链滴

OpenSSL 签名认证机制

作者：linyiheng

原文链接：https://ld246.com/article/1503241553471

来源网站：链滴

生成CA证书

```
#!/bin/sh
#Generate the key.
openssl genrsa -out private/ca.key 2048
#Generate a certificate request.
openssl req -new -extensions v3_ca -key private/ca.key -out private/ca.csr -config "/root/ca/c
nf/openssl.conf"
#Self signing key is bad... this could work with a third party signed key... registeryfly has them
on for $16 but I'm too cheap lazy to get one on a lark.
#I'm also not 100% sure if any old certificate will work or if you have to buy a special one that
you can sign with. I could investigate further but since this
# service will never see the light of an unencrypted Internet see the cheap and lazy remark.
# So self sign our root key.
echo "01" > serial
echo "01" > crlnumber
touch index.txt
openssl ca -extensions v3_ca -selfsign -days 36500 -in private/ca.csr -keyfile private/ca.key -o
t private/ca.crt -config "/root/ca/conf/openssl.conf"
# openssl x509 -extensions v3_ca -days 36500 -in private/ca.csr -signkey private/ca.key -out p
ivate/ca.crt
# Setup the first serial number for our keys... can be any 4 digit hex string... not sure if there a
e broader bounds but everything I've seen uses 4 digits.
# echo FACE > serial
# Create the CA's key database.
# touch index.txt
# Create a Certificate Revocation list for removing 'user certificates.'
# openssl ca -gencrl -out /usr/nginx-1.8.0/ca/private/ca.crl -crldays 7 -config "/usr/nginx-1.8.
/ca/conf/openssl.conf"

#生成服务端证书

# Create us a key. Don't bother putting a password on it since you will need it to start apache.
If you have a better work around I'd love to hear it.
openssl genrsa -out server/server.key 2048
# Take our key and create a Certificate Signing Request for it.
openssl req -new -key server/server.key -out server/server.csr
# Sign this bastard key with our bastard CA key.
openssl ca -extensions v3_server -in server/server.csr -cert private/ca.crt -keyfile private/ca.ke
 -out server/server.crt -config "/root/ca/conf/openssl.conf"
```

生成客户端证书

```
#!/bin/sh
# The base of where our SSL stuff lives.
base="/root/ca"
# Were we would like to store keys... in this case we take the username given to us and store
verything there.
mkdir -p $base/client/

# Let's create us a key for this user... yeah not sure why people want to use DES3 but at least l
t's make us a nice big key.
openssl genrsa -des3 -out $base/client/client.key 2048
```

原文链接：OpenSSL 签名认证机制

```
# Create a Certificate Signing Request for said key.
openssl req -new -key $base/client/client.key -out $base/client/client.csr
# Sign the key with our CA's key and cert and create the user's certificate out of it.
openssl ca -in $base/client/client.csr -cert $base/private/ca.crt -keyfile $base/private/ca.key -
ut $base/client/client.crt -config "/root/ca/conf/openssl.conf"

# This is the tricky bit... convert the certificate into a form that most browsers will understand
KCS12 to be specific.
# The export password is the password used for the browser to extract the bits it needs and i
sert the key into the user's keychain.
# Take the same precaution with the export password that would take with any other passwo
d based authentication scheme.
openssl pkcs12 -export -clcerts -in $base/client/client.crt -inkey $base/client/client.key -out $
ase/client/client.p12
```

OpenSSL配置文件

```
HOME                    = .
RANDFILE                = $ENV::HOME/.rnd

# Extra OBJECT IDENTIFIER info:
#oid_file               = $ENV::HOME/.oid
oid_section             = new_oids

# To use this configuration file with the "-extfile" option of the
# "openssl x509" utility, name here the section containing the
# X.509v3 extensions to use:
# extensions            =
# (Alternatively, use a configuration file that has only
# X.509v3 extensions in its main [= default] section.)

[ new_oids ]

# We can add new OIDs in here for use by 'ca', 'req' and 'ts'.
# Add a simple OID like this:
# testoid1=1.2.3.4
# Or use config file substitution like this:
# testoid2=${testoid1}.5.6

# Policies used by the TSA examples.
tsa_policy1 = 1.2.3.4.1
tsa_policy2 = 1.2.3.4.5.6
tsa_policy3 = 1.2.3.4.5.7

[ req ]
default_bits            = 2048
default_md              = sha1
distinguished_name      = req_distinguished_name
string_mask = default
x509_extensions = v3_req

[ req_distinguished_name ]
countryName                     = Country Name (2 letter code)
countryName_default             = CN
```

```
countryName_min                 = 2
countryName_max                 = 2

stateOrProvinceName             = State or Province Name (full name)
stateOrProvinceName_default     = ShangHai

localityName                    = Locality Name (eg, city)
localityName_default            = ShangHai

0.organizationName              = Organization Name (eg, company)
0.organizationName_default      = IdealGroup

# we can do this but it is not needed normally :-)
#1.organizationName             = Second Organization Name (eg, company)
#1.organizationName_default     = World Wide Web Pty Ltd

organizationalUnitName          = Organizational Unit Name (eg, section)
organizationalUnitName_default  = IdealMobile

commonName                      = Common Name (e.g. server FQDN or YOUR name)
commonName_max                  = 64

emailAddress                    = Email Address
emailAddress_max                = 64

[ v3_server ]
subjectKeyIdentifier=hash
authorityKeyIdentifig=keyid:always,issuer
basicConstraints = CA:false
keyUsage = di`
italSignature, keyEncipherment
extendedKeyUsage = serverAuth
[ v3_ca ]
subjectKeyIdentifier=hash
authorityKeyIdentifier=keyid:always,issuer
basicConstraints = CA:true
keyUsage = cRLSign, keyCertSign
#keyUsage = digitalSignature, keyEncipherment
#extendedKeyUsage = serverAuth

#subjectAltNam=DNS:win.linyiheng.cn
[ v3_req ]
basicConstraints = CA:FALSE


[ ca ]
default_ca      = foo                # The default ca section

[ foo ]
dir           = /root/ca        # top dir
database      = /root/ca/index.txt          # index file.
new_certs_dir  = /root/ca/newcerts          # new certs dir

certificate    = /root/ca/private/ca.crt        # The CA cert
```

原文链接：OpenSSL 签名认证机制

```
serial          = /root/ca/serial             # serial no file
private_key     = /root/ca/private/ca.key  # CA private key
RANDFILE        = /root/ca/private/.rand      # random number file

default_days    = 3650                  # how long to certify for
default_crl_days = 3000                   # how long before next CRL
default_md      = sha1                  # message digest method to use
unique_subject = no                     # Set to 'no' to allow creation of
                                # several ctificates with same subject.
policy          = policy_any            # default policy

[ policy_any ]
countryName = optional
stateOrProvinceName = optional
organizationName = optional
organizationalUnitName = optional
localityName            = optional
commonName              = supplied
emailAddress            = optional
```