2) Message Fragmentation. RFC 7383, November 2014.

[28] Valery Smyslov. Intermediate Exchange in the Internet Key Exchange Protocol Version 2 (IKEv2). RFC 9242, May 2022.

[29] Valery Smyslov. Mixing Preshared Keys in the IKE_INTERMEDIATE and CREATE_CHILD_SA Exchanges of the Internet Key Exchange Protocol Version 2 (IKEv2) for Post-Quantum Security. RFC 9867, November 2025.

[30] Jerome Solinas and David E. Fu. Elliptic Curve Groups modulo a Prime (ECP Groups) for IKE and IKEv2. RFC 5903, June 2010.

[31] C. Tjhai, M. Tomlinson, G. Bartlett, Scott Fluhrer, Daniel Van Geest, Oscar Garcia-Morchon, and Valery Smyslov. Multiple Key Exchanges in the Internet Key Exchange Protocol Version 2 (IKEv2). RFC 9370, May 2023.

Annexe A
Configuration strongSwan

strongSwan [4] est un logiciel open source qui implémente IKEv2 et IPsec.
Afin d'illustrer les règles et les recommandations formulées dans ce guide, cette annexe propose des exemples de configuration de la version 6.1.0 [5] de strongSwan permettant de respecter ces règles et recommandations.
Les configurations proposées dans cette annexe ne sont pas exhaustives et la conformité aux règles et recommandations peut parfois être atteinte par des options de configuration différentes.
Une configuration adéquate de strongSwan permet de respecter les règles et les recommandations de ce référentiel, à l'exception des règles Règle 1, Règle 2, et Règle 3 qui sont hors du périmètre de strongSwan. La reprise à l'identique des configurations proposées ne saurait, à elle seule, garantir le respect de l'ensemble des règles et recommandations du présent guide.

L' implémentation de strongSwan a été retenue en raison de son usage fréquent dans les environnements mettant en œuvre IPsec. Ce choix répond à un objectif pédagogique et ne constitue pas une recommandation spécifique d'utiliser strongSwan.

Règle 4 Ordonner la consultation des règles de la politique de sécurité

strongSwan n'implémente pas la gestion des politiques de sécurité au moyen des bases de données SPD comme dans la RFC 4301 [15]. La configuration par défaut de strongSwan peut respecter cette règle lorsque l'ordre attendu découle de la précision des sélecteurs de trafic. Sous Linux, par exemple, les politiques IPsec installées dans la SPD/XFRM sont consultées par priorité croissante et la clé connections.<conn>.children.<child>.priority du fichier /etc/swanctl/swanctl.conf vaut par défaut 0, ce qui laisse strongSwan calculer automatiquement leur priorité en privilégiant les politiques les plus spécifiques. En présence de politiques qui se chevauchent et dont l'ordre ne peut pas être déduit des sélecteurs, cette priorité doit être définie explicitement sur le composant qui applique la politique. Les valeurs à retenir dépendent de la politique globale de l'équipement et ne peuvent pas être recommandées de manière générique.

4. https://strongswan.org/
5. Version la plus récente lors de la rédaction de ce document

Règle 5 **Utiliser obligatoirement le protocole ESP**

La configuration par défaut de strongSwan respecte cette règle.
Configurer les suites cryptographiques pour le protocole ESP, avec <child>.esp_proposals pour chaque connections.<conn>.children, dans le fichier de configuration swanctl.conf. Ne pas configurer de suite cryptographique pour le protocole AH.

Règle 6 **Supporter et utiliser l'anti-rejeu dans le protocole ESP**

La configuration par défaut de strongSwan ne respecte pas cette règle.
Par défaut, strongSwan supporte une fenêtre anti-rejeu de taille 32 paquets.
strongSwan peut être configuré comme indiqué ci-dessous pour respecter cette règle, si Netlink (une famille de sockets des interface du noyau Linux) est supporté.
Afin de configurer une fenêtre anti-rejeu de taille 1024 paquets, Netlink doit être supporté, et la clé <child>.replay_window de chaque connections.<conn>.children doit être configurée à "1024" (<child>.replay_window = 1024), dans le fichier de configuration swanctl.conf.

