10.Nginx HTTPS 实践

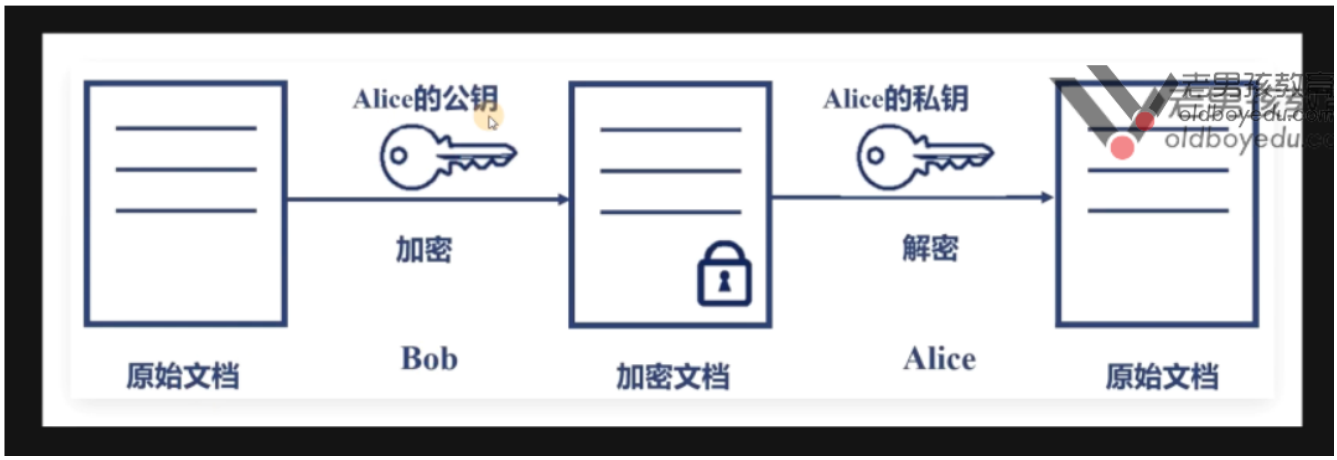
实战构建一个满足苹果要求的HTTPS后台服务

1.HTTPS安全证书基本概述

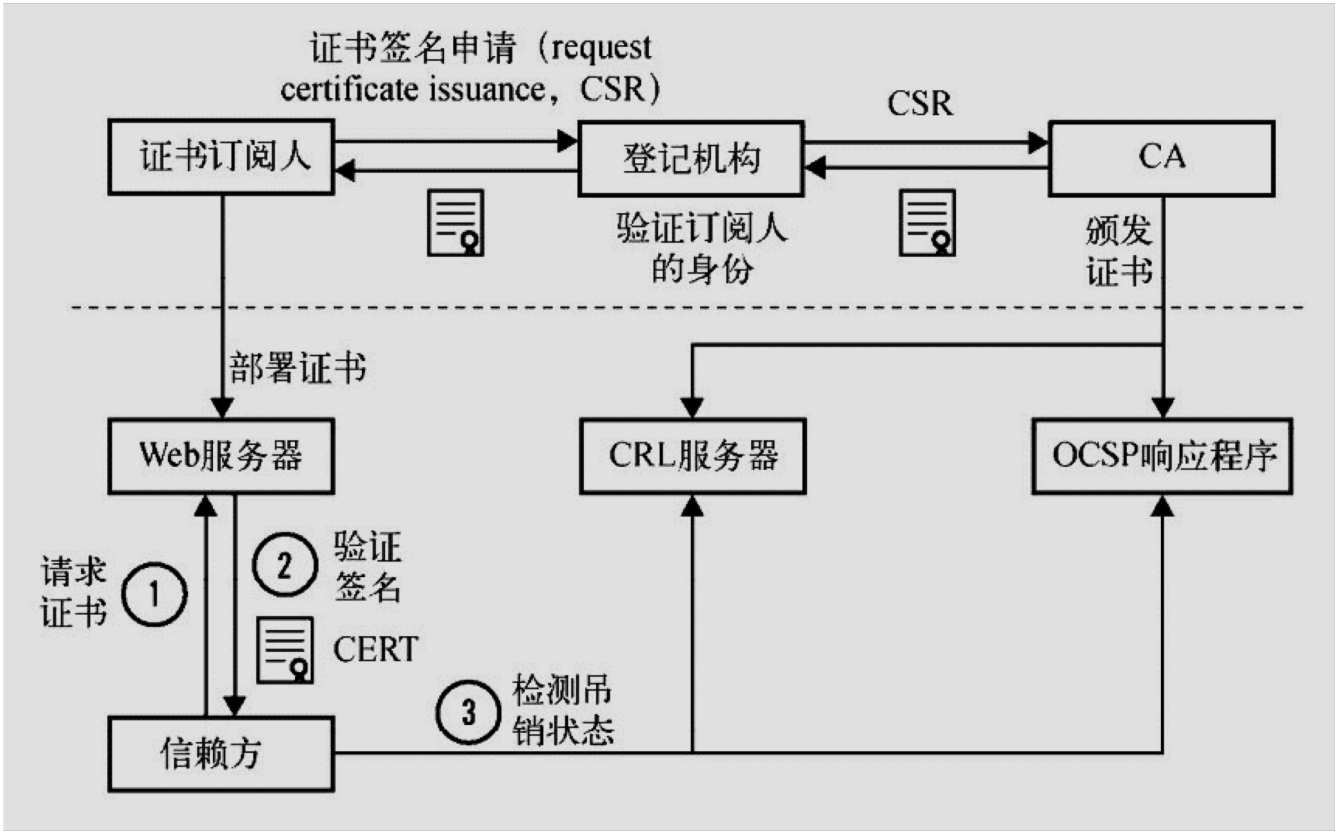
为什么需要使用HTTPS, 因为HTTP不安全。当我们使用http网站时，会遭到劫持和篡改，如果采用https协议，那么数据在传输过程中是加密的，所以黑客无法窃取或者篡改数据报文信息，同时也避免网站传输时信息泄露。

那么我们在实现https时，需要了解ssl协议，但我们现在使用的更多的是TLS加密协议。
那么TLS是怎么保证明文消息被加密的呢？在OSI七层模型中，应用层是http协议，那么在应用层协议之下，我们的表示层，是ssl协议所发挥作用的一层，它通过（握手、交换秘钥、告警、加密）等方式，使应用层http协议没有感知的情况下做到了数据的安全加密

