附录2：购买SSL认证

1. 先生成CSR文件 （CSR--Certificate Signing Request 是申请SSL认证certificate所需要的信息，包括 Public Key, 必须在自己的Server上生成）

$ openssl version -a

$ sudo apt install openssl #检查openssl版本， 如果没有，则安装

$openssl req -nodes -newkey rsa:2048 -keyout myserver.key -out server.csr

**Country Name** (2 letter code) [AU]:**CA**

State or **Province** Name (full name) [Some-State]:**ON**

Locality Name (eg, **city**) []:**Toronto**

Organization Name (eg, **company**) [Internet Widgits Pty Ltd]:EPLAN

Organizational Unit Name (eg, section) []:IT

Common Name (e.g. server FQDN or YOUR name) []: **www.udoido.com**

Email Address []:

A challenge password []:

An optional company name []:

其中： Common Name就是域名

直接回车： 使用缺省的值

输入“.”: 清空

其中命令行参数的含义：

\* req : 要求生成CSR文件

\* rsa:2048 RSA 2048位

\* -nodes ： no DES， 不要encrypt 到PKCS#12 文件

* -newkey rsa:2048 升级
* -new -newkey： 新的key

\* 指定key文件和csr文件的名称， （名称必须是域名）

-keyout udoido.com.key ： key文件名称

-out udoido.com.csr 文件名称

结果生成2个文件：

* 私钥： myserver.key （代表本机，无法找回。所以不可丢失，也不能分享给它人）
* CSR文件： server.scr：

其中私钥在生成之后也参与了CSR的生成。

1. 提交CSR文件给Sectigo( SSL 认证公司)
2. Sectigo认证提供的文件：

CER编码： CERT\_CC7148305.crt （i.e. CERT 文件， 下面写作： www\_udoido\_com.crt）

ca.pem

csr.pem

intermediate.pem

（必须再次要求Sectigo提供ca-bundle形式的文件：

www\_udoido\_com.ca-bundle

www\_udoido\_com.crt）

1. 交给NodeJS服务器：

上面自己生成的key文件和申请得到的crt、ca-bundle文件一起交给node作为https的启动参数

var https\_options = {

ca: fs.readFileSync("/path/to/mydomain.ca-bundle"), （一般网站不需要，）

key: fs.readFileSync("/path/to/server.key"), // 私钥，在生成申请资料CSR时采用的私钥

cert: fs.readFileSync("/path/to/mydomain.crt") // 认证得到的文件 XXX.crt

};

续购SSL：

Moniker会自动发起续购，但是Sectigo要求你输入一串字符以确认。

新证书发到moniker，必须再次人工要求提供ca-bundle，

只替换cer证书文件，（ca-bundle和key都 与首次一样）

、、、、、、、、、、、、、、、、、、、、、、、、、、、、、、、、、、、

Https (SSL) 认证的基本知识：

一个证书，有3种编码形式： 原始的cer,==》DER编码 ==》PEM编码

CER (或CRT）： certificate，最原始的Certificate数据。

DER： 编码方法，适合任意类别数据， 包括上述的CER数据。 DER和BER都可以编码ASN.1描述的数据.

PEM（Privacy Enhanced Mail）：编码binary为ASCII （Base64）， 包括DER信息

<pre>

var https\_options = {

ca: fs.readFileSync("/path/to/mydomain.ca-bundle"),

key: fs.readFileSync("/path/to/server.key"),

cert: fs.readFileSync("/path/to/mydomain.crt")

};

</pre>

<pre>

https.createServer(options, function (req, res) {

res.writeHead(200);

res.end("Welcome to Node.js HTTPS Servern");}).

listen(8443)

</pre>