RFC 9154 : Extensible Provisioning Protocol (EPP) Secure Authorization Information for Transfer

Stéphane Bortzmeyer
<stephane+blog@bortzmeyer.org>

Première rédaction de cet article le 31 décembre 2021

Date de publication du RFC : Décembre 2021

https://www.bortzmeyer.org/9154.html

Le protocole EPP d'avitaillement des noms de domaine permet, entre autres opérations, de **transférer** un domaine d'un client (typiquement un BE) à un autre. Cette opération ouvre de sérieux problèmes de sécurité, le transfert pouvant être utilisé pour détourner un nom de domaine. En général, la sécurisation de ce transfert est faite par un mot de passe stocké en clair. Notre RFC décrit une méthode pour gérer ces mots de passe qui évite ce stockage, et qui gêne sérieusement les transferts malveillants.

EPP est normalisé dans le RFC 5730 [1] et c'est ce document qui décrit le cadre général d'autorisation d'un transfert de domaines. La forme exacte que prend l'autorisation dépend du type d'objets qu'on gère avec EPP. Pour les domaines (RFC 5731), c'est un élément `<authInfo>`, qui peut contenir divers types de sous-élements mais le plus fréquent est un simple mot de passe (parfois appelé « code de transfert » ou « code d'autorisation », ou simplement « authinfo »), comme dans cet exemple, une réponse à une commande EPP `<info>`, où le mot de passe est « 2fooBAR » :

```
<domain:infData>
     <domain:name>example.com</domain:name>
...
     <domain:crDate>1999-04-03T22:00:00.0Z</domain:crDate>
...
     <domain:authInfo>
       <domain:pw>2fooBAR</domain:pw>
     </domain:authInfo>
</domain:infData>
```

1. Pour voir le RFC de numéro NNN, `https://www.ietf.org/rfc/rfcNNN.txt`, par exemple `https://www.ietf.org/rfc/rfc5730.txt`

L'utilisation typique de ce mot de passe est que le client (en général un BE) le crée, le stocke en clair, l'envoie au serveur, qui le stocke. Lors d'un transfert légitime, le BE gagnant recevra ce mot de passe (typiquement via le titulaire de l'objet, ici un nom de domaine) et le transmettra au registre (RFC 5731, section 3.2.4). Ici, le BE gagnant demande à son client le code d'autorisation qu'il a normalement obtenu via le BE perdant (si le client est vraiment le titulaire légitime) : Et ici le BE perdant indique à son client le code d'autorisation : ("*registar*" = BE)

Notez que la façon d'autoriser les transferts, et d'accéder aux informations d'autorisation, dépend de la politique du registre. Les RFC sur EPP normalisent la technique mais pas la politique. Par exemple, lorsqu'un BE demande un transfert sans fournir d'information d'autorisation, certains registres refusent immédiatement le transfert, tandis que d'autres le mettent en attente d'une acceptation ou d'un refus explicite. De même, certains registres permettent de récupérer l'information d'autorisation, comme dans l'exemple ci-dessus, alors que d'autres (comme CentralNic) refusent.

Même chose pour d'autres types d'objets comme les contacts (RFC 5733) même si en pratique la pratique du transfert est plus rare pour ces types. Le mot de passe étant typiquement stocké en clair chez le client, pour pouvoir être donné en cas de transfert, on voit les risques que cela pose en cas d'accès à la base du BE. Aujourd'hui, stocker un mot de passe en clair est nettement considéré comme une mauvaise pratique de sécurité.

À la place, notre RFC décrit, non pas une modification du protocole EPP, mais une nouvelle procédure, une façon créative de se servir du protocole existant pour gérer ces informations d'autorisation de manière plus sérieuse : l'objet (par exemple le nom de domaine) est créé sans information d'autorisation, le serveur EPP (par exemple le registre de noms de domaine) doit refuser le transfert si cette information est manquante. Lors d'un transfert légitime, le client (par exemple un BE) perdant va générer un mot de passe, le transmettre au registre et à son client (typiquement le titulaire du nom de domaine) et **ne pas le stocker**. Le registre stockera le mot de passe uniquement sous forme condensée et, lorsqu'il recevra la demande de transfert accompagnée d'un mot de passe, il pourra condenser ce mot et vérifier qu'il correspond bien à celui stocké. Le mot ne servira qu'une fois et l'information d'autorisation est détruite après le succès du transfert. Tout ceci ne nécessite pas de modification du protocole, mais, dans certains cas, une modification des pratiques des différents acteurs (par exemple, le serveur EPP doit accepter qu'un objet soit créé sans information d'autorisation, et doit considérer que cela vaut refus de tout transfert).

