[**OpenSSL主配置文件openssl.cnf**](http://www.cnblogs.com/f-ck-need-u/p/6091027.html)

虽说配置文件很多设置不用修改就能直接使用，但是了解它是配置openssl相关事项所必须的。而且要实现复杂多功能，必然要对配置相关了然于心。

**1.man config**

该帮助文档说明了openssl.cnf以及一些其他辅助配置文件的规范、格式及读取方式。后文中的所有解释除非特别指明，都将以openssl.cnf为例。

[复制代码](javascript:void(0);)

[root@xuexi ~]# whatis config

Config (3pm) - access Perl configuration information

config (5ssl) - OpenSSL CONF library configuration files

Config::Extensions (3pm) - hash lookup of which core extensions were built

config.guess [config] (1) - guess the build system triplet

config [openssl] (5ssl) - OpenSSL CONF library configuration files

config.sub [config] (1) - validate and canonicalize a configuration triplet

config-util (5) - Common PAM configuration file for configuration utilities

[复制代码](javascript:void(0);)

因此直接man config即可。

配置文件openssl.cnf中分成了多个段落，每个段落都使用中括号包围的方式"[section\_name]"来标识。section\_name可以包含字母、数字和下划线。

第一个section被解释为默认段落，默认段落一般（是一般不是一定）没有[section\_name]标识。当搜索某一个section时，将首先搜索有名称的section，然后还会搜索默认section，如果没有找到匹配的有名称的section，将直接读取默认section。

该配置文件中使用#开头来书写注释信息。每个section包含一些name以及它们的值，格式为name=value，name和value的前导或尾随空格被忽略，如果要包含空格应该使用引号包围。

在name部分可以包含字母、数字以及一些标点符号，如“.”“,”“;”或“\_”。

在value部分可以使用变量扩展。在每个section中可以定义变量，每个section的变量默认只作用于当前section，变量引用的格式有两种"$var"或"${var}"。如果想要引用其他section中的变量或name，可以使用"$section\_name::name"或"${section::name}"。

在value部分可以指定为其他section的指针。请参看下文的示例。

可以使用反斜线“\”转义，包括转义引号字符以及反斜线本身，也可以使用“\”来进入多行书写模式。另外\n、\r、\b、\t是能够被识别的。

以下为书写示例，注意其中的特性。

[复制代码](javascript:void(0);)

/\* This is the default section.\*/

HOME=/temp

RANDFILE= ${ENV::HOME}/.rnd

configdir=$ENV::HOME/config

[ section\_one ]

default\_value = section\_three

/\* Also you can refer section\_name by character "@" \*/

default\_value = @section\_three

[ section\_two ]

/\* We are now in section two. \*//\* Quotes permit leading and trailing whitespace \*/

any = " any variable name "

other = A string that can \

cover several lines \

by including \\ characters

message = Hello World\n

[section\_three]

greeting =

$section\_one::message

[复制代码](javascript:void(0);)

**2./etc/pki/tls/openssl.cnf**

该文件主要设置了证书请求、签名、crl相关的配置。主要相关的伪命令为ca和req。对于x509不用该配置文件。

该文件从功能结构上分为4个段落：默认段、ca相关的段、req相关的段、tsa相关的段。每个段中都以name=value的格式定义。

该文件中没有被引用的段被视为忽略段，不会起到任何作用。

每个段中可以书写哪些name以及它们的意义，可以man相关命令，如man ca可以查看ca相关段可以书写的name，man req可以查看req相关段可以书写的name。

**(1).默认段**

第一段是默认段，一般没有section\_name，但不是一定没有，可以自定义有名称的。

默认段中定义的是一些公共属性，当搜索一个给定名称的段时，将首先搜索有名称的段，当搜索不到匹配的段后会搜索默认段。

以下是默认段的内容。

HOME = .

