RFC 8221 : Cryptographic Algorithm Implementation Requirements and Usage Guidance for Encapsulating Security Payload (ESP) and Authentication Header (AH)

Stéphane Bortzmeyer
<stephane+blog@bortzmeyer.org>

Première rédaction de cet article le 18 octobre 2017

Date de publication du RFC : Octobre 2017

https://www.bortzmeyer.org/8221.html

Le protocole de cryptographie IPsec vient avec une liste d'obligations concernant les algorithmes cryptographiques qu'il **faut** inclure. Cette liste figure dans ce RFC 8221 [1] (qui remplace l'ancien RFC 7321, avec beaucoup de changements). Ainsi, les différentes mises en œuvre d'IPsec sont sûres d'avoir un jeu d'algorithmes corrects en commun, assurant ainsi l'interopérabilité. Cette nouvelle version marque notamment l'officialisation de l'algorithme ChaCha20.

Ce nouveau RFC concerne les deux services d'IPsec, ESP ("*Encapsulating Security Payload*", RFC 4303) et AH ("*Authentication Header*", RFC 4302). Les RFC normatifs sur IPsec se veulent stables, alors que la cryptographie évolue. D'où le choix de mettre les algorithmes dans un RFC à part. Par exemple, la section 3.2 du RFC 4303 note « "*The mandatory-to-implement algorithms for use with ESP are described in a separate RFC, to facilitate updating the algorithm requirements independently from the protocol per se*" » (c'était à l'époque le RFC 4305, remplacé depuis par le RFC 4835, puis par le RFC 7321, puis par notre RFC 8221, trois ans après son prédécesseur).

Ce RFC « extérieur » à IPsec spécifie les algorithmes obligatoires, ceux sur lesquels on peut toujours compter que le pair IPsec les comprenne, ceux qui ne sont pas encore obligatoires mais qu'il vaut mieux mettre en œuvre car ils vont sans doute le devenir dans le futur, et ceux qui sont au contraire déconseillés, en général suite aux progrès de la cryptanalyse, qui nécessitent de réviser régulièrement ce RFC (voir section 1.1). Cette subtilité (différence entre « obligatoire aujourd'hui » et « sans doute obligatoire demain ») mène à une légère adaptation des termes officiels du RFC 2119 : "*MUST-*" (avec le signe moins à la fin) est utilisé pour un algorithme obligatoire aujourd'hui mais qui ne le sera sans doute

1. Pour voir le RFC de numéro NNN, https://www.ietf.org/rfc/rfcNNN.txt, par exemple https://www.ietf.org/rfc/rfc8221.txt

plus demain, en raison des avancées cryptanalytiques, et "*SHOULD+*" est pour un algorithme qui n'est pas obligatoire maintenant mais le deviendra sans doute.

Les section 5 et 6 donnent la liste des algorithmes. Je ne les répète pas intégralement ici, d'autant plus que tous les algorithmes ne sont pas cités, seulement ceux qu'il **faut** mettre en œuvre, car ils sont fiables, et ceux qu'il ne **faut pas** mettre en œuvre, car trop cassés, et qui doivent donc être documentés pour mettre les programmeurs en garde. Parmi les points à noter :

— ESP a un mode de chiffrement intègre ("*authenticated encryption*" qu'on peut aussi traduire par chiffrement vérifié ou chiffrement authentifié, que je n'aime pas trop parce qu'on peut confondre avec l'authentification, cf. RFC 5116). Ce mode est préféré à celui où chiffrement et intégrité sont séparés. Ce mode a un algorithme obligatoire, AES-GCM, décrit dans le RFC 4106 (il était "*SHOULD+*" dans le RFC 7321). Il y aussi un algorithme recommandé ("*SHOULD*", pas "*SHOULD+*" car trop récent et pas encore assez souvent présent dans les mises en œuvres d'IPsec), ChaCha20-Poly1305 (normalisé dans le RFC 7539), qui fait son entrée dans la famille IPsec (cf. RFC 7634). Il semble bien parti pour remplacer AES-GCM un jour. Dans le RFC 7321, chiffrement intègre et chiffrement tout court étaient décrits séparément mais ils sont depuis dans le même tableau.