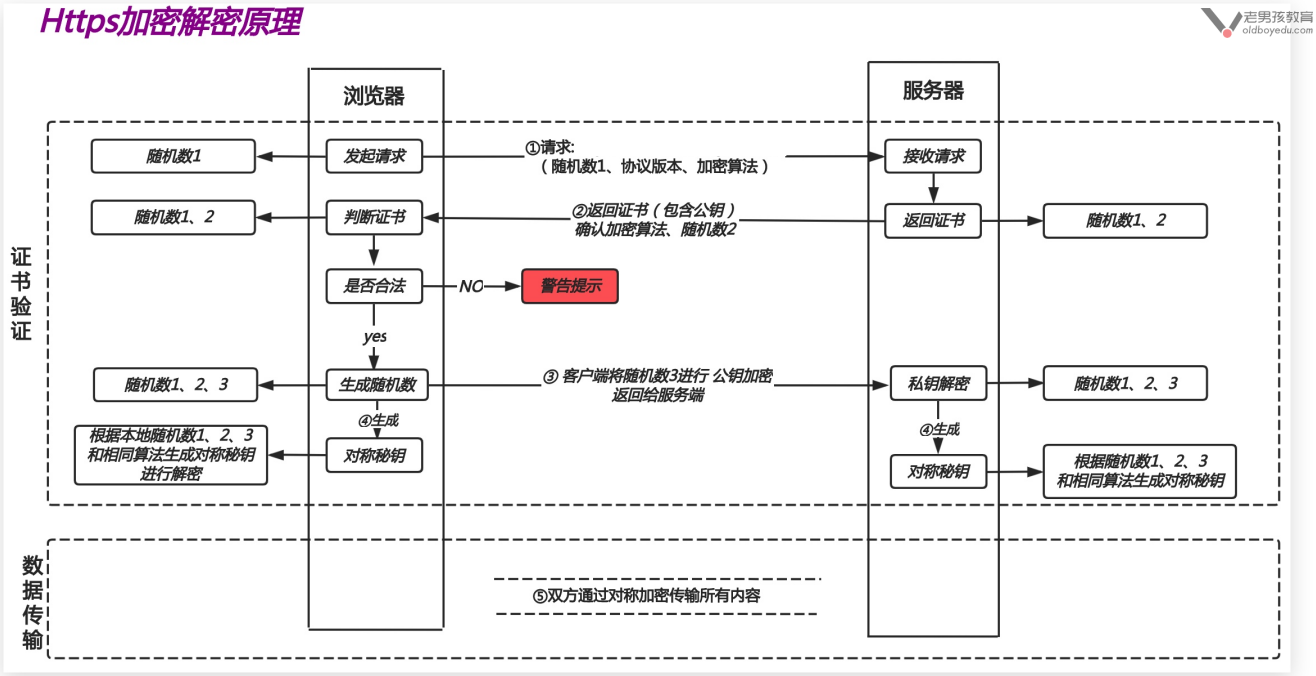
非对称

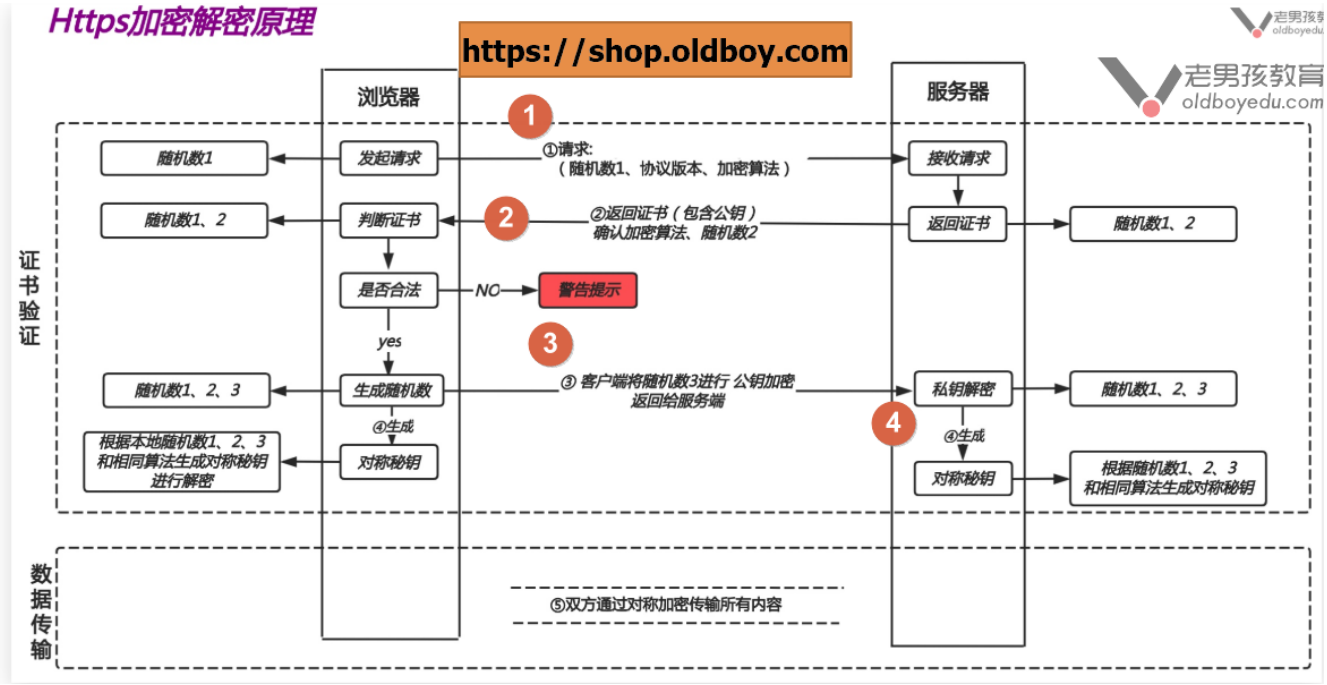


那么在数据进行加密与解密过程中，如何确定双方的身份，此时就需要有一个权威机构来验证双方省份。那么这个权威机构则是CA机构。那CA机构又是如何颁发证书



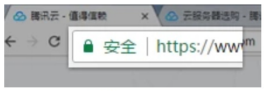
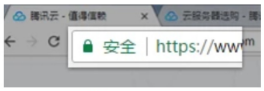
https全流程





那么证书究竟是怎样组成的呢，接下来我们看一下证书有哪几种类型?

- DV证书 个人使用,免费
- OV证书 中小企业使用(大部分)
- EV证书 (巨行企业使用)

对比	域名型 DV	企业型 OV	增强型 EV
绿色地址栏 (以chrome为例)	 小锁标记 + https	 小锁标记 + https	 小锁标记 + https + 公司名称
一般用途	个人站点和应用； 简单的Https加密需求	电子商务站点和应用； 中小型企业站点	大型金融平台； 大型企业和政府机构站点
审核内容	域名所有权验证	全面的企业身份验证； 域名所有权验证	最高等级的企业身份验证； 域名所有权验证
颁发时长	10分钟–24小时	3–5个工作日	5–7个工作日
单次申请年限	1 年	1–2 年	1–2 年
赔付保障金	—	125–175 万美金	150–175 万美金

HTTPS证书购买选择

保护1个域名 `www`

保护5个域名 `www images cdn test m`

通配符域名 `.oldboy.com`* 兼容 www bbs blog ... log url .....

HTTPS注意事项

Https不支持续费,证书到期需重新申请新并进行替换.

Https不支持三级域名解析，如 `test.m.oldboy.com` *m.oldboy.com abc.oldboy.com a.b.c.abc.oldboy.com*

*Https显示**绿色**，说明整个网站的url都是https的，并且都是安全的。*

*Https显示**黄色**，说明网站代码中有部分URL地址是http不安全协议的。*

*Https显示**红色**，要么证书是假的，要么证书已经过期。*

```
1
```

2.Nginx单台实现HTTPS实战

- ssl

1.环境准备

```
#nginx必须有ssl模块
[root@Nginx ~]# nginx -V
 --with-http_ssl_module

#创建存放ssl证书的路径
[root@Nginx ~]# mkdir -p /etc/nginx/ssl_key
[root@Nginx ~]# cd /etc/nginx/ssl_key
```

2.使用openssl命令充当CA权威机构创建证书(生产不使用此方式生成证书，不被互联网认可的黑户证书)

```
[root@Nginx ssh_key]# openssl genrsa -idea -out server.key 2048
Generating RSA private key, 2048 bit long modulus
.....+++
#记住配置密码，我这里是1234
Enter pass phrase for server.key:
Verifying - Enter pass phrase for server.key:
```

