HTTP&HTTPD

## http

超链接：可以在文本之间跳转的链接

HTTP : HyperText Transfer protocol: 超文本传输协议

诞生在欧洲量子实验室

web服务

http/0.9 : 仅纯文本（超链接） ASCII

HTML: HyperText Mark Language 超文本标记语言

URI ： Uniform Resouce Indentifier ,全局范围内。统一资源标识符

URL : Unifor Resource Locaction URI的子集，统一资源定位符

如：<http://www.magedu.com/download/log.gif> 资源

多个资源可以被整合一个html文档。这些资源都可以分散在不同的主机上。

web对象：

HTTP方法。 0.9 很弱，只有GET方法

HTTP1.0 post（提交） get（获取） delete （删除）

MIME ： Multipurpose Internet Mail Extention 多用途互联网邮件扩展

SMTP ：Simple Mail Transmission Protocal 最先应用于SMTP

MIME 可以将非文本数据在传输前重新编码为文本格式（二进制），接收方能够用相反的方式将其重新还原成原来的格式，还能够调用程序来打开这个文件。

如Base64编码

后来在HTTTP引入MIME。HTTP的内容从此就丰富起来了。

协议首部： image/jpeg 浏览器有插件解析这些文件格式。

动态网页：

服务器存储的文档非html格式，而是编程语言开发的脚本如php，jsp, asp 等

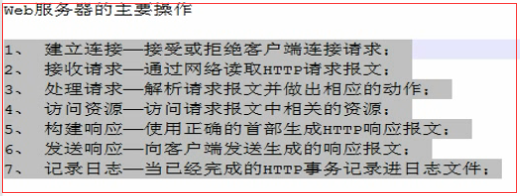
这些脚本被编译（解释）后生产html文档。在传输到客户端。



HTTP报文：

请求报文，响应报文





HTTP1.1 新特性

缓存 cache

长连接 keep-alive

单进程/单线程的机制 早期

多进程/多线程的机制

轮训机制： 水平触发 边缘处罚 事件驱动





## httpd

apache

NCSA, httpd进程，

web服务器早期的代表，

FSF : GNU,GPL

ASF:Apache software Foundation

FSF ASF 两大最著名的软件基金会

web: httpd

tomcat

hadoop 并行处理高性能的计算框架

httpd是一个纯粹的web server

2.2 2.3 2.4 1.3

httpd特性

1. 事先创建进程
2. 按需维持适当的进程
3. 模块化升级，核心比较小，各种功能都模块添加（包括php），支持运行时配置，支持单独编译模块
4. 支持虚拟主机，虚拟主机：物理服务器只有一个，web程序只有一个，却可以服务多个不同的站点。

IP + 端口 = socket

虚拟主机：

基于IP的虚拟主机 昂贵

基于端口的虚拟主机

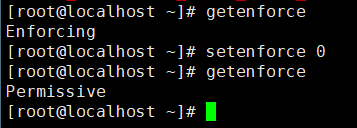
基于域名的虚拟主机

支持https协议，支持用户认证，支持基于IP或者主机名的ACL,支持每目录的访问的控制，支持URL重写。

## Install httpd

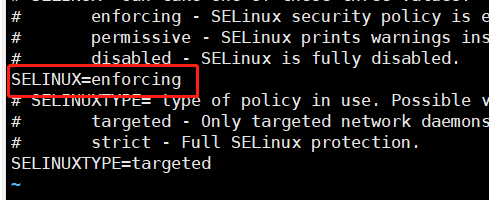
rpm 源码编译定制安装

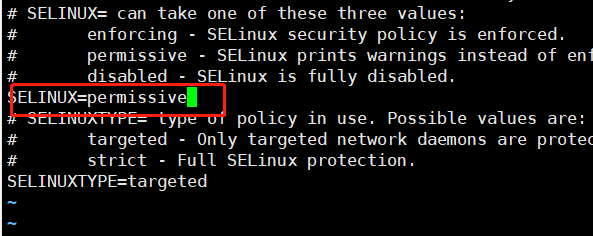
事先确保Selinux为关闭状态，这里设置后下次启动又会启动



当然也可以 vim /etc/sysconfig/selinux 或者

Vi /etc/selinux/config





/user/sbin/httpd（MPM:prefork）

httpd root root (master process)

httpd apache apache(worker process)

## config httpd

/etc/rc.d/init.d/httpd 红帽中具有的特性，可以控制httpd进程的启动

port: (tcp:80) (https:443)

/etc/httpd 工作根目录，相当于程序安装目录

/etc/httpd/conf : 配置文件目录

主配置文件：/etc/httpd/conf/httpd.conf

/etc/httpd/conf.d/\*.conf 也是组成部分 httpd.conf包含进来

/etc/httpd/modules 模块目录是一个链接，

/etc/httpd/logs ----->/var/log/httpd 日志目录

日志文件分为两类 ：访问日志 access\_log , 错误日志：err\_log

/var /www

html

cgi-bin 提供动态内容

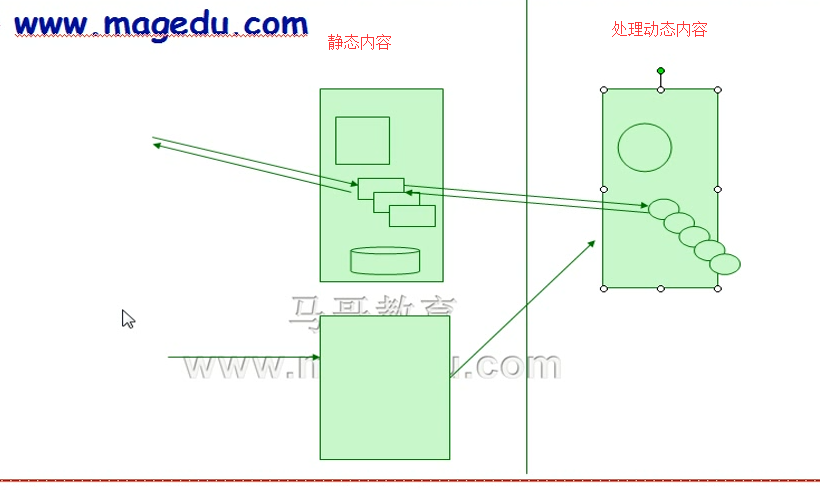
CGI ：Common Gateway Interface , 通用网关接口（协议）启动一个应用程序来处理脚本（动态），生成html资源

实现了CGI的语言： java（servlet）, perl , php , pathon，C#等

一个master process来专门负责进程的销毁和创建。称为FASTCGI

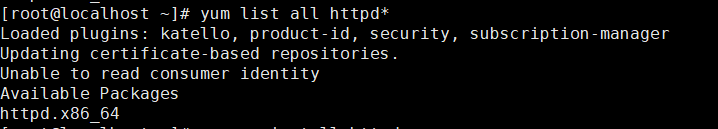
FastCgi工作在某个套接字上，专门维护进程的进程，可以分离静态内容和动态内容。

处理动态内容的服务器称为应用程序服务器。可以动静分离。静态服务器专门负责静态资源的读取和返回，如图片，js，css等

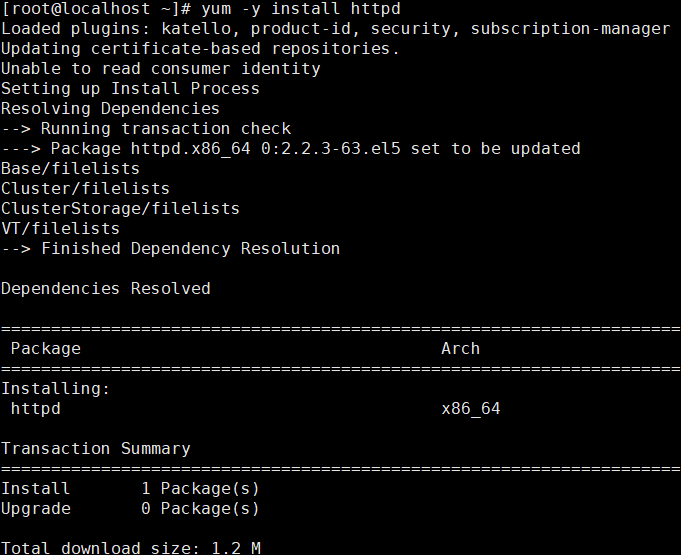


httpd安装：

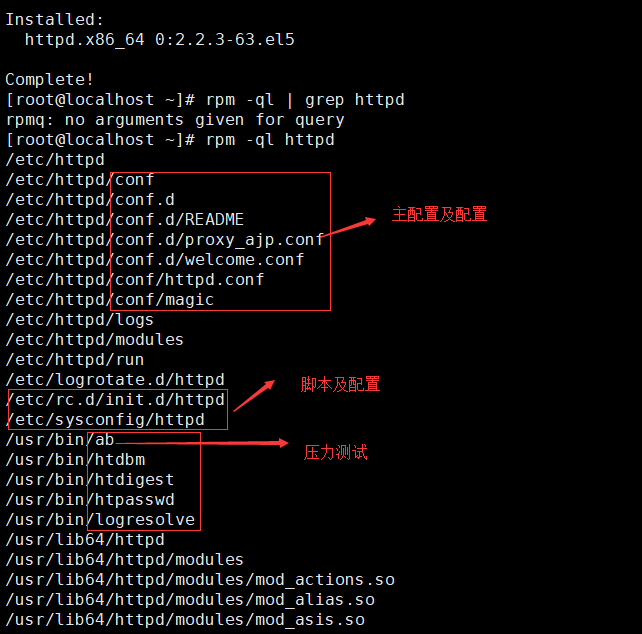
mount –r /dev/cdrom /mnt/cdrom

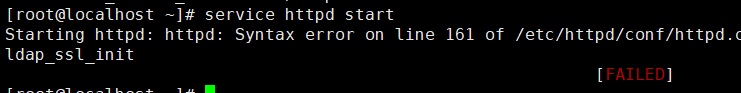


Yum –y install httpd



rpm –ql | grep httpd

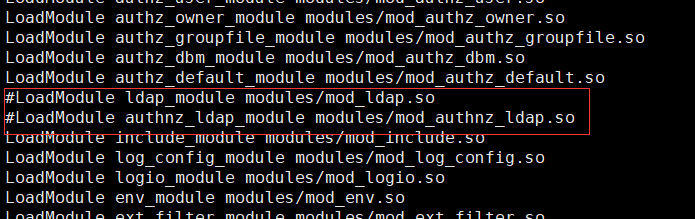


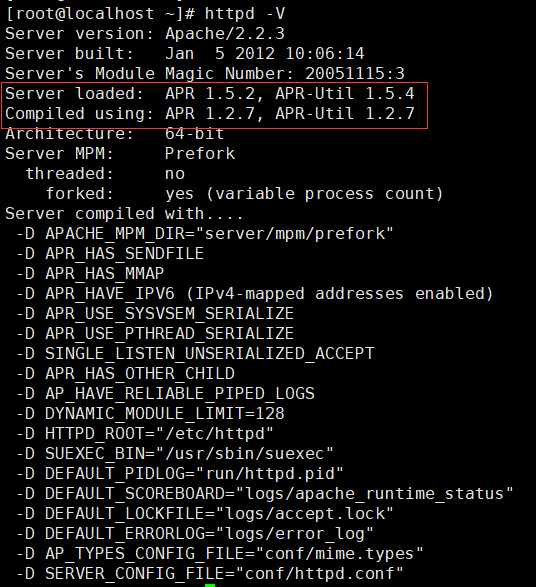


## 无法启动:undefined symbol: apr\_ldap\_ssl\_init

[root@test ~]# /etc/init.d/httpd start  
Starting httpd: httpd: Syntax error on line 161 of /etc/httpd/conf/httpd.conf: Cannot load /etc/httpd/modules/mod\_ldap.so into server: /etc/httpd/modules/mod\_ldap.so: undefined symbol: apr\_ldap\_ssl\_init