— Le mode le plus connu d'ESP, celui de chiffrement tout court (le chiffrement intègre est désormais préféré), a deux algorithmes obligatoires (sans compter AES-GCM, cité plus haut, qui permet le chiffrement intègre), ces deux algorithmes sont AES-CBC (RFC 3602) et le surprenant `NULL`, c'est-à-dire l'absence de chiffrement (RFC 2410 ; on peut utiliser ESP pour l'authentification seule, d'où cet algorithme). Il y a aussi plusieurs algorithmes notés "*MUST NOT*", comme DES-CBC (RFC 2405). Ils ne doivent pas être programmés, afin d'être sûr qu'on ne s'en serve pas.

— Le mode d'authentification (enfin, intégrité serait peut-être un meilleur mot mais c'est subtil) d'ESP a un "*MUST*", HMAC-SHA-256 (RFC 4868). L'étoile de SHA-1 (RFC 2404) baissant sérieusement, HMAC-SHA1 n'est plus que "*MUST-*". AES-GMAC, GMAC étant une variante de GCM, qui était en "*SHOULD+*" dans le vieux RFC, redescend en "*MAY*".

— Et AH, lui, a les mêmes algorithmes que ce mode d'authentification d'ESP.

La section 4 donne des conseils sur l'utilisation d'ESP et AH. AH ne fournit que l'authentification, alors qu'ESP peut fournir également le chiffrement. Bien sûr, le chiffrement sans l'authentification ne sert pas à grand'chose, puisqu'on risque alors de parler à l'homme du milieu sans le savoir (voir l'article de Bellovin, S. « "*Problem areas for the IP security protocols*" » dans les "*Proceedings of the Sixth Usenix Unix Security Symposium*" en 1996). Certaines combinaisons d'algorithmes ne sont pas sûres, par exemple, évidemment, ESP avec les algorithmes de chiffrement et d'authentification tous les deux à `NULL` (voir par exemple l'article de Paterson, K. et J. Degabriele, « "*On the (in)security of IPsec in MAC-then-encrypt configurations*" » à l'"*ACM Conference on Computer and Communications Security*" en 2010). Si on veut de l'authentification/intégrité sans chiffrement, le RFC recommande d'utiliser ESP avec le chiffrement `NULL`, plutôt que AH. En fait, AH est rarement utile, puisque ESP en est presque un sur-ensemble, et il y a même eu des propositions de le supprimer `<http://www.schneier.com/paper-ipsec.html>`. AH avait été prévu pour une époque où le chiffrement était interdit d'utilisation ou d'exportation dans certains pays et un logiciel n'ayant que AH posait donc moins de problèmes légaux. Aujourd'hui, la seule raison d'utiliser encore AH est si on veut protéger certains champs de l'en-tête IP, qu'ESP ne défend pas.

Le chiffrement intègre/authentifié d'un algorithme comme AES-GCM (RFC 5116 et RFC 4106) est la solution recommandée dans la plupart des cas `<http://blog.cryptographyengineering.com/2012/05/how-to-choose-authenticated-encryption.html>`. L'intégration du chiffrement et de la vérification d'intégrité est probablement la meilleure façon d'obtenir une forte sécurité.

Triple DES et DES, eux, ont des défauts connus et ne doivent plus être utilisés. Triple DES a une taille de bloc trop faible et, au-delà d'un gigaoctet de données chiffrées avec la même clé, il laisse fuiter des informations à un écoutant, qui peuvent l'aider dans son travail de décryptage. Comme, en prime, il est plus lent qu'AES, il n'y a vraiment aucune raison de l'utiliser. (DES est encore pire, avec sa clé bien trop courte. Il a été cassé avec du matériel dont les plans sont publics.)

Triple DES étant sorti, la « solution de remplacement », si un gros problème est découvert dans AES, sera ChaCha20. Il n'y a aucune indication qu'une telle vulnérabilité existe mais il faut toujours garder un algorithme d'avance.

