RFC 7671 : Updates to and Operational Guidance for the DANE Protocol

Stéphane Bortzmeyer
<stephane+blog@bortzmeyer.org>

Première rédaction de cet article le 21 octobre 2015

Date de publication du RFC : Octobre 2015

https://www.bortzmeyer.org/7671.html

Le protocole DANE (RFC 6698 [1]) de sécurisation des communications utilisant TLS est encore très peu déployé. Donc, il n'y a pas vraiment de retours d'expériences mais les premiers essais ont quand même permis de se faire des idées, et, pour aider ceux et celles qui vont déployer DANE dans les années à venir, ce nouveau RFC rassemble quelques conseils pratiques.

Donc, DANE permet d'améliorer l'authentification des sessions TLS en publiant dans le DNS, sécurisés par DNSSEC, des enregistrements nommés TLSA (non, ce n'est pas un acronyme, ce nom est à utiliser sans comprendre son étymologie). Ces enregistrements indiquent le certificat utilisé pour la connexion TLS. Ainsi, un gérant d'un serveur TLS n'est plus obligé de dépendre d'une tierce partie, genre AC.

J'ai simplifié car il y en a fait beaucoup de choix : DANE permet de publier des clés brutes, pas seulement des certificats, il permet de publier un certificat d'AC, pas seulement le certificat final, etc. Le RFC 6698 normalise trois champs dans l'enregistrement TLSA (cf. section 2 de notre RFC), « Utilisation du certificat » (quatre valeurs possibles), « Sélecteur » (deux valeurs possibles) et « Méthode de correspondance » (trois valeurs possibles). En tout, il existe vingt-quatre combinaisons, ce qui est probablement trop (combien de clients DANE ont-ils vraiment été testés avec les vingt-quatre combinaisons différentes ?). Notre RFC réduit ce choix par ses conseils.

Voici un exemple d'un enregistrement DANE à peu près conforme aux recommandations du RFC, celui de mon blog :

1. Pour voir le RFC de numéro NNN, https://www.ietf.org/rfc/rfcNNN.txt, par exemple https://www.ietf.org/rfc/rfc6698.txt

```
% dig TLSA _443._tcp.www.bortzmeyer.org

;; ->>HEADER<<- opcode: QUERY; status: NOERROR; id: 60895
;; Flags: qr rd ra; QUERY: 1; ANSWER: 1; AUTHORITY: 8; ADDITIONAL: 6

;; QUESTION SECTION:
;; _443._tcp.www.bortzmeyer.org. IN TLSA

;; ANSWER SECTION:
_443._tcp.www.bortzmeyer.org. 86400 IN TLSA 2 0 2 EB0AD84F11B4B08BF76C7866EF32842292BBB2862FB6FC49C0A3F80762
```

L'« utilisation du certificat » est DANE-TA (valeur 2, mais le RFC 7218 a introduit de jolis noms pour ces codes), le « sélecteur » est CERT (valeur : 0, on testera tout le certificat, pas juste la clé publique) et la « méthode de correspondance » 2, ce qui veut dire qu'on publie un condensat en SHA-512 de ce certificat.

Notre RFC 7671 ne fait pas qu'ajouter des conseils opérationnels au RFC 6698, il le modifie légèrement sur certains points (section 3 de notre RFC), par exemple en ajoutant qu'un client DANE doit utiliser l'extension TLS nommée SNI ("*Server Name Indication*", cf. RFC 6066).

Que mettre dans le champ « utilisation du certificat » de l'enregistrement TLSA ? La section 4 de notre RFC s'attaque à ce problème. Les quatre valeurs possibles sont PKIX-TA (le certificat utilisé pour TLS doit être vérifié par les mécanismes X.509 classiques **et** le certificat publié dans le TLSA doit être celui de l'AC), PKIX-EE (le certificat utilisé pour TLS doit être vérifié par les mécanismes X.509 classiques **et** le certificat publié dans le TLSA doit être celui du serveur TLS), DANE-TA (le certificat publié dans le TLSA est celui d'une AC à partir de laquelle on vérifie le certificat du serveur TLS) et enfin DANE-EE (le certificat publié dans le TLSA est celui du serveur TLS). Pour DANE-TA et DANE-EE, il n'y a pas besoin d'un magasin de certificats pré-existant sur le client TLS.