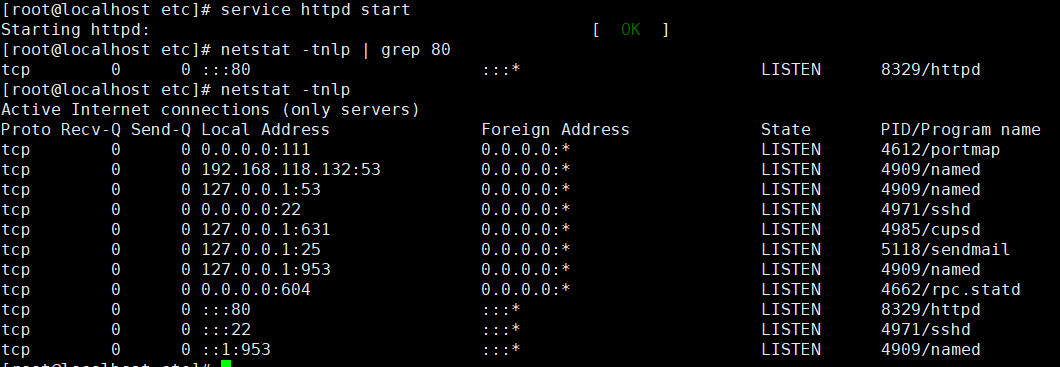
以前好用的好好的，重启httpd后，竟然无法启动了。。。  
根据错误提示，注释掉/etc/httpd/conf/httpd.conf中的“LoadModule ldap\_module modules/mod\_ldap.so  
LoadModule authnz\_ldap\_module modules/mod\_authnz\_ldap.so”后，htppd启动正常。

vi /etc/httpd/conf/httpd.conf

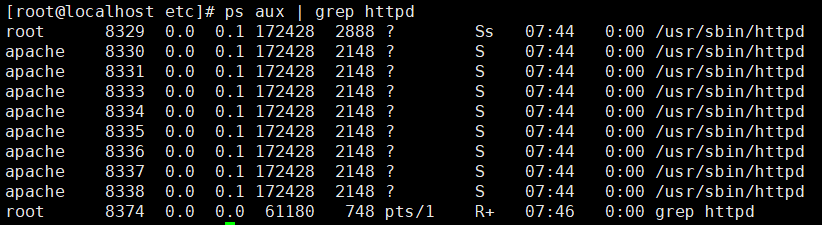
[root@test ~]#httpd -V  


原来是APR的问题（有多个路径不同的apr）。查看LD\_LIBRARY\_PATH，并没有异常。查看/etc/ld.so.conf，发现多了一个自定义路径，/usr/local/test  
 而这个路径下有libapr\*相关的文件，注释掉这个路径后，ldconfig一下，OK，httpd能正常启动了。

