OpenSSL - x509v3_config

Format de configuration des extensions de certificat X509 v3

Le format est le suivant :

extension_name=[critical,] extension_options

Si critical est présent, l'extension est critique. Il y'a 4 types d'extension : chaînes, multi-valué, raw, et arbitraire.

exemple chaîne :
nsComment="This is a Comment"
exemple multi-valué :
basicConstraints=critical,CA :true,pathlen :1
variante d'un multivalué :
basicConstraints=critical,@bs_section
[bs_section]
CA=true
pathlen=1

Extensions standard

Contraintes de base Extension multi-valué qui indique si un certificat est un certificat de CA

```
basicConstraints=CA:TRUE
basicConstraints=CA:FALSE
basicConstraints=critical,CA:TRUE, pathlen:0
```

Un certificat de CA doit avoir **basicContraints** à TRUE et optionnellement un **pathlen** qui indique le nombre maximum de CA qui peut apparaitre sous celle-ci dans une chaîne. A 0, peut seulement signer des certificats finaux.

Utilisation de clé Extension multi-valué consistant d'une liste de noms d'utilisation de clé permis Les noms supportés sont : **digitalSignature, nonRepudiation, keyEncipherment, dataEncipherment, keyAgreement, keyCertSign, cRLSign, encipherOnly et decipherOnly**.

exemple :

```
keyUsage=digitalSignature, nonRepudiation
keyUsage=critical, keyCertSign
```

Utilisation de clé étendue Consiste en une liste d'utilisation indiquant l'utilisation de la clé publique. Peut être un nom court, ou un OID :

Value Meaning

— ——-

serverAuth SSL/TLS Web Server Authentication.

clientAuth SSL/TLS Web Client Authentication.

codeSigning Code signing.

emailProtection E-mail Protection (S/MIME).

timeStamping Trusted Timestamping

msCodeInd Microsoft Individual Code Signing (authenticode)

msCodeCom Microsoft Commercial Code Signing (authenticode)

msCTLSign Microsoft Trust List Signing

msSGC Microsoft Server Gated Crypto

msEFS Microsoft Encrypted File System

nsSGC Netscape Server Gated Crypto

exemple :

```
extendedKeyUsage=critical,codeSigning,1.2.3.4
extendedKeyUsage=nsSGC,msSGC
```

Identifiant de clé du sujet Extention de type chaîne qui prend 2 valeurs : hash qui suit la rfc3280 ou une chaîne hexa donnant la valeur de l'extension à inclure

exemple :

```
subjectKeyIdentifier=hash
```

Identifiant de clé d'autorité Permet 2 options, keyid et issuer et peuvent prendre optionnellement la valeur always

Avec **keyid**, tente de copier le subject key identifier depuis le certificat parent. Avec always, une erreur est retourné si l'option échoue.

avec **issuer**, copie l'issuer et le numéro de série de l'issuer

exemple :

```
authorityKeyIdentifier=keyid,issuer
```

Subject Alternative Name L'extention subjectAltName permet d'inclure divers valeurs : email, URI, DNS, RID (ID :Object Identitifer), IP, dirName (un DN) et otherName.

email contient une options spécial 'copy' qui inclus automatiquement les emails contenus dans le sujet dans l'extension

dirName devrait pointer vers une section contenant un DN à utiliser.

otherName peut inclure des données arbitraires associées avec un OID

exemple :

```
subjectAltName=email:copy,email:my@other.address,URI :http://my.url.here/
subjectAltName=IP:192.168.7.1
subjectAltName=IP:13::17
subjectAltName=email:my@other.address,RID:1.2.3.4
subjectAltName=otherName:1.2.3.4;UTF8:some other identifier

subjectAltName=dirName:dir_sect

[dir_sect]
C=UK
O=My Organization
OU=My Unit
CN=My Name
```

Issuer Alternative Name Supporte toutes les options litéral de subjectAltName. Il ne supporte pas l'option mail :copy. Il supporte issuer :copy qui copie tous les subjectAltName du certificat de l'issuer

exemple :

```
issuerAltName = issuer:copy
```

Authority Info Access Extension d'accès aux informations de l'authorité donne des détails sur la mannière d'accéder à certaines informations lié à la CA. la syntaxe est **accessOID ;location**, où location a la même syntaxe qu'un subjectAltName (excepté **email :copy**) accessOID peut être un OID valide mais seul certaines valeurs ont une signification, par exemple OCSP et calssuers

exemple :

```
authorityInfoAccess = OCSP;URI:http://ocsp.my.host/
authorityInfoAccess = caIssuers;URI:http://my.ca/ca.html
```

Point de distribution de CRL Extension multi-valuée qui peut être un nom :valeur utilisant la même forme qu'un subjectAltName

Dans le cas d'une simple option, la section indiquée contient les valeurs pour chaque champs. dans cette section :

Si le nom est fullname la valeur devrait contenir le nom complet du point de distribution dans le même format que subjectAltName.