Règle 7 **Utiliser uniquement le mode tunnel**

La configuration par défaut de strongSwan ne force pas le respect de cette règle.
Si aucun mode n'est configuré, alors le mode tunnel sera négocié par défaut avec strongSwan.
strongSwan peut être configuré comme indiqué ci-dessous pour forcer le respect de cette règle.
Pour forcer l'utilisation du mode tunnel, configurer <child>.mode à "tunnel" (<child>.mode = tunnel) à la fois pour l'initiateur et le répondeur pour chaque connections.<conn>.children, dans le fichier de configuration de swanctl.conf.

Règle 8 **Utiliser le protocole IKEv2**

La configuration par défaut de strongSwan respecte cette règle.
La valeur par défaut de <conn>.version est "2" pour chaque connexion, dans le fichier de configuration swanctl.conf. Cette configuration par défaut n'autorise que l'usage du protocole IKEv2 pour la gestion des clés et des associations de sécurité dans IPsec.

Règle 9 **Proscrire les *nested* SA**

L'implémentation de strongSwan respecte cette règle. strongSwan ne supporte pas les *nested* SA.

Règle 10 Supporter le rafraîchissement des associations de sécurité

La configuration par défaut de strongSwan respecte cette règle.
La fréquence de rafraîchissement des associations de sécurité IKE SA est configurable avec la clé <conn>.rekey_time (configurée par défaut à 4h), dans le fichier de configuration swanctl.conf.
La fréquence de rafraîchissement des associations de sécurité Child SA est configurable avec la clé <child>.rekey_time (configurée par défaut à 1h), dans le fichier de configuration swanctl.conf.
D'autres paramètres de configuration ainsi que des informations concernant le rafraîchissement des associations de sécurité sont disponibles dans la documentation de strongSwan [a].

a. https://docs.strongswan.org/docs/latest/config/rekeying.html

Recommandation Reco 1 Utiliser le mécanisme *NAT-traversal*

La configuration par défaut de strongSwan respecte cette recommandation.
strongSwan implémente par défaut le mécanisme *NAT-traversal*. Egalement, strongSwan utilise par défaut le port 4500, car MOBIKE est activé par défaut dans le fichier de configuration swanctl.conf (la valeur par défaut de <conn>.mobike est "yes").

Recommandation Reco 2 Eviter la compression IP

La configuration par défaut de strongSwan respecte cette recommandation.
La configuration par défaut de strongSwan ne permet pas la compression IP.
Si le besoin de compression IP est clairement identifié alors la compression IP telle que définie dans la RFC 3173 peut être activée en configurant <child>.ipcomp à "yes" (valeur par défaut : "no") pour chaque connections.<conn>.children dans le fichier de configuration swanctl.conf.

Règle 11 Limiter les moyens d'authentification autorisés dans IKEv2

La configuration par défaut de strongSwan ne respecte pas cette règle.
strongSwan peut être configuré comme indiqué ci-dessous pour respecter cette règle.
Afin de restreindre les moyens d'authentification dans IKEv2 à l'authentification par signature, configurer la clé auth de connections.<conn>.local et de connections.<conn>.remote à "pubkey" dans le fichier de configuration swanctl.conf.
Afin de restreindre les moyens d'authentification dans IKE v2 à l'authentification par clé prépartagée : configurer la clé auth de connections.<conn>.local et connections.<conn>.remote à "psk" dans le fichier de configuration swanctl.conf.
Ne pas intégrer les plugins implémentant EAP (eap-tls, eap-dynamic, eap-gtc etc.).

Règle 12 **Proscrire l'authentification avec des certificats auto-signés**

Le refus d'une authentification avec un certificat auto-signé n'est pas configurable sur strongSwan.
Cependant, strongSwan rejette une authentification par certificat d'un pair si aucun certificat de la chaîne de certification du certificat du pair n'est présent dans un magasin de certificats de confiance de l'autre pair.
Ne pas configurer de certificat auto-signé autre que celui d'une autorité de certification racine dans le magasin de certificats de confiance de strongSwan.