netstat –tnlp | grep 80

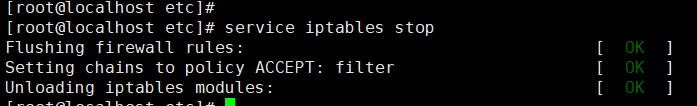


ps aux | grep httpd



关掉防火墙：

service iptables stop



启动成功

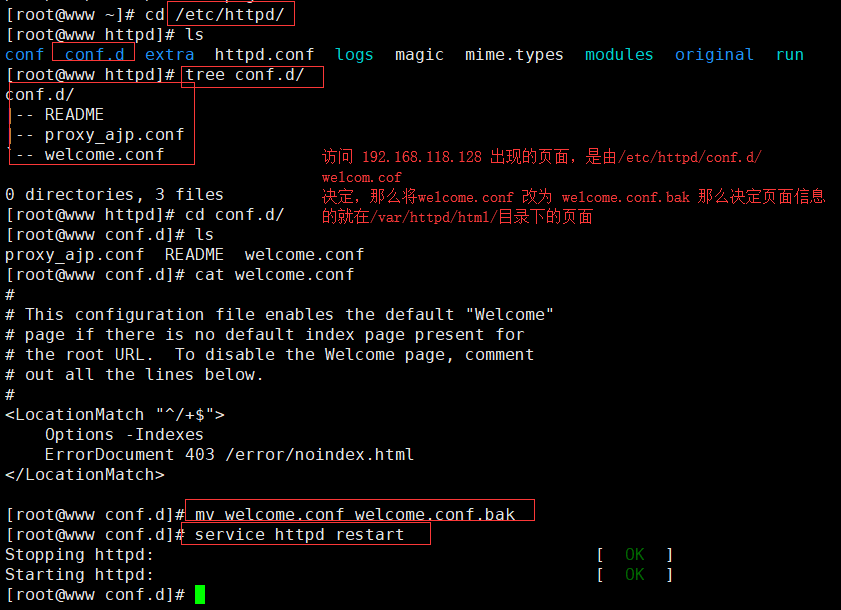


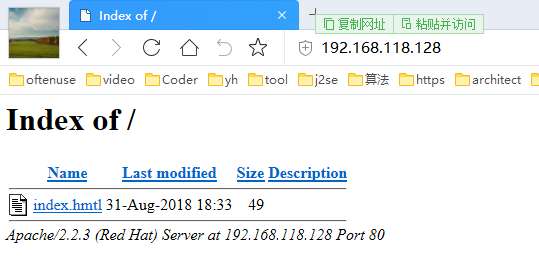
## Httpd config

http配置后测试有没有配置正确

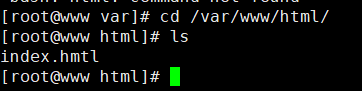
httpd –t

然后重启：service httpd restart





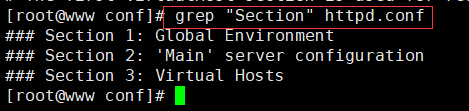
/var/www/html





配置文件分为3段：

grep “Section” httpd.conf



(1)全局段

(2)主服务器配置

(3)虚拟主机

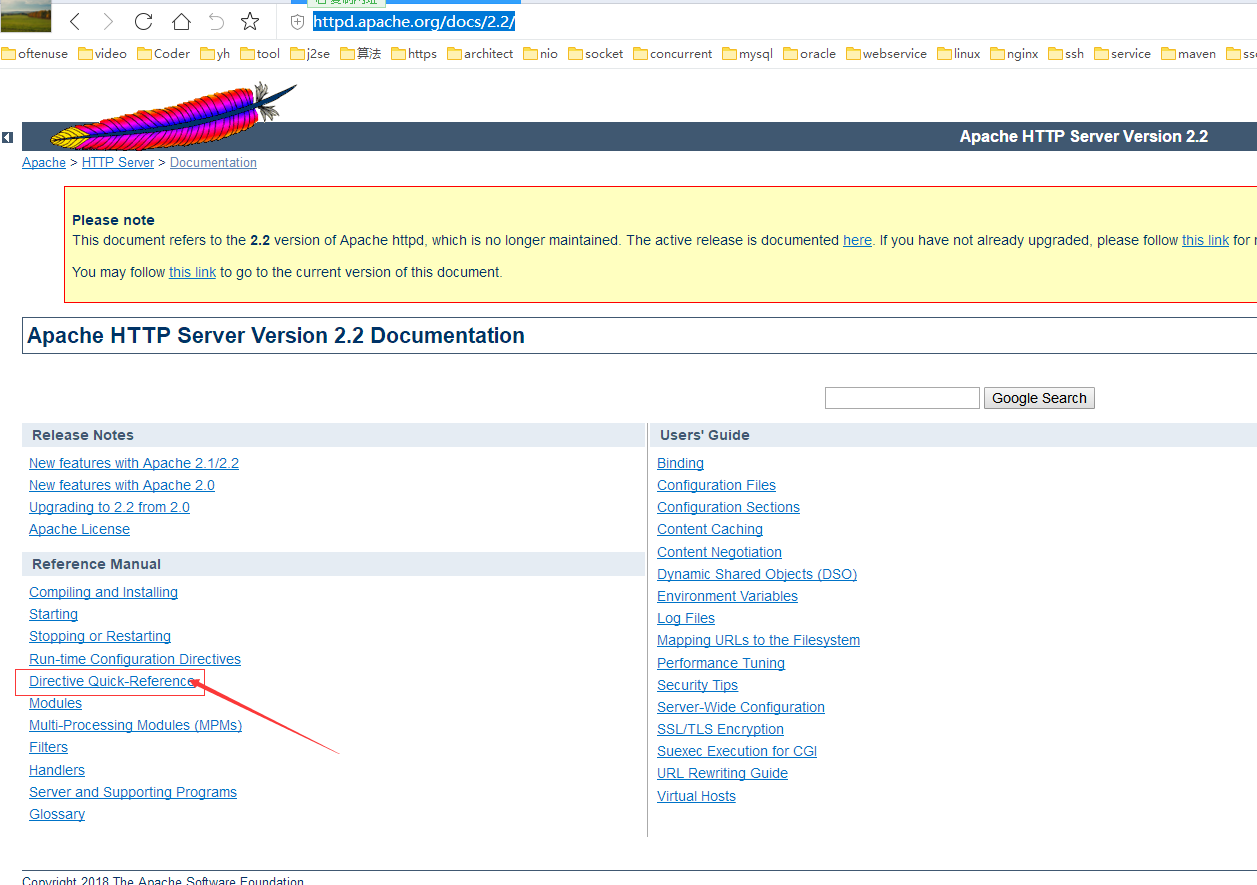
(2)和(3)不能同时使用，这里主服务器生效。虚拟主机能提供多个站点。

格式：指令 值

指令分大小写，值根据需要区分。

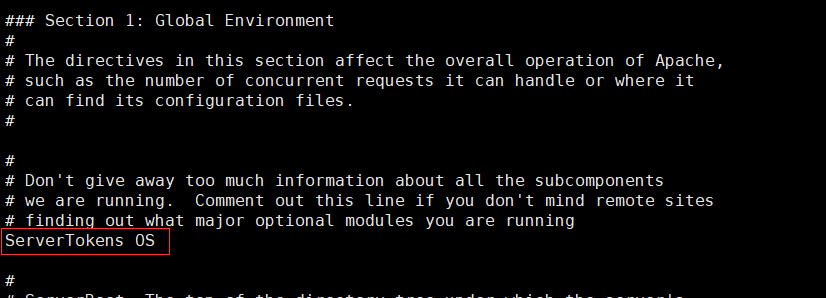
### Section 1 Gloabal Environment

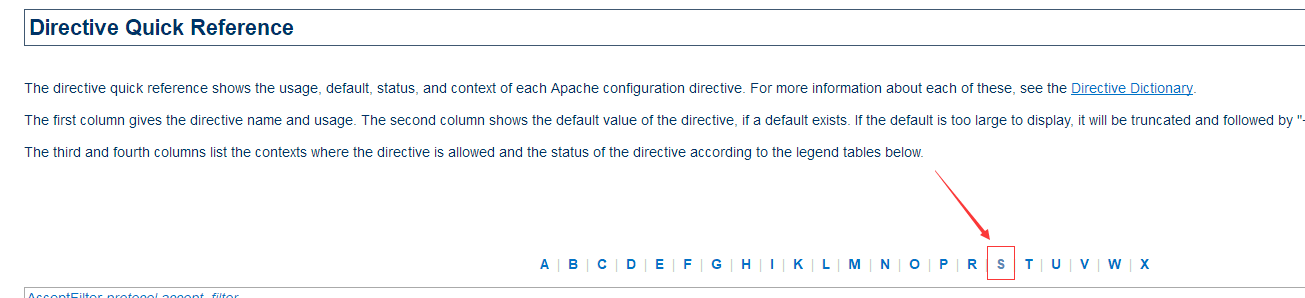
配置的值在 <http://httpd.apache.org/docs/2.2/> 文档

