RFC 8801 : Discovering Provisioning Domain Names and Data

Stéphane Bortzmeyer
<stephane+blog@bortzmeyer.org>

Première rédaction de cet article le 29 juillet 2020

Date de publication du RFC : Juillet 2020

https://www.bortzmeyer.org/8801.html

Vous vous demandez certainement, chère lectrice et cher lecteur, ce qu'est un PvD ("*ProVisioning Domain*"). Cette notion n'est pas nouvelle, mais elle n'avait été formalisée rigoureusement qu'avec le RFC 7556 [1]. Un PvD est un domaine de l'Internet où des règles cohérentes de configuration du réseau s'appliquent. Au sein, d'un PvD, il y a un résolveur DNS à utiliser, des adresses IP spécifiques, des règles de routage et de sécurité que n'ont pas forcément les autres PvD. Ce nouveau RFC décrit une option IPv6 qui permet au PvD de signaler explicitement son existence, via un RA ("*Router Advertisement*".)

La notion de PvD ("*ProVisioning Domain*") est utile pour les machines qui ont plusieurs connexions, à des réseaux (et donc peut-être des PvD) différents, par exemple un ordinateur connecté à la fois à un FAI grand public et au VPN de l'entreprise. Il y a des PvD **implicites** : une machine qui connait le concept de PvD affecte un PvD implicite à chaque réseau et, par défaut, ne considère pas que le résolveur DNS de l'un convient pour les autres. Et il y a des PvD **explicites**, et c'est à eux qu'est consacré ce RFC. Un PvD explicite a un **identificateur**, qui a la syntaxe d'un FQDN. Attention, même si l'identificateur est un nom de domaine, le terme de « domaine » (le D de PvD) ne doit pas être confondu avec le sens qu'il a dans le DNS. L'intérêt d'un PvD explicite est qu'il permet d'avoir plusieurs domaines distincts sur la même interface réseau, et qu'on peut y associer des informations supplémentaires, récupérées sur le réseau ou bien préconfigurées.

On l'a vu, le concept de PvD avait été décrit dans le RFC 7556. Ce que notre nouveau RFC 8801 rajoute, c'est :

— Une option aux RA ("*Router Advertisement*", RFC 4861) pour indiquer aux machines IPv6 du réseau le PvD explicite,

1. Pour voir le RFC de numéro NNN, https://www.ietf.org/rfc/rfcNNN.txt, par exemple https://www.ietf.org/rfc/rfc7556.txt

— Un mécanisme pour récupérer, en HTTPS, de l'information de configuration du réseau, pour les cas où cette information ne tiendrait pas dans le message RA. Elle est évidemment codée en JSON.

Le RA va donc indiquer le ou les identificateurs du ou des PvD du réseau. Notez qu'on peut avoir aussi le même PvD sur plusieurs réseaux, par exemple le Wi-Fi et le filaire chez moi font partie du même PvD et ont les mêmes règles et la même configuration.

Dans un monde idéal, l'utilisatrice dont l'ordiphone a à la fois une connexion 4G, une connexion Wi-Fi et un VPN par dessus, aurait un PvD pour chacun de ces trois réseaux différentes, et pourrait donc décider intelligemment quelle interface utiliser pour quel type de trafic. Par exemple, une requête DNS pour `wiki.private.example.com` (le wiki interne de l'entreprise) sera envoyée uniquement via le VPN, puisque les résolveurs des réseaux publics ne connaissent pas ce domaine. Autre exemple, l'information envoyée pour chaque PvD pourrait permettre aux applications de faire passer le trafic important (la vidéo en haute définition, par exemple), uniquement sur le réseau à tarification forfaitaire.

Bon, assez de considérations sur les PvD, passons aux nouvelles normes décrites dans ce RFC. D'abord, l'option RA. Elle est décrite dans la section 3. C'est très simple : une nouvelle option RA est créée, la 21, « PvD ID Router Advertisement Option » <`https://www.iana.org/assignments/icmpv6-parameters/icmpv6-parameters.xml#icmpv6-parameters-5`>, suivant le format classique des options RA, TLV (Type-Longueur-Valeur). Elle contient, entre autres :

— Un bit nommé H qui indique que le client peut récupérer des informations supplémentaires en HTTP (c'est détaillé en section 4 du RFC),
— Un bit nommé L qui indique que ce PvD est également valable pour IPv4,
— Un bit nommé R qui indique que le RA contient un autre RA (détails en section 5),
— Et bien sûr l'identificateur ("*PvD ID*") lui-même, sous la forme d'un FQDN.