Règle 13 **Supporter le format de certificat X.509 et proscrire *Hash and URL***

La configuration par défaut de strongSwan respecte cette règle. L'authentification par certificat d'un pair est rejetée si le certificat du pair est au format *Hash and URL* : la clé hash_and_url du démon charon est par défaut à "no" dans le fichier de configuration strongswan.conf.
En cas d'authentification par certificat d'un pair, alors ce pair doit être configuré avec un certificat au format X.509. De plus, le magasin de certificats de confiance de l'autre pair doit être configuré avec un certificat de la chaîne de certification du pair qui s'authentifie.

Règle 14 **Implémenter les mécanismes de révocation CRL et OCSP**

La configuration par défaut de strongSwan respecte cette règle.
Le plugin revocation permettant la vérification du statut de révocation d'un certificat X.509 par CRL ou OCSP est activé par défaut.
Il est possible de forcer l'échec d'une connexion si le statut de révocation n'est pas connu en configurant la clé revocation de connections.<conn>.remote à strict pour chaque connexion, dans le fichier de configuration swanctl.conf (valeur par défaut : relaxed, la connexion est refusée uniquement lorsque le certificat du pair est identifié comme révoqué).
Les CRLs peuvent être stockées localement (à ajouter dans le répertoire /etc/swanctl/x509crl).
Les certificats des répondeurs OCSP sont à ajouter dans le répertoire /etc/swanctl/x509ocsp.
Afin de permettre la récupération des CRLs auprès d'un CRL Distribution Point et la récupération des réponses OCSP auprès d'un répondeur OCSP, l'un des plugins suivants devra être activé : curl ou winhttp (ces deux plugins sont désactivés par défaut).

Règle 15 **Proscrire le chevauchement de requêtes IKEv2**

L'implémentation de strongSwan respecte cette règle.
La fenêtre de chevauchement de requête est de taille 1 et n'est pas configurable sur strongSwan. strongSwan ne supporte pas la notification SET_WINDOW_SIZE.

Règle 16 **Utiliser la négociation *childless* dans IKEv2**

La configuration par défaut de strongSwan ne respecte pas cette règle.
strongSwan peut être configuré comme indiqué ci-dessous pour respecter cette règle.
Afin d'imposer la négociation *childless initiation* définie dans la RFC 6023, configurer <conn>.childless à "force" (valeur par défaut à "allow") pour chaque connexion, dans le fichier de configuration swanctl.conf.

Règle 17 **Proscrire la suppression de l'association de sécurité IKE SA directement après un échec de création de Child SA**

L'implémentation de strongSwan respecte cette règle.
L'association de sécurité IKE SA en cours n'est pas supprimée en cas d'échec de la création d'une paire d'associations de sécurité Child SA pendant les échanges CREATE_CHILD_SA.

Règle 18 **Négocier la même liste de mécanismes cryptographiques lors d'une nouvelle tentative d'envoi de suites cryptographiques**

La configuration par défaut de strongSwan respecte cette règle.
Lors de la retransmission d'une requête IKEv2, strongSwan retransmet le message précédemment généré et conserve donc la même liste de mécanismes cryptographiques. Aucune option de configuration spécifique n'est nécessaire.
Lors d'une nouvelle tentative d'échange IKE_SA_INIT consécutive à une notification INVALID_KE_PAYLOAD, strongSwan conserve les propositions cryptographiques définies par la clé connections.<conn>.proposals dans le fichier de configuration swanctl.conf et adapte uniquement la charge utile d'échange de clés au groupe demandé par le pair. La configuration de la connexion ne doit pas être modifiée ou rechargée entre les tentatives d'une même négociation.

Règle 19 **Utiliser un mécanisme de *cookie* robuste dans IKEv2**

La configuration par défaut de strongSwan ne respecte pas cette règle.
strongSwan peut être configuré comme indiqué ci-dessous pour respecter cette règle.
Afin de forcer l'utiliser d'un mécanisme de *cookie* dans IKEv2, configurer la clé cookie_threshold du démon charon à la valeur "0" (valeur par défaut à "30") et la clé cookie_threshold_ip du démon charon à la valeur "0" (valeur par défaut à "3"). Ces configurations forcent l'utilisation des cookies pour chaque échange entre les pairs.
De plus, conserver la configuration par défaut de la clé dos_protection du démon charon à "yes".
La génération du cookie par strongSwan est conforme à cette règle.