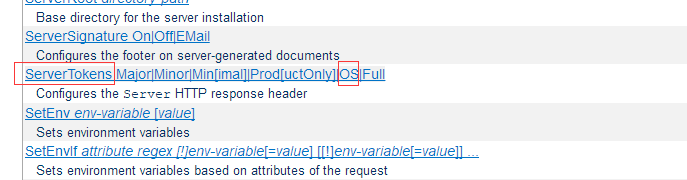


#### ServerTokens

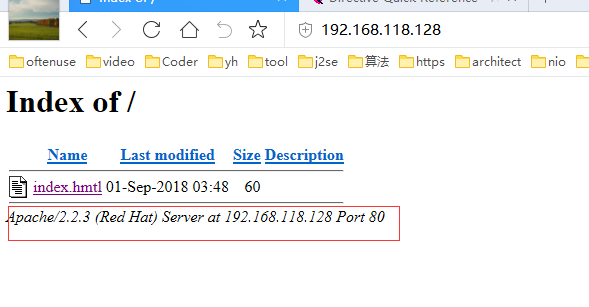
如果要找 ServerTokens 配置哪些值





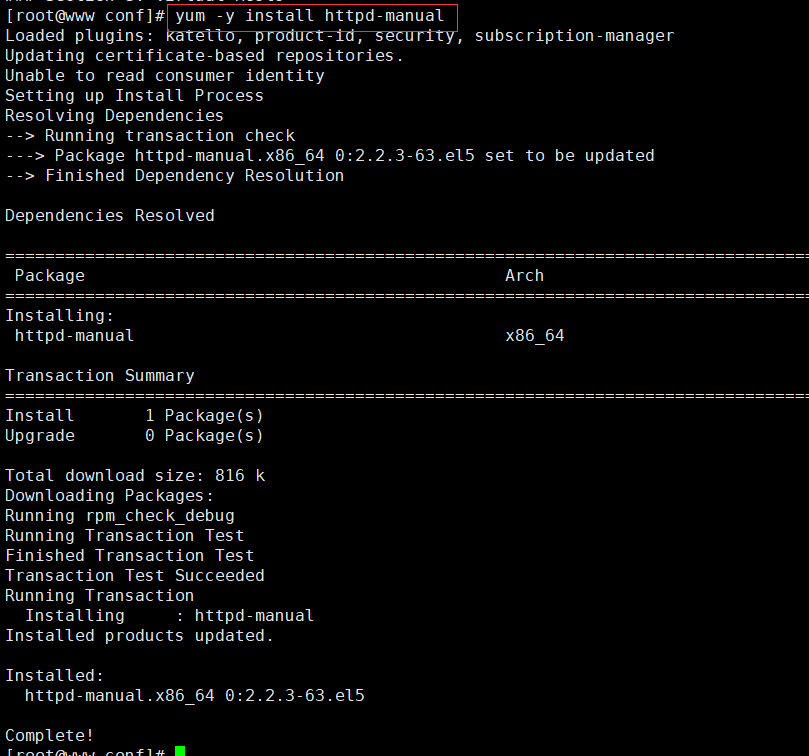


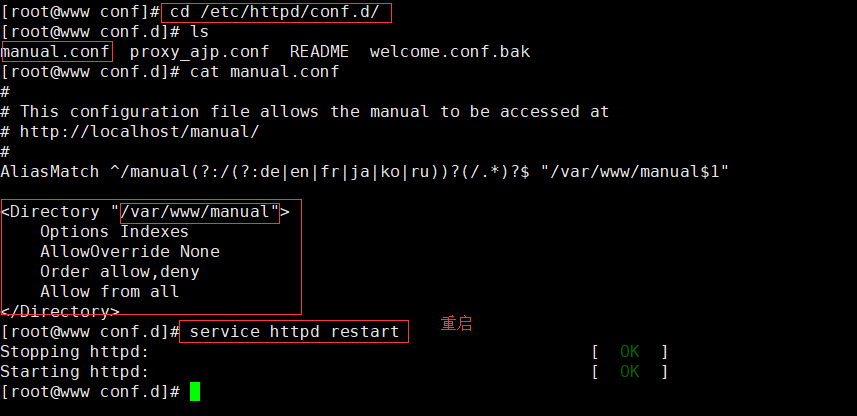
如果配置OS显示



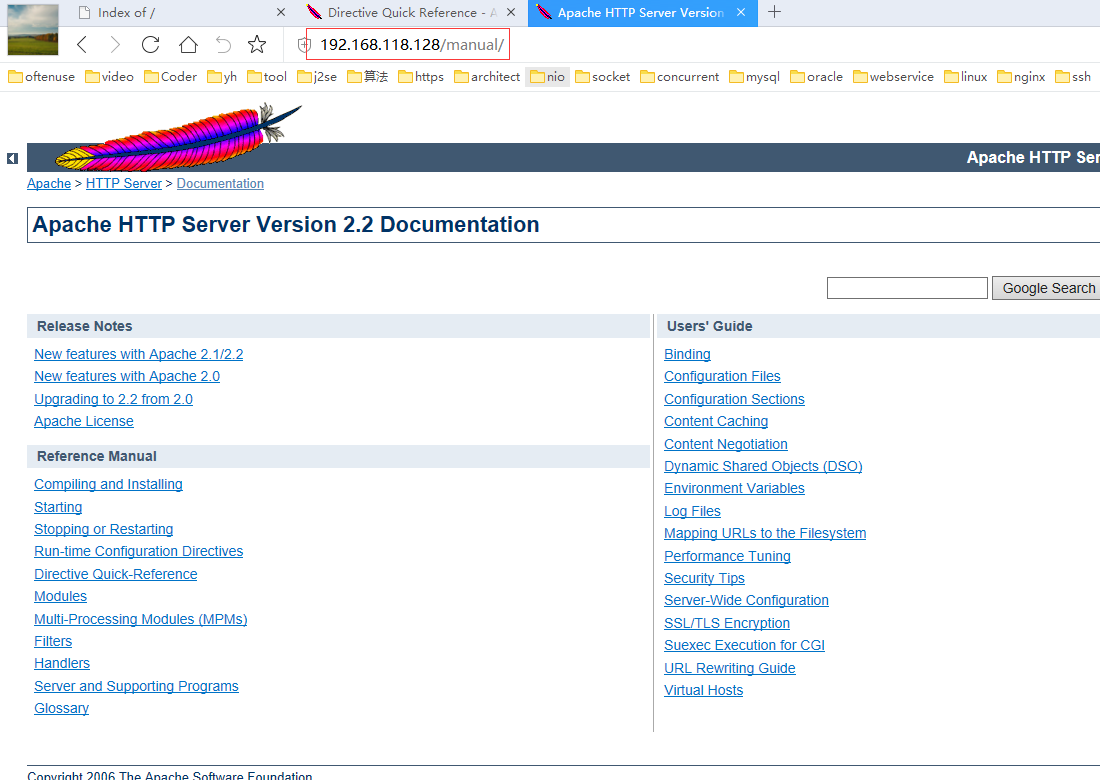
FULL 全部显示

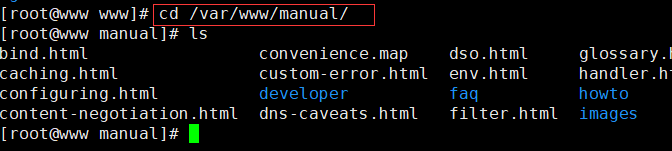
也可以装一个 httpd-manual



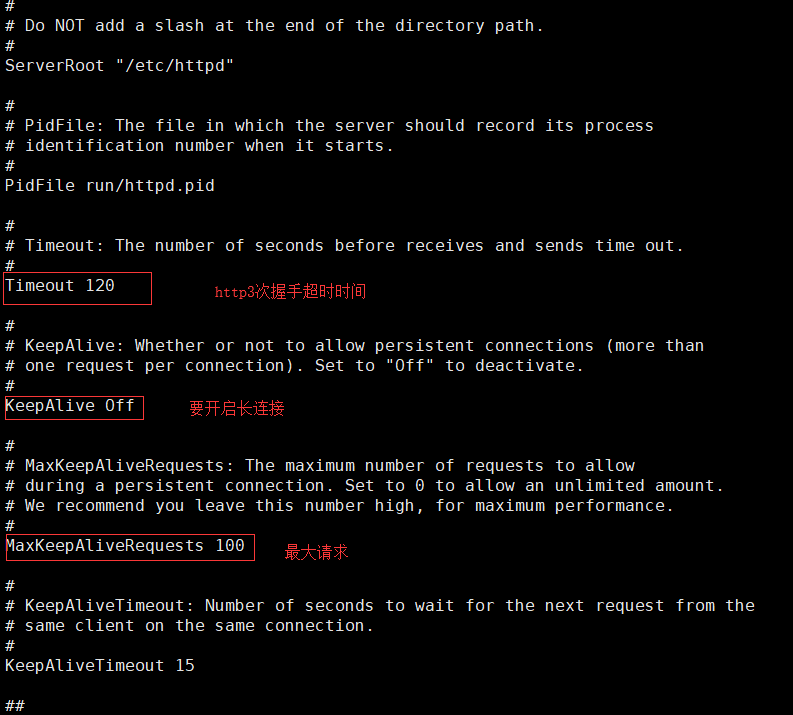


然后访问： http://192.168.118.128/manual/





#### Timeout



#### keepAliveTimeout

超时时间 15秒

#### MPM, Multi Path Modules

MPM：Multi Path Modules 多道处理模块

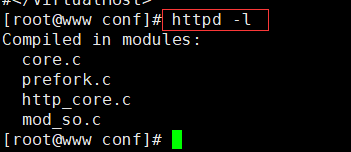
Mpm\_winnt windows上专用

Mpm\_prefork 预先生成进程，linux上专用 （一个进程用一个进程响应）apache2.2及以下，稳定可靠。性能比Event要差。

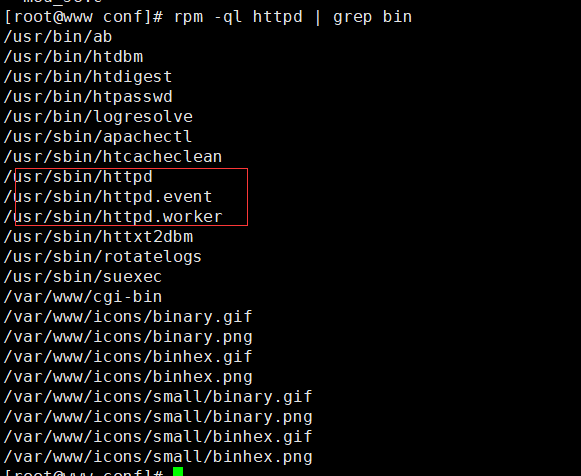
Worker （一个进程用一个线程响应，启动多个进程，每个进程多个线程）一个进程有多个线程，效率高，存在线程资源竞争（写的时候，要加锁，存在锁竞争）的情况下。Linux不是原生态支持线程，效率未必有prefork高。

Evnet 一个进程处理多个请求，基于事件驱动模型 apache2.4及以上默认使用，很强大。nginx也是使用event这种机制。

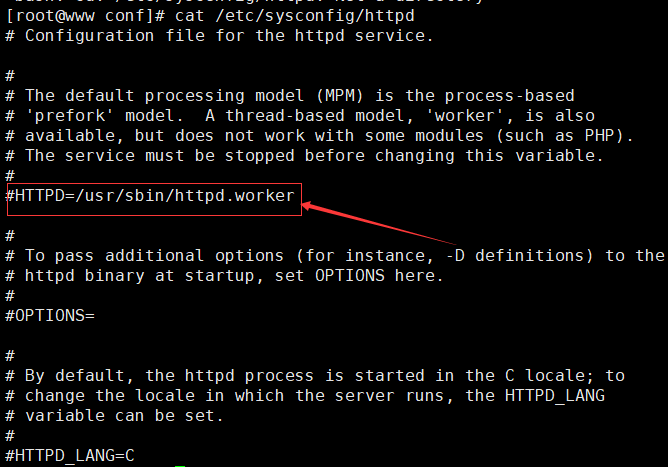
Httpd –l



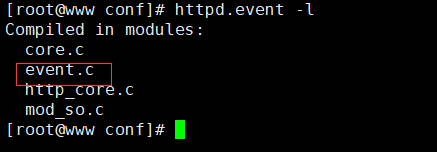
Rpm –ql httpd | grep bin



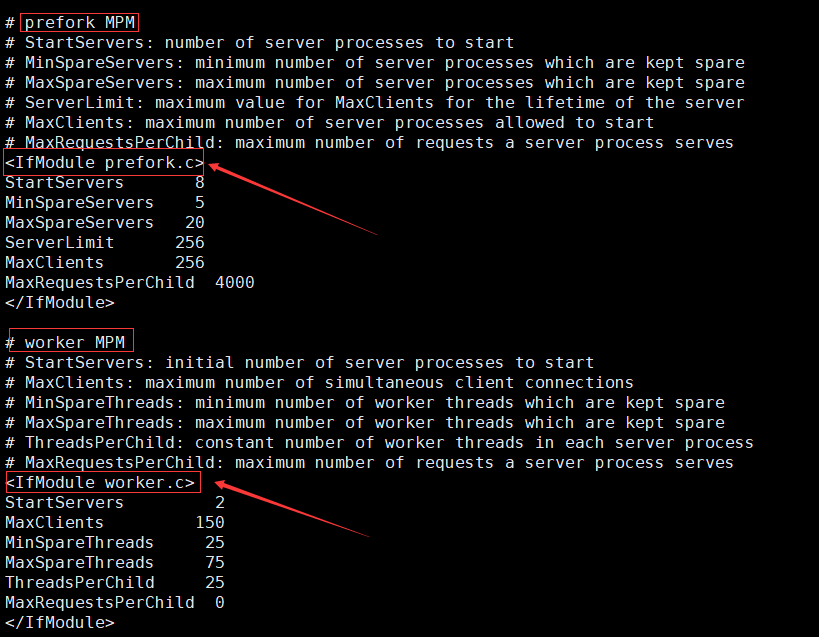
如果要使用可以在/etc/sysconfig/下的httpd文件中启用 #HTTPD=/usr/sbin/httpd.worker



Httpd.event -l



Apache2.2不建议使用httpd.event



Prefork模型

<IfModule prefork.c>

StartServers 8 服务器启动时，启动的空闲进程

MinSpareServers 5 最少空闲进程

MaxSpareServers 20 最大空闲进程，如果500个用户同时进来，请求结束，就产生很多空闲进程，浪费资源，所有要杀掉，最大只保留20

ServerLimit 256

MaxClients 256 最大客户端数。最多有多少个请求同时连进来。如果超过256就等待，如过要调整，那么 ServerLimit = MaxClients =512 ，httpd进程全部kill掉重启

