Drown, et quelques remarques sur la sécurité

Stéphane Bortzmeyer
<stephane+blog@bortzmeyer.org>

Première rédaction de cet article le 1 mars 2016. Dernière mise à jour le 5 mars 2016

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

Aujourd'hui a été publiée la faille de sécurité Drown <https://drownattack.com/>, touchant notamment la bibliothèque cryptographique OpenSSL. C'est l'occasion de se livrer à quelques réflexions sur Drown, TLS et la sécurité.

Le même jour, huit (!) failles de sécurité dans OpenSSL ont été publiées <https://www.openssl.org/news/secadv/20160301.txt> (Drown a été éclaté en deux failles, CVE-2016-0800 et CVE-2016-0703). Cela ne facilite pas le travail de ceux qui essaient de démêler tout cela mais Drown <https://drownattack.com/> est très bien documenté sur son site officiel (très complet, avec logo et tout). Je recommande également le papier officiel <https://drownattack.com/drown-attack-paper.pdf>, très clair. Il n'y a normalement donc pas besoin de ré-expliquer ce qu'est Drown.

Pourtant, je suis étonné de constater que certaines choses n'ont pas été comprises. Par exemple bien des gens croient que, puisque leur serveur HTTPS n'accepte pas SSLv2, ils sont en sécurité. Rien n'est plus faux et c'est bien expliqué dans la FAQ de Drown : pour effectuer cette attaque, il suffit que la clé du site visé soit disponible sur un autre site, qui, lui, accepte SSLv2. Et, c'est là un point qui m'a personnellement surpris, ce cas est assez fréquent. Comme le note l'article Drown, le modèle de financement des AC encourage à acheter le moins de certificats possibles, et donc à les utiliser sur plusieurs serveurs, ou bien sur le serveur de production et sur celui de développement.

Prenons un exemple : le serveur du journal quotidien Ouest-France n'est apparemment pas vulnérable, il ne gère pas SSLv2 :

```
%  openssl s_client -connect www.ouest-france.fr:443 -ssl2
CONNECTED(00000003)
write:errno=104
---
no peer certificate available
---
No client certificate CA names sent
---
SSL handshake has read 0 bytes and written 48 bytes
```

```
---
New, (NONE), Cipher is (NONE)
Secure Renegotiation IS NOT supported
Compression: NONE
Expansion: NONE
No ALPN negotiated
SSL-Session:
    Protocol  : SSLv2
    Cipher    : 0000
...
```

Mais on trouve, sur un tout autre réseau, une machine `93.17.51.145` qui, elle, acceptait (elle a été réparée depuis) SSLv2 et a un certificat `*.ouest-france.fr` (les certificats avec joker sont évidemment les plus vulnérables à Drown puisqu'on les achète justement pour les mettre sur plusieurs machines) :

```
%  openssl s_client -connect 93.17.51.145:443 -ssl2
CONNECTED(00000003)
...
subject=/C=FR/postalCode=35000/ST=Ille-et-Vilaine/L=RENNES/street=10 rue Du Breil/street=ZI Rennes Sud-Est/C
...
SSL handshake has read 1609 bytes and written 367 bytes
---
New, SSLv2, Cipher is RC2-CBC-MD5
Server public key is 2048 bit
Secure Renegotiation IS NOT supported
Compression: NONE
Expansion: NONE
No ALPN negotiated
SSL-Session:
    Protocol  : SSLv2
    Cipher    : RC2-CBC-MD5
    Session-ID: 6F1500003AE7F4EF421B7C964089D28B
```