Il n'y a pas forcément besoin, pour un client DANE, de gérer les quatre utilisations. S'il le fait, un attaquant qui arrive à modifier les enregistrements DNS peut remplacer un PKIK-EE par un DANE-EE et la sécurité supplémentaire de la vérification X.509 qu'impose X.509 est donc assez illusoire. Un client paranoïaque et traditionnaliste qui veut imposer des vérifications X.509 doit donc ignorer purement et simplement les enregistrements avec DANE-EE ou DANE-TA.

Mais notre RFC donne le conseil contraire : ne gérer **que** DANE-EE et DANE-TA, en raison du manque de confiance qu'on peut avoir dans les centaines d'AC existantes, dont beaucoup ont déjà fait la preuve qu'on ne pouvait pas leur faire confiance <http://www.01net.com/actualites/comment-le-ministere-des-finances-espionne-le-trafic-web-de-ses-collaborateurs-610140.html>. La vérification X.509 de ces AC n'apporte pas grand'chose en sécurité. En outre, il n'existe aucune liste standard d'AC reconnues, ce qui rend PKIX-TA et PKIX-EE très fragiles : le magasin du client doit être un sur-ensemble du magasin de tous les serveurs possibles ce qui mène à des listes d'AC délirantes (regardez celle de votre navigateur Web).

Depuis la publication de DANE, d'autres techniques visant à boucher les trous de X.509 sont apparues, comme les certificats au grand jour du RFC 6962. Comme la liste publique décrite dans ce RFC ne stocke que des certificats émis par des AC connues de la liste, les vérifications du RFC 6962 ne doivent pas être faites pour les utilisations DANE-TA et DANE-EE.

La section 5 rassemble ensuite des conseils spécifiques à chaque usage particulier du certificat publié avec DANE. D'abord, l'usage DANE-EE (valeur : 3). Dans cet usage, le certificat du serveur est publié

dans l'enregistrement TLSA et les vérifications X.509 (RFC 5280) ne doivent **pas** être faites. Le RFC 6698, qui normalisait DANE, ne disait apparemment pas ce qu'il fallait faire des autres informations contenues dans le certificat comme la date d'expiration. Notre RFC 7671 précise (et c'est un des points où il met à jour le RFC 6698, ne se contentant pas d'ajouter des conseils, cf. section 12 pour les autres cas) que les autres informations doivent être ignorées. Si le certificat est dans un enregistrement TLSA, il est bon, quelle que soit la date d'expiration. Idem pour le nom de serveur dans le certificat.

Le sélecteur recommandé, pour cet usage, est SPKI (valeur : 1) qui indique qu'on ne trouve pas dans l'enregistrement TLSA tout le certificat mais juste la clé publique. Raisons de cette recommandation :

— Cela évite de changer l'enregistrement TLSA quand les métadonnées dans le certificat changent (cas des certificats CAcert `<https://www.bortzmeyer.org/cacert.html>`, qui ne durent que six mois),
— C'est compatible avec les clés nues du RFC 7250,
— C'est cohérent avec l'obligation d'ignorer les données du certificat, à l'exception de la clé publique.

Il n'est pas recommandé d'envoyer le certificat entier (méthode de correspondance Full, valeur 0), notamment en raison des problèmes du DNS avec des enregistrements de grande taille (ce qui est souvent le cas des certificats). Il faut plutôt utiliser un condensat du certificat. En cas de doute, notre RFC recommande donc DANE-EE + SPKI + SHA-256 (notez que l'enregistrement TLSA de mon serveur HTTP, cité plus haut, ne suit pas cette recommandation mais ce que le RFC présente comme un second choix).

Mais ce n'est pas impératif. L'usage DANE-TA, où on publie dans l'enregistrement TLSA le certificat de l'AC et pas celui du serveur, est tout à fait légitime (c'est celui que j'utilise). Attention, il impose d'envoyer toute la chaîne de certificats dans la réponse TLS. (Avec OpenSSL et GnuTLS, il suffit de concaténer tous ces certificats, dans le bon ordre - du plus spécifique au plus général, cf. RFC 5246, section 7.4.2 - dans un seul fichier PEM, qui sera celui indiqué au serveur.) Cette nouvelle obligation a été ajoutée pour le cas où l'AC choisie par le gérant du serveur DANE ne soit pas dans le magasin du client (pour CAcert `<https://www.bortzmeyer.org/cacert.html>`, c'est fréquent).