MaxRequestsPerChild 4000 一个进程最多响应4000次，达到4000次数就kill掉。

</IfModule>

#### Work模型

<IfModule worker.c>

StartServers 2

MaxClients 150 最大客户端数量

MinSpareThreads 25 最小空闲线程

MaxSpareThreads 75 最大空闲线程

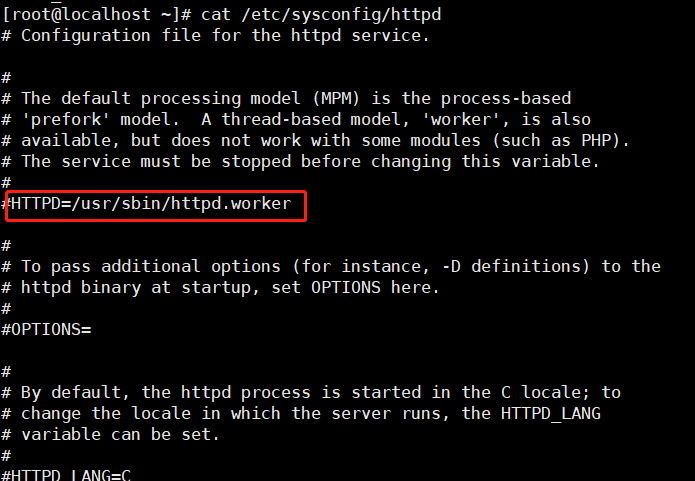
ThreadsPerChild 25 一个进程生成多少个线程

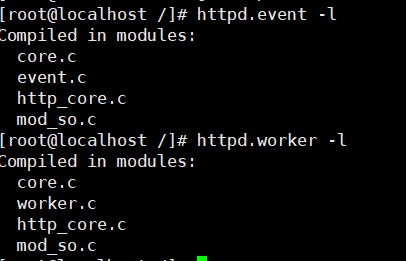
MaxRequestsPerChild 0

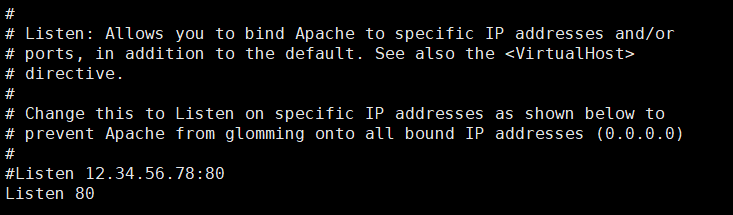
</IfModule>

如何定义使用的什么模板：

/etc/sysconfig/httpd







#

# Listen: Allows you to bind Apache to specific IP addresses and/or

# ports, in addition to the default. See also the <VirtualHost>

# directive.

#

# Change this to Listen on specific IP addresses as shown below to

# prevent Apache from glomming onto all bound IP addresses (0.0.0.0)

#

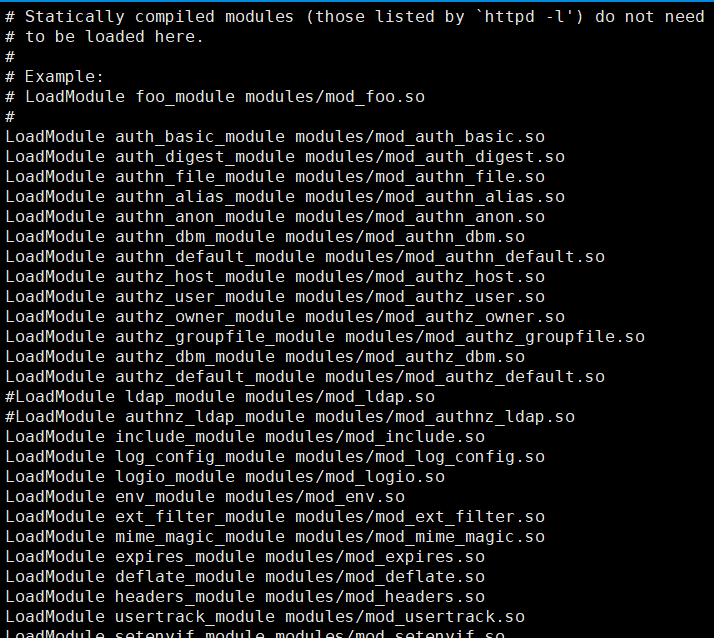
#Listen 12.34.56.78:80

Listen 80

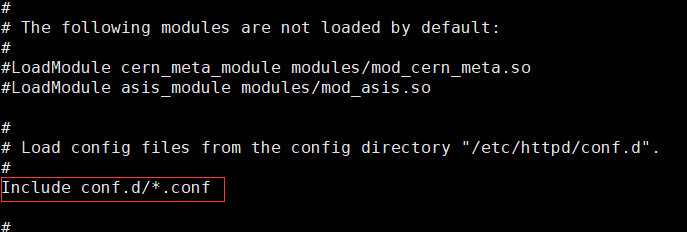
Listen 192.168.118.128:8080

这里可以配置多个端口

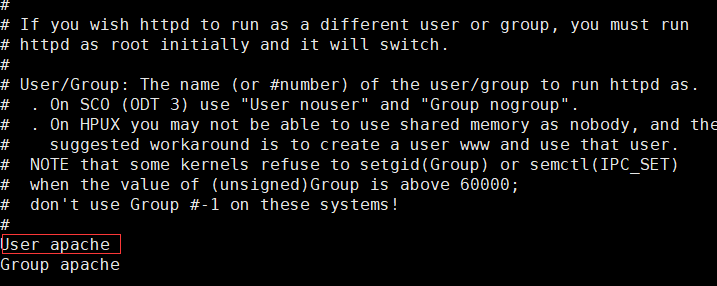
加载的模块



包含/etc/httpd/conf.d目录下的配置文件



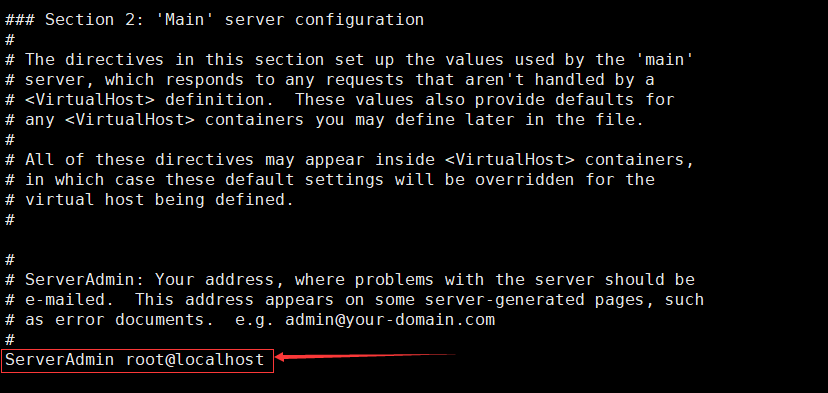
都以apache普通用户运行。



### Section 2 Main Server Config

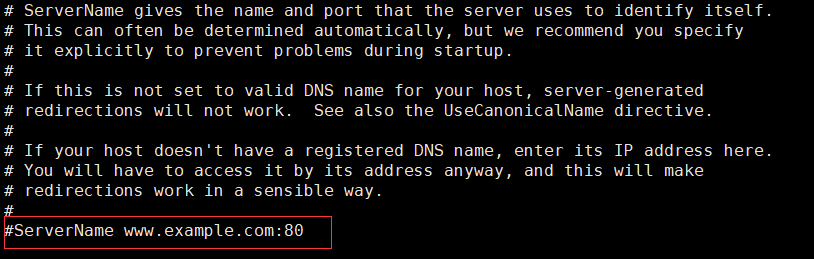
#### ServerAdmin

服务器管理员，每一个站点都需要一个



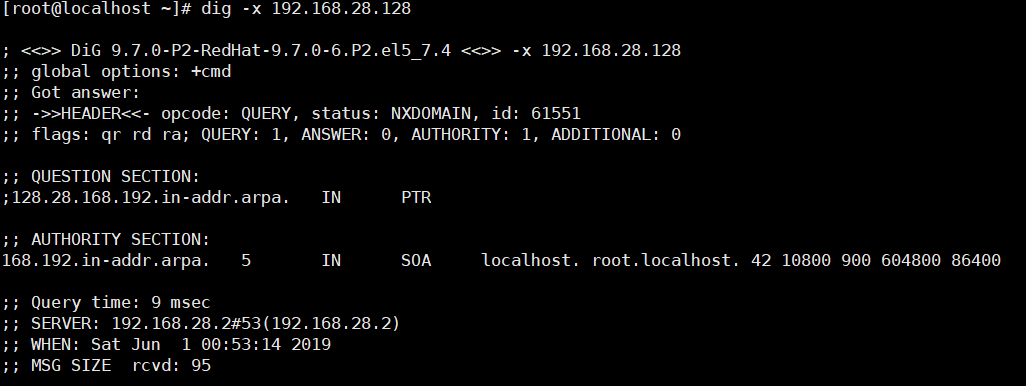
#### ServerName

服务器名称，服务启动的时候，DNS能够解析，虚拟主机的时候，这项必须配置。

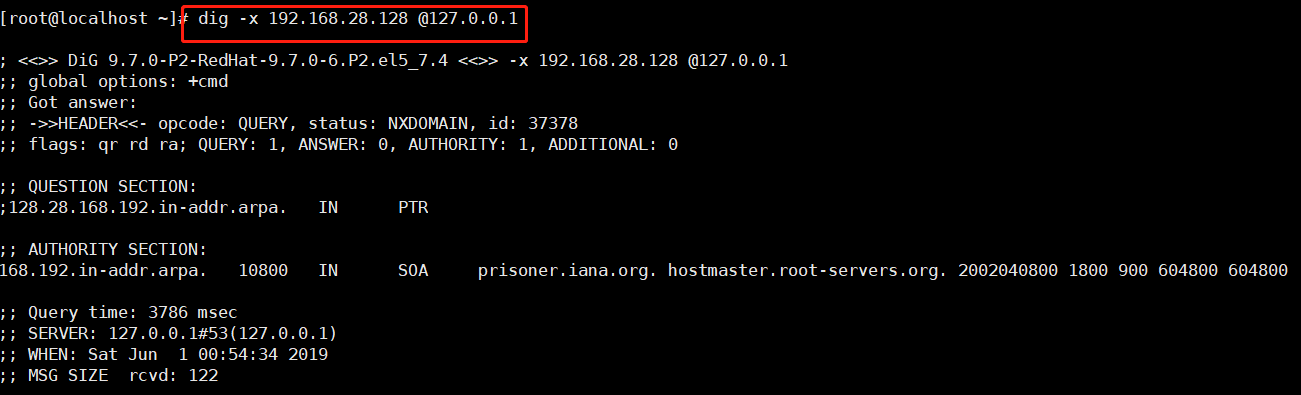


vi /etc/hosts 中配置

DNS能够解析 dig –x 192.168.28.128



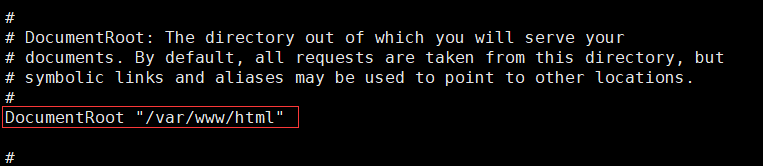
dig -x 192.168.28.128 @127.0.0.1



#### DocumentRoot

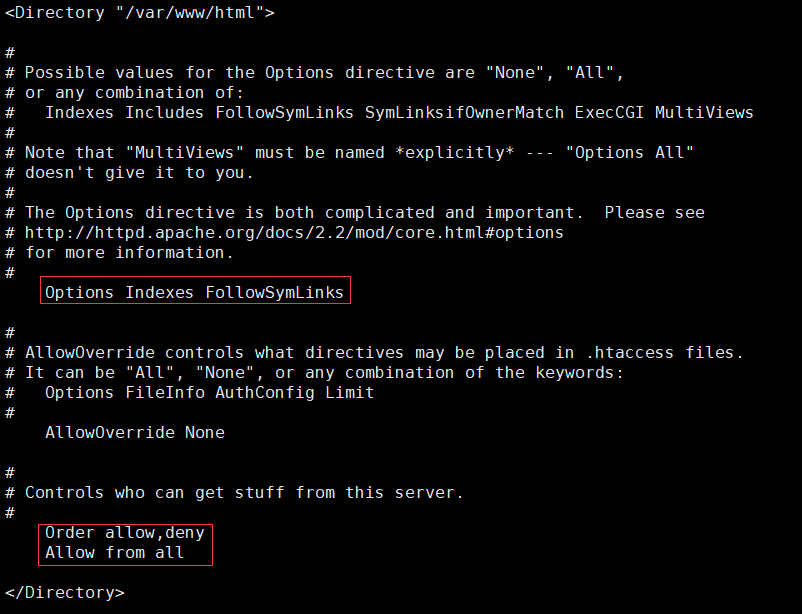
文档根目录，可以自己配置（相对路径：192.168.118.128/） / = /var/www/html