3.生成自签证书，同时去掉私钥的密码

```
[root@Nginx ssl_key]# openssl req -days 36500 -x509 \
-sha256 -nodes -newkey rsa:2048 -keyout server.key -out server.crt

Country Name (2 letter code) [XX]:CN
State or Province Name (full name) []:WH
Locality Name (eg, city) [Default City]:WH
Organization Name (eg, company) [Default Company Ltd]:edu
Organizational Unit Name (eg, section) []:SA
Common Name (eg, your name or your servers hostname) []:oldboy
Email Address []:oldboy@foxmail.com


# req  -->用于创建新的证书
# new  -->表示创建的是新证书
# x509 -->表示定义证书的格式为标准格式
# key  -->表示调用的私钥文件信息
# out  -->表示输出证书文件信息
# days -->表示证书的有效期
```

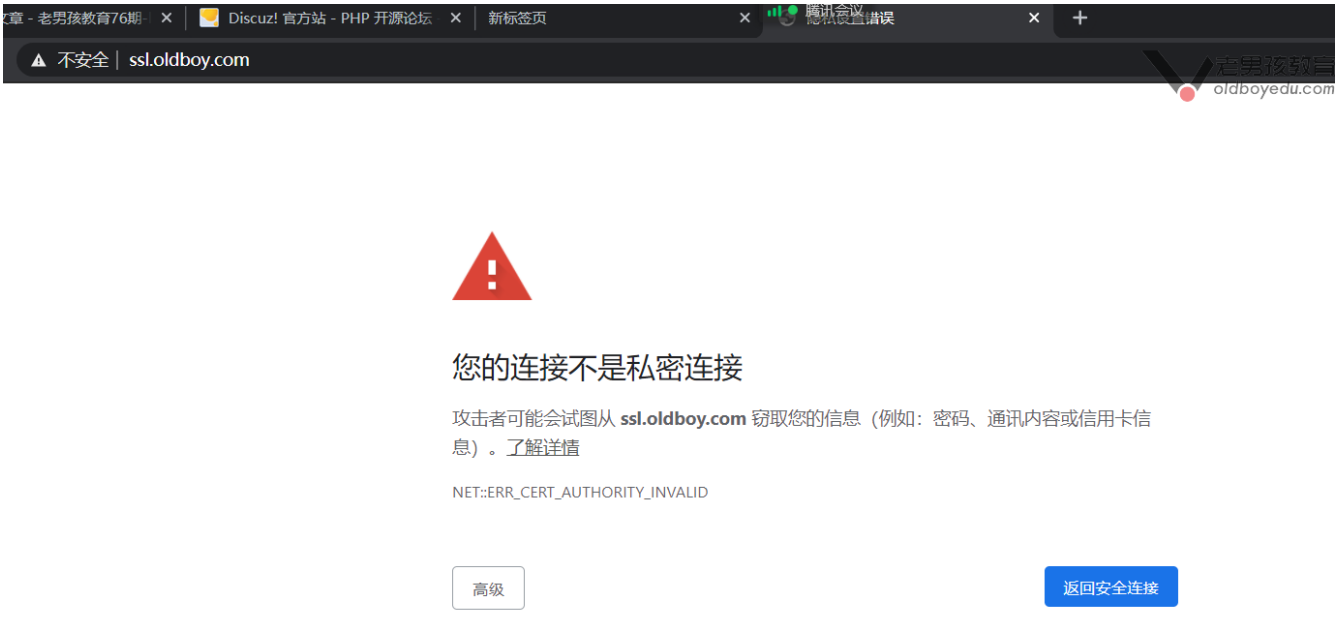
4.证书申请完成后需要了解Nginx如何配置Https

```
#启动ssl功能  过时 不在使用
Syntax: ssl on | off;  #1.15.0 之后不在使用 ， 请使用 listen 443 ssl; 进行替代
Default: ssl off;
Context: http, server

#证书文件
Syntax: ssl_certificate file;
Default: —
Context: http, server

#私钥文件
Syntax: ssl_certificate_key file;
Default: —
Context: http, server
```