Recommandation Reco 3 Implémenter le mécanisme de cookie proposé dans la RFC 7296

strongSwan ne respecte pas cette recommandation car strongSwan n'implémente pas exactement le mécanisme de la RFC 7296. Cependant, la génération du cookie par strongSwan est conforme à la Règle 19.

Recommandation Reco 4 Privilégier le silence dans les échanges IKEv2

La configuration par défaut de strongSwan paraît compatible avec cette recommandation sur certains points, sans qu'il soit possible d'affirmer que tous les échanges IKEv2 nécessitant un abandon silencieux sont traités sans exception.
Le démon charon ne fournit pas d'option générale imposant le silence pour toutes les erreurs, selon le protocole, certaines doivent être ignorées, tandis que d'autres peuvent ou doivent entraîner une notification.
Il est recommandé de conserver charon.send_vendor_id à "no" dans le fichier de configuration strongswan.conf, afin de ne pas émettre l'identifiant facultatif de l'implémentation, ce paramètre contribue à limiter les informations exposées, mais ne garantit pas à lui seul le respect de la recommandation dans tous les échanges IKEv2.

Règle 20 Implémenter la fragmentation dans IKEv2

La configuration par défaut de strongSwan respecte cette règle.
L'usage de la fragmentation des messages dans le protocole IKEv2, telle que définie dans la RFC 7383, est forcé par la valeur par défaut à "yes" de la clé <conn>.fragmentation pour les configurations présentes dans le fichier de configuration swanctl.conf.

Règle 21 Implémenter uniquement des échanges de clés hybrides dans IKEv2

L'implémentation de strongSwan respecte cette règle.
strongSwan supporte les échanges de clé multiples dans IKEv2 définis dans la RFC 9370 ainsi que le mécanisme des messages intermédiaires de la RFC 9242.

Règle 22 **Implémenter l'échange de clés hybride dans IKEv2 (IKE_SA_INIT)**

strongSwan peut être configuré pour respecter cette règle.
strongSwan implémente les RFC nécessaires à l'échange de clé hybride dans IKEv2 (RFC 9370 et RFC 9242). De plus strongswan supporte des suites cryptographiques permettant un échange de clé hybride avec un premier échange de clés basé sur le schéma Diffie-Hellman et un second échange de clés basé sur un KEM post-quantique.
La valeur de la clé <conn>.proposals des configurations présentes dans le fichier de configuration swanctl.conf doit être configurée avec un mécanisme Diffie-Hellman et un KEM post-quantique. Par exemple, cette liste de proposition de mécanismes cryptographiques pour IKEv2 respecte cette règle : aes256gcm16-prfsha256-x25519-ke1_mlkem768.

Règle 23 **Implémenter l'échange de clés hybride dans IKEv2 (CREATE_CHILD_SA)**

La configuration par défaut de strongSwan ne respecte pas cette règle.
strongSwan peut être configuré comme indiqué ci-dessous pour respecter cette règle.
strongSwan implémente les RFC nécessaires à l'échange de clé hybride dans IKEv2 (RFC 9370 et RFC 9242). De plus strongSwan supporte des suites cryptographiques permettant un échange de clé hybride avec un premier échange de clés basé sur le schéma Diffie-Hellman et un second échange de clés basé sur un KEM post-quantique. La valeur de la clé <child>.esp_proposals des configurations de CHILD_SA (connections.<conn>.children) présentes dans le fichier de configuration swanctl.conf doit être configurée avec un mécanisme Diffie-Hellman et un KEM post-quantique. Par exemple, cette liste de proposition de mécanismes cryptographiques pour IKEv2 respecte cette règle : aes256gcm16-prfsha256-x25519-ke1_mlkem768.