Pour l'authentification/intégrité, MD5, ayant des vulnérabilités connues (RFC 6151), question résistance aux collisions, est relégué en "*MUST NOT*". (Notez que des vulnérabilités à des collisions ne gênent pas forcément l'utilisation dans HMAC.) SHA-1 a des vulnérabilités analogues <https://security.googleblog.com/2017/02/announcing-first-sha1-collision.html> mais qui ne concernent pas non plus son utilisation dans HMAC-SHA1, qui est donc toujours en "*MUST-*" (le moins indiquant son futur retrait). Les membres de la famille SHA-2 n'ont pas ces défauts, et sont désormais "*MUST*" pour SHA-256 et "*SHOULD*" pour SHA-512.

Voilà, c'est fini, la section 10 sur la sécurité rappelle juste quelques règles bien connues, notamment que la sécurité d'un système cryptographique dépend certes des algorithmes utilisés mais aussi de la qualité des clés, et de tout l'environnement (logiciels, humains).

Ce RFC se conclut en rappelant que, de même qu'il a remplacé ses prédécesseurs comme le RFC 7321, il sera probablement à son tour remplacé par d'autres RFC, au fur et à mesure des progrès de la recherche en cryptographie.

Si vous voulez comparer avec un autre document sur les algorithmes cryptographiques à choisir, vous pouvez par exemple regarder l'annexe B1 du RGS, disponible en ligne <https://www.ssi.gouv.fr/entreprise/reglementation/confiance-numerique/le-referentiel-general-de-securite->.

Merci à Florian Maury pour sa relecture acharnée. Naturellement, comme c'est moi qui tiens le clavier, les erreurs et horreurs restantes viennent de ma seule décision. De toute façon, vous n'alliez pas vous lancer dans la programmation IPsec sur la base de ce seul article, non ?

La section 8 du RFC résume les changements depuis le RFC 7321. Le RFC a été sérieusement réorganisé, avec une seule section regroupant les algorithmes avec AEAD et sans (j'ai gardé ici la présentation séparée de mon précédent article <https://www.bortzmeyer.org/7321.html>). Les sections d'analyse et de discussion ont donc été très modifiées. Par exemple, le RFC 7321 s'inquiétait de l'absence d'alternative à AES, problème qui a été résolu par ChaCha20. Parmi les principaux changements d'algorithmes :

- AES-GCM prend du galon en passant de "*SHOULD+*" à "*MUST*" comme prévu,
- ChaCha20 et Poly1305 (RFC 7539 et RFC 7634) font leur apparition (ils n'étaient pas mentionnés dans le RFC 7321),
- SHA-1, lui, recule, au profit de SHA-2,
- Triple DES est désormais très déconseillé ("*SHOULD NOT*"),
- Un nouveau concept apparait, celui des algorithmes cryptographiques « pour l'IoT », un domaine qui pose des problèmes particuliers, en raisons des fortes contraintes pesant sur ces objets (peu de mémoire, peu de puissance électrique disponible, des processeurs lents). C'est ainsi qu'AES-CCM (RFC 4309) passe à "*SHOULD*" mais uniquement pour l'Internet des Objets, et qu'AES-XCBC (RFC 3566), qui semblait voué à disparaitre, refait surface spécifiquement pour les objets connectés (peut remplacer HMAC pour l'authentification, afin de ne pas avoir à implémenter SHA-2). Notez que XCBC est peu connu et n'a même pas de page dans le Wikipédia anglophone [Caractère Unicode non montré [2]]

En pratique, ce RFC est déjà largement mis en œuvre, la plupart des algorithmes cités sont présents (quand ils sont "*MUST*" ou "*SHOULD*") ou absents (quand ils sont "*MUST NOT*") dans les implémentations d'IPsec. Une exception est peut-être ChaCha20, qui est loin d'être présent partout. Malgré cela, ce RFC n'a pas suscité de controverses particulières, l'IETF avait un consensus sur ces nouveaux choix.

2. Car trop difficile à faire afficher par LaTeX