

# RFC 3490 : Internationalizing Domain Names in Applications (IDNA)

Stéphane Bortzmeyer  
<stephane+blog@bortzmeyer.org>

Première rédaction de cet article le 18 Décembre 2006

Date de publication du RFC : Mars 2003

<http://www.bortzmeyer.org/3490.html>

---

Sur la planète, des dizaines d'écritures différentes sont en service. Jusqu'à ce RFC, qui normalise les IDN, il n'était pas possible de les utiliser dans les noms de domaine où, pour des raisons diverses, seul un sous-ensemble d'US-ASCII était possible.

Contrairement à ce qu'on lit souvent (le nombre d'énormités publiées sur les IDN remplit beaucoup de disques durs chez Google), ce n'est pas la faute du DNS. Celui-ci est parfaitement compatible avec des encodages d'Unicode comme UTF-8 (c'est explicite dans le RFC 1035<sup>1</sup>, même si sa section 2.3.1 cite une *"Preferred name syntax"* mais le RFC 2181 a enfoncé le clou et rappelant en section 11 que *"Those restrictions aside, any binary string whatever can be used as the label of any resource record."*). BIND ou nsd acceptent n'importe quelle chaîne d'octets. PowerDNS est bogué de ce point de vue mais la résolution de cette bogue est en cours (<http://wiki.powerdns.com/cgi-bin/trac.fcgi/ticket/115>).

Non, le problème qui fait qu'on ne pouvait écrire que <http://www.maconneriegenerale.fr/> au lieu de <http://www.maconneriegnrale.fr/> était plus complexe (<http://www.bortzmeyer.org/pourquoi-idn-et-pas-un-dns-unicode.html>):

- les noms de **machines**, eux, sont soumis à des règles très restrictives, listées dans le RFC 1123.
- les serveurs DNS doivent effectuer une canonicalisation des noms demandés : par exemple, `chocolat.fr` et `Chocolat.FR` sont équivalents. Pour Unicode, une telle canonicalisation est bien plus complexe et la mettre en œuvre nécessiterait de changer tous les serveurs DNS...

---

<sup>1</sup>Pour voir le RFC de numéro NNN, <http://www.ietf.org/rfc/rfcNNN.txt>, par exemple <http://www.ietf.org/rfc/rfc1035.txt>

L'approche imposée par l'IESG a donc été de ne pas toucher à l'« infrastructure » (les serveurs de noms) mais de tout faire dans les applications. D'où le nom d'**IDNA**, pour "*IDN in Applications*"IDN.

Le principe d'IDN est le suivant : le nom en Unicode, par exemple `CAF.fr` est d'abord canonicalisé par l'algorithme `nameprep` (normalisé dans le RFC 3491, `nameprep` est un profil de `stringprep`, qui était normalisé dans le RFC 3454). Il devient ici `caf.fr` (les règles de `nameprep` sont évidemment bien plus complexes pour les écritures non-latines).

Ensuite, le nom est encodé en punycode, selon l'algorithme normalisé dans le RFC 3492. Il devient `xn--caf-dma.fr`, chaîne de caractères qu'il vaut évidemment mieux ne pas montrer à l'utilisateur. C'est cette chaîne, en pur ASCII qui va être passée au DNS. Les serveurs de noms n'ont donc besoin d'aucune mise à jour.

Aujourd'hui, il existe de nombreuses bibliothèques, y compris en logiciel libre, pour gérer les IDN, de façon à ce que l'auteur de l'application n'aie pas à le faire lui-même. Citons par exemple GNU `libidn` (<http://josefsson.org/libidn/>) ou comme celle qui est accessible (<http://docs.python.org/lib/module-encodings.idna.html>) aux programmeurs Python.

Et plusieurs TLD comme `.de` ou bien `.no` (ou évidemment les TLD chinois comme `.tw`) permettent l'enregistrement d'IDN.

Comme je l'ai indiqué, beaucoup d'erreurs ont été ou sont toujours colportées à propos des IDN. Par exemple, la page Web <http://cr.yz.to/djbdns/idn.html> montre une grave ignorance d'Unicode et une complète incompréhension de la question de la canonicalisation. Une autre erreur courante (elle est régulièrement commise par l'ICANN) est de confondre langue et écriture. IDN ne s'occupe que des écritures, pas des langues.