Surveillez les dates d'expiration de vos certificats X.509 !

Stéphane Bortzmeyer
<stephane+blog@bortzmeyer.org>

Première rédaction de cet article le 9 décembre 2012. Dernière mise à jour le 14 septembre 2017

https://www.bortzmeyer.org/tester-expiration-certifs.html

Un certain nombre de services sur l'Internet, notamment de serveurs Web, sont protégés par le protocole TLS, qui utilise presque toujours un certificat X.509 pour vérifier l'authenticité du serveur. Ces certificats ont une durée de vie limitée et doivent donc être renouvelés de temps en temps. Comme un certain nombre d'administrateurs ont oublié, au moins une fois, un tel renouvellement, il me semble utile de donner ce conseil : intégrez la surveillance des dates d'expiration dans votre système de supervision !

Pourquoi est-ce que les certificats expirent ? Les mauvaises langues vous diront que c'est parce que les AC veulent gagner davantage d'argent. Mais il y a quand même une bonne raison technique : comme le mécanisme de révocation de X.509 marche très mal <https://www.imperialviolet.org/2011/03/18/revocation.html>, la durée limitée du certificat est la seule vraie protection contre un vol de la clé privée : au moins, le voleur ne pourra pas jouir du fruit de son forfait éternellement.

Les certificats ont donc une date d'expiration, que les navigateurs Web peuvent montrer (dans les informations d'une connexion HTTPS), ici avec deux navigateurs différents : Si on préfère la ligne de commande, OpenSSL permet d'afficher l'information :

```
% openssl  s_client -servername www.dns-oarc.net -connect www.dns-oarc.net:443 < /dev/null > /tmp/oarc.pem
% openssl x509 -enddate -noout -in /tmp/oarc.pem
notAfter=Aug 27 15:28:19 2014 GMT
```

Ici, on voit que le certificat de l'OARC <https://www.dns-oarc.net/> n'expirera pas avant un an et demi.

Les clients TLS vérifient cette date et refusent normalement la connexion avec un serveur dont le certificat a expiré. Merci à François pour m'avoir signalé l'expiration de admin.lautre.net qui me permet de mettre de vrais exemples, ici avec Chrome :

Comment éviter ce sort tragique ? Le mieux serait que l'AC prévienne ses clients lorsque le certificat va bientôt expirer. CAcert <https://www.bortzmeyer.org/cacert.html> le fait, par exemple, mais toutes les AC ne le font pas. Bref, cela vaut la peine de tester l'approche de l'expiration, depuis son système de supervision habituel. On sera ainsi prévenu **avant** le problème.

Si on utilise Nagios, ou un de ses compatibles, comme Icinga, les programmes de tests standards savent en général tester. Par exemple, check_http <https://www.monitoring-plugins.org/doc/man/check_http.html> a l'option `-C` qui prend deux paramètres, un obligatoire qui précise le nombre de jours où le certificat doit être valide avant que le logiciel de supervision ne lance un avertissement et un facultatif qui donne le nombre de jours où le certificat doit être valide avant que ce logiciel ne lance une alarme critique (par défaut, c'est lorsque le certificat a expiré que cette alarme se déclenche). Voici des exemples avec le certificat de l'OARC qui a encore 625 jours à vivre :

```
% /usr/share/nagios/libexec/check_http -H www.dns-oarc.net --ssl=1 -C 700,100
WARNING - Certificate '*.dns-oarc.net' expires in 625 day(s) (08/27/2014 15:28).
```

Ici, on a un avertissement ("*WARNING*") car on a demandé que le certificat ait encore 700 jours (une valeur irréaliste, mais c'est pour faire une démo de ce programme de test). Avec des valeurs plus raisonnables, check_http nous dit évidemment que tout va bien :

```
% /usr/share/nagios/libexec/check_http -H www.dns-oarc.net --ssl=1 -C 15,7
OK - Certificate '*.dns-oarc.net' will expire on 08/27/2014 15:28.
```

Par contre, avec le certificat expiré vu plus haut :

```
% /usr/share/nagios/libexec/check_http -H admin.lautre.net --ssl=1 -C 15,7
CRITICAL - Certificate '*.lautre.net' expired on 12/08/2012 23:59.
```

On peut alors mettre ce test dans sa configuration Nagios, Icinga, etc :

```
define service{
use          generic-service
host_name      my-web
service_description HTTP
check_command      check_http!-H www.example.net --ssl -C 15,7
}
```

Ou, avec un Icinga 2, plus récent :

```
vars.http_vhosts["web-cert"] = {
    http_uri = "/"
    http_vhost = "www.example.net"
    http_ssl = true
    http_sni = true
    http_certificate = "15,7"
  }
```

On sera désormais prévenu par les mécanismes habituels de son logiciel de supervision. Notez bien que d'autres programmes de test de Nagios ont une telle option, mais pas toujours la même (c'est `-D` pour check_imap <https://www.monitoring-plugins.org/doc/man/check_imap.html>, par exemple).

