RÉPUBLIQUE FRANÇAISE
Liberté
Égalité
Fraternité

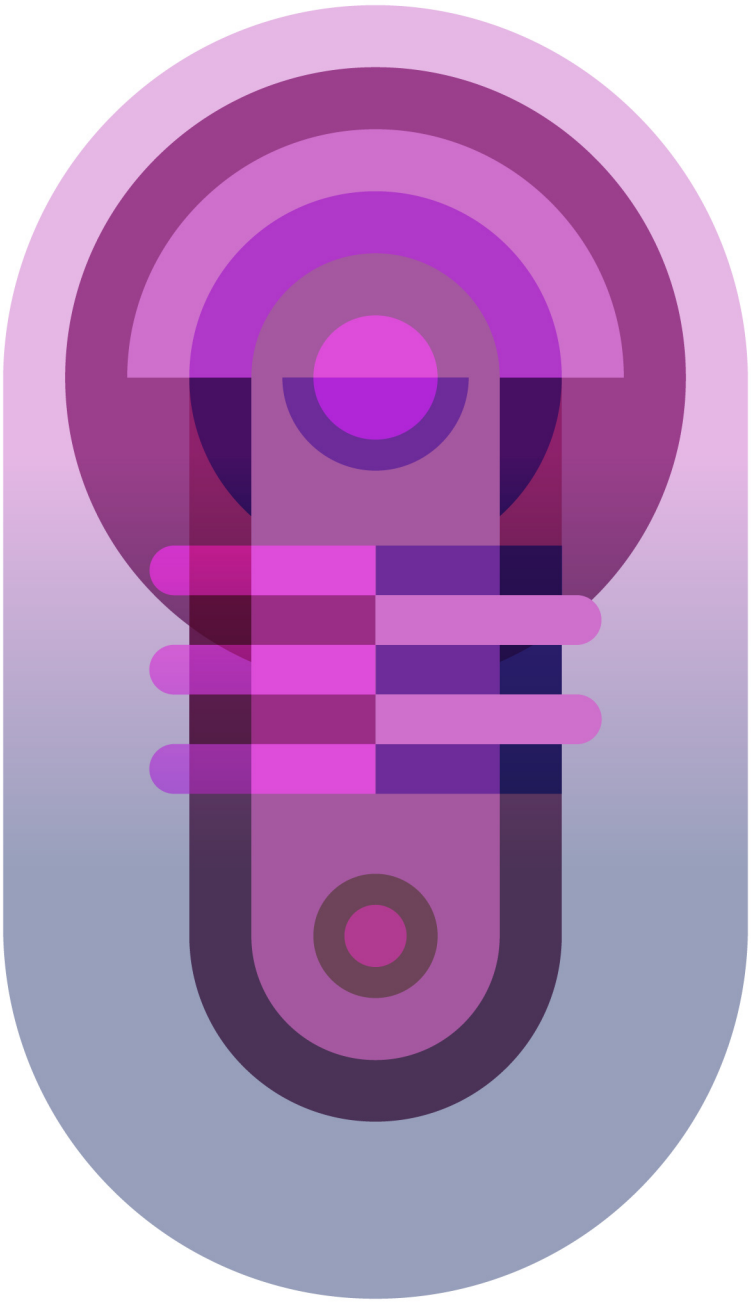
TRANSITION POST-QUANTIQUE DE SSHV2

FICHE TECHNIQUE ANSSI

ANSSI-FT-116
02/02/2026

PUBLIC VISÉ :

Développeur | Administrateur | RSSI | DSI | Utilisateur



Informations

Attention

Ce document rédigé par l'ANSSI s'intitule **« Transition Post-Quantique de SSHv2 »**. Il est téléchargeable sur le site `cyber.gouv.fr`.

