Peut-on se passer des métadonnées dans les protocoles Internet ?

Stéphane Bortzmeyer
<stephane+blog@bortzmeyer.org>

Première rédaction de cet article le 12 novembre 2013

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

Dans les discussions sur les risques d'espionnage des utilisateurs sur l'Internet, le terme de métadonnées revient souvent. Ce sont les données qui, sans faire partie du contenu de la communication, servent à son acheminement. On sait qu'un espion qui n'aurait accès qu'à ces métadonnées peut récolter plein d'informations intéressantes. Peut-on empêcher cela ?

Par exemple, dans un paquet IP, les métadonnées sont entre autres les adresses IP source et destination. Dans un message de courrier électronique, les adresses de l'expéditeur et du destinataire (`MAIL FROM` et `RCPT TO` dans le dialogue SMTP). Pourquoi parle-t-on souvent des métadonnées lors de discussions sur les risques de l'espionnage et la protection de la vie privée ? Parce qu'elles sont particulièrement difficiles à sécuriser et parce qu'elles donnent certes moins d'informations que le contenu des messages mais que c'est encore trop.

Elles sont difficiles à sécuriser car les équipements intermédiaires en ont besoin pour acheminer les informations. La protection la plus évidente (et, effectivement, souvent la plus efficace) contre l'espionnage est le chiffrement. Celui-ci rend les données inaccessibles à l'espion. Mais il laisse les métadonnées. Dans le cas d'IP, l'utilisation d'une technique comme IPsec (RFC 4301 [1]) masque le contenu du paquet mais laissent intactes les adresses de source et de destination. Les techniques comme SSH (RFC 4251) ou TLS (RFC 5246) laissent encore plus de métadonnées visibles (le protocole de couche 4, et les ports). Dans le cas du courrier électronique, chiffrer le message avec PGP (RFC 9580) laisse les métadonnées SMTP (`MAIL FROM`, l'adresse de l'expéditeur, `RCPT TO`, celle du destinataire) en clair (ainsi que celles de l'en-tête du message, cf. RFC 5322).

Bien sûr, c'est moins intéressant pour l'espion que d'avoir tout le contenu du message et le chiffrement reste donc une excellente idée <https://www.bortzmeyer.org/crypto-protection.html>. Mais l'accès aux métadonnées reste un puissant outil. Il permet l'analyse de trafic.

1. Pour voir le RFC de numéro NNN, https://www.ietf.org/rfc/rfcNNN.txt, par exemple https://www.ietf.org/rfc/rfc4301.txt

Alors, comment éviter cela ? Supprimer les métadonnées ? Certaines sont implicites et ne sont pas modifiables facilement. Par exemple, l'écart temporel entre les messages est une métadonnée et elle peut être une source utile d'informations. Si on voit trois connexions HTTPS vers trois serveurs donnés, dans un intervalle très court, on pourra se dire que c'est sans doute la même page Web et on pourra chercher quelle page a des ressources sur ces trois serveurs (un jeu d'enfant pour Google). Autre exemple d'une métadonnée utile à l'écoutant, la taille des messages transmis. On peut en déduire, par exemple lors d'une connexion HTTPS s'il y a eu lecture d'une page Web ou bien envoi d'un fichier. Mais changer la taille d'un message n'est pas trivial. Comme on ne peut pas la réduire à volonté, la seule méthode est de bourrer en ajoutant des données aux messages trop courts. Cela a un coût en matière d'occupation de la capacité réseau. On trouve là un cas classique en sécurité : la nécessité de faire des compromis, ici entre vie privée et développement durable.

D'autres métadonnées sont explicites comme les adresses d'expéditeur et de récepteur. Comment les supprimer ? Parfois, c'est relativement facile. Par exemple, si on chiffre son courrier avec PGP, l'objet du message (champ `Subject:`) reste en clair. Il n'y a aucune bonne raison pour que cela soit le cas, à part la compatibilité avec le courrier traditionnel. Les logiciels qui chiffrent avec PGP devraient chiffrer tout le message (type `message/rfc822`) et le mettre comme partie MIME d'un message dont l'objet serait banalisé (`Subject: this is a message`). Mais d'autres métadonnées explicites sont là pour de bonnes raisons et vont être plus dures à éradiquer.

