RFC 3339 : Date and Time on the Internet: Timestamps

Stéphane Bortzmeyer
<stephane+blog@bortzmeyer.org>

Première rédaction de cet article le 30 août 2009

Date de publication du RFC : Juillet 2002

https://www.bortzmeyer.org/3339.html

Comment représente-t-on une date sur l'Internet ? Il existe beaucoup de RFC qui ont besoin d'un format de date et, si certaine restent fidèles à des vieux formats, la plupart des RFC modernes se réfèrent à ce RFC 3339 [1] qui définit un sous-ensemble d'ISO 8601 pour la représentation <https://www.bortzmeyer.org/representation-texte.html> des dates.

Un des exemples les plus connus des vieux formats est le RFC 5322 qui garde, pour le courrier électronique, l'ancien format (initialement normalisé dans le RFC 724) difficile à analyser, et qui ressemble à `Date: Thu, 27 Aug 2009 08:34:30 +0200`. Ce format n'a été gardé que parce qu'on ne peut plus changer la norme du courrier sur un point aussi fondamental. Mais les RFC récents définissent un autre format, celui de notre RFC 3339. C'est, par exemple, le cas du RFC 4287 qui dit que les éléments XML d'un flux de syndication Atom doivent utiliser ce format, par exemple <published>2009-08-25T00:00:00Z</pu

Pourquoi un RFC sur ce sujet et ne pas simplement se référer à la norme ISO 8601 ? Il y a plusieurs raisons. L'une est politique, c'est le fait que l'ISO ne distribue pas librement le texte de la plupart de ses normes (une version provisoire non redistribuable traine sur le serveur de l'ISO <http://www.loc.gov/standards/datetime/iso-tc154-wg5_n0039_iso_wd_8601-2_2016-02-16.pdf>). L'autre est que ISO 8601, comme beaucoup de normes ISO, est mal spécifiée, très complexe et pleine de détails inutiles (cf. section 5.5). L'IETF a donc choisi la voie d'un **sous-ensemble** de ISO 8601, le format du RFC 3339.

La section 1 du RFC résume le problème général de la transmission et de l'affichage des dates. Beaucoup de programmes se sont cassés le nez sur ce sujet (je me souviens d'un logiciel commercial de gestion de tickets, très cher, qui envoyait des messages avec des dates invalides, ce qui décidait un autre logiciel commercial très cher, chargé de la lutte contre le spam, à les jeter).

1. Pour voir le RFC de numéro NNN, https://www.ietf.org/rfc/rfcNNN.txt, par exemple https://www.ietf.org/rfc/rfc3339.txt

Le RFC est bâti sur quelques principes, exposés dans cette section 1. On utilise évidemment le calendrier grégorien. Comme le but est de dater des événements Internet, on se limite aux dates de l'ère actuelle (i.e. après Jésus-Christ), on suppose que le décalage entre l'horloge locale et UTC est connu, etc.

Le problème de l'heure locale fait l'objet de la section 4. Les règles définissant l'heure légale étant très compliquées et changeantes (en raison de la stupide heure d'été), le choix est de s'appuyer sur UTC (section 4.1). Puisque le décalage entre l'heure locale et UTC est supposé connu, il est prévu de permettre d'indiquer ce décalage (section 4.2). Cela se fait toujours sous forme numérique, les noms alphabétiques des fuseaux horaires n'étant pas normalisés. Enfin, la section 4.4 réenfonce le clou au sujet des horloges qui ne connaissent que l'heure locale et pas son décalage avec UTC : de telles horloges sont inutilisables sur l'Internet puisqu'elles les indications temporelles transmises à partir d'elles sont fausses 23 fois sur 24 (dès qu'on n'est pas dans le même fuseau horaire).

Les historiens notent que c'était le cas des systèmes d'exploitation Microsoft pendant longtemps : l'horloge du PC ne stockait que l'heure locale, ce qui a créé d'innombrables problèmes lorsque Windows a dû se résigner à accepter les connexions avec l'Internet.