Évolutions du document :

VERSION	DATE	NATURE DES MODIFICATIONS
1.0	02/02/2026	Version initiale

Table des matières

1
Introduction

Ce document vise à présenter un état des lieux de la transition post-quantique du protocole Secure Shell (SSH). La transition s'applique à la version v2, définie dans une série de RFC [7, 8, 9, 10, 11]. Le document présente d'abord le fonctionnement général du protocole. Ensuite, il identifie les parties du protocole concernées par la menace quantique et des solutions de transition associées. Il présente également les travaux existants sur le sujet. Ce document vient en complément de l'avis de l'ANSSI sur la transition post-quantique [1, 2].

Les mécanismes de cryptographie asymétrique utilisés aujourd'hui sont vulnérables à la menace quantique. Ainsi, le but de la transition post-quantique est de les remplacer par des mécanismes de cryptographie asymétrique dite post-quantique, supposée résistante à des attaques qui seraient réalisées sur un ordinateur classique et sur un ordinateur quantique. En particulier, les signatures de cryptographie classique sont remplacées par des signatures de cryptographie post-quantique, et les échanges de clés, souvent basés sur le schéma Diffie-Hellman (DH), sont remplacés par des échanges de clés basés sur des mécanismes d'encapsulation de clés (ou *Key Encapsulation Mechanism* (KEM)) post-quantiques. Pendant la transition post-quantique, l'hybridation est recommandée par l'ANSSI [1, 2]. Cela signifie l'utilisation d'un mécanisme de cryptographie asymétrique classique éprouvé, mais vulnérable à des attaques quantiques, combiné avec un mécanisme de cryptographie asymétrique supposé résistant aux attaques quantiques, mais pour lequel l'assurance de robustesse vis-à-vis des attaques réalisées à l'aide d'ordinateurs classique et quantique est moindre. Une conséquence de l'utilisation des mécanismes de cryptographie post-quantique dans les protocoles est une augmentation importante de la taille des messages échangés pendant les phases d'échange de clés et d'authentification. En effet, les mécanismes de cryptographie post-quantique existants possèdent des tailles de clés, de chiffrés et de signatures très grandes par rapport aux mécanismes de cryptographie classique. Quant aux mécanismes de cryptographie symétrique, la robustessse de ces derniers n'est pas menacée par l'ordinateur quantique. Toutefois, l'ANSSI recommande [1, 2] d'augmenter les tailles des clés du chiffrement symétrique et des sorties des fonctions de hachage.

2
Présentation de SSHv2

Le protocole SSHv2 (Secure Shell) permet à un client d'accéder à un serveur distant en protégeant les communications. Les usages les plus communs de SSHv2 sont l'administration à distance, le transfert sécurisé de données et la redirection de flux. SSHv2 assure principalement la confidentialité et l'intégrité des données, ainsi que l'authentification du client et du serveur. Il est composé de trois protocoles comme présenté dans la Figure 1 : Transport Layer, User Authentication et Connection. Les parties impactées par la transition post-quantique sont indiquées en rouge, il s'agit des protocoles Transport Layer et User Authentication. SSHv2 fonctionne au-dessus du protocole de transport TCP.

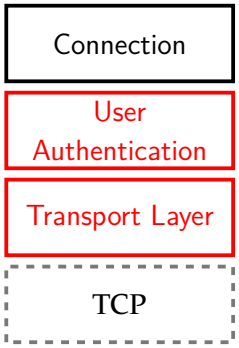


FIGURE 1 – Protocole SSHv2

- Le protocole Transport Layer permet d'établir la session SSHv2 et de protéger les données transmises.
  - Transport Layer exécute les premiers échanges (*handshake*) entre le client et le serveur. Pendant ces échanges, le client et le serveur négocient les suites cryptographiques devant être utilisées et établissent un secret partagé, tout en authentifiant le serveur avec une signature. Les clés symétriques de chiffrement et d'intégrité sont dérivées à partir de ce secret partagé.

Attention

La partie *handshake* du protocole Transport Layer est impactée par la menace quantique. En effet, des mécanismes asymétriques sont utilisés pour l'échange de clés (DH) et l'authentification par signature du serveur [1, 2].

