# KCM, gérer des clés cryptographiques sans autorité de certification

Stéphane Bortzmeyer
<stephane+blog@bortzmeyer.org>

Première rédaction de cet article le 28 octobre 2008

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

---

Dans tout système utilisant la cryptographie, la gestion des clés est la principale plaie. La cryptographie garantit qu'un tiers ne pourra pas modifier, ni même écouter la communication mais comment savoir si l'interlocuteur auquel on parle n'est pas justement celui dont on veut se protéger ? Dans le monde physique, on reconnait sa voix et son visage, mais une clé cryptographique n'a pas de voix. Le mécanisme KCM ("*Key Continuity Management*") s'attaque à ce problème.

Prenons pour simplifier le cas du courrier électronique. Pour authentifier un message et pour s'assurer de sa confidentialité, on le chiffre avec la clé publique du destinataire. Mais comment savoir si c'est la bonne ? Les deux normes les plus répandues de chiffrement du courrier, PGP (RFC 3156 [1] et RFC 4880) ou S/MIME (RFC 3850) utilisent deux approches opposées.

PGP utilise des clés qui sont signées par qui le veut, souvent à l'occasion de sessions de signature <https://www.bortzmeyer.org/pgp-session.html>. Si une clé a été signée par quelqu'un à qui on fait confiance, on l'utilise. Sinon, on prend un risque, celui d'utiliser une clé qui est peut-être celle d'un imposteur. C'est donc à l'utilisateur de prendre des décisions de sécurité (≪ La clé de Jeanne est signée par Paul, et celle de Paul par moi. Mais Paul signe t-il après des vérifications sérieuses ? ≫) Le mécanisme est simple, ne nécessite pas d'autorité de certification centrale et marche bien... au sein de petites communautés solides. Il n'a jamais fonctionné pour l'ensemble de l'Internet.

S/MIME, au contraire, repose sur X.509 et dépend donc d'autorités de certification qui signent les messages. L'utilisateur n'a plus de décisions à prendre. Mais ces autorités de certification sont chères, complexes et il n'est pas évident qu'elles fournissent effectivement une bonne sécurité.

En se tournant vers d'autres protocoles, on peut trouver d'autres idées. Ainsi, SSH (RFC 4251) utilise par défaut un mécanisme dit TOFU pour ≪ "*Trust On First Use*" ≫. Les clés des machines auxquelles on

1. Pour voir le RFC de numéro NNN, https://www.ietf.org/rfc/rfcNNN.txt, par exemple https://www.ietf.org/rfc/rfc3156.txt

se connecte ne sont quasiment jamais vérifiées lors de la première connexion mais elles sont stockées dans le fichier `/.ssh/known_hosts` et, si elles changent par la suite, SSH interrompt la connexion. Si on a la chance que le méchant ne soit pas en train de mener un détournement du DNS <https://www.bortzmeyer.org/comment-fonctionne-la-faille-kaminsky.html> ou de BGP <https://www.bortzmeyer.org/faille-bgp-2008.html> juste au moment de la première connexion, on peut être tranquille pour la suite. (Le terme plus formel pour TODU est « "*leap of faith*" » qu'on trouve souvent dans la littérature sur la sécurité des réseaux. On parle aussi du mécanisme « bébé-oie » par allusion au fait que l'oie sortie de l'œuf considère le premier être vu comme sa mère) Ce mécanisme de TOFU a des limites (et il existe des projets pour l'améliorer comme Perspectives <https://www.bortzmeyer.org/perspectives-ssh.html>) mais marche « suffisamment bien » pour l'instant.

Peut-on l'étendre au courrier ? Oui, répondent Simson Garfinkel et Robert Miller dans leur article Johnny 2 : A User Test of Key Continuity Management with S/MIME and Outlook Express <http://cups.cs.cmu.edu/soups/2005/2005proceedings/p13-garfinkel.pdf>. L'idée de base est simple : à chaque configuration d'une nouvelle identité de courrier, le logiciel crée automatiquement une nouvelle paire de clés et envoie la clé publique avec le message. Au premier message reçu de cette adresse, il faudra décider si on l'accepte ou pas mais, ensuite, la clé est conservée et, si elle change, cela indique probablement qu'un usurpateur tente d'utiliser cette adresse, sans avoir la bonne clé.

Le mécanisme a été testé lors d'une experiénce nommé « Johnny 2 » (ainsi nommée en référence au célèbre article « "*Why can't Johnny encrypt ?*" <http://cups.cs.cmu.edu/courses/ups-sp06/notes/060202.pdf> » qui concluait que les problèmes d'utilisabilité étaient au cœur des difficultés de déploiement de la cryptographie). Le résultat est très positif, montrant que les utilisateurs se débrouillent bien avec les différents cas de figure (premier message d'un correspondant, message suivant, message d'un usurpateur).