RFC 6726 : FLUTE - File Delivery over Unidirectional Transport

Stéphane Bortzmeyer
<stephane+blog@bortzmeyer.org>

Première rédaction de cet article le 7 novembre 2012

Date de publication du RFC : Novembre 2012

https://www.bortzmeyer.org/6726.html

Peut-on envoyer un fichier d'un ordinateur à un autre lorsque la liaison est **unidirectionnelle**, c'est-à-dire que les paquets ne voyagent que dans un sens ? Pas question de se servir de FTP ou de HTTP, qui reposent sur le protocole de transport TCP, qui a besoin de bidirectionnalité (car il dépend du retour des accusés de réception). Le protocole FLUTE ("*File Delivery over Unidirectional Transport*") normalisé dans ce RFC, fournit un ensemble de mécanismes pour effectuer ce transfert. La principale application concerne le "*multicast*", où une machine unique diffuse un fichier à plusieurs autres, qu'elle ne connait pas forcément et qui ne peuvent pas lui écrire. Un exemple typique est la distribution des mises à jour d'un gros logiciel à des milliers, voire millions, de machines dans le monde. Parmi les usages possibles, distribuer de la vidéo <http://www.etsi.org/deliver/etsi_ts/102400_102499/102472/01.03.01_60/ts_102472v010301p.pdf>, une nouvelle version d'un jeu très populaire, etc.

Pour cela, FLUTE s'appuie sur le système ALC (RFC 5775 [1]) + LCT ("*Layered Coding Transport*", RFC 5651), qui fait l'essentiel du travail. ALC+LCT sert de protocole de transport d'objets binaires, FLUTE se charge de la sémantique. FLUTE dépend aussi d'autres protocoles de la famille "*multicast*", comme FEC (RFC 5052), qui assure la fiabilité du transport, en permettant au récepteur de détecter les erreurs.

Transférer des gros fichiers sur un réseau ouvert comme l'Internet implique également un mécanisme de contrôle de la congestion, pour éviter que l'émetteur ne noie le réseau sous les paquets, sans se rendre compte que les tuyaux sont pleins. Comme on n'a pas TCP pour limiter la congestion en détectant les paquets perdus, il faut trouver un autre mécanisme. Dans le cas de FLUTE, les récepteurs souscrivent plusieurs abonnements "*multicast*", éventuellement de débits différents, et jonglent avec ces divers abonnements. (Sur des réseaux fermés, le contrôle de congestion de FLUTE peut se faire par encore d'autres mécanismes comme le "*shaping*".)

Les métadonnées (les propriétés du fichier) sont transmises dans un élément XML. On y trouve entre autres :

1. Pour voir le RFC de numéro NNN, https://www.ietf.org/rfc/rfcNNN.txt, par exemple https://www.ietf.org/rfc/rfc5775.txt

— Un identifiant du fichier, un URI. C'est juste un identifiant, et rien ne garantit qu'on puisse récupérer le fichier à cet « endroit ».
— Un type de contenu, par exemple `text/plain`.
— La taille du fichier.
— Un encodage utilisé pendant le transport, par exemple la compression avec zlib (RFC 1950).
— D'éventuelles informations de sécurité comme une empreinte cryptographique ou une signature numérique.

L'élément XML `<File>` qui contient tout cela est mis ensuite dans un élément `<FDT-Instance>` (FDT = "*File Delivery Table*") qui contient également les informations externes au fichier (comme la date limite pendant laquelle cette information sera valable). Voici l'exemple que donne le RFC, pour le fichier `track1.mp3` :

```
<?xml version="1.0" encoding="UTF-8"?>
<FDT-Instance xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
     xsi:schemaLocation="urn:ietf:params:xml:ns:fdt
                         ietf-flute-fdt.xsd"
     Expires="2890842807">
     <File
       Content-Location="http://www.example.com/tracks/track1.mp3"
       Content-Length="6100"
       Content-Type="audio/mp3"
       Content-Encoding="gzip"
       Content-MD5="+VP5IrWploFkZWc11iLDdA=="
       Some-Private-Extension-Tag="abc123"/>
</FDT-Instance>
```

Cet élément XML est transmis avec le fichier. Les informations liées à l'émission du fichier (adresse IP source qui émettra le fichier, début de l'émission, etc) sont récupérées par le destinataire par un moyen non spécifié ici (cf. section 6). Cela a pu être une description SDP (RFC 4566), une ressource sur le Web ou bien d'autres méthodes, même une configuration manuelle. Le point important est que, une fois toutes ces informations en sa possession, le récepteur FLUTE peut se préparer à recevoir le fichier, et peut vérifier sa bonne délivrance (l'émetteur, lui, n'en saura rien, puisque la liaison est unidirectionnelle).

L'envoi unidirectionnel de fichiers soulève des problèmes de sécurité particuliers (section 7, très détaillée). Si le fichier transmis est confidentiel, il doit être chiffré avant, que ce soit par un chiffrement global du fichier (par exemple avec PGP) ou en route par IPsec.

Et si l'attaquant veut modifier le fichier ? On peut utiliser une signature numérique du fichier, avant son envoi ou bien, pendant le trajet authentifier les paquets ALC avec le RFC 6584 ou bien TESLA ("*Timed Efficient Stream Loss-Tolerant Authentication*", RFC 4082 et RFC 5776) ou encore, là aussi, IPsec. À noter que certaines de ces techniques demandent une communication bidirectionnelle minimale, par exemple pour récupérer les clés publiques à utiliser.

Ce RFC est la version 2 de FLUTE, la version 1 était normalisée dans le RFC 3926, qui avait le statut expérimental. Avec ce nouveau document, FLUTE passe sur le chemin des normes IETF. La liste des changements importants figure en section 11. Notez tout de suite que FLUTE v2 est un protocole différent, incompatible avec FLUTE v1.

Je ne connais pas les implémentations de FLUTE mais il en existe plusieurs (pour la version 1) par exemple jFlute `<http://jflute.berlios.de/>`, l'ancien MCL v3 `<http://planete-bcast.inrialpes.fr/rubrique.php3?id_rubrique=1>` (plus maintenue en version libre mais Expway `<http://expway.com/>` en a une version commerciale fermée) ou MAD-FLUTE `<http://mad.cs.tut.fi/>` fait à l'université de Tampere.

Merci à Vincent Roca pour sa relecture attentive.