RFC 5784 : Sieve Email Filtering: Sieves and display directives in XML

Stéphane Bortzmeyer
<stephane+blog@bortzmeyer.org>

Première rédaction de cet article le 11 mars 2010

Date de publication du RFC : Mars 2010

https://www.bortzmeyer.org/5784.html

Le langage de filtrage de courrier Sieve, normalisé dans le RFC 5228[1] a une syntaxe qui lui est propre et qui nécessite le développement d'outils spécifiques pour l'analyser. Or, il existe un format générique pour lequel d'innombrables outils ont été développés, XML. Ce RFC décrit donc un moyen de représenter les scripts Sieve en XML, ainsi que les règles de conversion de et vers le format traditionnel.

La syntaxe officielle de Sieve est celle d'un fichier texte, format simple et facilement traitable par des programmes. Mais la plupart des utilisateurs de Sieve, n'ayant pas d'expertise avec les langages formels, ne lisent et n'écrivent pas le Sieve dans ce format. Ils se servent d'outils spécialisés qui leur présentent le script sous une forme plus claire pour le non-spécialiste.

Le format XML étant très répandu, des facilités existent pour développer de tels outils, si la syntaxe d'entrée est du XML (section 1 du RFC). Comme ce n'est pas le cas de Sieve, notre RFC 5784 normalise une correspondance entre le format classique de Sieve et XML, permettant les conversions dans les deux sens. En outre, ce schéma XML ajoute la possibilité de donner des indications sur la présentation visuelle souhaitée, un point qui manquait dans la syntaxe Sieve classique (et qui était en général remplacé par des commentaires structurés spécifique à un logiciel d'édition).

La section 3 du RFC rappelle les particularités de la grammaire de Sieve : pour permettre les extensions futures, le langage n'a pas de mots réservés. La syntaxe est fixe (les extensions ne peuvent pas la modifier ou l'étendre). Toute extension peut ajouter des mots-clés. Et il existe déjà de très nombreuses extensions.

Un script Sieve est constitué de **commandes**, qui sont des **actions** ou des **contrôles** (mais la syntaxe ne fournit aucun moyen de distinguer les deux, seule la connaissance du mot-clé le permet). La commande est composée d'un mot-clé, suivi d'arguments, d'un test optionnel et d'un bloc de commandes optionnel. Voici quelques exemples :

1. Pour voir le RFC de numéro NNN, https://www.ietf.org/rfc/rfcNNN.txt, par exemple https://www.ietf.org/rfc/rfc5228.txt

```
stop;   /* Un contrôle tout nu, sans argument ou bloc */
require "fileinto"; /* Un contrôle avec un seul argument */
if true {stop;}   /* Un contrôle avec un test et un bloc de
   commandes */
discard;   /* Une action toute nue, sans argument */
fileinto "folder";  /* Une action avec un argument */
```

La section 4 définit la représentation en XML. Les contrôles sont représentés par l'élement `<control>` et les actions par l'élément `<action>`. Les exemples ci-dessus seraient donc, en XML :

```
<control name="stop"/>
<control name="require"><str>fileinto</str></control>
<control name="if">
  <test name="true"/><control name="stop"/>
</control>

<action name="discard"/>
<action name="fileinto"><str>folder</str></action>
```

Les directives permettant de contrôler la représentation visuelle du script Sieve sont en section 4.1. Par exemple, l'élement `<displayblock>` permet de groupe ensemble des commandes Sieve :

```
<displayblock name="File filter list mail" order="1"
              group="FILE_TO_FOLDER" enable="true">
  <control name="if">
    <test name="header">
      <tag>is</tag>
      <str>Sender</str>
      <str>owner-ietf-mta-filters@imc.org</str>
    </test>
    <action name="fileinto">
      <str>filter</str>
    </action>
  </control>
</displayblock>
```

et l'élément `<displaydata>` d'indiquer du texte qui doit être montré à l'utilisateur lors de l'édition.

Pour permettre l'aller-retour dans les traductions de XML en Sieve, la section 4.2 spécifie des commentaires Sieve structurés permettant de garder l'information de présentation lors de la traduction. Ainsi, le code XML ci-dessus serait « sauvegardé » en :

```
/* [* name="File filter list mail" order="1" group="FILE_TO_FOLDER" enable="true" */
if header :is "Sender" "owner-ietf-mta-filters@imc.org"
...
```

où [* indique le début d'un `displayblock`.

Afin de permettre la validation des scripts Sieve en XML, notre RFC propose deux schémas, un en langage W3C Schema (annexe B), l'autre en Relax NG (annexe C). Prenons le schéma en RelaxNG ((en ligne sur `https://www.bortzmeyer.org/files/xmlsieve.rnc`)) et testons un script :

```
% rnv xmlsieve.rnc test2.xml
test2.xml
```

Parfait, il est bien valide.

L'annexe A contient un exemple complet d'un script Sieve (celui de la section 9 du RFC 5228) en XML. Sa version XML est disponible en (en ligne sur `https://www.bortzmeyer.org/files/rfc5228-sec9.xml`) et sa version sieve en (en ligne sur `https://www.bortzmeyer.org/files/rfc5228-sec9.sieve`).

Voici autre un exemple de script Sieve en XML complet. Ce script jette tous les messages qui n'ont pas de champ `Date:` ou de champ `From:`, ainsi que les messages dont l'expéditeur est `foobar@example.org` :

```
<?xml version="1.0" encoding="utf-8"?>
<sieve xmlns="urn:ietf:params:xml:ns:sieve">
   <control name="if">
     <test name="anyof">
       <test name="not">
         <test name="exists">
           <list><str>From</str><str>Date</str></list>
         </test>
       </test>
       <test name="header">
         <tag>contains</tag>
         <str>from</str>
         <str>foobar@example.org</str>
       </test>
     </test>
     <action name="discard"/>
   </control>
</sieve>
```

Pour le traduire en script Sieve classique, on peut utiliser le programme XSLT qui figure dans l'annexe D du RFC (et qui est également disponible en (en ligne sur `https://www.bortzmeyer.org/files/xml2sieve.xslt`)) :

```
% xsltproc xml2sieve.xslt test.xml

if anyof ( not ( exists [ "From", "Date" ] ),
   header :contains "from" "foobar@example.org" )
{
  discard;
}
```

En dehors de ce programme XSLT (qui ne va que dans un seul sens, depuis XML vers Sieve), je ne connais pas encore de mise en œuvre de ce RFC. Michel Sébastien me signale que JSieve `<http://james.apache.org/jsieve/>` semble avoir en projet une telle option `<http://james.apache.org/jsieve/util/index.html#Sieve%20In%20Xml>`.