RFC 10042 : Post-Quantum/Traditional Hybrid Key Exchange with the Module-Lattice-Based Key-Encapsulation Mechanism for Use in SSH

Stéphane Bortzmeyer
<stephane+blog@bortzmeyer.org>

Première rédaction de cet article le 31 août 2026

Date de publication du RFC : Août 2026

https://www.bortzmeyer.org/10042.html

Tout le monde (et son chat) se met à la cryptographie post-quantique en ce moment et plus précisement à des systèmes hybrides, c'est-à-dire utilisant à la fois un algorithme traditionnel (comme ECDH) et un algorithme post-quantique (comme ML-KEM). Ce RFC décrit l'échange de clés <https://www.bortzmeyer.org/echange-cles.html> avec un système hybride dans le protocole SSH.

Au début d'une connexion SSH (RFC 4251 [1], section 1), les deux parties qui communiquent procèdent en effet à un échange de clés <https://www.bortzmeyer.org/echange-cles.html> (section 7 du RFC 4253) afin de parvenir à un accord sur la clé partagée qui sera utilisée pour chiffrer la communication. Jusqu'à récemment, cet échange de clés était fait en utilisant un algorithme reposant sur les courbes elliptiques (RFC 5656 et RFC 8731), algorithme qui est vulnérable aux futurs calculateurs quantiques. Si des calculateurs quantiques réellement utilisables (les CRQC, "*Cryptographically Relevant Quantum Computers*") apparaissent un jour, l'ancien algorithme devra être abandonné. Et on ne peut pas attendre pour voir si des CRQC deviennent réellement disponibles : des attaquants sont peut-être aujourd'hui en train d'enregistrer des sessions SSH, afin de les décrypter plus tard lorsqu'on aura enfin des CRQC (RFC 9958, section 1). Bref, il vaut mieux passer à des algorithmes post-quantiques dès maintenant.

Mais les algorithmes post-quantiques existants ne sont pas forcément sûrs face à la cryptanalyse ; on manque de recul à ce sujet. D'où l'idée de systèmes hybrides (RFC 9794), ou PQ/T (pour "*Post-Quantum/Traditional*") combinant algorithme post-quantique et algorithme traditionnel, et obligeant l'attaquant à casser les deux s'il veut décrypter la communication. (Si vous voulez creuser la question

1. Pour voir le RFC de numéro NNN, https://www.ietf.org/rfc/rfcNNN.txt, par exemple https://www.ietf.org/rfc/rfc4251.txt

de la sécurité des systèmes hybrides, regardez « "*Security of Hybrid Key Encapsulation*" <https://eprint.iacr.org/2020/1364> » et « "*Security of Hybrid Key Establishment using Concatenation*" <https://eprint.iacr.org/2023/972> ».) C'est ce genre de système que normalise notre RFC. Il est sûr face à un attaquant actif, même si ce dernier dispose d'un calculateur quantique ou bien s'il a trouvé une faille dans l'algorithme post-quantique.

Plus précisement, ce RFC va utiliser l'algorithme post-quantique ML-KEM avec un choix d'algorithmes traditionnels.

La section 2 du RFC décrit concrètement l'échange de clés avec le système hybride (relire les RFC 4251 et RFC 4253 peut être utile). Le client SSH, au lieu de commencer par `SSH_MSG_KEXDH_INIT` (RFC 4253) ou `SSH_MSG_KEX_ECDH_INIT` (RFC 5656), envoie un message `SSH_MSG_KEX_HYBRID_INIT` (valeur 30 dans le protocole) qui contient la concaténation des deux clés du client. Et la réponse, au lieu de `SSH_MSG_KEXDH_REPLY` ou `SSH_MSG_KEX_ECDH_REPLY` sera un `SSH_MSG_KEX_HYBRID_REPLY` (valeur 31 dans le protocole) contenant la concaténation des deux clés du serveur. Le choix du mécanisme hybride utilisé est indiqué dans le message `SSH_MSG_KEXINIT` (RFC 4253, section 7.1) et peut valoir `mlkem768nistp256-sha256`, `mlkem1024nistp384-sha384` ou `mlkem768x25519-sha256` (désormais enregistrés à l'IANA <https://www.iana.org/assignments/ssh-parameters/ssh-parameters.xml#ssh-parameters-16>). Rappelez-vous qu'on parle d'échange de clés <https://www.bortzmeyer.org/echange-cles.html> (symétriques), pas d'authentification, la réponse `SSH_MSG_KEX_HYBRID_REPLY` contient aussi la clé publique du serveur, celle qui sert à vérifier qu'on parle au bon serveur, mais celle-ci n'est pas concernée par ce RFC.

Le mécanisme hybride va nous donner deux secrets partagés, un pour chaque algorithme (le post-quantique et le traditionnel) et le secret qui servira (après dérivation) de clé pour le chiffrement symétrique est la concaténation de ces deux secrets (c'est très proche de ce que fait TLS, dans le RFC 9954).

Un problème classique de la cryptographie post-quantique est la taille des clés, bien plus importante qu'avec les algorithmes traditionnels. (L'hybride est même un tout petit peu plus grand, avec la clé traditionnelle.) SSH a des limites de taille (section 6.1 du RFC 4253) mais les algorithmes spécifiés dans notre RFC ne les atteignent pas.

La section 6 du RFC détaille les conséquences de ce schéma hybride sur la sécurité. Elle conseille par exemple d'utiliser X25519 (RFC 8731) plutôt que les courbes NIST comme P256, pour sa rapidité et sa meilleure résistance aux attaques par canal auxiliaire, sauf si on veut de la conformité plutôt que de la sécurité et qu'on doit utiliser les courbes NIST.

Côté mise en œuvre de ce système hybride, OpenSSH le connait (`ssh -Q kex` pour avoir la liste des algorithmes connus) et, depuis la version 10.4, « "*the hybrid post-quantum algorithm mlkem768x25519-sha256 is now used by default for key agreement*" ». Par exemple :

```
% ssh -v something.bortzmeyer.org
debug1: OpenSSH_10.4p1, OpenSSL 3.6.3 9 Jun 2026
...
debug1: kex: algorithm: mlkem768x25519-sha256
debug1: kex: host key algorithm: ssh-ed25519
debug1: kex: server->client cipher: chacha20-poly1305@openssh.com MAC: <implicit> compression: none
debug1: kex: client->server cipher: chacha20-poly1305@openssh.com MAC: <implicit> compression: none
debug1: expecting SSH2_MSG_KEX_ECDH_REPLY
debug1: SSH2_MSG_KEX_ECDH_REPLY received
...
```

Hmmm, ça aurait dû être `SSH_MSG_KEX_HYBRID_REPLY`, bizarre…