RFC 5351 : An Overview of Reliable Server Pooling Protocols

Stéphane Bortzmeyer
<stephane+blog@bortzmeyer.org>

Première rédaction de cet article le 30 septembre 2008

Date de publication du RFC : Septembre 2008

https://www.bortzmeyer.org/5351.html

Il existe de très nombreux cas où une application cliente peut utiliser plusieurs serveurs parmi un groupe (le "*pool*"). L'idée de la suite de protocoles "*Reliable Server Pooling*" (RSerPool) est de fournir un moyen standard de faire ce choix. Ce RFC explique les concepts de base de ce protocole, normalisé dans d'autres RFC.

La section 1 du RFC résume les raisons pour lesquelles on veut disposer d'un groupe, un "*pool*", de serveurs plutôt que d'un serveur unique, par exemple, pour atteindre des objectifs de haute disponibilité ou de répartition de charge. Les nouveaux protocoles devront fournir notamment les services suivants :

— indépendance par rapport à l'application,
— liaison directe entre le client et le serveur, sans machine intermédiaire,
— séparation des fonctions de disponibilité et de changement en cas de panne, des fonctions propres à l'application.

Comment sont structurés les protocoles de la famille RSerPool ? (Le cahier des charges était le RFC 3237 [1] et prévoyait un protocole relativement léger, interne à un domaine administratif, moins ambitieux, donc, que les grilles.) Le groupe ("*pool*") a un identificateur, le PH ("*Pool Handle*"). Les serveurs du groupe se nomment les PE ("*Pool Element*"). Chaque PE s'enregistre auprès d'un serveur ENRP ("*Endpoint haNdlespace Redundancy Protocol*") en utilisant le protocole ASAP ("*Aggregate Server Access Protocol*"), décrit dans le RFC 5352. Le client se nomme, lui, le PU ("*Pool User*"). Il utilise également ASAP pour parler au serveur ENRP, obtenant ainsi la correspondance entre le PH et l'adresse IP du serveur effectif. Le PU parlera ensuite directement au PE, utilisant un protocole spécifique à l'application. Enfin, le serveur ENRP peut se synchroniser avec d'autres serveurs ENRP (RFC 5353) pour fournir également un service à haute disponibilité.

1. Pour voir le RFC de numéro NNN, https://www.ietf.org/rfc/rfcNNN.txt, par exemple https://www.ietf.org/rfc/rfc3237.txt

Ces protocoles sont décrits dans différents RFC, le RFC 5352 pour ASAP, le RFC 5353 pour ENRP, le RFC 5354 pour le format des paramètres, le RFC 5356 pour les politiques de sélection des serveurs et le RFC 5355 pour l'analyse des risques de sécurité.

La section 2 décrit ASAP ("*Aggregate Server Access Protocol*"), le protocole de communication entre le PU et le serveur ENRP, ainsi qu'entre ce dernier et les PE, les membres du groupe. ASAP est normalisé dans le RFC 5352. Il permet aux PE de s'enregistrer auprès du serveur ENRP (section 2.2) et au PU d'obtenir l'adresse d'un membre du groupe (section 2.3). Dès qu'un PE s'enregistre sous un identificateur de groupe (un PH), ce groupe existe et, lorsque le dernier PE se désenregistre, le groupe disparait (section 2.1).

Comme la communication entre le PU, le client et le PE, le serveur, est directe (ce qui permet au protocole d'être indépendant de l'application), il n'y a pas d'état et il peut donc être difficile de passer d'un PE à un autre en cas de panne, le nouveau PE ne se souvenant pas du contexte de l'application. La famille de protocoles RSerPool fournit donc quelques mécanismes limités pour conserver un état (section 2.5). Ainsi, la section 2.5.1 permet un système de petits gâteaux, envoyés par le PE au PU et que celui-ci doit renvoyer lorsqu'il change de PE, permettant ainsi de garder trace de l'état de l'application.

La section 3 décrit ENRP, "*Endpoint haNdlespace Redundancy Protocol*", le protocole de communication entre les serveurs ENRP. Ce protocole, normalisé dans le RFC 5353, permet aux serveurs ENRP d'un domaine de se synchroniser et de garder trace des PE enregistrés dans tel ou tel groupe.

La section 4, enfin, donne des exemples d'usage de la famille RSerPool. L'exemple de la section 4.1 est relativement simple, RSerPool y est utilisé pour sélectionner un serveur. La résolution d'un PH en adresse IP d'un serveur est donc le seul service invoqué. Cette section décrit les modifications nécessaires au code source de l'application :

— Spécifier un PH (identificateur de groupe) au lieu d'un nom (ou d'une adresse IP),
— Au lieu d'utiliser la fonction classique de résolution de nom, `getaddrinfo`, utiliser une fonction RSerPool de résolution,

À noter qu'il n'existe pas d'API standard pour les protocoles de la famille RSerPool. À noter également que, dans un exemple aussi simple, RSerPool n'était pas forcément nécessaire, les enregistrements DNS SRV du RFC 2782 auraient suffi.

L'exemple de la section 4.2 est plus complexe, faisant appel à toutes les fonctions de RSerPool, y compris pour la transmission de données entre le client, le PU et le serveur, le PE.

Il existe au moins une mise en œuvre de RSerPool. Elle est disponible en <http://tdrwww.iem.uni-due.de/dreibholz/rserpool/>. Elle est décrite dans la thèse "*Reliable Server Pooling – Evaluation, Optimization and Extension of a Novel IETF Architecture*" <http://duepublico.uni-duisburg-essen.de/servlets/DerivateServlet/Derivate-16326/Dre2006-final.pdf>. La page de l'auteur <http://tdrwww.exp-math.uni-essen.de/dreibholz/rserpool/> contient beaucoup d'autres informations sur la famille Rserpool.