Règle 24 **Utiliser des clés éphémères à chaque nouvel échange de clés**

La configuration par défaut de strongSwan ne respecte pas cette règle.
strongSwan peut être configuré comme indiqué ci-dessous pour respecter cette règle.
La configuration d'un mécanisme d'échange de clé hybride pour les suites cryptographiques d'ESP(*ESP proposals*) entraîne l'utilisation de clés éphémères pour chaque échange de clés dans le protocole IKEv2. La valeur de la clé <child>.esp_proposals des configurations de CHILD_SA (connections.<conn>.children) présentes dans le fichier de configuration swanctl.conf doit être configurée avec un mécanisme Diffie-Hellman et un KEM post-quantique. Par exemple, cette liste de proposition de mécanismes cryptographiques pour IKEv2 respecte cette règle : aes256gcm16-prfsha256-x25519-ke1_mlkem768.

Règle 25 Proscrire l'utilisation de clés pré-partagées pour assurer la confidentialité des échanges dans le protocole IKEv2

La configuration par défaut de strongSwan respecte cette règle.
strongSwan implémente la RFC 8784 et n'implémente pas la RFC 9867.
Ne pas configurer secrets.ppk<suffix> dans le fichier de configuration swanctl.conf.

Règle 26 Mécanismes de chiffrement des messages obligatoires

La configuration par défaut de strongSwan ne respecte pas les règles et les recommandations concernant les suites cryptographiques.
strongSwan peut être configuré comme indiqué ci-dessous pour respecter ces règles et ces recommandations.
Les suites cryptographiques proposées pour IKEv2 et ESP doivent être configurées obligatoirement avec les mécanismes de chiffrement suivants : aes256gcm16 et aes256ctr.

Règle 27 Mécanismes de chiffrement des messages optionnels

La configuration par défaut de strongSwan ne respecte pas les règles et les recommandations concernant les suites cryptographiques.
strongSwan peut être configuré comme indiqué ci-dessous pour respecter ces règles et ces recommandations.
Les suites cryptographiques proposées pour IKEv2 et ESP peuvent être configurées avec les mécanismes de chiffrement suivants : aes256, aes192, aes192ctr, aes192ccm16, aes256ccm16, aes192gcm16.

Règle 28 Mécanisme d'intégrité des messages obligatoire

La configuration par défaut de strongSwan ne respecte pas les règles et les recommandations concernant les suites cryptographiques.
strongSwan peut être configuré comme indiqué ci-dessous pour respecter ces règles et ces recommandations.
Les suites cryptographiques proposées pour IKEv2 et ESP doivent être configurées obligatoirement avec le mécanisme d'intégrité suivant : sha256 ou sha2_256 (deux noms pour le même mécanisme).

Règle 29 **Mécanismes d'intégrité des messages optionnels**

La configuration par défaut de strongSwan ne respecte pas les règles et les recommandations concernant les suites cryptographiques.
strongSwan peut être configuré comme indiqué ci-dessous pour respecter ces règles et ces recommandations.
Les suites cryptographiques proposées pour IKEv2 et ESP peuvent être configurées avec les mécanismes d'intégrité suivants : sha384 (ou sha2_384) et sha512 (ou sha2_512) (sha384 et sha2_384 désignent le même mécanisme d'intégrité AUTH_HMAC_SHA2_384_192, sha512 et sha2_512 désignent le même mécanisme d'intégrité AUTH_HMAC_SHA2_512_256).

Règle 30 **Fonction pseudo-aléatoire obligatoire**

La configuration par défaut de strongSwan ne respecte pas les règles et les recommandations concernant les suites cryptographiques.
strongSwan peut être configuré comme indiqué ci-dessous pour respecter ces règles et ces recommandations.
Les suites cryptographiques proposées pour IKEv2 et ESP doivent être configurées obligatoirement avec la fonction pseudo-aléatoire suivante : prfsha256.

Règle 31 **Fonctions pseudo-aléatoires optionnelles**

La configuration par défaut de strongSwan ne respecte pas les règles et les recommandations concernant les suites cryptographiques.
strongSwan peut être configuré comme indiqué ci-dessous pour respecter ces règles et ces recommandations.
Les suites cryptographiques proposées pour IKEv2 et ESP peuvent être configurées avec les fonction pseudo-aléatoire suivantes : prfsha384, prfsha512.

