RFC 8753 : Internationalized Domain Names for Applications (IDNA) Review for New Unicode Versions

Stéphane Bortzmeyer
<stephane+blog@bortzmeyer.org>

Première rédaction de cet article le 16 avril 2020

Date de publication du RFC : Avril 2020

https://www.bortzmeyer.org/8753.html

La norme IDN, qui permet depuis 2003 les noms de domaine en Unicode était autrefois liée à une version particulière d'Unicode. La révision de cette norme en 2010 a libérédélivré <https://www.youtube.com/watch?v=wQP9XZc2Y_c> IDN d'une version spécifique. La liste des caractères autorisés n'est plus statique, elle est obtenue par l'application d'un algorithme sur les tables Unicode. Tout va donc bien, et, à chaque nouvelle version d'Unicode, il suffit de refaire tourner l'algorithme ? Hélas non. Car, parfois, un caractère Unicode change d'une manière qui casse la compatibilité d'IDN et qui, par exemple, rend invalide un nom qui était valide auparavant. La norme IDN prévoit donc un examen manuel de chaque nouvelle version d'Unicode pour repérer ces problèmes, et décider de comment les traiter, en ajoutant une exception spécifique, pour continuer à utiliser ce caractère, ou au contraire en décidant d'accepter l'incompatibilité. Ce nouveau RFC ne change pas ce principe, mais il en clarifie l'application, ce qui est d'autant plus nécessaire que les examens prévus n'ont pas toujours été faits, et ont pris anormalement longtemps, entre autre en raison de l'absence de consensus sur la question.

Le RFC à lire avant celui-ci, pour réviser, est le RFC 5892 [1]. Il définit l'algorithme qui va décider, pour chaque caractère Unicode, s'il est autorisé dans les noms de domaine ou pas. À chaque version d'Unicode (la dernière est la 13.0 <https://www.bortzmeyer.org/unicode-13-0.html>), on refait tourner l'algorithme. Bien qu'Unicode soit très stable, cela peut faire changer un caractère de statut. Si un caractère auparavant interdit devient autorisé, ce n'est pas grave. Mais si l'inverse survient ? Des noms qui étaient légaux cessent de l'être, ne laissant le choix qu'entre deux solutions insatisfaisantes, supprimer le nom, ou bien ajouter une exception manuelle (procédure qui n'est pas parfaitement définie dans le RFC 5892, qui sera faite, si tout va bien, par de nouveaux RFC.)

1. Pour voir le RFC de numéro NNN, https://www.ietf.org/rfc/rfcNNN.txt, par exemple https://www.ietf.org/rfc/rfc5892.txt

[Notez que ce RFC affirme que les registres sont responsables des noms qu'ils autorisent, point de vue politique et non technique, contestable et d'ailleurs contesté.]

En quoi consiste donc le nouveau modèle d'examen qui accompagne la sortie d'une nouvelle version d'Unicode ? La section 3 du RFC l'expose. D'abord faire tourner l'algorithme du RFC 5892, puis noter les caractères qui ont changé de statut. Cela doit être fait par un expert nommé par l'IESG (DE, pour "*Designated Expert*", actuellement Patrik F[Caractère Unicode non montré [2]]ltstr[Caractère Unicode non montré]m, un des auteurs du RFC.) Puis réunir un groupe d'experts (il faut un groupe car personne n'est compétent dans tous les aspects d'IDN à la fois, ni dans toutes les écritures normalisées, cf. annexe B ; si l'IETF n'a pas de problèmes à trouver des experts en routage, elle doit par contre ramer pour avoir des experts en écritures humaines.) La recommandation pour ce groupe est d'essayer de préserver le statut des caractères et donc, si nécessaire, de les ajouter à une table d'exceptions (le RFC 6452 avait pris la décision opposée.) Le fait d'avoir des experts explicitement nommés va peut-être permettre d'éviter le syndrome du « il faudrait que quelqu'un s'en occupe » qui avait prévalu jusqu'à présent.

La promesse de publier les tables résultant de l'algorithme du RFC 5892 n'ayant pas été tenue (ce qui a fait râler <https://www.icann.org/public-comments/proposed-iana-sla-lgr-idn-tables-2019-0 le RFC espère que cette fois, le travail sera fait. Pour l'instant, l'état actuel de la révision est dans l'"*Internet-Draft*" draft-faltstrom-unicode12-00.

L'annexe A résume les changements depuis le RFC 5892 (qui n'est pas remplacé, juste modifié légèrement) :
— Séparer explicitement l'examen des nouvelles versions d'Unicode en deux, la partie algorithmique et la partie plus politique,
— Désigner un groupe d'experts pour cette deuxième partie,
— Et quelques autres détails pratico-bureaucratiques sur l'organisation du travail.