Contrairement à l'usage DANE-EE, il est recommandé, avec cet usage DANE-TA, d'utiliser le sélecteur Cert (valeur : 0), car, cette fois, il faut tenir compte du contenu complet du certificat (comme les limites de longueur, par exemple, ou bien les restrictions sur les noms que l'AC peut approuver, cf. RFC 5280, section 4.2.1.10).

Un rappel de sécurité pour finir la section sur DANE-TA : cet usage vous protège contre les autres AC, malhonnêtes ou piratées, mais pas contre les autres utilisateurs de la même AC. Choisissez donc bien votre Autorité de Certification !

Deux autres usages possibles de l'enregistrement TLSA maintiennent des validations X.509 classiques. Ce sont les usages PKIK-EE (valeur : 1) et PKIX-TA (valeur : 0).

Un risque important de l'utilisation de DANE est la désynchronisation entre le ou les certificat(s) envoyés en TLS et ceux publiés dans le DNS, dans l'enregistrement TLSA (section 6 de notre RFC). Il faut absolument maintenir ces informations en cohérence, ce qui est d'autant plus difficile que les outils de supervision actuels n'offrent en général pas grand'chose pour DANE. Ainsi, si on utilise DANE-EE ou PKIX-EE, quand on change de clé publique sur le serveur TLS, il faut penser à mettre à jour l'enregistrement TLSA. Compte-tenu de l'importance de la mise en cache des données dans le DNS, cela nécessite de publier le nouveau certificat dans le DNS à l'avance (cf. section 8.4). Bref, notre RFC fait remarquer à juste titre qu'il est très recommandé que le DNS et le TLS soient gérés par la même organisation, autrement, le risque de cafouillage est trop élevé. (DANE est encore d'utilisation récente et on n'a pas encore vu beaucoup de cas de changement de certificat. Quand ça arrivera, il est à craindre qu'il y ait quelques malheurs, lorsque l'administrateur système oubliera qu'il avait publié un enregistrement TLSA.)

La section 8 met en garde les gérants de serveurs utilisant DANE contre le fait qu'on ne peut pas espérer que tous les clients DANE comprennent tous vos enregistrements TLSA. Ceux-ci peuvent spécifier un algorithme de condensation récent que tout le monde ne gère pas encore. Ou bien ils peuvent utiliser les clés nues du RFC 7250, que tout le monde n'accepte pas, loin de là. Attention donc, surtout lorsque vous décidez des paramètres.

La même section traite du difficile problème du remplacement : en raison des caches du DNS, les changements de clé ou de paramètres ne sont pas vus immédiatement partout. Il faut donc suivre quelques précautions quand on est hébergeur DNS d'un domaine qui utilise DANE :

— Publier à l'avance les nouveaux enregistrements TLSA (quelle avance ? Le TTL de l'ensemble TLSA.)
— Une fois le serveur TLS reconfiguré pour la nouvelle clé, virer les anciens enregistrements TLSA.
— Si on change les paramètres (par exemple on passe d'un sélecteur CERT à un sélecteur SPKI ou bien le contraire), bien vérifier que tous les cas sont couverts pour les anciennes et les nouvelles chaînes de certificat.

À noter que le problème des clients DANE ne gérant pas toutes les valeurs possibles des paramètres est d'autant plus agaçant qu'il n'y a pas forcément de règles précises sur le choix d'un enregistrement TLSA, si plusieurs conviennent. Imaginons que vous ayez deux enregistrements TLSA, identiques sauf la méthode de correspondance qui vaut SHA-256 pour l'un et SHA-512 pour l'autre. Lequel sera utilisé par le client DANE ? Il faudrait évidemment que ce soit le plus fort (SHA-512) mais le RFC 6698 ne dit pas comment. Notre nouveau RFC demande au contraire que tout client DANE ait un classement des algorithmes de condensation, idéalement configurable par l'utilisateur, et qui soit par défaut l'ordre des valeurs (SHA-256 : 1, SHA-512 : 2, etc). Le client DANE, lorsqu'il a le choix, doit prendre le mieux classé (ou la méthode Full - valeur : 0 - qui n'utilise pas d'algorithme de condensation cryptographique, si un enregistrement TLSA l'utilise).