Le RFC note que la norme EPP ne décrit, logiquement, que le protocole, c'est-à-dire l'interaction entre les deux machines, mais pas ce que fait chaque machine de son côté. Ainsi, la nécessité de stocker les mots de passe de manière sécurisée n'est pas imposée par EPP (mais est néanmoins une bonne pratique).

D'autre part, EPP ne prévoit pas explicitement de durée de vie pour les mots de passe (mais n'interdit pas non plus de les supprimer au bout d'un temps donné, ce qui va être justement la technique de notre RFC).

Petite révision sur les acteurs de l'avitaillement de noms de domaine en section 2 du RFC. La norme EPP parle de client et de serveur, notions techniques, mais du point de vue "*business*", il y a trois acteurs (cf. la terminologie dans le RFC 8499), le registre (qui gère le serveur EPP), le bureau d'enregistrement (BE, qui gère le client EPP) et le titulaire (qui se connecte à son BE via une interface Web ou une API). Dans beaucoup de domaines d'enregistrement, il n'y a aucun lien direct entre le titulaire et le registre, tout devant passer par le BE.

Maintenant, place à la description de la nouvelle manière de faire des transferts. Bien qu'elle ne change pas le protocole, qu'elle ne soit qu'une nouvelle façon d'utiliser ce qui existe déjà dans EPP, elle doit se signaler lors de la connexion EPP, avec l'espace de noms `urn:ietf:params:xml:ns:epp:secure-authin` (enregistré à l'IANA <`https://www.iana.org/assignments/xml-registry/xml-registry.xml#ns`>). En effet, la nouvelle manière a besoin que le serveur accepte des choses qui sont autorisées par EPP mais pas obligatoires, notamment :

— une information d'autorisation vide (représentée par exemple par le `NULL` dans une base SQL),
— la possibilité de supprimer l'information d'autorisation avec la commande EPP `<update>`,
— la possibilité de valider l'information d'autorisation avec la commande EPP `<info>`,
— le refus de tout transfert si l'information d'autorisation est vide,
— la remise à zéro de l'information d'autorisation lorsque le transfert a été réalisé (mot de passe à usage unique).

Le serveur, lui, en recevant `urn:ietf:params:xml:ns:epp:secure-authinfo-transfer-1.0`, peut compter que le client EPP saura :

— générer une information d'autorisation forte (aléatoire, par exemple),
— ne le faire que lorsqu'un transfert est demandé.

L'autorisation d'information dans l'élement XML `<domain:pw>` (RFC 5731, section 3.2.4) est un mot de passe qui doit être difficile à deviner par un attaquant. Idéalement, il doit être aléatoire ou équivalent (RFC 4086). Le RFC calcule que pour avoir 128 bits d'entropie, avec uniquement les caractères ASCII imprimables, il faut environ 20 caractères.

Pour compenser l'absence de la notion de durée de vie de l'information d'autorisation dans EPP, le client ne doit définir une information d'autorisation que lorsqu'un transfert est demandé, et supprimer cette information ensuite. La plupart du temps, le domaine n'aura pas d'information d'autorisation, et les transferts seront donc refusés.

L'information d'autorisation, comme tout mot de passe, ne doit plus être stockée en clair, mais sous forme d'un condensat. Le BE perdant ne doit pas la stocker (il la génère, la passe au titulaire et l'oublie ensuite). Le BE gagnant ne doit la stocker que le temps de finaliser le transfert. Évidemment, toute la communication EPP doit être chiffrée (RFC 5734). Lors d'une demande de transfert, le registre va vérifier qu'un condensat de l'information d'autorisation transmise par le BE gagnant correspond à ce que le BE perdant avait envoyé. L'information vide est un cas particulier, le registre ne doit pas tester l'égalité mais rejeter le transfert.

La section 4 explique en détail le processus de transfert avec cette nouvelle méthode :

— Quand le domaine est créé, l'information d'autorisation est vide (pas de « "*authinfo*" »),
— quand le titulaire veut transférer le nom à un nouveau BE, il demande au BE perdant l'information d'autorisation,
— le BE perdant génère un mot de passe, qu'il transmet au titulaire et au registre (qui peut répondre avec le code d'erreur EPP 2202 si ce mot de passe ne lui semble pas assez fort), puis qu'il oublie aussitôt,
— le titulaire donne le mot de passe au BE gagnant,
— le BE gagnant demande le transfert au registre, en fournissant le mot de passe, ce qui permet au transfert d'être accepté immédiatement,
— le registre (ou bien le BE perdant), efface le mot de passe.