RANDFILE = $ENV::HOME/.rnd

oid\_section = new\_oids

仅定义了当前目录变量，以及随机数的文件路径变量。

至于最后一行的oid\_section=new\_oids表示指向[new\_oids]段。以下为new\_oids段。oid是是对象标识符，干啥的我也不知道，反正没改过它。

[复制代码](javascript:void(0);)

[ new\_oids ]

tsa\_policy1 = 1.2.3.4.1

tsa\_policy2 = 1.2.3.4.5.6

tsa\_policy3 = 1.2.3.4.5.7

[复制代码](javascript:void(0);)

**(2).ca相关的段**

这些段定义ca相关的控制选项。以下为ca相关段内容。其中黄底加粗黑字的为必须项，黄底加粗红字的为建议设置或建议修改的项。

[复制代码](javascript:void(0);)

####################################################################

[ ca ]

default\_ca = CA\_default /\*The default ca section\*/

####################################################################

[ CA\_default ]

dir = /etc/pki/CA /\* Where everything is kept \*/

/\* #### 这是第一个openssl目录结构中的目录 \*/

**certs** = $dir/certs /\* Where the issued certs are kept(已颁发的证书路径，即CA或自签的) \*/

/\* #### 这是第二个openssl目录结构中的目录，但非必须 \*/

**crl\_dir** = $dir/crl /\* Where the issued crl are kept(已颁发的crl存放目录) \*/

/\* #### 这是第三个openssl目录结构中的目录\*/

**database** = $dir/index.txt /\* database index file \*/

#unique\_subject = no /\* 设置为no则database文件中的subject列不能出现重复值 \*/

/\* 即不能为subject相同的证书或证书请求签名\*/

/\* 建议设置为yes，但为了保持老版本的兼容性默认是no \*/

**new\_certs\_dir** = $dir/newcerts /\* default place for new certs(将来颁发的证书存放路径) \*/

/\* #### 这是第四个openssl目录结构中的目录 \*/

**certificate** = $dir/cacert.pem /\* The A certificate(CA自己的证书文件) \*/

**serial** = $dir/serial /\* The current serial number(提供序列号的文件)\*/

crlnumber = $dir/crlnumber /\* the current crl number(当前crl序列号) \*/

crl = $dir/crl.pem /\* The current CRL(当前CRL) \*/

**private\_key** = $dir/private/cakey.pem /\* The private key(签名时需要的私钥，即CA自己的私钥) \*/

RANDFILE = $dir/private/.rand /\* private random number file(提供随机数种子的文件) \*/

x509\_extensions = usr\_cert /\* The extentions to add to the cert(添加到证书中的扩展项) \*/

/\* 以下两行是关于证书展示格式的，虽非必须项，但推荐设置。一般就如下格式不用修改 \*/

**name\_opt** = ca\_default /\* Subject Name options\*/

**cert\_opt** = ca\_default /\* Certificate field options \*/

/\* 以下是copy\_extensions扩展项，需谨慎使用 \*/

# copy\_extensions = copy /\* 生成证书时扩展项的copy行为，可设置为none/copy/copyall \*/

/\* 不设置该name时默认为none \*/

/\* 建议简单使用时设置为none或不设置，且强烈建议不要设置为copyall \*/

# crl\_extensions = crl\_ext

default\_days = 365 /\* how long to certify for(默认的证书有效期) \*/

default\_crl\_days= 30 /\* how long before next CRL(CRL的有效期) \*/

default\_md = default /\* use public key default MD(默认摘要算法) \*/

**preserve** = no /\* keep passed DN ordering(Distinguished Name顺序，一般设置为no \*/

/\* 设置为yes仅为了和老版本的IE兼容)\*/

**policy** = policy\_match /\* 证书匹配策略,此处表示引用[ policy\_match ]的策略 \*/

/\* 证书匹配策略定义了证书请求的DN字段(field)被CA签署时和CA证书的匹配规则 \*/

/\* 对于CA证书请求，这些匹配规则必须要和父CA完全相同 \*/

[ policy\_match ]