La section 10 de notre RFC rassemble des conseils divers pour la conception de protocoles utilisant DANE. D'abord, la taille des enregistrements, importante car le DNS a souvent des limites de taille. Par exemple, la plupart des serveurs faisant autorité ont une taille maximale de 4 096 octets pour les réponses se faisant en UDP. Même si la réponse est inférieure à cette valeur, une taille plus grande que 1 500 octets a de grandes chances de mener à une fragmentation de la réponse UDP, avec des conséquences négatives sur les performances voire, pire, sur la fiabilité de l'acheminement de la réponse (pas mal de pare-feux bogués bloquent stupidement les fragments). En théorie, on peut alors utiliser TCP pour transmettre les données mais lui aussi est souvent bloqué par des logiciels configurés par des incompétents.

Avec les grands certificats qu'on trouve souvent dans le monde X.509, le risque de paquets trop grands est bien plus élevé qu'avec les données de petite taille qu'on trouve habituellement dans le DNS. D'où les conseils de notre RFC : n'utilisez pas ensemble le sélecteur CERT (valeur : 0, il indique qu'il faut comparer tout le certificat, pas juste la clé publique) et la méthode de correspondance FULL (valeur : 0, elle indique que l'enregistrement TLSA contient les données complètes, pas juste un condensat). En effet, cette combinaison oblige à mettre un certificat entier dans l'enregistrement TLSA, ce qui dépasse en général la taille raisonnable. (D'autant plus qu'en cas de remplacement, il faudra publier simultanément **deux** enregistrements TLSA.) Avec un sélecteur SPKI (valeur : 1, on ne vérifie que la clé publique), le problème est moins grave, néanmoins autant ne jamais utiliser la méthode de correspondance FULL.

Notre RFC contient aussi des conseils pour le certificat envoyé par TLS. Il y a normalement des règles précises sur le **nom** ("*subject*", en terminologie X.509) qui doit être mis dans le certificat, afin de permettre l'authentification. Néanmoins, ces règles sont complexes (donc pas toujours bien comprises) et ont changé dans le temps. Il peut être prudent de rajouter des noms « au cas où ». Ainsi, pour un service auquel on accès via un enregistrement SRV, le RFC 7673 prescrit de vérifier le nom en partie droite du SRV. Mais comme on ne peut pas être sûr de ce que feront tous les clients DANE, autant essayer de mettre aussi dans le certificat le nom qui est en partie gauche du SRV.

La section 11 se penche, elle, sur DNSSEC. DANE est complètement dépendant de DNSSEC : sans lui, plus aucun enregistrement TLSA n'est accepté. La sécurité de DNSSEC est donc cruciale pour DANE. Avec X.509, la sécurité de l'AC que vous choisissez a peu d'importance, puisque n'importe quelle AC peut émettre un certificat pour votre serveur. (Les « "*Name constraints*" » - section 4.2.1.10 du RFC 5280 - sont très peu utilisées.) Avec DNSSEC, au contraire, vous ne dépendez que d'acteurs que vous choisissez : votre hébergeur DNS, votre BE (souvent le même que l'hébergeur), votre registre...

DNSSEC dépendant du DNS, il va donc falloir sécuriser votre configuration DNS (cf. le bon guide de l'ANSSI <https://www.ssi.gouv.fr/guide/bonnes-pratiques-pour-lacquisition-et-lexploitation->). Par exemple, il est recommandé de placer un verrou ("*registry lock*") pour limiter les risques de détournement de votre domaine, ou de modification non autorisée.

Certaines opérations dans le DNS sont délicates à faire lorsqu'un domaine est signé : le passage d'un BE à l'autre lorsque chacun est également l'hébergeur DNS. Une technique parfois recommandée est de dé-signer la zone temporairement. On retire le DS, on attend l'expiration dans les caches, on migre, on met un nouveau DS. Cette méthode est sûre et simple mais elle laisse une fenêtre d'insécurité, où la zone n'est pas signée. Sans DANE, c'est un inconvénient supportable (il faut juste espérer que l'attaquant ne frappe pas à ce moment), avec DANE, c'est impossible car les enregistrements DANE seront ignorés pendant cette période d'insécurité. S'ils sont d'usage PKIK-EE ou PKIK-TA, ce n'est pas trop grave, le certificat sera toujours validé par X.509 mais s'ils sont d'usage DANE-TA ou surtout DANE-EE, on ne pourra plus se connecter au serveur. Voilà pourquoi il faut étudier les conditions de migration du domaine **avant** de déployer DANE.