  - Ensuite, Transport Layer prend en charge la transmission des données SSHv2. En particulier, Transport Layer reçoit les messages des protocoles Connection et User Authentication afin de les protéger. Un canal sécurisé est alors établi entre le client et le serveur, il assure une protection en confidentialité et intégrité des échanges, en utilisant des mécanismes symétriques, négociés dans les échanges du *handshake*. La protection des données par le protocole Transport

Layer n'est pas impactée par la menace quantique, puisqu'elle n'utilise pas de mécanismes asymétriques. Toutefois, la sécurité des clés utilisées repose sur la partie *handshake*, pendant laquelle la dérivation de clés est calculée.

- Le protocole User Authentication permet au client de s'authentifier auprès du serveur. Ce protocole est exécuté après que le canal sécurisé soit établi, grâce aux échanges du *handshake* du protocole Transport Layer. L'authentification peut se faire notamment par signature, mot de passe ou GSS-API.

Attention

Le protocole User Authentication permet au client de s'authentifier en utilisant des signatures. Il est donc impacté par la menace quantique.

- Le protocole Connection permet d'exécuter des opérations entre le client et le serveur (*e.g.*, récupérer/envoyer des fichiers, envoyer des commandes pour de l'administration à distance, etc). Les données de ce protocole sont transmises et protégées grâce au canal sécurisé établi dans le protocole Transport Layer. Le protocole Connection n'est pas impacté par la menace quantique. Il n'utilise pas de mécanismes de cryptographie et ne sera donc pas traité dans la suite de ce document. Toutefois, la sécurité des échanges repose sur le protocole Transport Layer, pendant lequel les messages sont chiffrés.

2.1 La partie handshake du protocole Transport Layer de SSHv2

Les échanges de la partie *handshake* du protocole Transport Layer sont rappelés dans la Figure 1. Une double flèche signifie que le message est envoyé par le client et le serveur, sans importance de l'ordre.

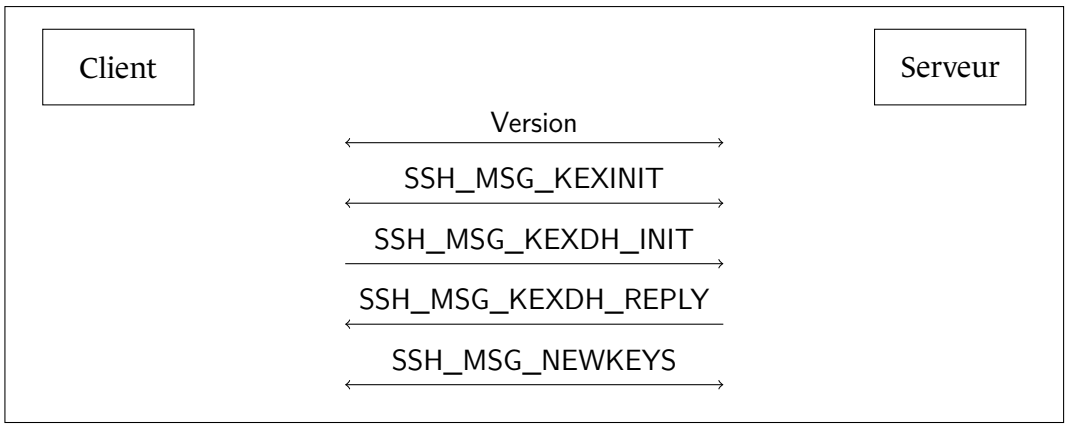


FIGURE 2 – Le protocole Transport Layer de SSHv2

- Les messages Version permettent au client et au serveur de vérifier la version du protocole (SSHv2).