Règle 32 **Mécanismes d'échange de clé obligatoires**

La configuration par défaut de strongSwan ne respecte pas les règles et les recommandations concernant les suites cryptographiques.
strongSwan peut être configuré comme indiqué ci-dessous pour respecter ces règles et ces recommandations.
Les suites cryptographiques proposées pour IKEv2 et ESP doivent être configurées obligatoirement avec les mécanismes d'échange de clé suivants : ecp256, ecp256bp, mlkem768, mlkem1024.

Règle 33 Mécanismes d'échange de clé optionnels

La configuration par défaut de strongSwan ne respecte pas les règles et les recommandations concernant les suites cryptographiques.
strongSwan peut être configuré comme indiqué ci-dessous pour respecter ces règles et ces recommandations.
Les suites cryptographiques proposées pour IKEv2 et ESP peuvent être configurées avec les mécanismes d'échange de clé suivants : ecp384, ecp521, ecp384bp, ecp512bp, curve25519 ou x25519, curve448 ou x448. Exemples de propositions de suites cryptographiques pour IKEv2 et ESP qui sont conformes aux recommandations Règle 26 à Règle 33 :
aes256gcm16-aes256ctr-aes256-aes192-aes192ctr-aes192ccm16-aes256ccm16-aes192gcm16-prfsha256-prfsha384-prfsha512-ecp256-ke1_mlkem768-ecp256bp-ke1_mlkem768-ecp256-ke1_mlkem1024-ecp256bp-ke1_mlkem1024-x25519-ke1_mlkem768-curve448-ke1_mlkem768

Exemples de proposition de suites cryptographiques minimales pour IKEv2 et ESP conformes aux règles et recommandations de ce document :
Exemple 1 : aes256gcm16-prfsha256-ecp256-ke1_mlkem768-ecp256bp-ke1_mlkem768-ecp256-ke1_mlkem1024-ecp256bp-ke1_mlkem1024
Exemple 2 : aes256gcm16-prfsha256-ecp256bp-ke1_mlkem768-ecp256bp-ke1_mlkem1024-ecp256-ke1_mlkem768-ecp256-ke1_mlkem1024

Règle 34 Mécanismes d'authentification obligatoires

La configuration par défaut de strongSwan ne respecte pas les règles et les recommandations concernant les suites cryptographiques.
strongSwan peut être configuré comme indiqué ci-dessous pour respecter ces règles et ces recommandations.
En cas d'authentification par signature, la clé auth de connections.<conn>.local et connections.<conn>.remote doit être configurée avec les mécanismes de chiffrement et d'intégrité suivants : ike :ecdsa-256-sha256.

Règle 35 Mécanismes d'authentification optionnels

La configuration par défaut de strongSwan ne respecte pas les règles et les recommandations concernant les suites cryptographiques.
strongSwan peut être configuré comme indiqué ci-dessous pour respecter ces règles et ces recommandations.
strongSwan ne supporte pas à ce jour ML-DSA (intégration en cours), ni SLH-DSA.
En cas d'authentification par signature, la clé auth de connections.<conn>.local et connections.<conn>.remote peut être configurée avec les mécanismes de signature et d'intégrité suivants : ecdsa-384, ecdsa-512, ecdsa-521, ed25519, ed448, sha384. Par exemple, ike :ecdsa-256-ecdsa-384-ecdsa-512-ecdsa-521-ed25519-ed448-sha256-sha384-sha512 respecte la Règle 34 et la Règle 35.

Version 1.0 - 17/11/2026 - ANSSI-PA-120
Licence ouverte / Open Licence (Étalab - v2.0)

AGENCE NATIONALE DE LA SÉCURITÉ DES SYSTÈMES D'INFORMATION

ANSSI - 51 boulevard de La Tour-Maubourg, 75700 PARIS 07 SP
cyber.gouv.fr / conseil.technique@ssi.gouv.fr

RÉPUBLIQUE FRANÇAISE
Liberté
Égalité
Fraternité