RFC 3403 : Dynamic Delegation Discovery System (DDDS) Part Three: The Domain Name System (DNS) Database

Stéphane Bortzmeyer
<stephane+blog@bortzmeyer.org>

Première rédaction de cet article le 2 février 2006

Date de publication du RFC : Octobre 2002

https://www.bortzmeyer.org/3403.html

Le système DDDS, décrit dans le RFC 3401 [1] permet d'associer à un nom de domaine l'adresse d'une ressource. Le RFC 3401 en donne une vision très générale, le RFC 3402 spécifie l'algorithme et notre RFC va préciser tout le reste pour le cas d'une base de données particulière, le DNS. C'est notamment dans notre RFC que se trouve décrits les enregistrements NAPTR.

Le RFC 3401 n'impose pas une base de données particulière pour DDDS. Mais toutes les applications, aujourd'hui, se servent du DNS, selon les règles édictées dans notre RFC (qui remplace donc le RFC 2915).

Le RFC 3403 normalise donc les enregistrements NAPTR du DNS. Ces enregistrements ressemblent à ceci :

```
cid.urn.arpa.  IN NAPTR 100   10   ""  ""  "!^urn:cid:.+@([^\.]+\.)(.*)$!\2!i"  .
```

où `cid.urn.arpa` est le nom de domaine, 100 l'ordre (au cas où il y aie plusieurs règles), 10 la préférence (nommé priorité dans le RFC 3402), les deux chaînes vides sont les options ("*flags*") et le service (le protocole utilisé), les deux derniers termes étant l'expression et le remplacement (ici vide).

Suivant l'exemple (**hypothétique**) de notre RFC, inspiré du RFC 2276 sur les URN : on veut construire un service qui va permettre de trouver un message en connaissant son Content-ID (un en-tête MIME normalisé dans le RFC 1873). Prenons `urn:cid:199606121851.1@bar.example.org` comme exemple d'URN. La règle bien connue (elle doit être spécifiée dans la définition de l'application DDDS) dit que la première clé est comprise entre le premier et le deuxième signe **:**, donc ici, "cid" et on y ajoute `urn.arpa`.

La première règle trouvé dans le DNS pour ce nom est :

1. Pour voir le RFC de numéro NNN, `https://www.ietf.org/rfc/rfcNNN.txt`, par exemple `https://www.ietf.org/rfc/rfc3401.txt`

```
cid.urn.arpa.  IN NAPTR 100   10   ""    ""  "!^urn:cid:.+@([^\.]+\.)(.*)$!\2!i"    .
```

L'expression rationnelle extrait juste le nom de domaine (en retirant le premier label) : `\2` désigne le second groupe de parenthèses, ici `example.org`. Comme il n'y a pas d'options ("*flags*" est une chaîne vide), la règle n'est pas terminale et on recherche donc un nouvel enregistrement NAPTR en utilisant `example.org` comme clé. Supposons que cela donne :

```
example.org. IN NAPTR 100  50  "s"    "http"  ""   www.example.org.
```

On saurait alors qu'il faut utiliser le protocole HTTP pour demander à `www.example.org` ce message.

Aujourd'hui, les NAPTR sont surtout utilisés dans le contexte d'ENUM, par exemple `1.9.7.9.0.7.8.6.8.0.3.` est un nom de domaine qui a deux enregistrements NAPTR nous donnant le numéro de téléphone de son titulaire (en Allemagne) et son adresse SIP. Mais on voit aussi des NAPTR dans les domaines `http.uri.arpa` et `ftp.uri.arpa` pour résoudre les URI (ici, avec l'outil dig) :

```
% dig NAPTR http.uri.arpa
http.uri.arpa.          20969   IN      NAPTR   0 0 "" "" "!^http://([^:/?#]*).*$!\\1!i" .
```

où ils extraient simplement le nom de machine de l'URI.

Je ne connais pas de mise en œuvre de NAPTR en logiciel libre, malheureusement (le module Perl `Net::DNS::RR::NAPTR` permet leur analyse mais ne met pas en œuvre l'algorithme ; même chose pour la classe NAPTR de DNS Python <`https://www.bortzmeyer.org/dnspython.html`>).

Une bonne synthèse expliquant les NAPTR <`http://blog.nominet.org.uk/tech/2006/07/14/naptr-records/`> figure dans le blog de Nominet.