5.配置Nginx配置Https实例

```
[root@Nginx ~]# cat /etc/nginx/conf.d/ssl.conf
server {
    listen 443;
    server_name s.oldboy.com;
    ssl on;
    ssl_certificate   ssl_key/server.crt;
    ssl_certificate_key  ssl_key/server.key;
    location / {
        root /code/ssl;
        index index.html;
    }
}

#准备对应的站点目录，并重启Nginx服务
[root@Nginx ~]# mkdir -p /code
[root@Nginx ~]# echo "Https" > /code/index.html
[root@Nginx ~]# systemctl restart nginx
```

6.浏览器输入`https://s.oldboy.com`访问,由于该证书非第三方权威机构颁发，而是我们自己签发的，所以浏览器会警告

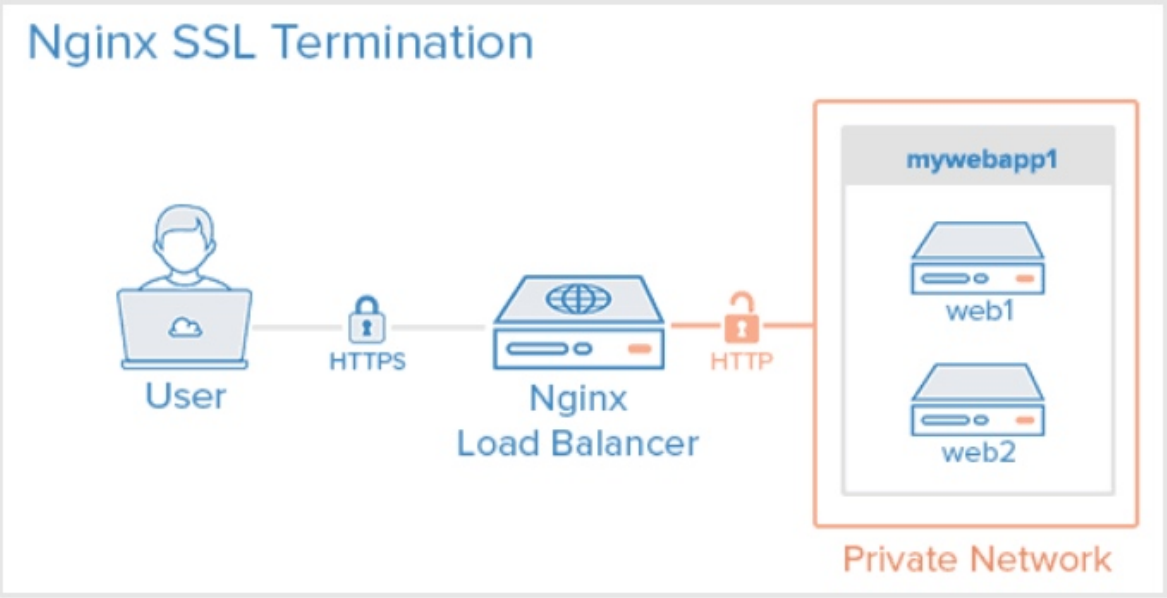


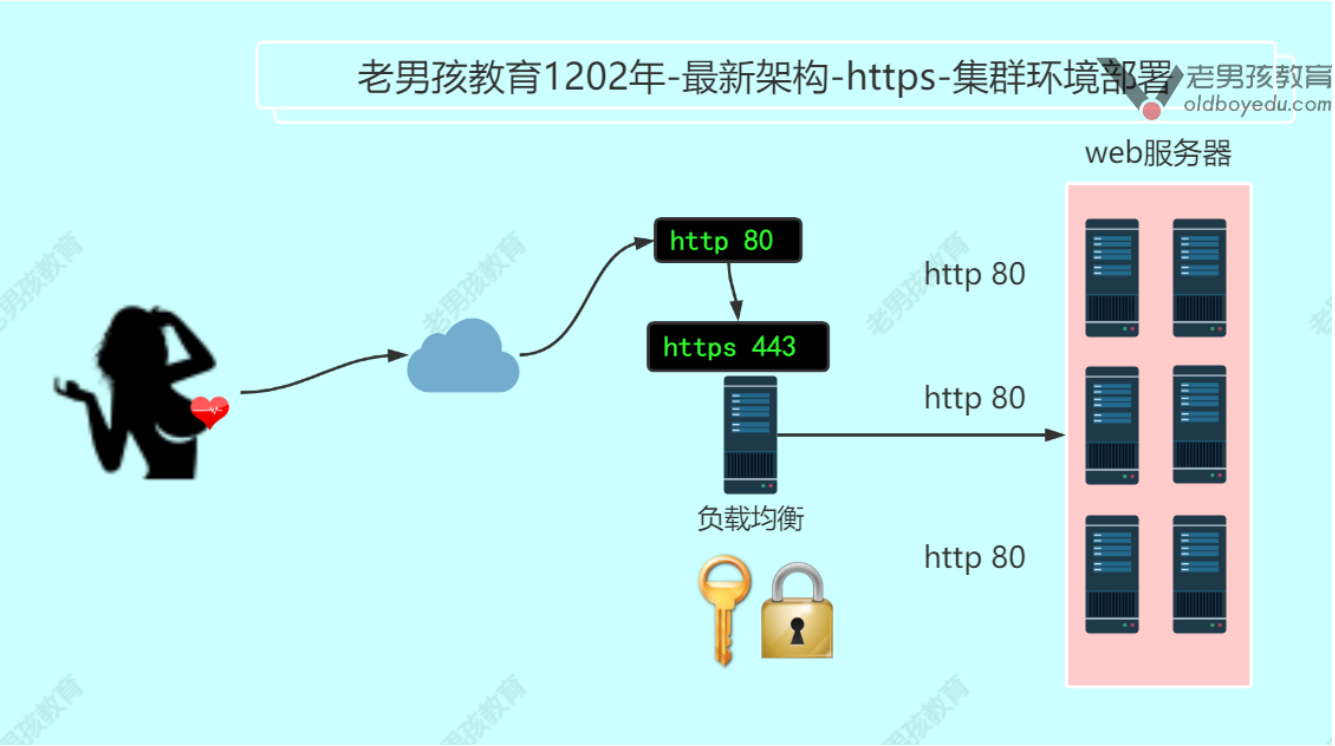
7.以上配置如果用户忘记在浏览器地址栏输入 https:// 那么将不会跳转至https，建议配置将用户访问 http 请求强制跳转https

```
[root@Nginx ~]# cat /etc/nginx/conf.d/ssl.conf
server {
    listen 443;
    server_name s.oldboy.com;
    ssl on;
    ssl_certificate   ssl_key/server.crt;
    ssl_certificate_key  ssl_key/server.key;
    location / {
        root /code/ssl;
        index index.html;
    }
}
server {
        listen 80;
        server_name s.oldboy.com;
        rewrite ^(.*) https://$server_name$1 redirect;  #rewrite跳转方式