- Les messages SSH_MSG_KEXINIT servent à négocier les suites cryptographiques employées dans la suite des échanges. Les messages incluent notamment :
  - une liste de groupes Diffie-Hellman (DH) pour l'échange de clés et de fonctions de hachage,
  - une liste de mécanismes de signature pour l'authentification du serveur :
    - le client envoie la liste des algorithmes qu'il supporte,
    - le serveur envoie la liste des algorithmes pour lesquels il possède une clé de signature,
  - une liste de mécanismes symétriques assurant la confidentialité et l'intégrité des données envoyées du client vers le serveur,
  - une liste de mécanismes symétriques assurant la confidentialité et l'intégrité des données envoyées du serveur vers le client.

  Chaque mécanisme négocié correspond à celui qui figure dans les listes du client et du serveur, par ordre de préférence du client.
- Le message SSH_MSG_KEXDH_INIT contient la clé publique DH du client, pour le groupe négocié.
- Le message SSH_MSG_KEXDH_REPLY contient la clé publique DH du serveur. Il contient également le certificat du serveur et une signature calculée avec la clé privée du serveur, associée à la clé publique présente dans son certificat [1]. Cette signature est calculée notamment sur l'empreinte (en utilisant la fonction de hachage négociée) des messages échangés précédemment, les clés publiques DH et le secret DH partagé.
- Les messages SSH_MSG_NEWKEYS dans les deux sens indiquent la fin des échanges. Tous les messages par la suite sont protégés en confidentialité et intégrité avec les mécanismes symétriques négociés. Les clés correspondantes sont dérivées à partir du secret DH partagé.

Information

Pour renouveler les clés de session, le *handshake* peut être de nouveau déclenché plus tard pendant la session.

2.2 Le protocole User Authentication de SSHv2

Le client s'authentifie grâce au protocole User Authentication. L'authentification peut se faire par plusieurs méthodes (*e.g.*, signature, mot de passe, GSS-API). Parmi les moyens d'authentification possibles, uniquement l'authentification par signature est vulnérable à la menace quantique. Avec ce moyen d'authentification, présenté dans la Figure 3, le client envoie sa clé publique (ou le certificat de son hôte dans le cas où il authentifie plutôt l'identité de ce dernier). Il envoie également une signature calculée avec la clé privée correspondante à sa clé publique (ou correspondante à la clé publique dans le certificat de son hôte). La signature est calculée sur la concaténation de plusieurs données (*e.g.*, l'identifiant de la session en cours, le nom utilisateur) [2].

1. Le serveur peut envoyer sa clé publique de vérification de signature au lieu d'un certificat. Le client doit vérifier que la clé publique ou le certificat reçu appartient au serveur avec lequel il souhaite communiquer. La gestion des clés publiques des serveurs sur le client n'est pas traitée dans ce document.
2. Le serveur doit pouvoir vérifier la clé publique du client en utilisant par exemple une base de données locale. La gestion des clés publiques des clients autorisés à se connecter sur le serveur n'est pas traitée dans ce document.

