Nginx搭建HTTPS服务器

2013-01-20 18:47 6082人阅读 评论(6) 收藏 举报

分类： Ubuntu (43) nginx (18) 运维 (58)

HTTPS简介

HTTPS(Hyper Text Transfer Protocol Secure)，是一种基于SSL/TLS的HTTP，所有的HTTP数据都是在SSL/TLS协议封装之上进行传输的。HTTPS协议是在HTTP协议的基础上，添加了SSL/TLS握手以及数据加密传输，也属于应用层协议。Https使用的默认端口是443。更多HTTPS原理可以参考阮一峰老师的文章：http://www.ruanyifeng.com/blog/2014/02/ssl_tls.html

SSL证书

证书类型简介

要设置安全服务器，使用公共钥创建一对公私钥对。大多数情况下，发送证书请求（包括自己的公钥），你的公司证明材料以及费用到一个证书颁发机构(CA)。CA验证证书请求及您的身份，然后将证书返回给您的安全服务器。

但是内网实现一个服务器端和客户端传输内容的加密，可以自己给自己颁发证书，只需要忽略掉浏览器不信任的警报即可！

由CA签署的证书为您的服务器提供两个重要的功能：

- 浏览器会自动识别证书并且在不提示用户的情况下允许创建一个安全连接。
- 当一个CA生成一个签署过的证书，它为提供网页给浏览器的组织提供身份担保。
- 多数支持ssl的web服务器都有一个CA列表，它们的证书会被自动接受。当一个浏览器遇到一个其授权CA并不在列表中的证书，浏览器将询问用户是否接受或拒绝连接。

制作CA证书

ca.key CA私钥：

```
[html] view plain copy print ?
01. openssl genrsa -des3 -out ca.key 2048
```

```
12:05:32-ZhengYi/etc/nginx$ sudo openssl genrsa -des3 -out ca.key 2048
Generating RSA private key, 2048 bit long modulus
.....................................................................+++
....+++
e is 65537 (0x10001)
Enter pass phrase for ca.key:
Verifying - Enter pass phrase for ca.key:
```


ca.crt CA根证书（公钥）：

openssl req -new -x509 -days 365 -key ca.key -out ca.crt

```
12:20:52-ZhengYi/etc/nginx$ sudo openssl req -new -x509 -days 365 -key ca.key -out ca.crt
Enter pass phrase for ca.key:
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [AU]:CN
State or Province Name (full name) [Some-State]:BeiJing
Locality Name (eg, city) []:BeiJing
Organization Name (eg, company) [Internet Widgits Pty Ltd]:LCZH
Organizational Unit Name (eg, section) []:CS
Common Name (e.g. server FQDN or YOUR name) []:wangzhengyi
Email Address []:wzyll1314@gmail.com
```


制作网站的证书并用CA签名认证

这里，假设网站域名为www.example.com，生成com.example.com证书私钥：

```
[plain] view plain copy print ?
01. openssl genrsa -des3 -out www.example.com.pem 1024
```

```
12:26:01-ZhengYi/etc/nginx$ sudo openssl genrsa -des3 -out www.example.com.pem 1024
Generating RSA private key, 1024 bit long modulus
.........++++++
..............++++++
e is 65537 (0x10001)
Enter pass phrase for www.example.com.pem:
Verifying - Enter pass phrase for www.example.com.pem:
```


制作解密后的www.example.com证书私钥：

```
[plain] view plain copy print ?
01. openssl rsa -in www.example.com.pem -out www.example.com.key
```

```
12:28:21-ZhengYi/etc/nginx$ sudo openssl rsa -in www.example.com.pem -out www.example.com.key
Enter pass phrase for www.example.com.pem:
writing RSA key
```


生成签名请求：

```
[plain] view plain copy print ?
01. openssl req -new -key www.example.com.pem -out www.example.com.csr
```

```
12:31:46-ZhengYi/etc/nginx$ sudo openssl req -new -key www.example.com.pem -out www.example.com.csr
Enter pass phrase for www.example.com.pem:
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [AU]:CN
State or Province Name (full name) [Some-State]:BeiJing
Locality Name (eg, city) []:BeiJing
Organization Name (eg, company) [Internet Widgits Pty Ltd]:LCZH
Organizational Unit Name (eg, section) []:CS
Common Name (e.g. server FQDN or YOUR name) []:*.example.com
Email Address []:wzyll1314@gmail.com

Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []:
An optional company name []:
```


