RFC 5730 : Extensible Provisioning Protocol (EPP)

Stéphane Bortzmeyer
<stephane+blog@bortzmeyer.org>

Première rédaction de cet article le 23 novembre 2009

Date de publication du RFC : Août 2009

https://www.bortzmeyer.org/5730.html

Les registres, par exemple les registres de noms de domaine fonctionnent parfois sur un modèle "*registry/registrar*" c'est-à-dire où le client final doit passer par un intermédiaire, le bureau d'enregistrement ("*registrar*") pour enregistrer son nom de domaine. Le "*registrar*" souhaite en général avoir un protocole de communication avec le registre afin d'automatiser ses opérations, dans son langage de programmation favori. EPP, décrit dans ce RFC, est un de ces protocoles d'**avitaillement** ("*provisioning*", et merci à Olivier Perret pour la traduction). (EPP n'emploie pas le terme de "*registrar*" mais celui de "*sponsoring client*", plus neutre. EPP peut donc en théorie être utilisé en cas de vente directe.)

Notre RFC remplace le second RFC sur EPP, le RFC 4930 [1], mais les changements sont mineurs (cf. annexe C, le principal étant que le RFC rend plus clair le fait que la liste des codes de retour est limitative : un registre ne peut pas créer les siens, ce qui est une des plus grosses limitations de EPP). EPP est désormais une norme complète ("*Full Standard*").

EPP a été réalisé sur la base du cahier des charges dans le RFC 3375. Au lieu de s'appuyer sur les protocoles classiques de communication comme XML-RPC ou SOAP, ou même sur l'architecture REST, EPP crée un protocole tout nouveau, consistant en l'établissement d'une connexion (authentifiée) puis sur l'échange d'éléments XML, spécifiés dans le langage W3C Schemas.

Par exemple, l'ouverture de la connexion se fait en envoyant l'élément XML `<login>` (section 2.9.1.1) :

1. Pour voir le RFC de numéro NNN, `https://www.ietf.org/rfc/rfcNNN.txt`, par exemple `https://www.ietf.org/rfc/rfc4930.txt`

