RFC 5891 : Internationalized Domain Names in Applications (IDNA): Protocol

Stéphane Bortzmeyer
<stephane+blog@bortzmeyer.org>

Première rédaction de cet article le 22 août 2010

Date de publication du RFC : Août 2010

https://www.bortzmeyer.org/5891.html

L'ensemble <https://www.bortzmeyer.org/idnabis.html> des RFC sur la deuxième version d'IDN couvre la terminologie (RFC 5890 [1]), les tables de caractères autorisés (RFC 5892), les justifications des choix effectués (RFC 5894) et, dans ce RFC 5891, le protocole lui-même, utilisé pour l'enregistrement et la lecture des noms de domaine.

Ces RFC « IDNAbis » remplacent donc les RFC 3490 et RFC 3491 mais pas le RFC 3492, Punycode continuant à être l'encodage utilisé pour les IDN, avec le même préfixe, xn--. Le principe reste le même : comme les règles pour les noms de machine (mais pas pour les noms de domaine <https://www.bortzmeyer.org/pourquoi-idn-et-pas-un-dns-unicode.html>) imposent l'utilisation des seuls caractères US-ASCII, comme il n'est pas question de mettre à jour toutes les plate-formes utilisées, IDN fonctionne en encodant les noms de domaines Unicode en ASCII, grâce à l'algorithme Punycode. L'infrastructure n'est donc pas modifiée mais on peut présenter aux utilisateurs le vrai nom en Unicode, quitte à le traduire lors du passage sur le réseau. (Officiellement, IDNAbis est nommé IDNA 2008 mais ce nom est incorrect, le protocole n'ayant pas été terminé en 2008.)

La section 3 précise ce fonctionnement en exigeant qu'un nom de domaine utilisé comme tel (i.e. comme élément d'un protocole, pas lorsqu'il est simplement cité dans du texte libre), doit être encodé en ASCII lorsqu'on parle aux applications non-IDN, et que la comparaison entre deux noms de domaine (pour voir s'ils sont égaux) doit se faire entre deux formes Unicode ou deux formes ASCII mais pas en mélangeant les deux. Dans IDNAbis, le passage de la forme Unicode ("*U-label*") à la forme ASCII ("*A-label*") et en sens inverse se fait sans perte d'informations. Les deux comparaisons sont donc

1. Pour voir le RFC de numéro NNN, https://www.ietf.org/rfc/rfcNNN.txt, par exemple https://www.ietf.org/rfc/rfc5890.txt

équivalentes. Comme beaucoup de protocoles utilisent des noms de domaine, les IDN affectent potentiellement beaucoup de monde. Sauf si le protocole le prévoit explicitement (ce qui est le cas des IRI du RFC 3987), les IDN doivent donc être mis sous leur forme ASCII. Idem pour les requêtes et réponses effectives faites avec le DNS.

Petit détail au passage. On trouve souvent des noms de domaine dans la partie **droite** d'un enregistrement DNS, par exemple dans le champ `RNAME` d'un SOA, qui indique l'adresse de courrier du responsable de la zone). Comme le rappelle la section 3.2.2, IDN ne change pas ces champs qui restent actuellement en pur ASCII, en attendant un futur RFC (le RFC 6530 ne suffit pas).

Passons maintenant aux deux protocoles utilisés par IDNAbis, le protocole d'enregistrement et le protocole de résolution. Le premier est décrit dans la section 4. Il concerne l'enregistrement d'un nom de domaine Unicode auprès d'un registre (attention, un registre ne gère pas forcément un TLD, ce protocole concerne tous les registres, même, par exemple, celui de `bortzmeyer.org`).

Donc, première étape, le registre reçoit un nom en Unicode (section 4.1). Il doit être normalisé en NFC et peut être encore normalisé selon des règles locales (cf. RFC 5895). Ce nom peut être transmis directement en Unicode (« "*U-label*" ») ou bien encodé en Punycode (« "*A-label*" »). Le RFC recommande de joindre la forme Punycode (voire uniquement celle-ci) mais il n'y a aucune justification technique à ce choix, c'est juste du FUD anti-Unicode.

Le registre doit ensuite vérifier que le nom est correct techniquement (section 4.2). Ainsi, il faut tester que la conversion "*U-label*" -¿ "*A-label*" et retour redonne bien le meme nom et, si uniquement une forme Punycode est reçue, qu'elle est bien légale (toute chaîne de caractères commençant par `xn--` n'est pas forcément du Punycode). Les caractères interdits doivent être absents (cf. RFC 5892).