Voici en EPP quelques messages pour réaliser ces différentes opérations. D'abord, la création d'un nom (notez le mot de passe vide) :

```
<create>
  <domain:create
    xmlns:domain="urn:ietf:params:xml:ns:domain-1.0">
    <domain:name>example.test</domain:name>
    <domain:authInfo>
      <domain:pw/>
    </domain:authInfo>
  </domain:create>
</create>
```

Ici, la mise à jour de l'information d'autorisation par le BE perdant, lorsque le titulaire lui a annoncé le départ du domaine ; le mot de passe est `LuQ7Bu@w9?%+_HK3cayg$55$LSft3MPP` (le RFC rappelle fortement l'importance de générer un mot de passe fort, par exemple en utilisant des sources bien aléatoires, comme documenté dans le RFC 4086) :

```
<update>
  <domain:update
    xmlns:domain="urn:ietf:params:xml:ns:domain-1.0">
    <domain:name>example.test</domain:name>
    <domain:chg>
      <domain:authInfo>
        <domain:pw>LuQ7Bu@w9?%+_HK3cayg$55$LSft3MPP</domain:pw>
      </domain:authInfo>
    </domain:chg>
  </domain:update>
</update>
```

Le BE perdant devra peut-être également supprimer l'état `clientTransferProhibited`, si le domaine était protégé contre les transferts.

Le BE gagnant peut également vérifier l'information d'autorisation sans déclencher un transfert, avec une requête `<info>`, qui lui renverra l'information d'autorisation. Pour plusieurs exemples par la suite, j'ai utilisé le logiciel Cocca `<https://cocca.org.nz/srs/>`. Cocca, par défaut, ne stocke pas l'autorisation d'information en clair et ne peut donc pas la renvoyer. Ou bien le client EPP peut envoyer une commande `<info>` en indiquant l'information d'autorisation. S'il obtient une erreur EPP 2202 (RFC 5730, section 3), c'est que cette information n'était pas correcte. Ici, la réponse EPP de Cocca lorsqu'on lui envoie un `<info>` avec information d'autorisation correcte :

```
Client : <info xmlns="urn:ietf:params:xml:ns:epp-1.0"><info xmlns="urn:ietf:params:xml:ns:domain-1.0"><name>

Serveur : ... <ns1:authInfo><ns1:pw>Authinfo Correct</ns1:pw></ns1:authInfo> ...
```

Et si cette information est incorrecte :

```
Serveur : ... <ns1:authInfo><ns1:pw>Authinfo Incorrect</ns1:pw></ns1:authInfo>
```

(Mais Cocca répond quand même avec un code EPP 1000, ce qui n'est pas correct.)

Et enfin, bien sûr, voici la demande de transfert elle-même :

```
<transfer op="request">
  <domain:transfer
    xmlns:domain="urn:ietf:params:xml:ns:domain-1.0">
    <domain:name>example1.com</domain:name>
    <domain:authInfo>
      <domain:pw>LuQ7Bu@w9?%+_HK3cayg$55$LSft3MPP</domain:pw>
    </domain:authInfo>
  </domain:transfer>
</transfer>
```

Et si c'est bon :

```
<ns0:response xmlns:ns0="urn:ietf:params:xml:ns:epp-1.0" xmlns:ns1="urn:ietf:params:xml:ns:domain-1.0" xmlns:xsi
<ns1:trStatus>serverApproved</ns1:trStatus> ...
```

Et avec une mauvaise information d'autorisation :

```
<ns0:response xmlns:ns0="urn:ietf:params:xml:ns:epp-1.0"><ns0:result code="2202"><ns0:msg>Invalid authorization
```

La section 6 du RFC décrit le problème de la transition depuis l'ancien modèle d'autorisation vers le nouveau. Notez que certains registres peuvent avoir une partie du nouveau système déjà en place. Le registre qui désire transitionner doit d'abord s'assurer que l'information d'autorisation absente ou vide équivaut à un rejet. Il doit ensuite permettre aux BE de mettre une information d'autorisation vide, permettre que la commande `<info>` puisse tester une information d'autorisation, s'assurer que l'acceptation d'un transfert supprime l'information d'autorisation, etc.

L'extension à EPP décrite dans ce RFC a été enregistrée dans le registre des extensions EPP `<https://www.iana.org/assignments/epp-extensions/epp-extensions.xml#epp-extensions-1>`. Quelles sont les mises en œuvre de ce RFC ? Cocca `<https://cocca.org.nz/srs/>`, déjà cité, le fait partiellement (par exemple en ne stockant pas les mots de passe en clair). Je n'ai pas testé avec ce logiciel ce qui se passait avec une information d'autorisation vide. Sinon, CentralNic a déjà ce mécanisme en production. Et Verisign l'a mis dans son SDK `<https://www.verisign.com/en_US/channel-resources/domain-registry-products/epp-sdks/index.xhtml>`.