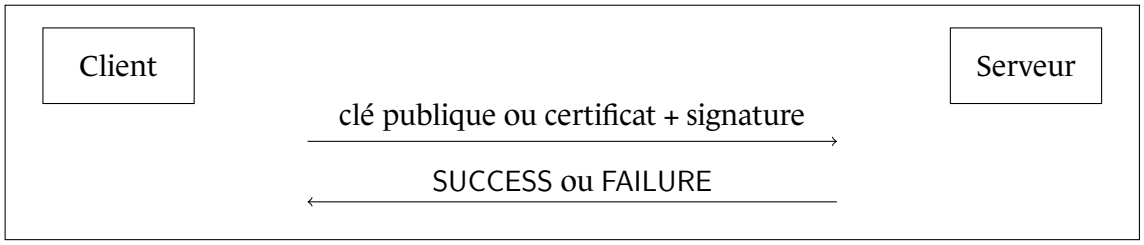


FIGURE 3 – Authentification du client par signature dans SSHv2

3
Transition post-quantique de SSHv2

Comme mentionné précédemment, les protocoles Transport Layer (uniquement la partie *handshake*) et User Authentication de SSHv2 mettent en œuvre des mécanismes asymétriques pour l'échange de clés et l'authentification par signature. Ces derniers sont vulnérables à la menace quantique. La transition post-quantique peut généralement être réalisée en utilisant des clés pré-partagées, ou en utilisant une méthode d'hybridation qui est la solution privilégiée par l'ANSSI [1, 2]. Dans le cas des protocoles Transport Layer et User Authentication, ces derniers ne supportent pas l'utilisation de clés pré-partagées (*c.f.*, Section 3.1), mais peuvent être hybridés (*c.f.*, Section 3.2). Notons que la transition post-quantique pour la confidentialité des échanges est plus urgente que celle pour l'authentification, notamment à cause des attaques *"store-now decrypt-later"*.

3.1 Sécurité post-quantique par clé pré-partagée

Comme précisé dans l'avis sur la transition post-quantique de l'ANSSI [1, 2], une clé pré-partagée peut être utilisée dans le cadre de la transition post-quantique, notamment si la confidentialité et l'intégrité (classiques et post-quantiques) de la clé pré-partagée sont assurées. Néanmoins, ceci n'est pas nativement supporté dans le protocole SSHv2. Il n'existe pas de travaux permettant l'utilisation de clés pré-partagées dans le protocole.

3.2 Sécurité post-quantique par hybridation

L'hybridation de SSHv2 implique la modification de plusieurs messages. En particulier, l'échange de clés basé sur le schéma DH et l'authentification par signature du serveur sont effectués dans les échanges du protocole Transport Layer. Ensuite, l'authentification par signature du client est effectuée dans les échanges du protocole User Authentication.

3.2.1 Hybridation de l'échange de clés

L'échange de clés a lieu dans la partie *handshake* du protocole Transport Layer. L'hybridation consiste à exécuter deux échanges de clés : un échange de clés avec un mécanisme classique (par exemple le schéma DH) et un échange de clés avec un mécanisme post-quantique (par exemple ML-KEM [12]). Ensuite, les deux secrets partagés sont combinés pour dériver le secret final ayant une sécurité classique et post-quantique.

Attention

Le mécanisme de combinaison des secrets partagés issus des deux échanges de clés (classique et post-quantique) pour dériver le secret partagé final doit assurer une sécurité classique et post-quantique. Des manières de combiner les deux secrets sont citées dans l'avis sur la transition post-quantique de l'ANSSI [1, 2].

L'hybridation de l'échange de clés de la partie *handshake* du protocole Transport Layer implique les modifications suivantes :

- dans les messages SSH_MSG_KEXINIT, le client et le serveur envoient des listes de groupes DH. Dans le cas d'un échange de clés hybride, ces listes présentent des noms de mécanismes pour des échanges de clés hybrides (groupe DH et KEM post-quantique).
- Dans le message SSH_MSG_KEXDH_INIT, le client envoie sa clé publique DH pour le groupe négocié. Dans le cas d'un échange de clés hybride, le client envoie sa clé publique DH et sa clé publique du KEM post-quantique, qui peuvent être concaténées.
- Dans le message SSH_MSG_KEXDH_REPLY, le serveur répond en particulier avec sa clé publique DH pour le groupe négocié. Dans le cas d'un échange de clés hybride, le serveur répond avec sa clé publique DH et le chiffré résultant de l'encapsulation sur la clé publique du KEM post-quantique du client, qui peuvent être concaténés.

Documents. Un draft internet [6] définit l'échange de clés hybride dans SSHv2, en précisant les noms et les formats des différents champs des messages. La solution proposée dans le draft est similaire à la description dans cette section. Le document utilise l'échange de clés avec le schéma DH basé sur les courbes elliptiques et le KEM post-quantique ML-KEM [12]. Un autre draft internet [5] définit l'échange de clés hybride en utilisant le schéma DH basé sur les courbes elliptiques avec le KEM post-quantique NTRU Prime [4]. Pour l'instant, il n'y a pas de standards officiels.