countryName = match /\* match表示请求中填写的该字段信息要和CA证书中的匹配 \*/

stateOrProvinceName = match

organizationName = match

organizationalUnitName = optional /\* optional表示该字段信息可提供可不提供 \*/

commonName = supplied /\* supplied表示该字段信息必须提供 \*/

emailAddress = optional

/\* For the 'anything' policy\*/

/\* At this point in time, you must list all acceptable 'object' types. \*/

/\* 以下是没被引用的策略扩展，只要是没被引用的都是被忽略的 \*/

[ policy\_anything ]

countryName = optional

stateOrProvinceName = optional

localityName = optional

organizationName = optional

organizationalUnitName = optional

commonName = supplied

emailAddress = optional

/\* 以下是添加的扩展项usr\_cert的内容\*/

[ usr\_cert ]

basicConstraints=CA:FALSE /\* 基本约束，CA:FALSE表示该证书不能作为CA证书，即不能给其他人颁发证书\*/

/\* keyUsage = critical,keyCertSign,cRLSign # 指定证书的目的，也就是限制证书的用法\*/

/\* 除了上面两个扩展项可能会修改下，其余的扩展项别管了，如下面的 \*/

nsComment = "OpenSSL Generated Certificate"

subjectKeyIdentifier=hash

authorityKeyIdentifier=keyid,issuer

[复制代码](javascript:void(0);)

**(3).req相关的段**

[复制代码](javascript:void(0);)

[ req ]

default\_bits = 2048 /\* 生成证书请求时用到的私钥的密钥长度 \*/

default\_md = sha1 /\* 证书请求签名时的单向加密算法 \*/

default\_keyfile = privkey.pem /\* 默认新创建的私钥存放位置， \*/

/\* 如-new选项没指定-key时会自动创建私钥 \*/

/\* -newkey选项也会自动创建私钥 \*/

distinguished\_name = req\_distinguished\_name /\* 可识别的字段名(常被简称为DN) \*/

/\* 引用req\_distinguished\_name段的设置 \*/

x509\_extensions = v3\_ca /\* 加入到自签证书中的扩展项 \*/

# req\_extensions = v3\_req /\* 加入到证书请求中的扩展项 \*/

attributes = req\_attributes /\* 证书请求的属性，引用req\_attributes段的设置，可以不设置它 \*/

# encrypt\_key = yes | no /\* 自动生成的私钥文件要加密否？一般设置no，和-nodes选项等价 \*/

/\* 输入和输出私钥文件的密码，如果该私钥文件有密码，不写该设置则会提示输入 \*/

/\* input\_password = secret \*/

/\* output\_password = secret \*/

# prompt = yes | no /\* 设置为no将不提示输入DN field，而是直接从配置文件中读取，需要同时设置DN默认值，否则创建证书请求时将出错。 \*/

string\_mask = utf8only

[ req\_distinguished\_name ]

/\* 以下项均可指定可不指定，但ca段的policy中指定为match和supplied一定要指定。 \*/

/\* 以下选项都可以自定义，如countryName = C，commonName = CN \*/

countryName = Country Name (2 letter code) /\* 国家名(C) \*/

countryName\_default = XX /\* 默认的国家名 \*/

countryName\_min = 2 /\* 填写的国家名的最小字符长度 \*/

countryName\_max = 2 /\* 填写的国家名的最大字符长度 \*/

stateOrProvinceName = State or Province Name (full name) /\* 省份(S) \*/

/\* stateOrProvinceName\_default = Default Province \*/

localityName = Locality Name (eg, city) /\* 城市(LT) \*/

localityName\_default = Default City

0.organizationName = Organization Name (eg, company) /\* 公司(ON) \*/

0.organizationName\_default = Default Company Ltd

organizationalUnitName = Organizational Unit Name (eg, section) /\* 部门(OU) \*/

/\* organizationalUnitName\_default = \*/

/\* 以下的commonName(CN)一般必须给,如果作为CA，那么需要在ca的policy中定义CN = supplied \*/

/\* CN定义的是将要申请SSL证书的域名或子域名或主机名。 \*/

/\* 例如要为zhonghua.com申请ssl证书则填写zhonghua.com，而不能填写www.zhonghua.com \*/

/\* 要为www.zhonghua.com申请SSL则填写www.zhonghua.com \*/

/\* CN必须和将要访问的网站地址一样，否则访问时就会给出警告 \*/

/\* 该项要填写正确，否则该请求被签名后证书中的CN与实际环境中的CN不对应，将无法提供证书服务 \*/

commonName = Common Name (eg, your name or your server\'s hostname) /\* 主机名(CN) \*/