Si jamais il y a plusieurs PvD sur le réseau, le routeur doit envoyer des RA différents, il n'est pas autorisé à mettre plusieurs options PvD dans un même RA.

Que va faire la machine qui reçoit une telle option dans un RA ? Elle doit marquer toutes les informations de configuration qui étaient dans le RA comme étant spécifiques à ce PvD. Par exemple, si la machine a deux interfaces, et reçoit un RA sur chacune, ayant des PvD différents, et donnant chacun l'adresses d'un résolveur DNS (RFC 8106), alors ce résolveur ne doit être utilisé que pour le réseau où il a été annoncé (et avec une adresse IP source de ce réseau). Rappelez-vous que différents résolveurs DNS peuvent donner des résultats différents, par exemple un des réseaux peut avoir un Pi-hole qui donne des réponses mensongères, afin de bloquer la publicité. (En parlant du DNS, des détails sur cette question complexe figurent dans la section 5.2.1 du RFC 7556.) Autre exemple, si on utilise un VPN vers son employeur, les requêtes DNS pour les ressources internes (`congés.rh.example.com`...) doivent être envoyées au résolveur de l'employeur, un résolveur extérieur peut ne pas être capable de les résoudre. En fait, tous les cas où la réponse DNS n'est pas valable publiquement nécessitent d'utiliser le bon résolveur DNS. C'est aussi ce qui se passe avec NAT64 (RFC 6147).

Idem pour d'autres informations de configuration comme le routeur par défaut : elles doivent être associées au PvD. Une machine qui connait les PvD ne peut donc pas se contenter d'une table simple listant le résolveur DNS, le routeur par défaut, etc. Elle doit avoir une table par PvD.

Si la machine récupère par ailleurs des informations de configuration avec DHCP, elle doit également les associer à un PvD, par exemple en utilisant comme PvD implicite celui de l'interface réseau utilisée.

Et pour IPv4, si le bit L est à un ? Dans ce cas, l'information IPv4 reçue sur la même interface doit être associée à ce PvD. (Notez que si plusieurs PvD sur la même interface ont le bit L, on ne peut rien décider, à part que quelqu'un s'est trompé.)

Les machines qui partagent leur connexion ("*tethering*"), ce qui est fréquent pour les connexions mobiles comme la 4G (RFC 7278), doivent relayer le RA contenant l'option PvD vers le réseau avec qui elles partagent (comme elles relaient d'autres messages similaires, cf. RFC 4389). Cela permettra à des techniques qui facilitent le partage, comme la délégation de préfixe du RFC 8415, de bien fonctionner. Même si la machine partageuse ne reçoit pas d'option PvD, il est recommandé qu'elle en ajoute une pour le bénéfice des engins avec qui elle partage sa connectivité.

Tout cela, c'était pour une machine "*PvD-aware*", une machine qui sait ce qu'est un PvD et qui sait le gérer. Mais avec du vieux logiciel, écrit avant le concept de PvD, que se passe-t-il ? Une machine munie de ce logiciel va évidemment ignorer l'option PvD du RA et donc utiliser un seul résolveur DNS, un seul routeur par défaut pour tous les cas. Elle aura sans doute beaucoup de problèmes si plusieurs interfaces réseau sont actives et, en pratique, il vaudra mieux n'avoir qu'une seule interface à un moment donné.

Nous avons vu plus haut que, si le bit H dans l'option PvD est à 1, la machine peut obtenir davantage d'informations ("*PvD Additional Information*") en HTTP. Cette information sera encodée en I-JSON (RFC 7493.) Pourquoi ne pas avoir mis ces informations dans le RA ? Parce qu'elles sont trop grosses ou, tout simplement, pas critiques, et pouvant être ignorées. Et on les récupère où, ces informations ? Si le PvD est `radio.example.com`, l'URL à utiliser est `https://radio.example.com/.well-known/pvd` (préfixe `.well-known` désormais enregistré <`https://www.iana.org/assignments/well-known-uris/well-known-uris.xml#well-known-uris-1`>.) Attention, on ne doit tenter d'accéder à cet URL que si le bit H était à 1. C'est du HTTPS (RFC 2818) et il faut donc vérifier le certificat (cf. RFC 6125.) Vous voyez qu'on ne peut pas choisir un PvD au hasard : il faut qu'il soit un nom de domaine qu'on contrôle, et pour lequel on peut obtenir un certificat. Si on configure un des ces affreux portails captifs, il faut s'assurer que les clients auront le droit de se connecter au serveur HTTP indiqué.