        #return 302 https://$server_name$request_uri;   #return跳转方式
}
```

3.Nginx集群实现HTTPS实践

实战Nginx负载均衡+Nginx WEB配置HTTPS安全





1) 环境准备

主机名	外网IP(NAT)	内网IP(LAN)	角色
lb01	eth0:10.0.0.5	eth1:172.16.1.5	nginx-proxy
web01	eth0:10.0.0.7	eth1:172.16.1.7	nginx-web01
web02	eth0:10.0.0.8	eth1:172.16.1.8	nginx-web02

2) 配置后端两台web 节点监听80端口, 如已配置则无需修改

```
1
```

```
#web01 web02
[root@web01 ~]# cat  /etc/nginx/conf.d/ssl.oldboy.com.conf
server {
    listen 80;
    server_name ssl.oldboy.com;
    root /code/ssl;
    location  / {
      index index.html;
    }
}

[root@web01 ~]# echo  web01 ssl  > /code/ssl/index.html
```

```
[root@web01 conf.d]# cat blog.oldboy.com.conf
server {
    listen 80;
    server_name blog.oldboy.com;
    root /code/wordpress;
    index index.php index.html;

    location ~ \.php$ {
        root /code/wordpress;
        fastcgi_pass   127.0.0.1:9000;
        fastcgi_index  index.php;
        fastcgi_param  SCRIPT_FILENAME  $document_root$fastcgi_script_name;
        include        fastcgi_params;
    }
}
```