Attention, toutefois. La documentation de check_http <https://www.monitoring-plugins.org/doc/man/check_http.html> le dit, mais de manière peu claire (« "*when this option is used the URL is not checked*" ») : dès qu'on active les tests du certificat, d'autres tests sont coupés, comme celui du code de retour HTTP (200, 401, 403, etc) ou comme celui de la présence d'une chaîne de caractères dans le contenu (option `http_string`). Il faut donc la plupart du temps faire deux tests, un avec l'option `http_certificate` et l'autre sans.

Et si on n'utilise pas Nagios ou un compatible et qu'on veut faire un programme de test ? Il existe évidemment plein de programmes pour cela.

Le plus souvent cité est ssl-cert-check <http://prefetch.net/articles/checkcertificate.html>, qui s'appuie sur OpenSSL. On le télécharge et on l'utilise ainsi (merci à Nicolas Legrand pour ce serveur, qui a une date d'expiration proche) :

```
% ssl-cert-check -s  dio.obspm.fr -p 443

Host                                           Status       Expires      Days
---------------------------------------------- ------------ ------------ ----
dio.obspm.fr:443                               Valid        Feb 4 2013   57
```

Si le certificat a expiré, on a :

```
% ssl-cert-check  -s admin.lautre.net -p 443

Host                                           Status       Expires      Days
---------------------------------------------- ------------ ------------ ----
admin.lautre.net:443                           Expired      Dec 8 2012   -1
```

Il dispose de plusieurs options, par exemple pour indiquer quelle marge de manœuvre on accepte :

```
% ssl-cert-check -s  dio.obspm.fr -p 443 -x 60

Host                                           Status       Expires      Days
---------------------------------------------- ------------ ------------ ----
dio.obspm.fr:443                               Expiring     Feb 4 2013   57
```

Ses deux principaux inconvénients sont qu'il ne définit pas le code de retour (ce qui le rend difficile à intégrer dans une solution de supervision automatisée, sauf à utiliser son option `-n`, conçue pour Nagios) et qu'il ne semble pas connaître le SNI ("*Server Name Indication*", RFC 6066 [1]), ce qui est pénible pour le cas où on a plusieurs serveurs derrière la même adresse IP <https://www.bortzmeyer.org/auth-x509-plusieurs-noms.html>.

1. Pour voir le RFC de numéro NNN, `https://www.ietf.org/rfc/rfcNNN.txt`, par exemple `https://www.ietf.org/rfc/rfc6066.txt`

Autre solution, un programme écrit par Kim-Minh Kaplan, (en ligne sur `https://www.bortzmeyer.org/files/check-tls.sh`). Il est conçu pour tester un groupe de serveurs (mis dans un fichier préalablement).

À noter que le programme sslscan <`http://sourceforge.net/projects/sslscan/`> est très utile pour faire un audit de son serveur mais ne semble pas adapté à la supervision. (Autre possibilité, mais je n'ai pas testé, SSLyze <`https://github.com/nabla-c0d3/sslyze`>.)

Patrick Mevzek me fait remarquer qu'il faudrait, en toute rigueur, tester la date d'expiration, non seulement du certificat final, celui du serveur, mais aussi celle de tous les certificats intermédiaires, qui peuvent aussi expirer. J'ignore si les logiciels ci-dessus pensent à cela ((en ligne sur `https://www.bortzmeyer.org/files/check-tls.sh`) ne le fait pas). Les certificats intermédiaires sont ceux de professionnels de la sécurité et, normalement, ils ont moins souvent des problèmes que les certificats finaux. Mais il peut arriver en effet qu'un serveur stocke des certificats intermédiaires qui ont été mis à jour par l'AC mais que l'administrateur du serveur a oublié de changer.

Bref, avec tous ces outils, l'administrateur système qui laisse un certificat expirer n'a vraiment aucune excuse.