

On peut générer de vrais/faux certificats

Stéphane Bortzmeyer

<stephane+blog@bortzmeyer.org>

Première rédaction de cet article le 22 octobre 2026

<https://www.bortzmeyer.org/emudhra.html>

Vous avez sans doute déjà lu des articles sur la faille de l'AC eMudhra (sinon, vous avez des liens à la fin). Une des conséquences de cette faille est que tout le monde peut générer des vrais/faux certificats. Exemple.

La faille (depuis éclipsée par deux grosses failles PHP) venait d'un générateur de nombres aléatoire trop faibles. Un certificat est une clé cryptographique publique signée par une AC. La clé privée correspondant à cette clé publique doit être choisie de manière à être imprévisible pour l'attaquant. (On dit souvent qu'elle doit être choisie aléatoirement ce qui, en pratique, est juste un peu plus fort que de dire « imprévisible ».) Mais un générateur imprévisible est difficile à faire sur une machine déterministe comme un ordinateur. Et les logiciels ont des bogues. Une bogue frappait apparemment l'AC eMudhra : les clés générées ne faisaient qu'un octet (le reste étant toujours nul), ce qui ne faisait que 256 clés possibles, rendant triviale une énumération de toutes les clés possibles. L'une d'elles était juste le chiffre 4 <<https://xkcd.com/221/>>, que vous trouverez souvent dans les articles sur cette faille.

Le fonctionnement du système X.509 étant ce qu'il est, si on peut générer les clés d'une AC qui est largement présente dans les magasins d'AC de plusieurs systèmes d'exploitation / logiciels / langages, on peut se faire passer pour n'importe qui.

Démonstration avec le service `emudhra.insecureoldshit.com:443`. Il n'a pas de lien avec l'AC eMudhra mais, comme n'importe qui peut générer les clés « privées » d'eMudhra, il peut se faire passer pour elle. Ne cherchez pas forcément à vous connecter en HTTPS, c'est déjà intéressant avec juste du TLS nu. Essayons avec GnuTLS :

```
% gnutls-cli emudhra.insecureoldshit.com:443
Processed 152 CA certificate(s).
Resolving 'emudhra.insecureoldshit.com:443'...
Connecting to '2.29.115.248:443'...
- Certificate type: X.509
- Got a certificate list of 3 certificates.
- Certificate[0] info:
  - subject 'CN=192.134.4.66:6621,OU=You,O=Penguin Association,C=AQ', issuer 'CN=EM OV TLS CA - G3C,O=eMudhra Inc
```

```

Public Key ID:
sha1:779bc70e16304e65523397a5e1ccdc14cb673c39
sha256:31ae0e5368eb9f27eb69532a7a1c3d02530544138675a4738808ef3c60f028a7
Public Key PIN:
pin-sha256:Ma40U2jrnyfraVMqehw9AlMFRBOGdaRziAjvPGDwKKc=

- Certificate[1] info:
- subject 'CN=EM OV TLS CA - G3C,O=eMudhra Inc,C=US', issuer 'CN=emSign Root TLS CA - G3,O=eMudhra Technol
- Certificate[2] info:
- subject 'CN=emSign Root TLS CA - G3,O=eMudhra Technologies Limited,C=IN', issuer 'CN=emSign ECC Root CA -
- Status: The certificate is trusted.
- Description: (TLS1.3-X.509)-(ECDHE-X25519)-(ECDSA-SECP256R1-SHA256)-(AES-128-GCM)
- Session ID: 8A:5F:E1:E3:6A:6D:00:01:5E:09:1E:78:5B:79:A8:BB:09:31:29:7F:9F:6B:C4:F3:6E:E0:72:F4:8E:45:7C:D
- Options:
- Handshake was completed

```

Notons plusieurs choses :

- Le certificat du client (celui de numéro 0) a comme titulaire (« *"subject"* ») CN=192.134.4.66:6621,OU=You, Association,C=AQ. À la syntaxe du RFC 4514¹ (qui normalise la forme texte du modèle de données de X.500), cela veut dire que le nom est 192.134.4.66 (l'adresse IP du client du service, ici moi), l'organisation est la fictive « *"Penguin Association"* » et le pays est l'Antarctique. (Rappelez-vous que c'est un vrai/faux certificat, généré lors de ma connexion.)
- Le certificat final est signé par l'AC eMudhra (CN=EM OV TLS CA - G3C,O=eMudhra Inc,C=US)...En fait, c'est une vraie/fausse signature, faite en récupérant une des clés privées de eMudhra, grâce à l'énumération des clés possibles.
- Le certificat a été accepté sur cette machine (une Debian, où le certificat d'eMudhra était dans le magasin du système) : « *"The certificate is trusted."* ».

