RFC 5485 : Digital Signatures on Internet-Draft Documents

Stéphane Bortzmeyer
<stephane+blog@bortzmeyer.org>

Première rédaction de cet article le 6 mars 2009. Dernière mise à jour le 13 décembre 2010

Date de publication du RFC : Mars 2009

https://www.bortzmeyer.org/5485.html

Lorsqu'un document, par exemple un "*Internet-Draft*" est publié et distribué via l'Internet, il peut être difficile de prouver qu'il est bien « authentique », qu'il a été validé et pas modifié ensuite. La technique la plus courante pour cela est la signature cryptographique et ce nouveau RFC explique comment elle s'applique aux "*Internet-Drafts*".

Le principe est le suivant : lors de la réception d'un nouvel "*Internet-Draft*" le secrétariat de l'IETF le signe au format CMS (RFC 3852 [1]), en utilisant un certificat X.509 (qui reste apparemment à publier). La signature, détachée de l'"*Internet-Draft*", au format PKCS#7, est publiée avec les "*Internet-Drafts*".

Pourquoi CMS et pas PGP tel que normalisé dans le RFC 4880 ? Je l'ignore mais c'est sans doute lié au fait que les défenseurs de X.509 sont plus bruyants, beaucoup d'argent est en jeu pour eux, avec les coûteuses AC.

Les détails, maintenant. CMS utilise ASN.1 (section 1.2) et ASN.1 a certains encodages qui permettent plusieurs représentations de la même donnée. C'est évidemment gênant pour la signature et notre RFC impose donc l'encodage DER qui n'a pas ce défaut.

Le texte de l'"*Internet-Draft*" devra être **canonicalisé**. Pour les "*Internet-Drafts*" au format texte seul, il faut notamment avoir une forme unique pour les fins de lignes, la forme standard CRLF (cf. annexe B du RFC 5198, la seule norme de cette forme standard). Pour ceux en XML (section 2.3), suivant la norme XML <http://www.w3.org/TR/2006/REC-xml-20060816>, ce sera LF qui marquera la fin de ligne. Les autres formats seront simplement traités comme du binaire.

1. Pour voir le RFC de numéro NNN, https://www.ietf.org/rfc/rfcNNN.txt, par exemple https://www.ietf.org/rfc/rfc3852.txt

CMS est une norme très complexe et la section 3 est consacrée à définir un profil de CMS pour nos signatures. C'est aussi l'occasion de spécifier quelques points pratiques comme l'importance pour le secrétariat qui publie les "*Internet-Drafts*" d'avoir des ordinateurs à l'heure, pour que l'attribut `Signing-Time` (section 3.2.3.3) soit correct.

Enfin, comme le but de toute l'affaire est la sécurité, le RFC se termine sur des "*Security Considerations*" (sections 5 et 6) qui recommandent l'usage de HSM pour stocker la clé privée de l'IETF.

Les signatures détachées portent l'extension `p7s` (section 2 et cf. RFC 3851). Les signatures peuvent être vérifiées avec OpenSSL (version 0.9.9 minimum car il faut la commande `cms`, et version 1.0 maximum, en raison d'un problème avec la 1.1, vous pouvez taper `openssl version` pour tester votre version). En attendant que j'installe cette version, j'ai fait les essais suivants avec la commande `smime`, puisque S/MIME et CMS ont beaucoup en commun.

Si je suis un simple lecteur d'"*Internet-Drafts*" et que je veux vérifier une signature (annexe A) :

```
% openssl smime -verify -CAfile $CERT -content ${id} \
         -inform DER -in ${id}.p7s -out /dev/null
```

où `id` est le nom de l'"*Internet-Draft*" et `CERT` celui du fichier `.pem` contenant le certificat de l'IETF. Si tout se passe bien, OpenSSL répondra :

```
Verification successful
```

et si un seul caractère de l'"*Internet-Draft*" a été modifié :

```
Verification failure
14024:error:04077068:rsa routines:RSA_verify:bad signature:rsa_sign.c:235:
14024:error:21071069:PKCS7 routines:PKCS7_signatureVerify:signature failure:pk7_doit.c:981:
14024:error:21075069:PKCS7 routines:PKCS7_verify:signature failure:pk7_smime.c:312:
```

Si je veux juste regarder le certificat contenu dans la signature :

```
% openssl pkcs7 -print_certs -inform DER -in ${id}.p7s
subject=/C=FR/L=Paris/O=Bidon/CN=Ca essai
issuer=/C=FR/L=Paris/O=Bidon/CN=CA essai
```

Si je suis le secrétariat de l'IETF et que je veux signer un "*Internet-Draft*" (bien sûr, dans la réalité, cela ne sera pas fait à la main avec la ligne de commande) :

```
% openssl smime -sign -in $id -inkey privkey.pem -outform DER -binary \
    -signer mycacert.pem -out ${id}.p7s -noattr
```

Si vous voulez faire l'essai complet, le moyen le plus rapide de créer l'infrastructure du secrétariat (les fichiers `privkey.pem` et `mycacert.pem`) est :

```
% openssl genrsa -out privkey.pem 1024
% openssl req -sha1 -new -x509  -key privkey.pem -out mycacert.pem -days 1095
```

À noter que cette technique ne s'applique qu'aux "*Internet-Draft*", pas aux RFC eux-même (et cela sera sans doute difficile en raison de l'extrême prudence du "*RFC editor*" <`http://www.rfc-editor.org/`>).

La mise en service de ce système a eu lieu (avec retard) en décembre 2010 et son mode d'emploi est désormais documenté <`https://www.ietf.org/standards/ids/idsignatures/`>.

Ce RFC a été légèrement mis à jour par la suite avec le RFC 8358.