Le problème avec l'adresse de récepteur, c'est qu'elle est indispensable au routage. Les équipements intermédiaires ont besoin de cette adresse pour acheminer le message. Pour ne pas avoir d'adresse de destinataire visible, la seule solution est, là encore, très coûteuse en ressources réseau : il faut diffuser à tout le monde, comme le fait BitMessage <`https://www.bortzmeyer.org/bitmessage.html`>.

Et pour l'adresse de l'expéditeur ? Elle n'est pas indispensable à l'acheminement, non ? On pourrait la supprimer (proposition dite « "*sourceless protocols*" »). Par exemple, sans changer le format des paquets IP, on pourrait imaginer une adresse source standard qui indiquerait « ce paquet est anonyme, il ne dira pas d'où il vient ». Mais ce n'est pas toujours aisé. Par exemple, parfois, il faut pouvoir envoyer un avis de non-remise comme les messages ICMP "*Destination Unreachable*", ou comme les messages de courrier du RFC 8098. Pour cela, il faut bien avoir une adresse à qui envoyer ces notifications. Autrement, l'expéditeur enverra dans le noir, en ne sachant jamais si son message est arrivé.

Et c'est encore plus vrai pour les protocoles qui exigent un dialogue alors que la couche basse ne fournit qu'un service de messages. Par exemple, le protocole de transport le plus utilisé dans l'Internet, TCP, nécessite que les messages puissent circuler dans les deux sens, ce qui impose que les adresses IP mises dans le paquet soient joignables.

Notez que les protocoles à connexion ne posent pas les mêmes problèmes car ils peuvent être « sans source ». Dans le vieux X.25 (ce n'est pas moi qui ai choisi l'exemple, c'est Rémi Després, lors d'une discussion/pizza), le serveur n'avait pas forcément l'adresse du client. Chaque routeur sur le trajet établissait une connexion avec le routeur suivant, ce qui fait qu'une machine ne connaissait que son prédécesseur. Il est amusant de noter que Tor fonctionne comme cela, mais avec bien des services en plus pour limiter encore davantage la fuite d'information (routeurs gérés par plusieurs organisations pour qu'ils ne puissent pas s'informer mutuellement, chiffrement systématique). Faudra-t-il donc remplacer IP par un protocole à connexion ?

Aujourd'hui, pour sécuriser le courrier électronique, il faudra combiner plusieurs techniques. Par exemple PGP pour du chiffrement de bout en bout (s'assurer que les serveurs intermédiaires ne pourront pas lire le message) **et** SMTP sur TLS (RFC 3207) pour chiffrer les métadonnées vis-à-vis d'un écoutant situé sur le trajet. Certes, SMTP sur TLS a des tas de limites, question sécurité (notamment, peu de MTA

vérifient le certificat du pair) mais c'est mieux que de laisser les messages PGP circuler en clair, avec les métadonnées.

À noter qu'un écoutant peut quand même savoir qu'il y a un trafic SMTP. Si la communication est entre deux serveurs SMTP personnels, ayant peu de comptes ou même un seul, il pourra, même si tout est chiffré avec TLS, savoir qu'il y échange de courrier entre deux personnes identifiées. On est mieux protégé contre ce genre d'espionnage si on utilise un gros serveur. L'écoutant ne sera guère avancé en sachant juste qu'un utilisateur de Yahoo écrit à un utilisateur de Gmail. Dans le cas de ces gros silos, le danger est évidemment ailleurs, dans leur envoi d'informations à la NSA. Il faudrait donc des gros serveurs, mais gérés par des organismes honnêtes. Pas facile, la sécurité.

Un dernier mot sur une technique qui peut aider à résoudre notre problème : le chiffrement homomorphique. Comme il permet de faire des opérations sur des données chiffrées, il offre (théoriquement, car je ne crois pas que cet usage ait déjà été exploré) la possibilité de combiner routage et chiffrement des métadonnées.