OpenSSL - rsautl

Utilitaire rsa

OPTIONS

-in filename Fichier d'entrée

-out filename Fichier de sortie

-inkey file Fichier de clé privée RSA en entrée

-pubin Le fichier d'entrée est une clé publique RSA

-certin L'entrée est un certificat contenant une clé publique RSA

-sign Signe la donnée en entrée et sort le résultat signé. requière une clé privée RSA

-verify Vérifie les données en entrée et sort les données récupérées

-encrypt Chiffre l'entrée en utilisant la clé publique RSA

-decrypt Déchiffre l'entrée en utilisant la clé privée RSA

-pkcs, -oeap, -ssl, -raw Le padding à utiliser (respectivement PKCS#1 v1.5, PKCS#1 OEAP, padding utilisé dans ssl v2, aucun padding). pour les signatures, seul -pkcs et -raw sont utilisé

-hexdump Dump en hexa les données en sortie

-asn1parse asn1parse la sortie. utile avec -verify

Notes

rsautl utilise directement l'algorithme RSA, il ne peut être utlisé que pour de petites portions de données.

Exemples

Signer des données en utilisant une clé privée :
openssl rsautl -sign -in file -inkey key.pem -out sig
Récupérer les données signées :
openssl rsautl -verify -in sig -inkey key.pem
Examiner les données signées en raw :
openssl rsautl -verify -in file -inkey key.pem -raw -hexdump

```
0000 - 00 01 ff ff ff ff ff ff-ff ff ff ff ff ff ff ff ................
0010 - ff ff ff ff ff ff ff ff-ff ff ff ff ff ff ff ff ................
0020 - ff ff ff ff ff ff ff ff-ff ff ff ff ff ff ff ff ................
0030 - ff ff ff ff ff ff ff ff-ff ff ff ff ff ff ff ff ................
0040 - ff ff ff ff ff ff ff ff-ff ff ff ff ff ff ff ff ................
0050 - ff ff ff ff ff ff ff ff-ff ff ff ff ff ff ff ff ................
0060 - ff ff ff ff ff ff ff ff-ff ff ff ff ff ff ff ff ................
0070 - ff ff ff ff 00 68 65 6c-6c 6f 20 77 6f 72 6c 64 .....hello world
```

Ce block est formaté PKCS#1. çà été fait en utilisant encrypt puis en déchiffrant le block detype 2 (le 2eme octet) et un padding aléatoire visible à la place des octets 0xff. Il est possible d'analyser la signature des certificats en utilisant rsautl avec asn1parse.

En considérant un exemple auto-signé dans certs/pca-cert.pem :
openssl asn1parse -in pca-cert.pem

```
0:d=0 hl=4 l= 742 cons : SEQUENCE
4:d=1 hl=4 l= 591 cons : SEQUENCE
8:d=2 hl=2 l= 3 cons : cont [ 0 ]
10:d=3 hl=2 l= 1 prim : INTEGER :02
13:d=2 hl=2 l= 1 prim : INTEGER :00
16:d=2 hl=2 l= 13 cons : SEQUENCE
18:d=3 hl=2 l= 9 prim : OBJECT :md5WithRSAEncryption
29:d=3 hl=2 l= 0 prim : NULL
31:d=2 hl=2 l= 92 cons : SEQUENCE
33:d=3 hl=2 l= 11 cons : SET
35:d=4 hl=2 l= 9 cons : SEQUENCE
37:d=5 hl=2 l= 3 prim : OBJECT :countryName
42:d=5 hl=2 l= 2 prim : PRINTABLESTRING :AU
....
599:d=1 hl=2 l= 13 cons : SEQUENCE
601:d=2 hl=2 l= 9 prim : OBJECT :md5WithRSAEncryption
612:d=2 hl=2 l= 0 prim : NULL
614:d=1 hl=3 l= 129 prim : BIT STRING
```

Le BIT STRING final contient la signature actuelle, et peut être extrait avec :
openssl asn1parse -in pca-cert.pem -out sig -noout -strparse 614
La clé publique du certificat peut être extrait avec :
openssl x509 -in test/testx509.pem -pubkey -noout >pubkey.pem
La signature peut être analysée avec :
openssl rsautl -in sig -verify -asn1parse -inkey pubkey.pem -pubin

```
0:d=0 hl=2 l= 32 cons : SEQUENCE
2:d=1 hl=2 l= 12 cons : SEQUENCE
4:d=2 hl=2 l= 8 prim : OBJECT :md5
14:d=2 hl=2 l= 0 prim : NULL
16:d=1 hl=2 l= 16 prim : OCTET STRING
0000 - f3 46 9e aa 1a 4a 73 c9-37 ea 93 00 48 25 08 b5 .F...Js.7...H%..
```

C'est la version parsée d'une structure ASN1 DigestInfo, ici avec un digest md5.
La partie du certificat qui a été signée peut être extraite avec :
openssl asn1parse -in pca-cert.pem -out tbs -noout -strparse 4
et son digest calculé avec :
openssl md5 -c tbs
MD5(tbs)= f3 :46 :9e :aa :1a :4a :73 :c9 :37 :ea :93 :00 :48 :25 :08 :b5
Qui peut être récupéré avec cette valeur.