Si maintenant je mets à jour mon système (les certificats des AC sont dans le paquetage Debian `ca-certificates`) :

```

% gnutls-cli emudhra.insecureoldshit.com:443
Processed 150 CA certificate(s).
Resolving 'emudhra.insecureoldshit.com:443'...
Connecting to '2.29.115.248:443'...
- Certificate type: X.509
- Got a certificate list of 3 certificates.
- Certificate[0] info:
- subject 'CN=192.134.4.66:8801,OU=You,O=Penguin Association,C=AQ', issuer 'CN=EM OV TLS CA - G3C,O=eMudhra
Public Key ID:
sha1:779bc70e16304e65523397a5e1ccdc14cb673c39
sha256:31ae0e5368eb9f27eb69532a7a1c3d02530544138675a4738808ef3c60f028a7
Public Key PIN:
pin-sha256:Ma40U2jrnyfraVMqehw9AlMFRBOGdaRziAjvPGDwKKc=

- Certificate[1] info:
- subject 'CN=EM OV TLS CA - G3C,O=eMudhra Inc,C=US', issuer 'CN=emSign Root TLS CA - G3,O=eMudhra Technol
- Certificate[2] info:
- subject 'CN=emSign Root TLS CA - G3,O=eMudhra Technologies Limited,C=IN', issuer 'CN=emSign ECC Root CA -
- Status: The certificate is NOT trusted. The certificate issuer is unknown.
*** PKI verification of server certificate failed...
*** Fatal error: Error in the certificate.

```

Vous noterez :

- Deux certificats de moins.
- Cette fois, le vrai/faux certificat est refusé, car le certificat d'eMudhra n'est plus dans le magasin.

1. Pour voir le RFC de numéro NNN, <https://www.ietf.org/rfc/rfcNNN.txt>, par exemple <https://www.ietf.org/rfc/rfc4514.txt>

Le certificat généré est disponible ici (en ligne sur <https://www.bortzmeyer.org/files/emudhra-false-cert.pem>) et voici son analyse par OpenSSL. Notez les dates de validité :

```
% openssl x509 -text -in ../files/emudhra-false-cert.pem
Certificate:
  Data:
    Version: 3 (0x2)
    Serial Number: 0 (0x0)
    Signature Algorithm: ecdsa-with-SHA384
    Issuer: C=US, O=eMudhra Inc, CN=EM OV TLS CA - G3C
    Validity
      Not Before: Oct 22 13:21:32 2026 GMT
      Not After : Oct 22 13:22:02 2026 GMT
    Subject: C=AQ, O=Penguin Association, OU=You, CN=192.134.4.66:63823
    Subject Public Key Info:
      Public Key Algorithm: id-ecPublicKey
      Public-Key: (256 bit)
      pub:
        04:95:61:11:86:61:1c:97:dd:0f:b5:e3:75:9b:07:
        ...
      ASN1 OID: prime256v1
      NIST CURVE: P-256
    X509v3 extensions:
      X509v3 Authority Key Identifier:
        34:EA:EA:0C:EB:C9:0A:DA:F9:D0:C5:E2:ED:9B:78:C9:87:9E:1B:6C
      X509v3 Subject Alternative Name:
        DNS:example.com, DNS:emudhra.insecureoldshit.com, DNS:*.emudhra.insecureoldshit.com, IP Address:
    Signature Algorithm: ecdsa-with-SHA384
    Signature Value:
      30:66:02:31:00:bd:8a:fa:91:fc:cd:c0:af:9f:85:0b:13:16:
      ...
-----BEGIN CERTIFICATE-----
MIICIZCCAaigAwIBAgIBADAKBggqhkJOPQQDAzBAMQswCQYDVQQGEwJVUzEUMBIG
...
```

Si vous tenez à faire du HTTP avec ce service, allez-y, il y a un gag.

D'autres ont utilisé cette possibilité pour permettre de générer de vrais/faux certificats, par exemple **mitmproxy** <<https://www.mitmproxy.org/>> avec ce code <<https://github.com/mitmproxy/mitmproxy/commit/169234499088fc2c414644a67efaed3d64ae2e3b>>.

Quelques lectures, si vous voulez apprendre ou approfondir :

- En français, un bon article de Next <<https://next.ink/262991/certificats-tls-nimporte-qui-pouva>>.
- Et en anglais, le bon article d'Ars Technica <<https://arstechnica.com/security/2026/10/the-internet-goes-on-high-alert-after-discovery-of-8-compromised-tls-certs/>>.
- L'article original <<https://emudhra.we-have-a-problem.com/>> (en anglais), qui n'indiquait pas encore la raison (la défaillance du générateur de nombres aléatoires).
- L'article technique de Scott Helme <<https://scotthelme.co.uk/eight-bits-of-trust-emudhras-ca-k>>.
- Le rapport de bogue chez Mozilla <https://bugzilla.mozilla.org/show_bug.cgi?id=2083449> (très détaillé, en anglais).
- Le communiqué officiel de l'AC <<https://emudhra.com/en/press-release/emudhra-statement-distr>>.

Plusieurs mathématiciens en ont profité pour faire remarquer que, contrairement à une simplification courante, 4 n'est pas « moins aléatoire » que, par exemple, 1012818637039180068205. Les deux ont la même probabilité d'être sélectionnés. Le problème vient du fait que les clés ne faisaient qu'un octet, pas spécialement du fait qu'elle avait la valeur 4.

<https://www.bortzmeyer.org/emudhra.html>