可以在Common Name中填入网站域名，即可生产该网站的证书。

用CA进行签名：

```
[plain] view plain copy print ?
01. openssl ca -policy policy_anything -days 365 -cert ca.crt -keyfile ca.key -in www.example.com.csr -out www.example.com.crt
```

可能执行签名时，会出现“I am unable to access the ./demoCA/newcerts directory”问题：

```
Using configuration from /usr/lib/ssl/openssl.cnf
Enter pass phrase for ca.key:
I am unable to access the ./demoCA/newcerts directory
./demoCA/newcerts: No such file or directory
```

解决方法：

```
[plain] view plain copy print ?
01. mkdir -p demoCA/newcerts
02. touch demoCA/index.txt
03. touch demoCA/serial
04. echo "01" > demoCA/serial
```

然后，再执行签名命令即可。

基于Nginx搭建HTTPS虚拟主机

虚拟主机配置文件

```
[html] view plain copy print ?
01. upstream sslfpm {
02.     server 127.0.0.1:9000   weight=10   max_fails=3 fail_timeout=20s;
03. }
04.
05. server {
06.     listen       192.168.1.*:443;
07.     server_name  192.168.1.*;
08.
09.     #为一个server开启ssl支持
10.     ssl                  on;
11.     #为虚拟主机指定pem格式的证书文件
12.     ssl_certificate      /home/wangzhengyi/ssl/wangzhengyi.crt;
13.     #为虚拟主机指定私钥文件
14.     ssl_certificate_key  /home/wangzhengyi/ssl/wangzhengyi_nopass.key;
15.     #客户端能够重复使用存储在缓存中的会话参数时间
16.     ssl_session_timeout  5m;
17.     #指定使用的ssl协议
18.     ssl_protocols  SSLv3 TLSv1;
```

```
    #指定许可的密码描述
    ssl_ciphers  ALL:!ADH:!EXPORT56:RC4+RSA:+HIGH:+MEDIUM:+LOW:+SSLv2:+EXP;
    #SSLv3和TLSv1协议的服务器密码需求优先级高于客户端密码
    ssl_prefer_server_ciphers   on;

    location / {
        root   /home/wangzhengyi/ssl/;
        autoindex on;
            autoindex_exact_size    off;
            autoindex_localtime on;
    }
        # redirect server error pages to the static page /50x.html
        #
        error_page   500 502 503 504  /50x.html;
        error_page   404 /404.html;

    location = /50x.html {
            root   /usr/share/nginx/www;
        }
    location = /404.html {
            root   /usr/share/nginx/www;
        }

        # proxy the PHP scripts to fpm
        location ~ \.php$ {
        access_log  /var/log/nginx/ssl/ssl.access.log  main;
        error_log /var/log/nginx/ssl/ssl.error.log;
        root /home/wangzhengyi/ssl/;
        fastcgi_param   HTTPS   on;
            include /etc/nginx/fastcgi_params;
            fastcgi_pass    sslfpm;
        }
}
```

HTTPS服务器优化

方法

SSL操作需要消耗CPU资源，所以在多处理器的系统，需要启动多个工作进程，而且数量需要不少于可用CPU的个数。最消耗CPU资源的SSL操作是SSL握手，有两种方法可以将每个客户端的握手操作数量降到最低：

1. 保持客户端长连接，在一个SSL连接发送多个请求
2. 在并发的连接或者后续的连接中重用SSL会话参数，这样可以避免SSL握手操作。

会话缓存用于保存SSL会话，这些缓存在工作进程间共享，可以使用**ssl_session_cache**指令进行配置。1M缓存可以存放约4000个会话。默认的缓存超时时间是5m，可以使用ssl_session_timeout加大它。

ssl_session_cache指令

```
[html] view plain  copy  print  ?
语法：ssl_session_cache off|none|builtin:size|shared:name:size
使用环境：main,server
缓存类型：
off -- 硬关闭，nginx明确告诉客户端这个会话不可重用
none -- 软关闭，nginx告诉客户端会话能够被重用，但是nginx实际上不会重用它们
bultin -- openssl内置缓存，仅可用于一个工作进程.可能导致内存碎片
shared -- 所有工作进程的共享缓存。（1）缓存大小用字节数指定（2）每个缓存必须拥有自己的名称（3）同名的缓存可用于多个虚拟主机
```

优化示例

```
[html] view plain  copy  print  ?
#优化ssl服务
ssl_session_cache   shared:wzy:10m;
#客户端能够重复使用存储在缓存中的会话参数时间
ssl_session_timeout  10m;
```