La section 2 du RFC rappelle l'histoire d'IDN. La norme originale, le RFC 3490 définissait l'acceptabilité de chaque caractère dans un nom de domaine. Lorsqu'une nouvelle version d'Unicode sortait, il n'y avait pas de mécanisme pour décider d'accepter ou non les nouveaux caractères. Il aurait fallu refaire un nouveau RFC pour chaque version d'Unicode (il y en a environ une par an.) La version 2 d'IDN, dans le RFC 5890, a changé cela, en décidant de normaliser l'algorithme et non plus la liste des caractères. L'algorithme, décrit en détail dans le RFC 5892, utilise les propriétés indiquées dans la base de données Unicode (cf. la norme <https://www.unicode.org/versions/Unicode13.0.0/>, sections 4 et 3.5) pour décider du sort de chaque caractère. Dans un monde idéal, ce sort ne change pas d'une version d'Unicode à l'autre. Si les propriétés Unicode des caractères sont stables, le statut (accepté ou refusé) restera le même, l'algorithme ne servant vraiment que pour les nouveaux caractères introduits par la nouvelle version d'Unicode. Mais, bien qu'Unicode fasse certaines promesses de stabilité <https://www.unicode.org/policies/stability_policy.html>, elles ne sont pas totales : un caractère peut donc se retrouver accepté une fois, puis refusé. Le RFC 5892 prévoyait donc, à juste titre, un examen manuel, et le RFC 5892 décrivait une table d'exceptions permettant de maintenir la compatibilité (section 2.7). Cette table aurait pu être remplie au fur et à mesure, pour conserver une certaine stabilité. Mais un tel examen n'a été fait qu'une fois, pour Unicode 6 <https://www.bortzmeyer.org/unicode-6-0.html>, dans le RFC 6452, suite à quoi il a été décidé de ne pas ajouter d'exceptions (et donc de laisser des noms devenir invalides.) En 2015, l'IAB avait demandé que les mises à jour soient suspendues <https://www.iab.org/documents/correspondence-reports-documents/2015-2/iab-statement-on-identifiers-and-unicode-7-0-0/>, suite au « problème » (dont tout le monde n'est pas convaincu) du caractère arabe b[Caractère Unicode non montré][Caractère Unicode non montré] avec hamza ([Caractère Unicode non montré], U+08A1). Puis, en 2018, avait insisté sur l'importance de terminer ce travail <https://www.iab.org/documents/correspondence-reports-documen 2018-2/iab-statement-on-identifiers-and-unicode/>. Deux ans après, ce n'est toujours pas fait. Donc, en pratique, IDNA version 2 n'a pas encore tenu sa promesse de pouvoir être mis à jour « presque » automatiquement.

2. Car trop difficile à faire afficher par LaTeX

À noter un autre problème avec la nouvelle version d'IDN : le fait qu'un caractère soit autorisé ou pas dépend d'un algorithme, et non plus d'une table statique. Mais, pour faciliter la vie des utilisateurs, l'IANA avait produit des tables en appliquant l'algorithme aux données Unicode. Il a toujours été prévu que ces tables soient juste une aide, qu'elles ne fassent pas autorité mais, malheureusement, certaines personnes les ont compris comme étant la référence, ce qui n'était pas prévu. Comme ces tables IANA n'ont pas été mises à jour au fur et à mesure des versions d'Unicode, le problème devient sérieux. Le RFC demande à l'IANA de modifier la description des tables <https://www.iana.org/assignments/idna-tables-6.3.0/idna-tables-6.3.0.xml> pour insister sur leur caractère non-normatif (« "*It should be stressed that these are not normative in that, in principle, an application can do its own calculations and these tables can change as IETF understanding evolves.*" ».)

J'ai écrit que tout le monde n'était pas convaincu de la nature du problème. Il y a en effet un désaccord de fond au sujet d'Unicode, entre ceux qui considèrent que toutes les lettres se valent, que toutes les écritures doivent avoir le même statut et, ceux, souvent des utilisateurs de l'alphabet latin, que la diversité dérange et qui voudraient mieux contrôler les caractères « dangereux », en utilisant des arguments de sécurité contestables <https://www.bortzmeyer.org/idn-et-phishing.html>. La question est d'autant plus sérieuse que les retards de mise à jour d'IDN (qui est toujours en version 6 alors qu'Unicode est en version 13 <https://www.bortzmeyer.org/unicode-13-0.html>) handicape surtout les utilisateurs des écritures les plus récemment ajoutées dans Unicode, en général des peuples minoritaires et peu présents dans le business international. Le retard n'est donc pas forcément gênant de la même manière pour tout le monde…