Le journal Ouest-France est hébergé par SDV <http://www.sdv.fr/> et cette `93.17.51.145` est apparemment dans les locaux du journal. Machine de développement ? Serveur utilisé pour des applications internes ? En tout cas, contrairement au site Web public, cette machine ne semble pas avoir fait l'objet de tests de sécurité et, à elle seule, elle rendait l'attaque Drown possible. En effet, il s'agit bien de la même clé (pas uniquement du même nom dans le certificat) :

```
% openssl s_client -connect 93.17.51.145:443 | openssl x509 -pubkey -noout
...
-----BEGIN PUBLIC KEY-----
MIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEAu83rhWk01bad5z55tfKN
TlqWSdXou/QXTNVsk0J47FFEYCvumcd5TgrLtsRDjz23K1+8T9KXgIzNN4wXwWUc
7O1Hu0ddX4w0OOiSaq2rxPcXdzeF98Mth+MtclUliYZTPa94TXEMSM/xPeELkfmi
sAqJjVFfGqacFcuzI0QryQPkAAU4ts0JtAoaUhHDNc+3Mz2UnHtaYsLEh9cHdkKq
QzpggPPN4emqcEr3fbvTPl0JIYossH/mOI8E4NTyVaZaScimi8efklxF3JEc+/6w
AqzGXt8CL2/ce+E4BojHCfilGrxU4WEeo7e9sMXHuhWCQ/fxT/6o3yktI+sLSv61
RQIDAQAB
-----END PUBLIC KEY-----

% openssl s_client -connect www.ouest-france.fr:443 | openssl x509 -pubkey -noout
-----BEGIN PUBLIC KEY-----
MIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEAu83rhWk01bad5z55tfKN
TlqWSdXou/QXTNVsk0J47FFEYCvumcd5TgrLtsRDjz23K1+8T9KXgIzNN4wXwWUc
7O1Hu0ddX4w0OOiSaq2rxPcXdzeF98Mth+MtclUliYZTPa94TXEMSM/xPeELkfmi
sAqJjVFfGqacFcuzI0QryQPkAAU4ts0JtAoaUhHDNc+3Mz2UnHtaYsLEh9cHdkKq
QzpggPPN4emqcEr3fbvTPl0JIYossH/mOI8E4NTyVaZaScimi8efklxF3JEc+/6w
AqzGXt8CL2/ce+E4BojHCfilGrxU4WEeo7e9sMXHuhWCQ/fxT/6o3yktI+sLSv61
RQIDAQAB
-----END PUBLIC KEY-----
```

On trouvait d'ailleurs cette clé sur plein d'autres machines du réseau `93.17.51.128/26`, presque toutes acceptant SSLv2 d'une façon ou d'une autre.

En revanche, si on regarde un autre domaine signalé sur le site Web de Drown, `blockchain.com`, on voit que le site Web public n'a pas du tout la même clé (ce n'est même pas le même algorithme) que la machine vulnérable (alors que le nom dans le certificat de la machine vulnérable est bien `*.blockchain.com`). Leur situation est donc moins grave (attention toutefois aux attaques actives) :

```
# Machine vulnérable car acceptant SSLv2 dans certaines conditions
% openssl s_client  -connect  50.87.196.92:443 | openssl x509 -pubkey -noout
-----BEGIN PUBLIC KEY-----
MIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEAsXMk8xOJaaqKmz88RicY
Mt4e4VlY0uTVaUS7mbgnUGgd15wQ/tLH91RL5S9c1C1FFrcjBpomODJXBdgTXjWi
x4OVRl0sEe6OjIX1FLqBl15n+Lx++933lQUWinMSInEVLnBBjcr9If5+ozJvscRS
eF/hikjhDYW/rDwF1f7ivTiiok8f12zTa2lhlxOH4KxcFPaFklXDa45DccDjdCIZ
uVLpFSngo5iS3nney+LPK4afYYffEWK9aN2Rj9NE8zRm3hGoGDva2vNmKGbewd3w
1huqr3o7fc3WZyW9HZlM7PjZF6YLhgJnAI09XXDHD3t+G4ijcS6F4Rh9B36whuoo
CQIDAQAB
-----END PUBLIC KEY-----

# Serveur Web public
% openssl s_client  -connect  www.blockchain.com:443 | openssl x509 -pubkey -noout
-----BEGIN PUBLIC KEY-----
MFkwEwYHKoZIzj0CAQYIKoZIzj0DAQcDQgAE8DGz/3DLSflRPxO5uHuiww7EbqEP
/S3VWfdw4PGs4WcuqWTtIGht1rwiuTjrU3Es1uxm3T+JfT1JOOhYut8IjA==
-----END PUBLIC KEY-----
```

Pour résumer ce point essentiel « **Même si votre serveur n'accepte pas SSLv2, vous êtes peut-être quand même vulnérable** ».