URL是相对于DocumentRoot而言的



<Directory>节点配置与DocumentRoot配置的路径配置相对应，可以配置/var/www/html的访问属性

#### Direcotry



Options 后面可以配置多个用空格隔开。

None: 不支持任何选项

Indexes 允许索引目录 生成环境中一定要关闭，文件服务器可以打开

Includes 允许执行服务端 包含（SSI）

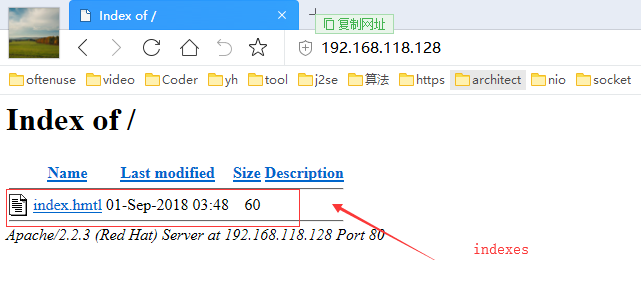
FollowSymLinks 允许访问符号链接指向的源文件

SymLinksifOwnerMatch

ExecCGI 允许CGI脚本

MultiViews 国际化

ALL：支持所有



这种做法，不能应用于生成环境。Options none

AllowOverride None 允许覆盖，覆盖下面服务器访问控制列表（Order allow）

Ordery 用于定义主机的访问功能的，IP，网络地址或主机定义的访问控制列表

Order allow,deny

Allow from all

Deny from all

例子：

仅仅允许 192.168.0.0/24 网络访问

Order allow,deny -- 除了这个网络其他都deny

Allow from 192.168.0.0/24

不允许这个网络访问

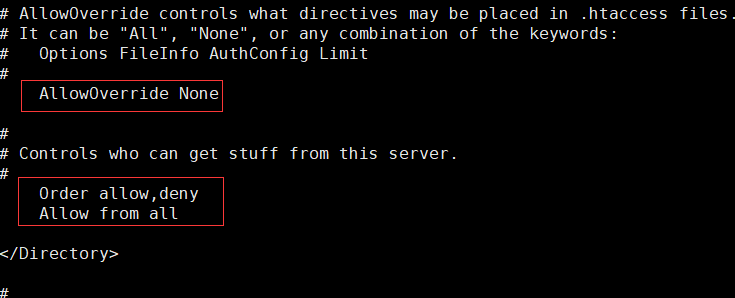
Order deny,allow -- deny是默认值,这里要改下次序，除了这个网段，其他都allow。

deny from 192.168.0.0/24

例如：我要拒绝 192.168.1.101 ，172.16.100.177

Ordery deny,allow

Deny from 192.168.1. 101 172.16.100.177



Options

FileInfo

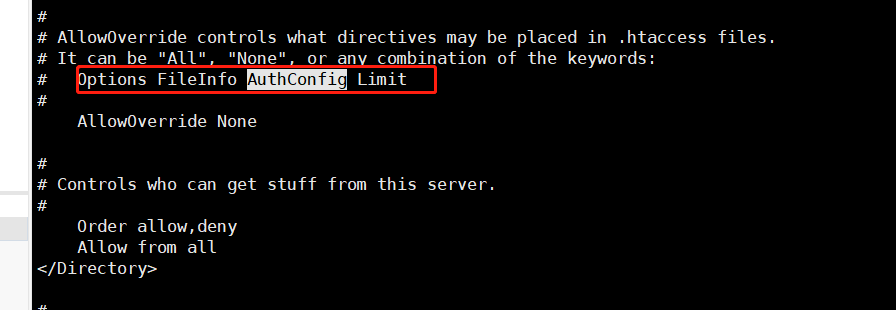
AuthConfig 常用 验证配置文件，

Limit

None

#### AllowOverride

覆盖 (Order , Allow)，用 AuthConfig来控制



##### 第一种基于用户

AllowOverride AuthConfig

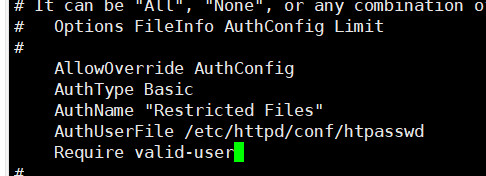
AuthType Basic

AuthName "Restricted Files"

AuthUserFile /etc/httpd/conf/htpasswd -- 密码

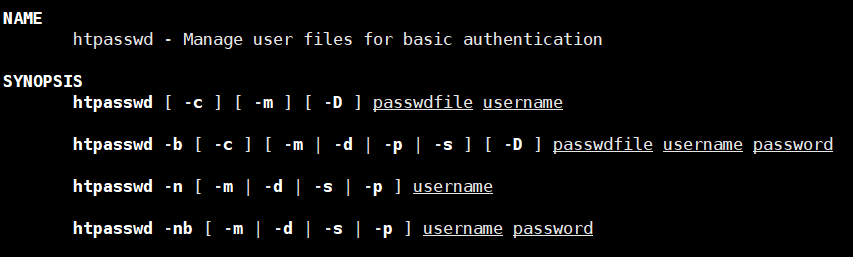
#Require user tom-- 只有tom用户可以登陆

Require valid-user

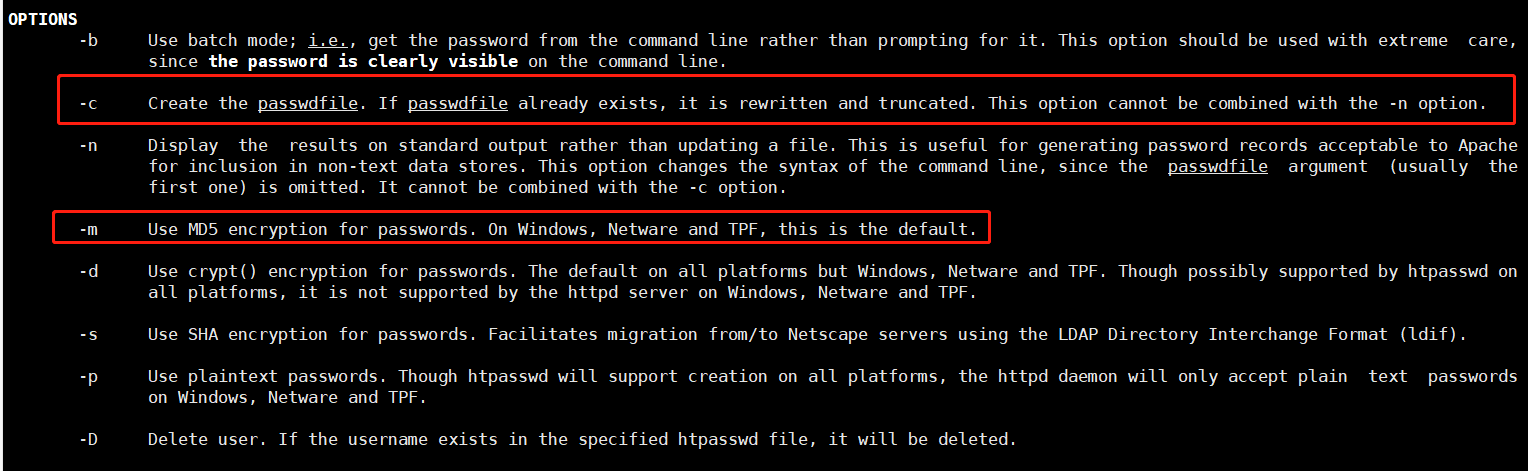


创建用户和密码：htpasswd

man htpasswd



Htpasswd 第一次创建文件用-c create，以后创建用户就不要使用-c –m使用MD5加密

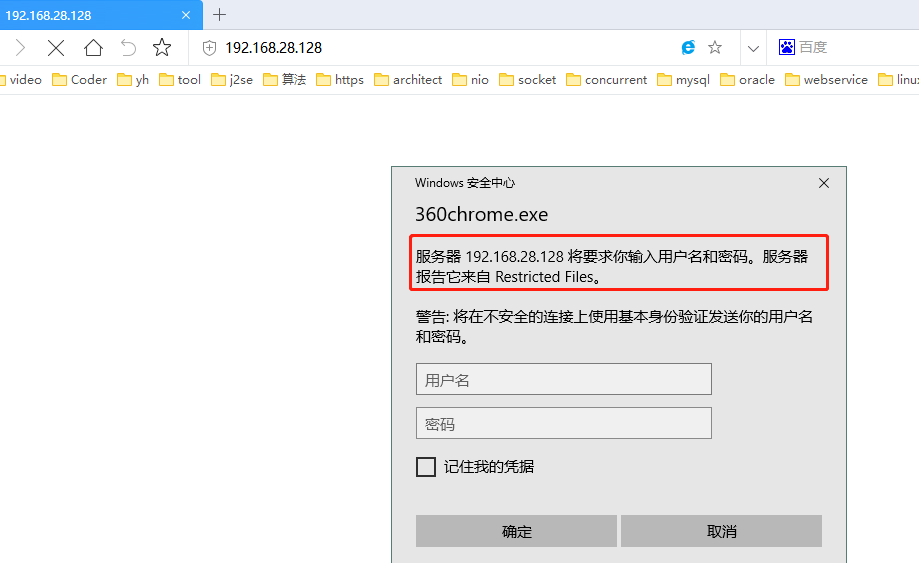


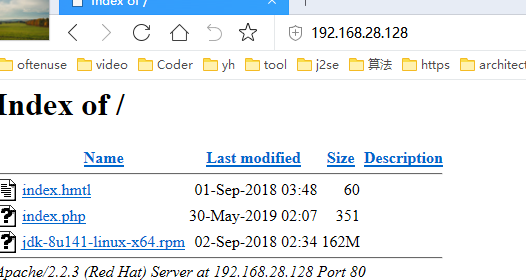
htpasswd -c -m /etc/httpd/conf/htpasswd tom



tom 123456

jerry 123456





##### 第二种基于组

AllowOverride AuthConfig

AuthType Basic

AuthName "Restricted Files"