À côté des règles absolues (« le caractère ; est interdit »), il existe également des règles contextuelles comme l'interdiction du caractère Katagana U+30FB sauf dans les écritures utilisées au Japon (cf. RFC 5892). Enfin, si le nom comporte des caractères dont la directionnalité est de droite à gauche (cas de l'écriture arabe), il faut également suivre les prohibitions du RFC 5893.

Notons que les interdictions de ce RFC 5891 ne sont qu'un minimum. Un registre peut toujours ajouter ses propres règles comme de n'accepter que les caractères qui sont utilisés pour la langue locale, ou bien pour interdire des mots considérés comme grossiers.

Après ce parcours du combattant, le nom est enregistré. Reste à le résoudre via une requête DNS. C'est l'objet de la section 5, sur le protocole de résolution. Ce dernier effectue moins de tests puisqu'ils sont censés avoir été faits à l'enregistrement. Notez toutefois que ce n'est pas un argument très solide : non seulement il peut exister des registres qui ne font pas les tests ou bien les font mal mais la seule existence des jokers dans le DNS (RFC 1034, section 4.3.3) permet à un nom non enregistré d'« exister » quand même.

Bref, pour résoudre un IDN, le client doit donc convertir en Unicode (en effet, l'environnement de départ n'utilisait pas forcément Unicode, cela pouvait être, par exemple, Latin-1) et le normaliser en NFC. Ensuite, il teste que les caractères Unicode présents sont bien autorisés (section 5.4). Il est à noter que le résultat de ce test dépend de la version d'Unicode utilisée par le client (probablement via des bibliothèques standard fournies par le système d'exploitation). Ainsi, un nom de domaine utilisant un caractère très récemment affecté par Unicode pourrait être refusé par beaucoup de clients IDN.

Enfin, le client IDN convertit le nom en Punycode et termine par une résolution DNS normale (sections 5.5 et 5.6).

La section sur la sécurité, obligatoire dans les RFC, mentionne le risque de confusion entre des caractères similaires (un FUD classique contre IDN <https://www.bortzmeyer.org/idn-et-phishing.html>) mais ne fournit pas de solution dans le protocole. Elle compte sur les registres pour ne pas accepter les noms problématiques.

L'annexe A dresse la liste des différences avec la première version d'IDN. Je cite notamment :

— Au lieu de dépendre d'une version spécifique d'Unicode (la 3.2 pour IDNA 1), le protocole est désormais indépendant de la version : tout changement dans Unicode est automatiquement disponible.
— Les protocoles pour l'enregistrement d'un nom et sa résolution sont désormais séparés (et légèrement différents).
— Les caractères de ponctuation et les symboles sont désormais presque tous exclus.
— Il n'y a plus d'étape de normalisation standard comme l'était le nameprep du RFC 3491. Seul reste le NFC. D'une manière générale, chaque application est désormais libre d'effectuer la correspondance ("*mapping*") entre ce qu'a tapé ou sélectionné l'utilisateur et l'IDN (cf. RFC 5895 pour un exemple).
— Le modèle de sélection des caractères autorisés est passé de « entièrement manuel, caractère par caractère » à « essentiellement algorithmique, fondé sur les propriétés Unicode - avec un peu d'exceptions manuellement ajoutées ». C'est ce qui permet l'indépendance par rapport aux versions d'Unicode.
— La validité d'un caractère peut désormais dépendre du contexte (ce qui complique sérieusement la vérification).
— Nouvelles règles « bidi ».
— Largement compatible avec l'ancien IDN (même Punycode, même préfixe, beaucoup de règles communes) mais pas totalement (certains noms légaux deviennent invalides).

Pour un point de vue critique sur IDNA bis, on peut consulter le "*Unicode Technical Standard #46, « Unicode IDNA Compatibility Processing »*" <http://www.unicode.org/reports/tr46/>, qui remet notamment en cause le soi-disant rôle d'Unicode dans le hameçonnage <https://www.bortzmeyer.org/idn-et-phishing.html>. Une critique de cette critique a été publiée en "*Review of Unicode TR#46*" <http://stupid.domain.name/node/955>.