commonName\_max = 64

emailAddress = Email Address /\* Email地址，很多时候不需要该项的 \*/

emailAddress\_max = 64

[ req\_attributes ] /\* 该段是为了某些特定软件的运行需要而设定的， \*/

/\* 现在一般都不需要提供challengepassword \*/

/\* 所以该段几乎用不上 \*/

/\* 所以不用管这段 \*/

challengePassword = A challenge password

challengePassword\_min = 4

challengePassword\_max = 20

unstructuredName = An optional company name

[ v3\_req ]

/\* Extensions to add to a certificate request \*/

basicConstraints = CA:FALSE

keyUsage = nonRepudiation, digitalSignature, keyEncipherment

[ v3\_ca ]

/\* Extensions for a typical CA \*/

subjectKeyIdentifier=hash

authorityKeyIdentifier=keyid:always,issuer

basicConstraints = CA:true

# keyUsage = cRLSign, keyCertSign /\* 典型的CA证书的使用方法设置，由于测试使用所以注释了 \*/

/\* 如果真的需要申请为CA/\*么该设置可以如此配置 \*/

[复制代码](javascript:void(0);)

可以自定义DN(Distinguished Name)段中的字段信息，注意ca段中的policy指定的匹配规则中如果指定了match或这supplied的则DN中必须定义。例如下面的示例：由于只有countryName、organizationName和commonName被设定为match和supplied，其余的都是optional，所以在DN中可以只定义这3个字段，而且在DN中定义了自定义的名称。

[复制代码](javascript:void(0);)

[policy\_to\_match]

countryName = match

stateOrProvinceName = optional

organizationName = match

organizationalUnitName = optional

commonName = supplied

emailAddress = optional

[DN]

countryName = "C"

organizationName = "O"

commonName = "Root CA"

[复制代码](javascript:void(0);)

**(4).配置文件示例**

以下是一个配置文件的示例。假设该配置文件路径为/ssl/ssl.conf。

[复制代码](javascript:void(0);)

[default]

name = root-ca /\* 变量\*/

default\_ca = CA\_default

name\_opt = ca\_default

cert\_opt = ca\_default

[CA\_default]

home = . /\* 变量\*/

database = $home/db/index

serial = $home/db/serial

crlnumber = $home/db/crlnumber

certificate = $home/$name.crt

private\_key = $home/private/$name.key

RANDFILE = $home/private/random

new\_certs\_dir = $home/certs

unique\_subject = no

copy\_extensions = none

default\_days = 3650

default\_crl\_days = 365

default\_md = sha256

policy = policy\_to\_match

[policy\_to\_match]

countryName = match

stateOrProvinceName = optional

organizationName = match

organizationalUnitName = optional

commonName = supplied

emailAddress = optional

[CA\_DN]

countryName = "C"

contryName\_default = "CN"

organizationName = "O"

organizationName\_default = "jmu"

commonName = "CN"

commonName\_default = "longshuai.com"

[req]

default\_bits = 4096

encrypt\_key = no

default\_md = sha256

utf8 = yes

string\_mask = utf8only

# prompt = no /\* 测试时该选项导致出错，所以将其注释掉\*/

distinguished\_name = CA\_DN

req\_extensions = ca\_ext

[ca\_ext]

basicConstraints = critical,CA:true

keyUsage = critical,keyCertSign,cRLSign

subjectKeyIdentifier = hash

[复制代码](javascript:void(0);)

根据该配置文件示例，进行自建根CA、签名等的操作方法请看：http://www.cnblogs.com/f-ck-need-u/p/6091105.html

参考

http://www.cnblogs.com/f-ck-need-u/p/6091027.html