AuthUserFile /etc/httpd/conf/htpasswd -- 密码

AuthGroupFile /etc/httpd/conf/htgroup – 并不存在需要自己建

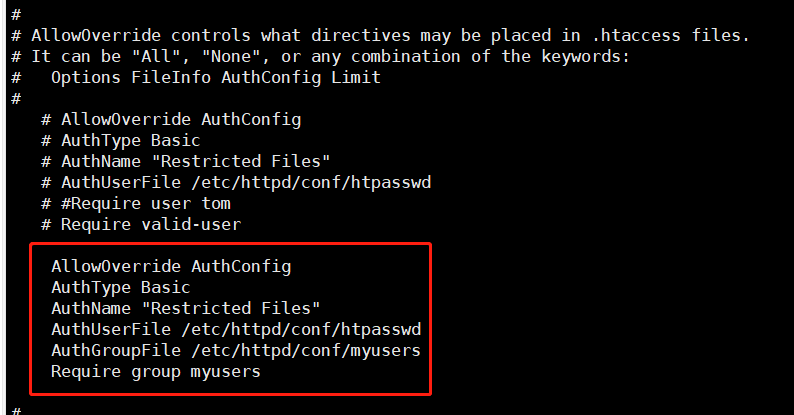
Require group myusers



用户需要使用htpasswd 先定义好



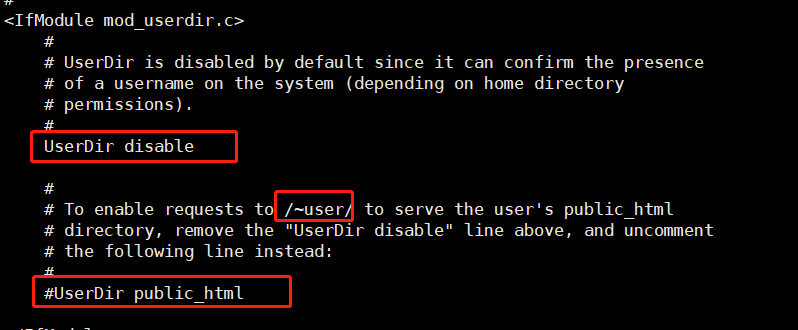




Htpasswd –D jerry 删除用户

#### Mod\_useridr

用户家目录中定义用户自己的网页

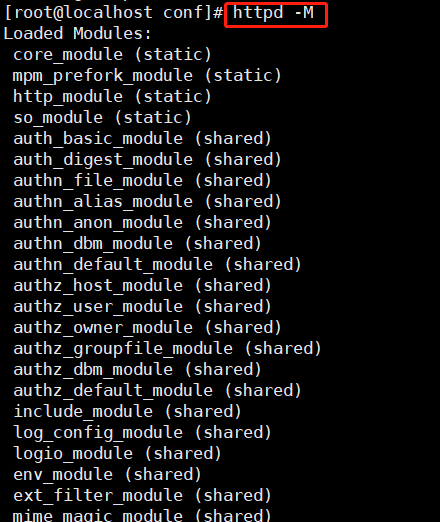


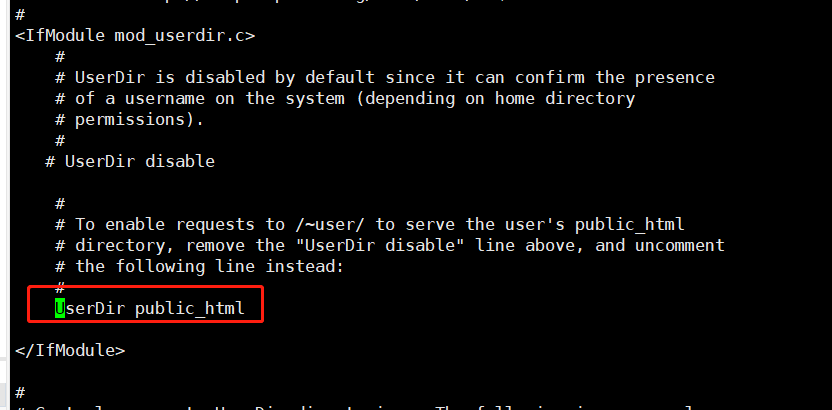
如果定义了 <http://192.168.28.128/~tom> 就可以访问自己的网页

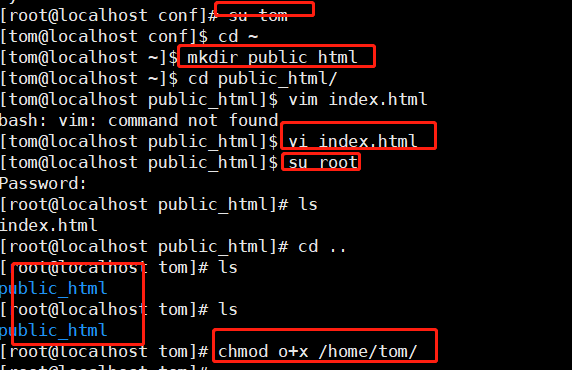
UserDir public\_html

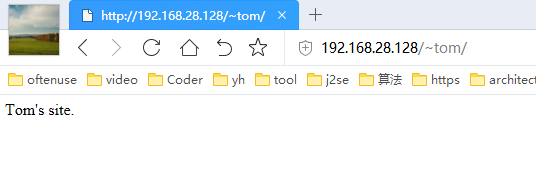
Httpd –M

额外装载的模块：



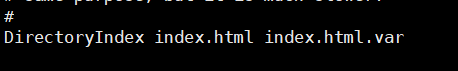






#### DirectoryIndex

默认访问页面



PV：页面访问量 每天统计

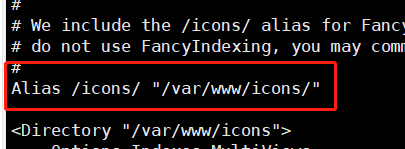
UV：用户访问量 每天的独立IP访问量

#### Alias

路径别名

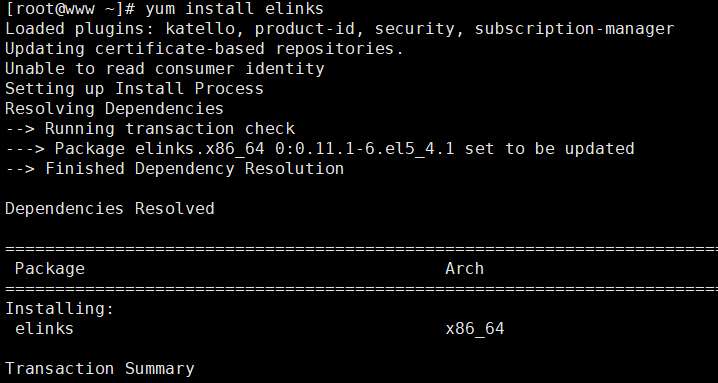
Alias /luntan/ “/bbs/forum/” 或者

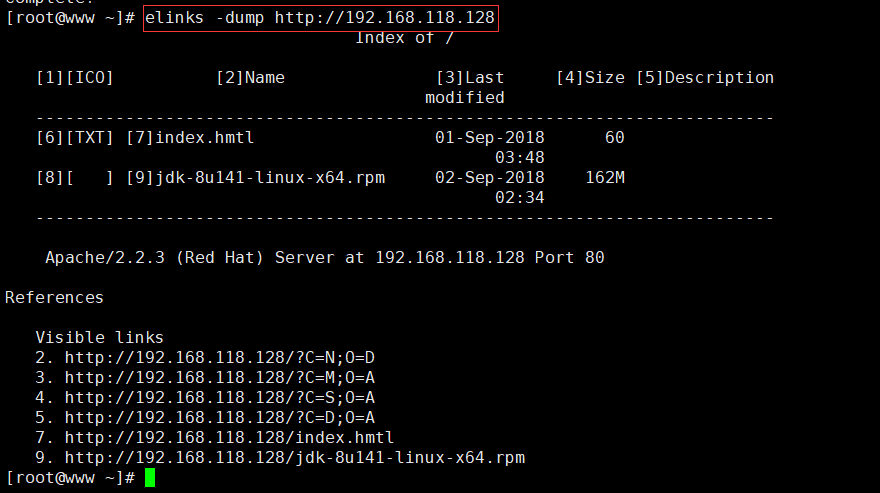
Alias /luntan “/bbs/forum” 要么都要/ 或者都不要



#### Elinks

纯文本下访问网页。elinks <http://192.168.118.128>

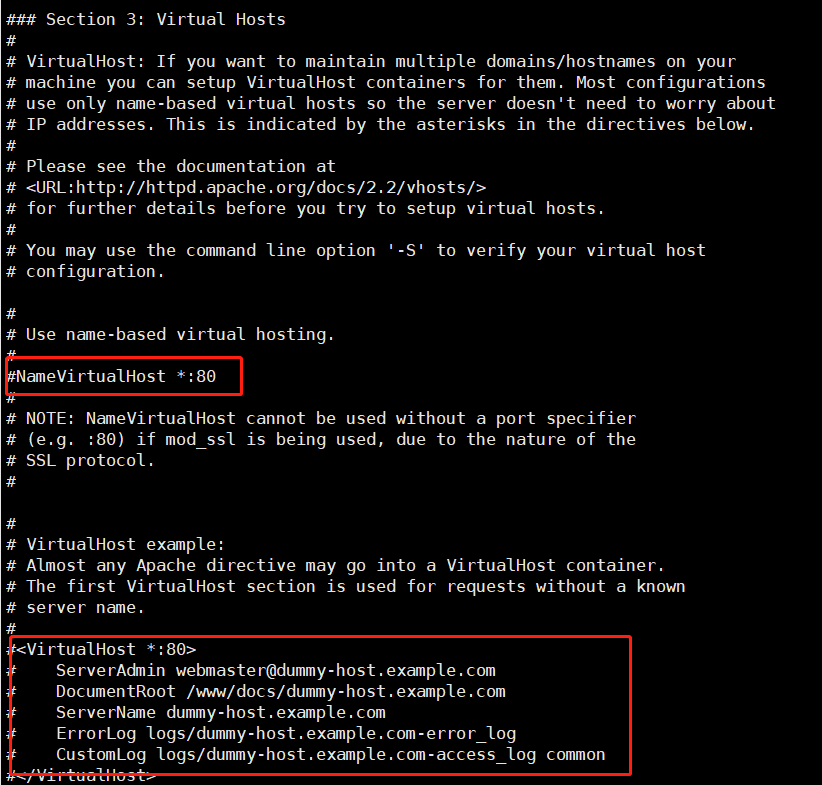




### Section 3 Virtual Hosts

Apache服务于多个不同的站点，提供不尽相同的资源访问

中心主机和虚拟主机不能同时使用



虚拟主机：

基于IP （昂贵）

IP1：80

IP2：80

基于端口：（公司内部使用，对外部提供站点不方便用户）

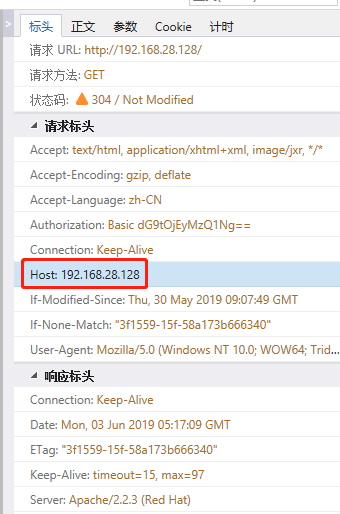
IP： 8080

IP： 8081

基于域名：

主机名不同： [www.hc.com](http://www.hc.com) , [www.hc.org](http://www.hc.org), [www.hc.net](http://www.hc.net)

HTTP首部： HOSTS节点，会携带 主机信息



Apache2.2 需要启用 NameVirtualHost

Apache2.4则不需

虚拟主机对外提供不同的访问站点,最起码有以下不一样的配置.

1. serverName 服务器名称
2. ServerAlias 服务器别名
3. DocumentRoot 文档根目录
4. 既然DocuemntRoot不一样,那么相应的directory也不相同.

