Représentation sous forme texte de ce qui passe sur le réseau

Stéphane Bortzmeyer
<stephane+blog@bortzmeyer.org>

Première rédaction de cet article le 25 août 2008. Dernière mise à jour le 18 mai 2020

https://www.bortzmeyer.org/representation-texte.html

Dans beaucoup de protocoles réseau, ce qui passe « sur le câble » est dans un format binaire incompréhensible pour un humain. Il est souvent utile de pouvoir le représenter sous forme texte, par exemple lorsque des programmes comme tcpdump ou Wireshark le décodent. C'est encore mieux si ce format texte est normalisé, cela permet la communication facile entre humains. Regardons quelques exemples surtout empruntés à la famille TCP/IP.

Logiquement, on va commencer par les adresses IP. Il semble que la représentation texte des adresses IPv4 soit normalisée, non ? Quatre nombres séparés par des points comme `192.0.2.25`, non ? Mais c'est plus compliqué que cela. Ni le RFC 791 [1], qui normalise IPv4, ni les RFCs suivants ne prévoient une forme textuelle standard. La représentation ci-dessus n'est qu'une habitude. Mais les bibliothèques qui convertissent depuis cette forme texte vers le binaire acceptent d'autres syntaxes. Voyons des exemples avec le programme telnet :

```
% telnet 3221226009
Trying 192.0.2.25...
...
% telnet 0xc0000219
Trying 192.0.2.25...
...
```

Bref, il existe bien des façons de représenter une adresse IPv4. Dans certains contextes seulement, une forme est préférée (par exemple RFC 4001 pour les MIB). Notons toutefois que, contrairement à ce que prétendent certains pour justifier leur conservatisme, par exemple le refus de noms de domaine entièrement en chiffres, il n'y a pas de risque de confusion entre noms et adresses. Le RFC 1123 est très clair dans sa section 2.1 : « *"The host SHOULD check the string syntactically for a dotted-decimal number before*

1. Pour voir le RFC de numéro NNN, `https://www.ietf.org/rfc/rfcNNN.txt`, par exemple `https://www.ietf.org/rfc/rfc791.txt`

looking it up in the Domain Name System." Donc, `192.0.2.25` n'est pas ambigu, même si le domaine de tête `.25` était créé.

La documentation peut être trompeuse. Sur une machine Ubuntu, le `manueldegetaddrinfo()` dit que l'adresse doit être exprimée sous forme "*dotted-decimal format for IPv4*" alors qu'en fait d'autres formes sont acceptés. Tout programme qui utilise `getaddrinfo()` (la grande majorité) acceptera alors, peut-être sans s'en rendre compte, ces autres formats, comme le fait telnet dans l'exemple ci-dessus. Par exemple, avec wget :

```
% wget http://0xc0000219/
--19:12:10--  http://0xc0000219/
          => 'index.html'
Resolving 0xc0000219... 192.0.2.25
Connecting to 0xc0000219|192.0.2.25|:80 connected.
HTTP request sent, awaiting response... 200 OK
...
```

Plus drôle, une représentation en Unicode, sans caractères ASCII, peut être acceptée. Ainsi avec l'adresse `[Caractère Unicode non montré [2]][Caractère Unicode non montré][Caractère Unicode non montré][Caractère Unicode non montré][Caractère Unicode non montré][Caractère Unicode non montré][Caractère Unicode non montré][Caractère Unicode non montré][Caractère Unicode non montré][Caractère Unicode non montré][Caractère Unicode non montré][Caractère Unicode non montré][Caractère Unicode non montré][Caractère Unicode non montré][Caractère Unicode non montré]` (regardez bien : ce ne sont pas les chiffres ASCII habituels, mais les chiffres dits « mathématiques en gras », comme [Caractère Unicode non montré ] alias U+[Caractère Unicode non montré ]1D7D3 MATHEMATICAL BOLD DIGIT FIVE `<https://r12a.github.io/uniview/?char=1D7D3>`). Essayez `ping [Caractère Unicode non montré][Caractère Unicode non montré][Caractère Unicode non montré][Caractère Unicode non montré][Caractère Unicode non montré][Caractère Unicode non montré][Caractère Unicode non montré][Caractère Unicode non montré][Caractère Unicode non montré][Caractère Unicode non montré][Caractère Unicode non montré][Caractère Unicode non montré][Caractère Unicode non montré][Caractère Unicode non montré]` dans votre terminal, ou bien l'URL avec votre navigateur. Vous trouverez d'autres exemple amusants sur le site de Magnus Bodin `<https://x42.com/active/ip32.mpl?host=www.bortzmeyer.org>`.