Implémentations. La version 9.9 d'OpenSSH implémente deux échanges de clés hybrides. Le premier utilise l'échange de clés avec le schéma DH basé sur les courbes elliptiques et le KEM post-quantique NTRU Prime [4]. Le second utilise l'échange de clés avec le schéma DH basé sur les courbes elliptiques et le KEM post-quantique ML-KEM [12]. Ce dernier est utilisé par défaut dans l'implémentation pour l'échange de clés depuis la version 10.0 d'OpenSSH. Ces deux échanges de clés correspondent aux spécifications dans [5, 6]. L'implémentation d'OpenSSH rend facile l'intégration d'autres échanges de clés hybrides.

3.2.2 Hybridation de l'authentification par signature

L'hybridation de l'authentification par signature nécessite l'hybridation des signatures et des certificats utilisés dans SSHv2. Il existe différents formats pour hybrider les certificats (authentifiant les clés de signature classique et post-quantique), par exemple avec deux certificats distincts (un pour la clé de signature classique, et un autre pour la clé de signature post-quantique), ou un seul certificat authentifiant la concaténation des clés de signature, etc. L'hybridation concerne aussi les chaînes de certification. Pour l'instant, il n'existe pas de standards pour l'authentification hybride.

L'hybridation de l'authentification par signature du serveur implique les modifications suivantes dans la partie *handshake* du protocole Transport Layer :

- dans les messages SSH_MSG_KEXINIT, le client et le serveur envoient des listes contenant des mécanismes de signature. Dans le cas de l'hybridation, ces mécanismes correspondent à des mécanismes de signature hybrides (composés d'une signature classique et d'une signature post-quantique).
- Dans le message SSH_MSG_KEXDH_REPLY, le serveur s'authentifie en envoyant son certificat et sa signature. Dans le cas de l'hybridation, le certificat ainsi que la signature sont hybrides.

L'hybridation de l'authentification par signature du client implique les modifications suivantes dans le protocole User Authentication :

- le client s'authentifie en envoyant sa clé publique ou le certificat de son hôte, ainsi que la signature correspondante. Dans le cas de l'hybridation, ces données sont hybrides, similairement à l'authentification du serveur.

Il n'existe pas pour l'instant de standards permettant l'intégration de l'authentification par signature hybride dans SSHv2.

Information

L'utilisation des mécanismes de cryptographie post-quantique augmente la taille des messages échangés lors des phases de l'échange de clés et de l'authentification par signature. Outre les impacts sur les performances, l'augmentation de la taille des messages oblige les protocoles à gérer la fragmentation de ces messages. Le protocole SSHv2 fonctionne au-dessus de TCP, un protocole de transport fiable, qui assure la segmentation et le réassemblage des grands messages.

4
Conclusion

La transition post-quantique du protocole SSHv2 concerne deux protocoles : le protocole Transport Layer (la partie *handshake*), où le client et le serveur négocient les suites cryptographiques, effectuent un échange de clés basé sur le schéma DH tout en authentifiant le serveur auprès du client, ainsi que le protocole User Authentication où le client s'authentifie auprès du serveur. L'ANSSI recommande d'utiliser l'hybridation pour les échanges de clés et l'authentification par signature. Le protocole SSHv2 ne propose pas l'utilisation d'une clé pré-partagée. La mise en œuvre de cette solution pour assurer une confidentialité et une authentification résistantes à la menace quantique n'est donc pas possible dans ce protocole.

L'hybridation de l'échange de clés dans la partie *handshake* du protocole Transport Layer nécessite un échange de clés basé sur un KEM post-quantique, en plus de l'échange de clés basé sur le schéma DH. Il existe un draft internet [6] qui décrit l'échange de clés hybride dans SSHv2 et l'instancie avec le schéma DH basé sur les courbes elliptiques et ML-KEM [12]. Un autre draft [5] décrit l'échange de clés hybride en utilisant l'échange de clés avec le schéma DH basé sur les courbes elliptiques et le KEM post-quantique NTRU Prime [4]. Pour l'instant, il n'y a pas de standards officiels. De plus, la version 9.9 d'OpenSSH implémente deux échanges de clés hybrides, en utilisant le schéma DH basé sur les courbes elliptiques avec ML-KEM [12] dans le premier, et le schéma DH basé sur les courbes elliptiques avec NTRU Prime [4] dans le second. L'échange de clés basé sur le schéma DH et ML-KEM est utilisé par défaut dans l'implémentation depuis la version 10.0 d'OpenSSH. Enfin, un travail établit des preuves de sécurité du protocole SSHv2 en y intégrant des échanges de clés hybrides [3].

