RFC 7539 : ChaCha20 and Poly1305 for IETF protocols

Stéphane Bortzmeyer
<stephane+blog@bortzmeyer.org>

Première rédaction de cet article le 15 mai 2015

Date de publication du RFC : Mai 2015

https://www.bortzmeyer.org/7539.html

Les algorithmes de cryptographie ChaCha20 et Poly1305 n'avaient jamais fait l'objet d'une spécification stable, permettant leur référencement pour les protocoles IETF. C'est désormais fait avec ce RFC (depuis remplacé par le RFC 8439 [1]), qui normalise aussi bien leur utilisation isolée (ChaCha20 seul ou Poly1305 seul) que leur combinaison, qui fournit du chiffrement intègre (AEAD).

Aujourd'hui, la référence en matière de chiffrement symétrique est AES, entre autres en raison de ses performances très élevées sur du matériel dédié. AES est quatre à dix fois plus rapide que 3DES, qui était la précédente référence. Il est donc logique que AES soit utilisé partout, et notamment qu'il chiffre la quasi-totalité des sessions TLS. Le problème alors est qu'on dépend trop d'AES : si une faille est découverte par la cryptanalyse, on est fichu (AES est par exemple vulnérable dans certains cas <http://research.microsoft.com/pubs/64024/aes-timing.pdf>). Il serait plus rassurant d'avoir des alternatives sérieuses à AES (n'obligeant pas à revenir à 3DES), pour que d'éventuelles percées en cryptanalyse ne cassent pas toute la crypto d'un coup. D'autant plus qu'AES a un défaut : rapide sur du matériel spécialisé, il l'est moins sur du matériel généraliste.

D'où les algorithmes décrits formellement dans ce RFC. Inventés par Bernstein, ils ont déjà fait l'objet d'un certain nombre d'analyses de sécurité (« *"New Features of Latin Dances : Analysis of Salsa, ChaCha, and Rumba"* <http://cr.yp.to/rumba20/newfeatures-20071218.pdf> » et « *"Latin Dances Revisited : New Analytic Results of Salsa20 and ChaCha"* <https://eprint.iacr.org/2012/065.pdf> »). ChaCha20 est un algorithme de chiffrement symétrique, plus rapide qu'AES sur un matériel générique (mise en œuvre purement en logiciel), Poly1305 est un MAC, et les deux peuvent être combinés pour faire du chiffrement intègre (et cela figure désormais dans le registre sur AEAD <https://www.iana.org/assignments/aead-parameters/aead-parameters.xml>).

1. Pour voir le RFC de numéro NNN, https://www.ietf.org/rfc/rfcNNN.txt, par exemple https://www.ietf.org/rfc/rfc8439.txt

La section 2 du RFC décrit les algorithmes (je ne la reprends pas ici, la crypto, c'est trop fort pour moi), et ajoute du pseudo-code, des exemples et des vecteurs de test (il y en a d'autres dans l'annexe A). À l'origine, Poly1305 était décrit comme lié à AES, pour obtenir, par chiffrement du numnique, une chaîne de bits unique et secrète. Mais, en fait, n'importe quelle fonction de chiffrement convient, pas uniquement AES. (L'article du Wikipédia anglophone sur Poly1305 continue à colporter cette erreur.)

En cryptographie, ce sont plus souvent les mises en œuvre que les algorithmes qui ont une faille. La section 3 est donc consacrée aux avis aux programmeurs, pour éviter qu'une erreur de leur part n'affaiblisse ces beaux algorithmes. Poly1305 nécessite de travailler avec des grands nombres et le RFC déconseille d'utiliser la plupart des bibliothèques existantes de gestion des grands nombres comme celle d'OpenSSL. Celles-ci sont trop souvent vulnérables à des attaques par mesure du temps écoulé et le RFC conseille d'utiliser uniquement des bibliothèques qui font leur travail en un temps constant, comme NaCl. Un exemple de mise en œuvre de Poly1305 est poly1305-donna <https://github.com/floodyberry/poly1305-donna>.

La section 4, sur la sécurité, détaille les points importants à suivre pour ne pas se faire casser sa jolie crypto. Pour ChaCha20, avant tout, il faut utiliser un numnique (numnique ? <https://www.bortzmeyer.org/nonce.html>) vraiment unique. On peut utiliser un compteur, on peut utiliser un LFSR, mais il doit être unique.

ChaCha20 et Poly1305 n'utilisent que des opérations simples, normalement faciles à implémenter en un temps constant, qui ne permettra pas à l'attaquant de déduire quoi que ce soit de la mesure des temps de réponse. Attention, programmeurs, certaines fonctions comme la classique `memcmp()` ne s'exécutent pas en un temps constant, et, si elles sont utilisées, permettent certaines attaques.

À noter que ces algorithmes ne sont pas encore dans le registre IANA des algorithmes TLS <https://www.iana.org/assignments/tls-parameters/tls-parameters.xhtml#tls-parameters-4>. Leur mise en œuvre dans Chrome et chez CloudFlare <https://blog.cloudflare.com/do-the-chacha-bett> utilise une valeur non-standard et non interopérable pour identifier l'algorithme. Les discussions se poursuivent pour intégrer ChaCha20 à Firefox <https://bugzilla.mozilla.org/show_bug.cgi?id=917571>. Notez aussi un bon article d'explication de CloudFlare <https://blog.cloudflare.com/it-takes-two-to-chacha-poly/>.

La norme pour ces algorithmes est, depuis mai 2018, le RFC 8439.