3) 配置第二台web 节点

```
[root@web02 ~]# yum install -y nginx
[root@web01 ~]# scp -rp /etc/nginx/ssl_key/ root@172.16.1.8:/etc/nginx/
[root@web01 ~]# scp -rp /etc/nginx/conf.d/ root@172.16.1.8:/etc/nginx/
```

4) 重启两台后端web 节点Nginx

```
[root@web01 ~]# systemctl restart nginx
[root@web02 ~]# systemctl restart nginx
```

5) Nginx负载均衡先 生成证书

```
[root@lb01 ~]# mkdir /etc/nginx/ssl_key -p
[root@lb01 ~]# cd /etc/nginx/ssl_key
[root@lb01 ~]# openssl genrsa -idea -out server.key 2048
[root@lb01 ~]# openssl req -days 36500 -x509 -sha256 -nodes -newkey rsa:2048 -keyout server.key -out server.crt
```

6) Nginx负载均衡配置文件如下

```
[root@lb01 ~]# cat /etc/nginx/conf.d/proxy.conf
# 定义后端资源池
upstream site {
    server 172.16.1.7:80 max_fails=2 fail_timeout=10s;
    server 172.16.1.8:80 max_fails=2 fail_timeout=10s;
}
# https配置
server {
    listen 443;
    server_name blog.oldboy.com;
    ssl on;
    ssl_certificate  ssl_key/server.crt;
    ssl_certificate_key  ssl_key/server.key;
    location / {
        proxy_pass http://site;
        include proxy_params;
    }
}
# 用户http请求跳转至https
server {
    listen 80;
    server_name blog.oldboy.com;
    return 302 https://$server_name$request_uri;
}
```

7) 重启Nginx 负载均衡

```
[root@lb01 ~]# nginx -t
[root@lb01 ~]# systemctl restart nginx
```

8) wordpress早期安装如果是使用http方式, 那开启https后会导致, wordpress出现加载或无法登陆问题。

```
#web节点增加此参数
location ~ \.php$ {
    ...
    fastcgi_param  HTTPS on;
    ...
}
```

4. https应用场景

- 纯静态页面(展示页面) 只使用 http协议 (静态不涉及动态请求)
- 只要有动态信息,推荐使用https

5. https优化

- 官方最佳实践
- ssl模块中

```
# 设置    worker_processes  等于 cpu核心总数 lscpu
set the number of worker processes equal to the number of processors,

```

```
# 开启 keepalived长连接 keepalive_timeout 设置长一些  70
        keepalive_timeout   70;

# enable the shared session cache, 开启内存中的共享空间,存放session缓存
# disable the built-in session cache 关闭内置session缓存

#增加session过期时间
and possibly increase the session lifetime (by default, 5 minutes):
        ssl_session_cache   shared:SSL:10m;  #10MB
        ssl_session_timeout 10m;             #会话超时时间


```

- ssl模块指令补充:

```

       ssl_protocols       TLSv1 TLSv1.1 TLSv1.2;   #指定 tls版本
       ssl_ciphers         AES128-SHA:AES256-SHA:RC4-SHA:DES-CBC3-SHA:RC4-MD5;  #指定可以用加密格式


```

今日总结

思维导图总结: https://www.processon.com/view/link/60868b77f346fb0e35c51ac7

- https 认证流程
- 申请证书
- **部署单台nginx**
- **部署单台tomcat**
- **部署集群nginx和tomcat**
- **云环境clb（https 80-->443) +ecs**

- openvpn