RFC 8999 : Version-Independent Properties of QUIC

Stéphane Bortzmeyer
<stephane+blog@bortzmeyer.org>

Première rédaction de cet article le 28 mai 2021

Date de publication du RFC : Mai 2021

https://www.bortzmeyer.org/8999.html

Ce RFC, qui fait partie de la première fournée <https://www.bortzmeyer.org/quic.html> sur QUIC, décrit les invariants du protocole QUIC, les choses qui ne changeront pas dans les futures versions de QUIC.

Comme tous les protocoles Internet, QUIC doit faire face à l'ossification, cette tendance à rendre les changements impossibles, en raison du nombre de composants du réseau qui refusent tout comportement qu'ils n'attendaient pas, même si c'était une évolution prévue ou en tout cas possible. Une des méthodes pour limiter cette ossification est la documentation d'invariants, des parties du protocole dont on promet qu'elles ne changeront pas dans les futures versions. Tout le reste peut bouger et les équipements comme les "*middleboxes*" peuvent donc s'appuyer sur ce RFC pour faire les choses proprement, en sachant ce qui durera longtemps, et ce qui est susceptible d'évoluer.

La section 15 du RFC 9000 [1] explique le principe des versions de QUIC. QUIC est actuellement à la version 1. C'est la seule pour l'instant. Mais si une version 2 (puis 3, puis 4...) apparait un jour, il faudra négocier la version utilisée entre les deux parties (section 6 du RFC 9000). Ces nouvelles versions pourront améliorer le protocole, répondant à des phénomènes qui ne sont pas forcément prévisibles. Par exemple QUIC v1 utilise forcément TLS pour la sécurité (RFC 9001), mais les futures versions pourront peut-être utiliser un autre protocole cryptographique, plus sûr, ou plus rapide. L'expérience de nombreux protocoles IETF est qu'il est difficile de déployer une nouvelle version d'un protocole déjà en place, car il y a toujours des équipements dans le réseau qui faisaient des suppositions sur le protocole, qui ne sont plus vraies avec la nouvelle version (et, souvent même pas avec l'ancienne...) et qui perturberont, voire couperont la communication. D'où cette idée de documenter les invariants, en indiquant donc clairement que tout ce qui n'est pas invariant...peut changer. Un exemple d'invariant est « "*A QUIC packet with a long header has the high bit of the first byte set to 1. All other bits in that byte are version*

1. Pour voir le RFC de numéro NNN, https://www.ietf.org/rfc/rfcNNN.txt, par exemple https://www.ietf.org/rfc/rfc9000.txt

specific." ». Actuellement (version 1), le deuxième bit vaut forcément 1 (RFC 9000, section 17.2), mais ce n'est pas un invariant : les autres versions feront peut-être différemmment.

Concevoir des invariants est tout un art. Trop d'invariants et QUIC ne peut plus évoluer. Pas assez d'invariants et ils ne servent plus à rien. Ainsi, il semble difficile de faire un répartiteur de charge qui marche avec toutes les futures versions de QUIC (pas assez d'invariants pour lui <https://github.com/quicwg/load-balancers/issues/22>).

Et dans les paquets, qu'est-ce qui sera invariant ? (Petit rappel au passage, un datagramme UDP peut contenir plusieurs paquets QUIC.) Il y a deux sortes de paquets, les longs et les courts. (Plus rigoureusement, ceux à en-tête long et ceux à en-tête court.) On les distingue par le premier bit. Tous les autres bits du premier octet sont spécifiques d'une version particulière de QUIC, et ne sont donc pas invariants. Ce premier octet est suivi d'un numéro de version sur 32 bits (aujourd'hui, forcément 1, sauf en cas de négociation de version, non encore spécifiée), puis du "*connection ID*" de la destination (attention : longueur variable, dans les paquets longs, il est encodé sous forme longueur puis valeur, cela permettra d'avoir des identificateurs de connexion très longs dans le futur) puis, mais seulement pour les paquets longs, du "*connection ID*" de la source. Tout le reste du paquet dépend de la version de QUIC utilisée.

Notez que la longueur des paquets n'étant pas dans les invariants, on ne peut même pas trouver combien il y a de paquets QUIC dans un datagramme de manière indépendante de la version.

L'identificateur de connexion est une donnée opaque : la façon de le choisir pourra varier dans le futur.

Bien sûr, spécifier rigoureusement les invariants n'empêchera pas les "*middleboxes*" de tirer des fausses conclusions et, par exemple, de généraliser des comportements de la version 1 de QUIC à toutes les versions ultérieures (section 7 du RFC). L'annexe A donne une liste (sans doute incomplète) des suppositions qu'un observateur pourrait faire mais qui sont **erronées**. QUIC essaie de réduire ce qu'on peut observer, en chiffrant au maximum, afin de limiter ces suppositions erronées, mais il reste des choses visibles, qui ne sont pas forcément invariantes. Le RFC écrit ces suppositions de manière neutre, en notant qu'elles sont fausses. J'ai préféré rédiger en insistant sur leur fausseté. Donc, rappelez-vous que, notamment dans les futures versions de QUIC :

— QUIC n'utilise pas forcément TLS,
— les paquets longs peuvent apparaitre même après l'établissement de la connexion,
— il n'y aura pas toujours de phase de connexion au début,
— le dernier paquet avant une période de silence n'est pas forcément uniquement un accusé de réception,
— les numéros de paquet ne croissent pas forcément de un à chaque paquet,
— ce n'est pas toujours le client qui parle le premier,
— les "*connection ID*" peuvent changer très vite,
— le second bit du premier octet, parfois appelé à tort le « QUIC bit », n'est pas forcément à un, et il avait même été proposé de le faire varier <https://mailarchive.ietf.org/arch/msg/quic/Rd_SbFEMU3WXLms9aqw5v69MIKc/> (ce qu'on appelle le graissage, cf. RFC 8701),
— et plusieurs autres points.

À noter que le numéro de version n'apparait que dans les paquets longs. Un programme qui observe des paquets QUIC de versions différentes ne peut donc analyser les paquets courts que s'il a mémorisé les numéros de versions correspondant à chaque "*connection ID*" de destination qu'il voit.

Autre point important : il n'y a pas d'invariant qui identifie un paquet QUIC. Pas de nombre magique ou de chose équivalente. Donc, aucun moyen d'être raisonnablement sûr que ce qui passe est du QUIC. C'est bien sûr volontaire, pour augmenter les chances que la neutralité <https://www.bortzmeyer.org/neutralite.html> soit respectée. Mais cela peut amener des "*middleboxes*" agressives à essayer de deviner si c'est bien du QUIC ou pas, en se trompant. (Le « QUIC bit » n'est évidemment pas une preuve suffisante, il peut s'agir d'un autre protocole où ce bit vaut 1.)