(Au passage, petite note technique : certains systèmes d'exploitation, et à juste titre, compilent OpenSSL sans SSLv2. La commande ci-dessus vous dira alors « "*unknown option -ssl2*" ». Il faut utiliser un système non sécurisé, comme Arch Linux, ou bien prendre un OpenSSL qu'on a compilé soi-même. C'est à cause de ce problème que l'excellent script `testssl.sh` vient avec son propre OpenSSL.)

On voit d'ailleurs un effet négatif de la "*gamification*" popularisée, dans le monde HTTPS, par SSLlabs <`https://www.ssllabs.com/ssltest/`>. Pendant que tout le monde s'amuse à obtenir la meilleure note possible pour s'en vanter sur les rézosocios (« j'ai un A sur SSLlabs alors que la banque Machin a uniquement un B, je suis trop un geek crypto-expert »), des machines comme ce pauvre serveur restent ignorés.

Autre question posée par Drown, pourquoi est-ce qu'il y a autant de machines qui acceptent encore SSLv2 ? Il a été officiellement abandonné en 2011 par le RFC 6176[1]. Et le RFC est sorti bien tard, tous les experts savaient depuis longtemps que SSLv2 était cassé sans espoir de réparation. Mais les mises à jour ne se font, sur l'Internet, qu'à un rythme glacial (voir nul). Le problème n'est pas technique, il vient du fait que les mises à jour ne rapportent rien, n'ont pas de retour sur investissement, et ne sont donc en général jamais faites. Tout le monde s'indigne lorsqu'une faille comme Drown est publiée mais, en temps normal, personne ne s'en préoccupe. La prévention est toujours le parent pauvre de la sécurité. L'opérationnel et la sécurité concrète n'intéressent personne.

1. Pour voir le RFC de numéro NNN, `https://www.ietf.org/rfc/rfcNNN.txt`, par exemple `https://www.ietf.org/rfc/rfc6176.txt`

Compte-tenu de cela, est-ce que Drown représente un réel danger aujourd'hui, et faut-il patcher d'urgence ? Bien sûr, il faut patcher, c'est le B. A. BA de la sécurité pour une machine connectée à l'Internet. Mais le danger exact de Drown n'est pas facile à évaluer. On est dans le cas classique en sécurité du verre que l'optimiste voit à moitié plein alors que le pessimiste le voit à moitié vide. L'optimiste va faire remarquer que l'exploitation effective de Drown est très difficile, dans les conditions du monde réel (en laboratoire, ça marche toujours mieux). Le pessimiste rétorquera que les attaques ne vont toujours qu'en s'améliorant. Si Drown est difficile à exploiter aujourd'hui, cela ne durera pas éternellement. À noter que, pour la raison expliquée plus haut, cela ne sert à rien de patcher votre serveur principal : il faut patcher **toutes** les machines qui ont le certificat du serveur principal.

Je suggère d'ailleurs d'en tirer une leçon : ne jamais partager un certificat (et donc une clé privée) entre des machines qui ont des administrateurs différents : le risque est en effet très élevé que l'une d'elles soit moins sécurisée que les autres, permettant une attaque.

Ah, notez aussi que patcher OpenSSL ne servira que pour le futur : si un attaquant a utilisé Drown contre vous, et a réussi à obtenir des informations tels que vos mots de passe, il serait prudent de les changer...

Merci à Léo pour avoir signalé une grosse approximation dans l'article original.

---

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