# RFC 5424 : The syslog Protocol

Stéphane Bortzmeyer
<stephane+blog@bortzmeyer.org>

Première rédaction de cet article le 10 mars 2009

Date de publication du RFC : Mars 2009

https://www.bortzmeyer.org/5424.html

---

Mettant à jour l'ancienne description, voici la nouvelle spécification du protocole syslog, protocole de transmission d'informations sur les événements observés par un système.

syslog est un très ancien protocole, qui, comme souvent sur l'Internet, n'avait pas été normalisé pendant longtemps. Seul l'examen des sources du programme `syslogd` ou bien l'étude des paquets passant sur le réseau, permettaient de décrire le protocole. Le premier RFC à formaliser syslog était le RFC 3164 [1], qui vient d'être remplacé par notre RFC. Au contraire de son prédécesseur, qui décrivait l'existant, ce nouvel RFC et ses compagnons normalisent un nouveau protocole, en étendant l'ancien syslog, le "*BSD syslog*" (l'annexe A.1 discute des différences entre les deux protocoles). Parmi les changements du nouveau protocole, notons une description modulaire, qui sépare le format utilisé (qui fait l'objet de notre RFC) du protocole utilisé pour le transport des données (voir par exemple le RFC 5426).

syslog sert à transmettre des rapports sur des événements survenus dans un système. Le programme client ("*originator*") qui signale les événements transmet à un serveur syslog ("*collector*"), situé sur la même machine ou bien ailleurs sur le réseau. Le serveur syslog, typiquement configuré sur Unix via le fichier `/etc/syslog.conf` va ensuite enregistrer ces événements, par exemple dans un fichier comme `/var/log/mail.log`. Ces fichiers, véritables journaux de bord du système, permettent à l'ingénieur de suivre tout ce qui se passe. On y trouvera des informations telles que la date, le nom de la machine où à été noté l'événement, un court texte décrivant celui-ci, etc. Par exemple :

```
Nov 29 21:18:11 ludwigVI dhcpd: DHCPREQUEST for 172.19.1.25 from 00:1c:23:00:6b:7f via eth0
```

où le serveur DHCP a signalé une requête pour une adresse IP, ou bien :

1. Pour voir le RFC de numéro NNN, `https://www.ietf.org/rfc/rfcNNN.txt`, par exemple `https://www.ietf.org/rfc/rfc3164.txt`

```
Nov 29 21:06:17 foobar named[2418]: lame server resolving 'oasc08a.247realmedia.com' (in 'oasc08a.247realmed
```

où le serveur de noms BIND note un problème avec un domaine.

syslog est un immense succès. Non seulement tous les démons Unix l'utilisent pour signaler les événements qu'ils observent (tournant en permanence, sans console, sans utilisateur qui les suit, ils n'ont que ce canal pour communiquer) mais tout routeur, tout commutateur réseau a aussi un client syslog, qu'on peut configurer pour envoyer les messages à un serveur, souvent Unix, qui écoute sur le réseau et note tout. (Avec IOS, par exemple, cela se fait avec les commandes `logging facility local3` - ou un autre service que local3 - et `logging 192.0.2.84` pour indiquer l'adresse IP du serveur syslog.)

On peut aussi utiliser syslog pour noter manuellement <https://www.bortzmeyer.org/doc-by-syslog.html> des événements qu'on détecte.

L'architecture complète peut être plus complexe qu'un simple client/serveur, la section 4.1 donnant plusieurs exemples.

Rien ne garantit la bonne délivrance des messages, syslog est typiquement unidirectionnel (les sections 5.1 et 8.5 discutent plus en détail cette question).

La section 6 discute en détail du format des messages syslog, format conçu pour rester compatible avec le précédent, tout en permettant davantage de structuration (l'ancien format avait très peu de structure et il était donc difficile d'en extraire automatiquement des informations, par exemple pour le filtrage des événements avec un programme comme Swatch <http://swatch.sourceforge.net/>). Le premier champ d'un en-tête syslog, nommé `PRI` est écrit entre ¡ et ¿ et code le **service** ("*facility*") et la **gravité** ("*severity*") du message. Le service indique quelle partie du système a noté l'événements (2, le sous-système de courrier, 4, la sécurité, etc). Notons que ce système est très rigide et ne permet pas une grande finesse dans le classement. La gravité permet, elle, de distinguer les messages de débogage (gravité 7), de simple information (gravité 6) et ainsi de suite jusqu'aux extrêmes urgences (gravité 0). Service et gravité sont encodés dans un seul nombre en multipliant le premier par 8, donc, si j'émets un message du service 16 (≪ usage local 0 ≫) avec une gravité de 3 (erreur), ici avec la commande `logger` :

```
% logger -p local0.error "Test syslog"
```

Le message sur le réseau commencerait par `<131>` (16 * 8 + 3).

L'en-tête permet ensuite d'indiquer le numéro de version de syslog, la date, le nom de la machine émettrice, le nom de l'application (`logger`, cité plus haut, met par défaut le nom de l'utilisateur), un numéro qui peut identifier une instance de l'application (sur Unix, c'est typiquement le numéro de processus mais le RFC note à juste titre que ce champ ne peut pas avoir de signification standard)...

Une nouveauté de ce RFC est la présence de données structurées, après l'en-tête et avant le message. Ces données sont formatées de telle façon qu'un serveur syslog de l'ancien protocol peut toujours les traiter comme du texte. Elles s'écrivent sous forme de doublets attribut-valeur placés entre crochets. Les noms d'attributs possibles sont décrits dans la section 7 et font l'objet d'un registre IANA <https://www.iana.org/assignments/syslog-parameters> (section 9.2). Un exemple est `[meta language="fr"]` pour indiquer que le texte est en français ou bien `[timeQuality tzKnown="1" isSynced="1"]` pour indiquer que la machine qui a émis le message connait son fuseau horaire et que son horloge est synchronisée, par exemple par NTP.

Enfin, le message se termine par du texte libre, encodé en UTF-8. Le RFC cite ainsi comme exemple un message complet avec son en-tête :

```
<34>1 2003-10-11T22:14:15.003Z mymachine.example.com su - ID47 - BOM'su root' failed for lonvick on /dev/pts/8
```

syslog étant un protocole assez primitif, fonctionnant souvent sur le simple UDP, il n'est pas étonnant qu'il aie connu quelques problèmes de sécurité et que la section 8, consacrée à ce sujet, soit très détaillée. Parmi les nombreuses questions discutées, on peut citer par exemple le rejeu (section 8.4), contre lequel syslog n'a pas de protection, l'absence de garantie de délivrance des messages (section 8.5, qui note qu'on peut avoir de bonnes raisons de ne pas vouloir cette garantie), l'absence de contrôle de congestion, qui rend possible une noyade d'une machine sous les messages syslog (section 8.6), etc.

Une mise en œuvre comme rsyslog <http://www.rsyslog.com/> gère les principales nouveautés de ce RFC (comme le transport sur TCP, avec ou sans TLS).