Si le nom est **relativename**, la valeur devrait contenir un nom de section dont le contenu représente un DN

Le nom **CRLIssuer** devrait contenir une valeur pour ce champ au format subjectAltName

Si le nom est **reasons**, la valeur devrait consister en un champ contenant les raisons : **keyCompromise, CACompromise, affiliationChanged, superseded, cessationOfOperation, certificateHold, privilegeWithdrawn, AACompromise**.

exemple :

```
crlDistributionPoints=URI:http://myhost.com/myca.crl
crlDistributionPoints=URI:http://my.com/my.crl,URI:http://oth...
crlDistributionPoints=crldistpoint1_section

[crldistpoint1_section]
fullname=URI:http://myhost.com/myca.crl
CRLissuer=dirName:issuer_sect
reasons=keyCompromise, CACompromise

[issuer_sect]
C=UK
O=Organisation
CN=Some Name
```

Issuing Distribution Point Devrait seulement apparaître dans les CRL. Extension multi-valuée dont la syntaxe est similaire à la section pointé par l'extension de point de distribution de CRL avec quelques différences

Les noms **reasons** et **CRLIssuer** ne sont pas reconnus

Le nom **onlysomereasons** est accepté. La valeur est au même format que le champ **reasons** des crlDistributionPoints. Les noms **onlyuser, onlyCA, onlyAA et indirectCRL** sont aussi acceptés et sont des booléen.

exemple :

```
issuingDistributionPoint=critical, @idp_section
[idp_section]
fullname=URI:http://myhost.com/myca.crl
indirectCRL=TRUE
onlysomereasons=keyCompromise,CACompromise
[issuer_sect]
C=UK
O=Organisation
CN=Some Name
```

Certificate Policies Extension raw

Si on suit la recommandation PKIX, en utilisant des OID :

```
certificatePolicies= 1.2.4.5, 1.1.3.4
```

Si on inclus des qualifiers, les OID de stratégie et les qualifieurs doivent être spécifiés dans une section séparée (sous la forme **@section**) La section référée doit inclure l'OID de stratégie en utilisant **policyIdentifier**. les qualifiers **cPSuri** peuvent être inclus avec cette syntaxe :

```
CPS.nnn=value
```

les qualifiers userNotice peuvent être définis avec :

```
userNotice.nnn=@notice
```

Cette section peut inclure explicitText, organization et noticeNumbers.

exemple :

```
certificatePolicies=ia5org,1.2.3.4,1.5.6.7.8,@polsect
[polsect]
policyIdentifier = 1.3.5.8
CPS.1="http://my.host.name/";
CPS.2="http://my.your.name/";
userNotice.1=@notice
[notice]
explicitText="Explicit Text Here"
organization="Organisation Name"
noticeNumbers=1,2,3,4
```

Si **userNotice** est utilisé avec IE5, il faut l'option **ia5org** avant pour modifier l'encodage. Cette options modifie le type de champs **organization** dans la rfc2459, il peut seulement être de type **DisplayText**. Dans la rfc3280 **IA5String** est aussi permis.

Policy Constraints Extension multivaluée consistant de nom **requireExplicitPolicy** ou **inhibitPolicyMapping** et une valeur entière non négative Au moins un composant doit être présent.

exemple :

```
policyConstraints = requireExplicitPolicy:3
```

Inhibit Any Policy Extension chaîne consistant de valeurs non négatives

exemple :

```
inhibitAnyPolicy = 2
```

Name Constraints Extension multi-valuée. Le nom devrait commencer avec le mot **permitted** ou **excluded**, suivi par un ' ;' le reste suivi la syntaxe subjectAltName excepté **email :copy** et **IP**

exemple :

```
nameConstraints=permitted;IP:192.168.0.0/255.255.0.0
nameConstraints=permitted;email:.somedomain.com
nameConstraints=excluded;email:.com
issuingDistributionPoint = idp_section
```

OCSP No Check Extension chaîne, mais sa valeur est **ignored**

exemple :

```
noCheck = ignored
```

Extensions arbitraires

Si une extension n'est pas supportée par OpenSSL, elle doit être encodé en utilisant le format arbitraire. Il est possible d'utiliser ce format pour les extensions connues.

Il y'a 2 manières d'écrire des extensions de type arbitraire :

Utiliser le mot ASN1 suivi de l'extension en utilisant la même syntaxe que ASN1_generate_nconf(3) :

```
1.2.3.4=critical,ASN1:UTF8String:Some random data
1.2.3.4=ASN1:SEQUENCE:seq_sect
[seq_sect]
field1 = UTF8:field1
field2 = UTF8:field2
```

Utiliser le mot DER pour inclure les donnée brutes :

```
1.2.3.4=critical,DER:01:02:03:04
1.2.3.4=DER:01020304
```

La valeur après DER est un dump DER d'une extension. Une extension peut être placée dans cette forme pour remplacer le mode par défaut :

```
basicConstraints=critical,DER:00:01:02:03
```

Notes

Si une extension est multi-valué et un champs doit contenir un ',', la forme longue doit être utilisée sinon le séparateur sera mal interprété :

```
subjectAltName=URI:ldap://somehost.com/CN=foo,OU=bar
```

va produire une erreur, mais ceci est valide :

```
subjectAltName=@subject_alt_section
[subject_alt_section]
subjectAltName=URI:ldap://somehost.com/CN=foo,OU=bar
```

Due à la librairie conf, le même nom de champ ne peut se trouver qu'une seul fois. Celà signifie que :

```
subjectAltName=@alt_section
[alt_section]
email=steve@here
email=steve@there
```

Va seulement reconnaître la dernière valeur, alors que ceci reconnaîtra toutes les valeurs :

```
[alt_section]
email.1=steve@here
email.2=steve@there
```