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<epp xmlns="urn:ietf:params:xml:ns:epp-1.0"
     xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
     xsi:schemaLocation="urn:ietf:params:xml:ns:epp-1.0
     epp-1.0.xsd">
  <command>
    <login>
      <clID>ClientX</clID>
      <pw>foo-BAR2</pw>
      <newPW>bar-FOO2</newPW>
      <options>
        <version>1.0</version>
        <lang>fr-CA</lang>
      </options>
      <svcs>
        <objURI>urn:ietf:params:xml:ns:obj1</objURI>
        <objURI>urn:ietf:params:xml:ns:obj2</objURI>
        <objURI>urn:ietf:params:xml:ns:obj3</objURI>
      </svcs>
    </login>
    <clTRID>ABC-12345</clTRID>
  </command>
</epp>
```

Les espaces de noms utilisés, comme `urn:ietf:params:xml:ns:epp-1.0` sont enregistrés dans le registre IANA `<https://www.iana.org/assignments/xml-registry/ns.html>` (cf. section 6). Le `<clTRID>` est un identifiant de transaction (celui-ci est choisi par le client) et il facilite la traçabilité des opérations EPP. L'exécution de ces éléments doit être atomique (section 2 du RFC).

Chaque commande EPP entraîne une réponse (section 2.6) qui inclus notamment un code numérique à quatre chiffres (section 3) résumant si la commande a été un succès ou un échec (reprenant le schéma expliqué dans la section 4.2.1 du RFC 5321). Le premier chiffre indique le succès (1) ou l'échec (2) et le second la catégorie (syntaxe, sécurité, etc). Par exemple 1000 est un succès complet, 1001 un succès mais où l'action n'a pas encore été complètement effectuée (par exemple parce qu'une validation asynchrone est nécessaire), 2003, une erreur syntaxique due à l'absence d'un paramètre obligatoire, 2104 une erreur liée à la facturation (crédit expiré, par exemple), 2302, l'objet qu'on essaie de créer existe déjà, etc.

EPP est donc typiquement un protocole synchrone. Toutefois, il dispose également d'un mécanisme de notification asynchrone. La commande EPP `<poll>` (section 2.9.2.3) permet d'accéder à des messages qui avaient été mis en attente pour un client, et lui donnaient l'état des traitements asynchrones.

Le RFC 5730 spécifie une dizaine de commandes EPP comme `<check>`, qui permet de s'assurer qu'un objet peut être créé, `<info>`, qui permet d'accéder aux informations disponibles à propos d'un objet existant (comme whois sauf que whois est typiquement public et EPP typiquement réservé aux bureaux d'enregistrement), `<create>` pour créer de nouveaux objets (par exemple réserver un nom de domaine), etc.

Comme tout objet géré par EPP a un « "*sponsoring client*" », l'entité qui le gère (dans le cas d'un registre de noms de domaine, il s'agit typiquement du bureau d'enregistrement), EPP fournit une commande pour changer ce client, `<transfer>` (section 2.9.3.4). Des options de cette commande permettent d'initier le transfert, d'annuler une demande en cours, de l'approuver explicitement (pour le client sortant), etc.

La syntaxe formelle des éléments XML possibles est décrite dans la section 4, en utilisant le langage W3C schema.

Le transport de ces éléments XML sur le réseau peut se faire avec différents protocoles, s'ils respectent les caractéristiques listées en section 2.1. Par exemple, le RFC 5734 normalise EPP reposant directement sur TCP.

Un des points délicats de la conception d'un tel protocole est que chaque registre a sa propre politique d'enregistrement et ses propres règles. Quoi que dise ou fasse l'ICANN, cette variété va persister. Par exemple, l'expiration automatique d'un domaine existe dans `.com` mais pas dans `.eu` ou `.fr`. Le protocole EPP ne prévoit donc pas d'éléments pour gérer telle ou telle catégorie d'objets (les domaines mais aussi les serveurs de noms ou les contacts). Il doit être complété par des "*mappings*", des schémas définissant une classe d'objets. Certains de ces "*mappings*" sont spécifiés pas l'IETF comme le "*domain mapping*" (gestion des domaines) dans le RFC 5731. L'annexe A fournit un exemple de "*mapping*". Plusieurs registres utilisent des "*mappings*" non-standard, pour s'adapter à leurs propres règles d'enregistrement, ce qui limite la portabilité des programmes EPP (le cadre d'extension de EPP fait l'objet de la section 2.7). C'est ce qu'ont fait les français <http://www.afnic.fr/doc/interface/epp_fr>, les brésiliens <http://registro.br/epp/rfc-EN.html> ou les polonais <http://www.dns.pl/porozumienie/draft-zygmuntowicz-epp-pltld-02.txt>.

Dans tous les cas, chaque objet manipulé avec EPP reçoit un identificateur unique (par exemple `SB68-EXAMPLEREP`), dont les caractéristiques sont normalisées dans la section 2.8. Les dépôts possibles sont enregistrés dans un registre IANA <https://www.iana.org/assignments/epp-repository-ids>.

Complexe, EPP n'a pas été reçu avec enthousiasme chez les registres existants, sauf ceux qui refaisaient complètement leur logiciel comme `.be`. On notera que certains gros TLD comme `.de` n'utilisent pas EPP (Denic utilise son protocole MRI/RRI). Il existe d'autres protocoles d'avitaillement comme :

— Le registre Adams' Names a son protocole <http://www.adamsnames.tc/api/xmlrpc.html> au dessus de XML-RPC,
— Le "*registrar*" Gandi utilise son protocole <https://api.ote.gandi.net/>, au dessus de XML-RPC pour ses revendeurs,
— Le "*registrar*" BookMyName utilise son protocole <http://api.doc.free.org/>, au dessus de SOAP pour ses revendeurs,
— OVH a aussi une API SOAP <http://www.ovh.com/soapi/fr/?group=domains> très riche,

et beaucoup d'autres qui ne semblent pas forcément documentés publiquement.

Il existe plusieurs mises en œuvres d'EPP en logiciel libre par exemple le serveur EPP OpenReg <https://www.isc.org/software/openreg> de l'ISC, le logiciel Fred <http://fred.nic.cz> du registre de `.cz` ou bien le client EPP Net : :DRI <http://search.cpan.org/~pmevzek/Net-DRI-0.95/> de Patrick Mevzek. Si vous voulez expérimenter avec EPP, vous devez installer votre propre serveur, il n'existe pas de serveur public pour tester.