RFC 9773 : ACME Renewal Information (ARI) Extension

Stéphane Bortzmeyer
<stephane+blog@bortzmeyer.org>

Première rédaction de cet article le 18 juin 2025

Date de publication du RFC : Juin 2025

https://www.bortzmeyer.org/9773.html

Si vous utilisez Let's Encrypt (et tout le monde l'utilise, de nos jours), vous avez sans doute reçu les messages « "*Let's Encrypt Expiration Emails Update*" » qui vous préviennent que cette AC n'enverra plus de rappels <https://letsencrypt.org/2025/01/22/ending-expiration-emails/> que vos certificats vont bientôt expirer. C'est parce qu'un meilleur système est maintenant disponible, ARI ("*ACME Renewal Information*"). ARI permet à une AC utilisant le protocole ACME d'indiquer à ses clients des suggestions sur le renouvellement des certificats. Il est décrit dans ce RFC.

ACME est normalisé dans le RFC 8555[1] et est surtout connu via le succès de Let's Encrypt. Les certificats sont de courte durée (aujourd'hui trois mois mais ça va diminuer) et il faut donc les renouveler souvent. On peut le faire automatiquement via cron, ou bien analyser le certificat et renouveler quand sa date d'expiration approche. L'un des problèmes est que cela peut mener à ce que plusieurs demandes de renouvellement arrivent en même temps sur l'AC. Le pic d'activité qui en résulterait pourrait charger l'AC inutilement. L'idée est donc que ce soit le serveur ACME, l'AC, qui planifie les renouvellements, et pas le client ACME. Cela permettrait aussi des changements de planning, comme une réduction des durées de validité, ou bien une révocation.

Donc, ARI ajoute un nouvel URL à la description d'une AC, `renewalInfo`. Voici celui de Let's Encrypt (qui met en œuvre ARI depuis deux ans <https://letsencrypt.org/2023/03/23/improving-resliiency-
>) :

1. Pour voir le RFC de numéro NNN, https://www.ietf.org/rfc/rfcNNN.txt, par exemple https://www.ietf.org/rfc/rfc8555.txt

```
% curl https://acme-v02.api.letsencrypt.org/directory
{
  "keyChange": "https://acme-v02.api.letsencrypt.org/acme/key-change",
  "meta": {
    "caaIdentities": [
      "letsencrypt.org"
    ],
    "profiles": {
          ...
    },
    "termsOfService": "https://letsencrypt.org/documents/LE-SA-v1.5-February-24-2025.pdf",
    ...
  },
  "newAccount": "https://acme-v02.api.letsencrypt.org/acme/new-acct",
  "newNonce": "https://acme-v02.api.letsencrypt.org/acme/new-nonce",
  "newOrder": "https://acme-v02.api.letsencrypt.org/acme/new-order",
  "renewalInfo": "https://acme-v02.api.letsencrypt.org/draft-ietf-acme-ari-03/renewalInfo",
  "revokeCert": "https://acme-v02.api.letsencrypt.org/acme/revoke-cert"
```

Ce nouveau membre a été ajouté au registre IANA <https://www.iana.org/assignments/acme/acme.xml#acme-resource-types>.

Et comment on obtient quelque chose à cet URL `https://acme-v02.api.letsencrypt.org/draft-ietf-a` On doit passer un identificateur du certificat. Il est formé par la concaténation de l'identifiant de l'AC et du numéro de série du certificat (pour garantir qu'il soit unique). Je passe sur les détails de construction (lisez la section 4 du RFC pour cela) mais ce petit script Python vous fait le calcul. Utilisons-le sur l'actuel certificat de ce blog :

```
% openssl x509 -text -in cert.pem
[Notez les valeurs]

% python ari-make-path.py 93:27:46:98:03:A9:51:68:8E:98:D6:C4:42:48:DB:23:BF:58:94:D2 00:06:7d:03:91:c6:49:b
kydGmAOpUWiOmNbEQkjbI79YlNI.AAZ9A5HGSbiDuuXG2orcvk0zeA

% curl -i https://acme-v02.api.letsencrypt.org/draft-ietf-acme-ari-03/renewalInfo/kydGmAOpUWiOmNbEQkjbI79YlN
...
retry-after: 21600
...
{
  "suggestedWindow": {
    "start": "2025-06-23T03:50:00Z",
    "end": "2025-06-24T23:00:50Z"
  }
}
```

Voilà, on a fait un URL en concaténant la valeur de `renewalInfo` avec celle obtenue à partir du certificat et on sait désormais quand est-ce que Let's Encrypt suggère de renouveler ce certificat. Le format de sortie est clair mais vous avez les détails dans la section 4.2. Les dates sont évidemment au format du RFC 3339. Au passage, la date d'expiration est le 24 juillet 2025 donc Let's Encrypt n'attend pas le dernier moment. (Comme le dit un commentaire dans le code source du serveur, « "*calculate a point 2/3rds of the way through the validity period (or halfway through, for short-lived certs)*" ».)

Le RFC précise qu'un membre `explanationURL` peut être ajouté mais Let's Encrypt ne le fait pas. Les membres possibles figurent dans un nouveau registre IANA <https://www.iana.org/assignments/acme/acme.xml#acme-renewal-info-object-fields>, auquel on pourra ajouter de nouveaux membres, en fournissant une spécification (« Spécification nécessaire » du RFC 8126).

Et on utilise cet intervalle entre deux dates comment ? Le RFC recommande de choisir une date au hasard dans l'intervalle. Le client ACME ne doit pas dormir jusqu'à la date sélectionnée, il doit réessayer de temps en temps car l'AC a pu changer la date (c'est tout le principe d'ARI). Mais attention à respecter le `Retry-After:` (six heures, ici), cf. RFC 9110, section 10.2.3.

Notre RFC ajoute également un membre à la description d'une commande de certificat : `replaces` indique quel certificat on est censé remplacer. (En utilisant le même identificateur de certificat qu'indiqué plus haut.) Il a été ajouté au registre IANA <https://www.iana.org/assignments/acme/acme.xml#acme-order-object-fields>.

ARI est mis en œuvre dans le serveur ACME de Let's Encrypt, Boulder <https://github.com/letsencrypt/boulder>. Regardez `core/objects.go` et notamment la fonction `RenewalInfoSimple`. Côté client, Lego <https://go-acme.github.io/lego/>, acmez <https://github.com/mholt/acmez/> et le CertMagic <https://pkg.go.dev/github.com/caddyserver/certmagic> de Caddy <https://caddyserver.com/> ont ARI mais certbot <https://certbot.eff.org/> ou dehydrated <https://github.com/dehydrated-io/dehydrated> ne gèrent pas ARI. Si vous voulez vous y mettre, Let's Encrypt a écrit un guide d'intégration d'ARI <https://letsencrypt.org/2024/04/25/guide-to-integrating-ari-into-existing-acme-clients/>.