Les données auront le type `application/pvd+json`. Elles contiennent obligatoirement :

— Le PvD,
— une date d'expiration, au format RFC 3339,
— les préfixes IP du PvD.

Ainsi que, si on veut :

— les domaines où chercher les noms qui ne sont pas des FQDN,
— une indication que ce PvD n'a pas de connexion à l'Internet.

D'autres membres peuvent être présents, un registre IANA <`https://www.iana.org/assignments/pvds/pvds.xml#additional-information-pvd-keys`> existe et, pour y ajouter des termes, c'est la procédure « Examen par un expert » du RFC 8126.

Voici un exemple, avec le PvD `smart.mpvd.io` (qui a servi à un hackathon IETF) :

```
% curl  https://smart.mpvd.io/.well-known/pvd
{
"name": "PvD for smart people",
"prefixes": ["2001:67c:1230:101::/64", "2001:67c:1230:111::/64"],
"noInternet": false,
"metered": false,
"captivePortalURL" : "https://smart.mpvd.io/captive.php"
}
```

On notera que le membre `expires` manque, ce qui n'est pas bien, et que le membre `identifier`, qui devrait indiquer le PvD, est également absent, ce qui est encore plus mal. On voit qu'il s'agissait d'une expérimentation. Voici ce qu'aurait dû être l'objet JSON :

```
{
        "identifier": "smart.mpvd.io",
        "expires": "2020-04-08T15:30:00Z",
"prefixes": ["2001:67c:1230:101::/64", "2001:67c:1230:111::/64"],
"noInternet": false,
"name": "PvD for smart people",
"metered": false,
"captivePortalURL" : "https://smart.mpvd.io/captive.php"
}
```

Si vous connaissez d'autres serveurs d'information PvD, indiquez-les moi. Mais il est probable que la plupart ne seront accessibles que depuis un réseau interne.

Voilà, vous savez l'essentiel désormais. La section 5 du RFC ajoute quelques considérations pratiques. D'abord, le bit R qui permettait à une option PvD d'inclure un RA. À quoi ça sert ? L'idée est que, pendant longtemps, la plupart des réseaux accueilleront un mélange de machines qui connaissent les PvD, et de machines qui ne sont pas conscientes de ce concept. Le bit R sert à envoyer des informations qui ne seront comprises que des premières. Par exemple, on met un préfixe dans le RA et, dans l'option PvD, un autre RA qui contient un autre préfixe. Les machines qui connaissent les PvD verront et utiliseront les deux préfixes, celles qui ne connaissent pas les PvD ignoreront l'option et n'auront que le premier préfixe. On peut également envoyer deux RA, un ne contenant pas d'option PvD, et un en contenant une (plus le « vrai » RA).

Un peu de sécurité, pour finir (section 6). D'abord, il faut bien se rappeler que les RA, envoyés en clair et non authentifiés, n'offrent aucune sécurité. Il est trivial, pour un méchant connecté au réseau, d'envoyer de faux RA et donc de fausses options PvD. En théorie, des protocoles comme IPsec ou SEND (RFC 3971) peuvent sécuriser les RA mais, en pratique, on ne peut guère compter dessus. Les seules solutions de sécurisation effectivement déployées sont au niveau 2, par exemple 802.1X combiné avec le RFC 6105. Ce n'est pas un problème spécifique à l'option PvD, c'est la sécurité (ou plutôt l'insécurité) des RA.

Ah, et un petit mot sur la vie privée (section 7 du RFC). La connexion au serveur HTTP va évidemment laisser des traces, et il est donc recommandé de la faire depuis une adresse source temporaire (RFC 8981), et sans envoyer `User-Agent:` ou "*cookies*".

Question mise en œuvre, il y a eu du code libre pour des tests <`https://github.com/IPv6-mPvD`> (incluant une modification du noyau Linux, et un dissecteur pour Wireshark) mais je ne sais pas ce que c'est devenu et où en sont les systèmes d'exploitation actuels.