L'hybridation de l'authentification par signature dans les protocoles Transport Layer et User Authentication nécessite l'hybridation des certificats utilisés pour authentifier le client et le serveur, ainsi que les signatures envoyées dans les messages d'authentification. L'hybridation de l'authentification nécessite plus de travaux. Il n'existe pas pour l'instant de standards officiels permettant l'authentification par signature et certificat hybrides dans SSHv2.

La transition post-quantique du protocole SSHv2 étant un sujet très étudié, il est nécessaire de suivre l'évolution des travaux et des implémentations.

Bibliographie

[1] ANSSI. Avis de l'anssi sur la migration vers la cryptographie post-quantique. https://cyber.gouv.fr/publications/avis-de-lanssi-sur-la-migration-vers-la-cryptographie-post-quantique.

[2] ANSSI. Avis de l'anssi sur la migration vers la cryptographie post-quantique. https://cyber.gouv.fr/publications/avis-de-lanssi-sur-la-migration-vers-la-cryptographie-post-quantique-0.

[3] Benjamin Benčina, Benjamin Dowling, Varun Maram, and Keita Xagawa. Post-quantum cryptographic analysis of ssh. *Cryptology ePrint Archive*, 2025.

[4] Daniel J Bernstein, Chitchanok Chuengsatiansup, Tanja Lange, and Christine Van Vredendaal. Ntru prime. https://ntruprime.cr.yp.to/.

[5] Markus Friedl, Jan Mojzis, and Simon Josefsson. Secure Shell (SSH) Key Exchange Method Using Hybrid Streamlined NTRU Prime sntrup761 and X25519 with SHA-512 : sntrup761x25519-sha512. Internet-Draft draft-ietf-sshm-ntruprime-ssh-03, Internet Engineering Task Force, May 2025. Work in Progress.

[6] Panos Kampanakis, Douglas Stebila, and Torben Hansen. PQ/T Hybrid Key Exchange in SSH. Internet-Draft draft-ietf-sshm-mlkem-hybrid-kex-02, Internet Engineering Task Force, April 2025. Work in Progress.

[7] Chris M. Lonvick and Sami Lehtinen. The Secure Shell (SSH) Protocol Assigned Numbers. RFC 4250, January 2006.

[8] Chris M. Lonvick and Tatu Ylonen. The Secure Shell (SSH) Authentication Protocol. RFC 4252, January 2006.

[9] Chris M. Lonvick and Tatu Ylonen. The Secure Shell (SSH) Connection Protocol. RFC 4254, January 2006.

[10] Chris M. Lonvick and Tatu Ylonen. The Secure Shell (SSH) Protocol Architecture. RFC 4251, January 2006.

[11] Chris M. Lonvick and Tatu Ylonen. The Secure Shell (SSH) Transport Layer Protocol. RFC 4253, January 2006.

[12] NIST. Module-lattice-based key-encapsulation mechanism standard. https://csrc.nist.gov/pubs/fips/203/final.

Version 1.0 - 02/02/2026 - ANSSI-FT-116

AGENCE NATIONALE DE LA SÉCURITÉ DES SYSTÈMES D'INFORMATION
ANSSI - 51 boulevard de La Tour-Maubourg 75000 PARIS 07 SP
cyber.gouv.fr / conseil.technique@ssi.gouv.fr

RÉPUBLIQUE FRANÇAISE
Liberté
Égalité
Fraternité