Au fait, quelle est la représentation **binaire** des adresses IPv4 ? C'est un nombre de 32 bits, gros boutien (section 3.2 et annexe B du RFC 791). En revanche, le RFC ne précise pas directement que ces nombres sont non signés...

Et pour IPv6 ? Cette fois, il existe une représentation normalisée : le RFC 4291 la fixe dans sa section 2.2, la représentation préférée étant celle avec les deux-points, par exemple `2001:DB8::8:800:200C:417A` (une norme plus stricte a été ensuite définie dans le RFC 5952). Mais le RFC ne donne pas de grammaire, par exemple en ABNF, de cette syntaxe. Plusieurs RFC en contiennent une (par exemple le RFC 3986, section 3.2.2 ou bien le RFC 5321, section 4.1.3), mais il n'existe pas de grammaire formelle standard.

La question de la représentation texte peut susciter des polémiques, notamment lorsque il existe déjà des programmes qui analysent ces représentations texte et qu'une réforme de celle-ci peut invalider ces programmes. Ainsi, il a fallu du temps pour adopter une forme texte standard pour les numéros de systèmes autonomes (AS pour "*Autonomous System*") lorsque ceux-ci comportent quatre octets, comme permis par le RFC 4893, prédécesseur du RFC 6793. Un "*Internet-Draft*" avait été écrit, `draft-michaelson-4byte-as-representation-05`, mais avait échoué devant l'IESG. Il existe en

2. Car trop difficile à faire afficher par LaTeX

effet deux écoles pour la représentation des numéros d'AS, celle qui dit qu'un numéro d'AS ne doit être qu'un identificateur **opaque** et que sa syntaxe texte n'a donc pas besoin d'être lisible ou mémorisable, donc que le nombre décimal, par exemple 196613, suffit (on parle d'ASPLAIN). Et une autre école dit que les numéros d'AS sont échangés entre humains, par téléphone ou par courrier, et doivent donc être des identificateurs utilisables par des humains. Cette école écrirait donc le même numéro d'AS 3.5 (3 fois 65536 plus 5, notation dite ASDOT). Avec les anciens numéros d'AS, sur deux octets, la question n'avait pas une grande importance, puisque tous les numéros étaient relativement petits, inférieurs à 65535. Ainsi, les programmes qui les traitaient ne s'attendent pas à une syntaxe structurée. Outre les questions de principe ci-dessus, la syntaxe structurée, avec le point comme séparateur, est critiquée par ceux qui maintiennent un tel programme. Le RFC 4893 est donc sorti sans représentation texte standard et il a fallu attendre le RFC 5396 pour en avoir une (ASPLAIN).

Par contre, une forme texte existe depuis toujours pour les filtres LDAP (RFC 2254). Leur représentation binaire est gouvernée par les règles d'encodage d'ASN.1 et est donc inaccessible à l'humain normal. Si, pour une adresse IPv4, le passage de la forme texte à la forme binaire est faisable « de tête », pas question d'en faire autant avec BER. D'où la forme texte des filtres maintenant bien connue comme `&(cn=latham 47)(state=lost)`.

Le dernier exemple de cet article concernera le DNS. La représentation texte des noms de domaine est bien connue et on trouve désormais de tels noms même sur la devanture des boulangeries (`www.boulangerie-maeder.f`
Mais cette représentation texte (RFC 1034, section 3.1) n'a rien à voir avec ce qui circule « sur le câble ». En effet, la forme binaire de ces noms (RFC 1035, section 3.1) n'utilise pas le point comme séparateur mais un encodage longueur-valeur, où un octet indiquant la longueur est suivi par le label (les noms de domaine sont composés de labels). Le décodage de cette forme est détaillée dans « Décoder les paquets DNS capturés avec pcap <https://www.bortzmeyer.org/pcap-decodage-dns.html> ».

On peut donc créer des noms de domaines comportant des points, même s'ils seront difficiles à manipuler.