Voici pour le concept de temps. Et le format ? Quel est le cahier des charges précis ? La section 5 commence par énumérer les caractéristiques souhaitées. Les temps doivent pouvoir être comparées lexicographiquement (section 5.1), par exemple avec `strcmp()` en C ou `sort` avec le shell Unix. Contrairement à une idée répandue, ISO 8601 ne garantit pas cette possibilité (voir la section 5.1 pour une liste des règles à observer pour qu'une telle comparaison soit possible).

Autre critère important, le format doit être compréhensible par un humain (section 5.2). C'est une caractéristique fréquente des protocoles Internet ; les formats binaires sont relativement rares. Ceci dit, un format trop lisible peut mener à des erreurs. Par exemple, le format de date du RFC 5322 a mené certains programmeurs à croire qu'ils pouvaient traduire les noms de jour et de mois, puisque ceux-ci apparaissent sous une forme non-numérique (c'était l'erreur du logiciel commercial dont je parlais plus tôt). Le format de notre RFC est donc compréhensible mais pas forcément familier à tout être humain et les logiciels doivent donc se préparer à l'afficher sous une forme plus conforme aux habitudes locales.

Autre erreur du format du RFC 5322, la redondance de l'information (section 5.4). Inclure le nom du jour est redondant (puisqu'on peut le calculer à partir de la date, l'annexe B donne un exemple de code) et augmente donc les risques d'erreur (que faire si ce jour n'est pas correct, qui croire ?).

Assez de préliminaires, quel format utiliser, maintenant ? La section 5.6 va le décrire rigoureusement, en utilisant le langage ABNF (RFC 5234). Il existe quelque options comme le fait que le temps peut être écrit en heure locale, si le décalage avec UTC est indiqué. La section 5.8 donne quelques exemples de temps corrects. Voici comment les produire avec GNU `date` (le fait de séparer la date et le temps par un espace et pas par la lettre T est une violation de la grammaire mais le RFC est ambigu sur ce point) :

```
% date --rfc-3339='ns'
2009-07-12 11:03:08.434344377+02:00

% date --rfc-3339='seconds'
2009-07-12 11:03:12+02:00

% date --rfc-3339='date'
2009-07-12
```

et voici un programme Python qui affiche la date actuelle en UTC (donc avec le Z à la fin, Z étant l'équivalent de `+00:00`) :

```
import time

# date+time but no fractional seconds
rfc_3339_nosecfrac = "%Y-%m-%dT%H:%M:%SZ"

# gmtime = UTC (old name)
print time.strftime(rfc_3339_nosecfrac,
                    time.gmtime(time.time()))
```

Et, en C, voici un exemple possible :

```
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
#include <sys/time.h>

#define MAX_SIZE 256
#define RFC_3339_NOFRAC_UTC "%Y-%m-%dT%H:%M:%SZ"

int
main()
{
    char           *result = malloc(MAX_SIZE);
    struct timeval  tv;
    struct tm      *time;
    size_t          result_len;
    (void) gettimeofday(&tv, NULL);
    time = gmtime(&tv.tv_sec);
    result_len = strftime(result, MAX_SIZE, RFC_3339_NOFRAC_UTC, time);
    if (result_len != 0) {
        printf("%s\n", result);
        exit(0);
    } else {
        printf("Error in strftime\n");
        exit(1);
    }
}
```

Y a t-il des problèmes de sécurité cachés derrière l'affichage de la date ? Oui, la section 7 en cite un : la connaissance de l'heure locale d'un système peut permettre à un attaquant de déterminer les heures où le système est le plus suceptible d'être surveillé et le paranoïaque peut donc avoir intérêt à n'émettre que des temps en UTC…

Notre RFC compte plusieurs annexes intéressantes. Par exemple, l'annexe A est une rétro-ingénierie de la norme ISO 8601. Comme celle-ci ne spécifie pas formellement de grammaire pour le format des dates, ce sont les auteurs du RFC qui en ont établi une, à partir de la norme. Elle est bien plus riche (avec beaucoup plus d'options) que celle de la section 5.6. L'annexe D donne les détails sur la gestion des secondes intercalaires, qu'on peut trouver en ligne <http://tycho.usno.navy.mil/leapsec.html>.