DNSSEC n'a pas de notion de révocation (là encore, sauf si on utilise les usages PKIX-EE ou PKIK-TA, où les vérifications de la révocation du certificat peuvent être faites). Attention donc de ne pas mettre des TTL trop longs à vos enregistrements TLSA (section 13 du RFC, au début) et surtout pas des périodes de validité des signatures trop longues pour éviter des attaques par rejeu. Pour ces périodes de validité, le RFC recommande quelques jours pour des clés importantes et quelques semaines pour les autres. Regardez par exemple l'enregistrement TLSA de mon serveur de courrier :

```
_25._tcp.mail.bortzmeyer.org. 86400 IN TLSA 2 0 1 (
FF2A65CFF1149C7430101E0F65A07EC19183A3B633EF
4A6510890DAD18316B3A )
_25._tcp.mail.bortzmeyer.org. 86400 IN RRSIG TLSA 8 5 86400 (
20151101223339 20151012192513 53605 bortzmeyer.org.
BEiYRQh4i4Z3RH0h3yZqDjltCmv...
```

Le TTL est d'une journée (ce qui fait qu'un changement d'urgence du certificat pourrait casser la vérification pendant une journée, dans le pire des cas). La durée de validité de la signature est de 20 jours (<Validity><Default>P20D</Default></Validity> dans la configuration OpenDNSSEC). Avec des courtes validités, on ne peut pas se permettre d'avoir un serveur DNS maître en panne très longtemps, et des périodes de quelques jours ne doivent donc être utilisées que si on dispose d'une équipe professionnelle et réactive.

Enfin, il est évidemment crucial de superviser <https://www.bortzmeyer.org/monitor-dnssec.html> sa configuration DNSSEC.

Enfin, dernier conseil, section 14 (mais on l'a déjà dit plus haut), attention au fait que certains enregistrement TLSA seront considérés comme inutilisables par certains clients DANE. En TLS « opportuniste » (RFC 7435), cela se traduira par une absence de vérification du certificat par le client. Mais

en mode plus sécurisé (très rare à l'heure actuelle), cela peut se traduire par une impossibilité de se connecter en TLS.

Terminons avec un peu de ligne de commande Unix. Comment on génère un enregistrement TLSA ? Par exemple, pour le mien, d'usage DANE-TA (valeur : 2), je prends le certificat de mon AC, CAcert <https://www.bortzmeyer.org/cacert.html> (wget https://www.cacert.org/certs/root.crt) et voici deux méthodes possibles, avec OpenSSL, puis avec hash-slinger <http://people.redhat.com/pwouters/hash-slinger/>. D'abord, avec OpenSSL (voir cet article <http://blog.huque.com/2012/10/dnssec-and-certificates.html>) :

```
%  openssl x509 -in  root.crt -outform DER | openssl sha256
(stdin)= ff2a65cff1149c7430101e0f65a07ec19183a3b633ef4a6510890dad18316b3a
```

Et on n'a plus qu'à mettre :

```
_25._tcp.MXSERVER.DOMAIN.  IN TLSA 2 0 1  ff2a65cff1149c7430101e0f65a07ec19183a3b633ef4a6510890dad18316b3a
```

dans sa configuration DNS. Avec hash-slinger :

```
%  tlsa --create --port  25 --certificat root.crt --usage 2 --selector 0 --mtype 1 \
             mail.bortzmeyer.org
_25._tcp.mail.bortzmeyer.org. IN TLSA 2 0 1 ff2a65cff1149c7430101e0f65a07ec19183a3b633ef4a6510890dad18316b3a
```

Troisième possibilité, le faire en ligne <https://www.huque.com/bin/gen_tlsa>.

Et si je voulais un enregistrement TLSA d'usage DANE-EE (valeur : 3) ? Avec OpenSSL, ce serait :

```
printf '_25._tcp.%s. IN TLSA 3 1 1 %s\n' \
        $(uname -n) \
        $(openssl x509 -in server.pem -noout -pubkey |
            openssl pkey -pubin -outform DER |
            openssl dgst -sha256 -binary |
            hexdump -ve '/1 "%02x"')
_25._tcp.www.foobar.example. IN TLSA 3 1 1 755b35d2c34c54729d92f97e25d8a8be020235d5b1c6b7751bfefc896b8e5164
```

Avec hash-slinger, il suffit de modifier les paramètres de la ligne de commande.

Si vous voulez tester vos enregistrements TLSA, ce qui est très recommandé :
— Pour un serveur HTTP, .
— Pour un serveur SMTP, .
Et pour lire davantage, vous pouvez voir le dossier thématique de l'AFNIC sur DANE <https://www.afnic.fr/fr/ressources/publications/dossiers-thematiques/securiser-les-communications-html>.