<Directory [www.hc.com](http://www.hc.com)>

Option

AllowOverride

</Directory>

1. Alias 路径别名
2. 访问日志不一样, errorLog, customLog
3. <Location “/imags”>

定义获取的方法

</Location>

要使用虚拟主机,得先注释中心主机,注释DocumentRoot

基于域名:

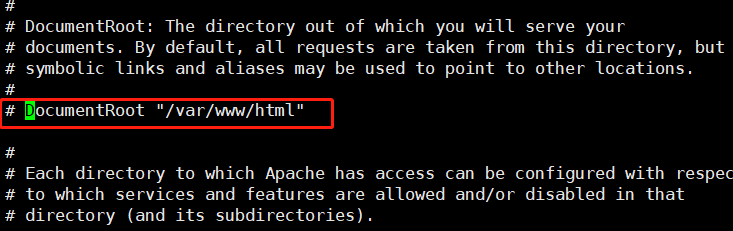
\*80

ServerName 不同

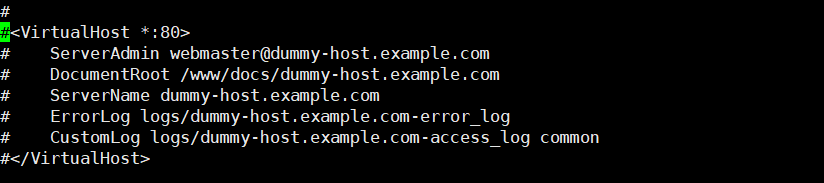
#### 如何配置虚拟主机

##### 基于IP的虚拟主机

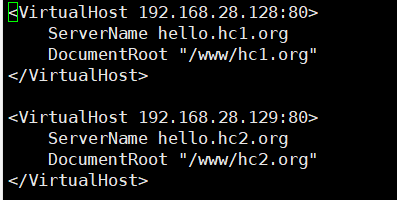
1.首先要取消中心主机，将DocumentRoot节点注释



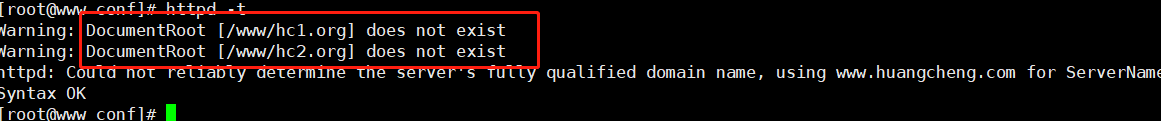
2.配置，要么打开VirtualHost进行配置，要么自己定义文件配置，这里我们自行配置，在 /etc/httpd/conf.d/目录下定义一个virtual.conf文件



Vi ../conf.d/virtual.conf

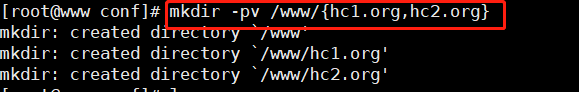


Httpd –t 语法分析



DocumentRoot不存在，手动创建目录

Mkdir –pv /www/{hc1.org,hc2.org}

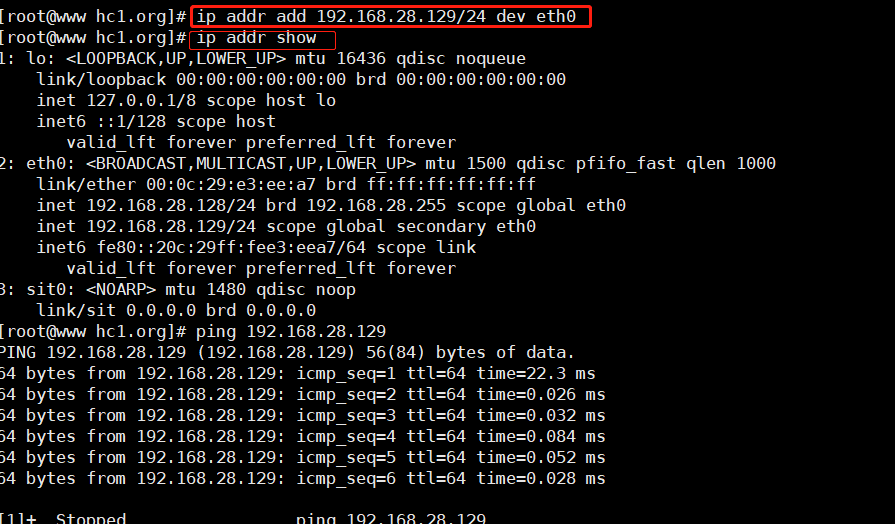


分别创建index.html

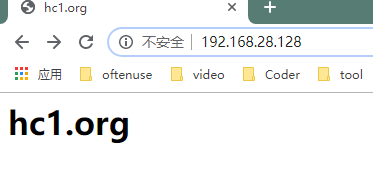


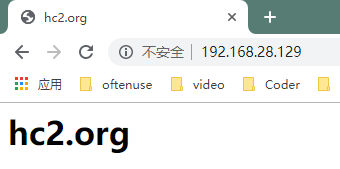
添加一个地址： ip addr add 192.168.129/24 dev eth0

查看： ip addr show



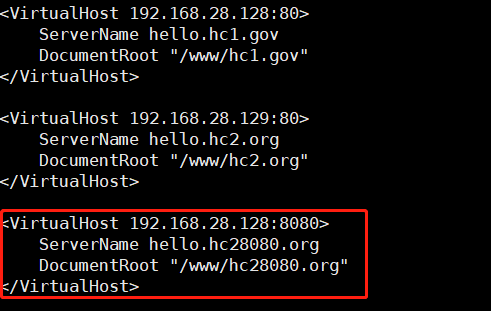
重启 service httpd restart





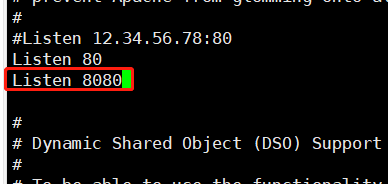
##### 基于端口的虚拟主机

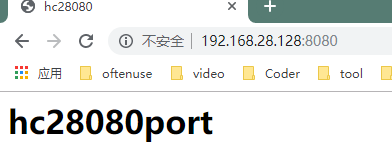
再添加一个基于端口的虚拟主机 （复制 4yy 表示复制4行， p表示粘贴）



修改主配置文件监听8080端口

Vi /etc/httpd/conf/httpd.conf

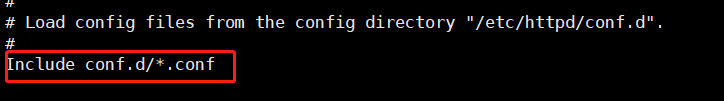


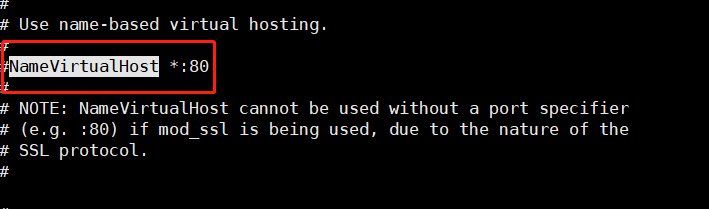


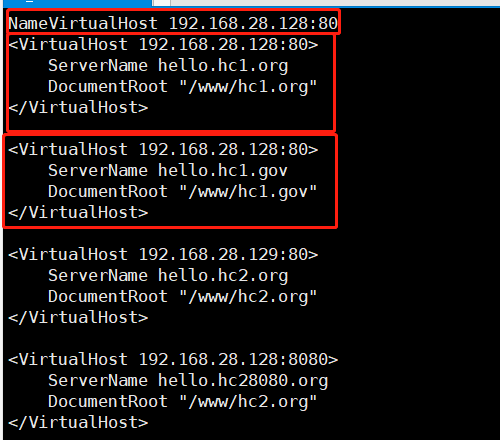
##### 基于主机名的虚拟主机

###### virtualHost配置

首先要在主配置文件中启用NameVirtualHost，或者在 /etc/httpd/conf.d/vitural.conf文件中配置。因为Include conf.d/\*.conf已经包含进来了



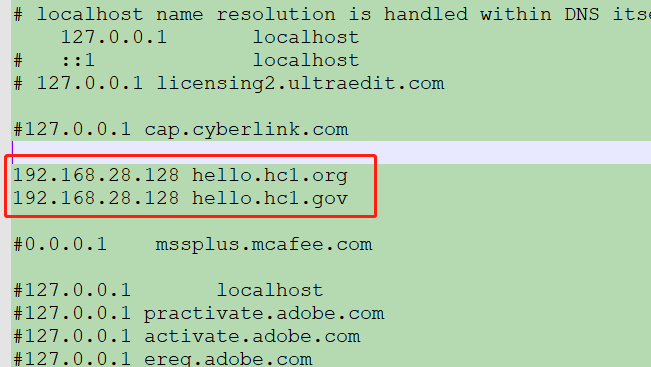


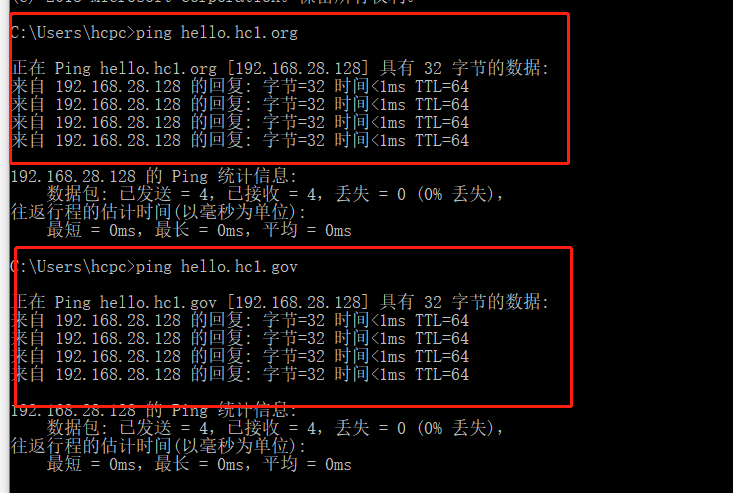


创建文件夹并添加index.html mkdir /www/hc1.gov

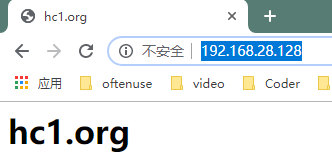
如果DNS不能解析，配置C:/windows/system32/drivers/hosts文件加入下列条目

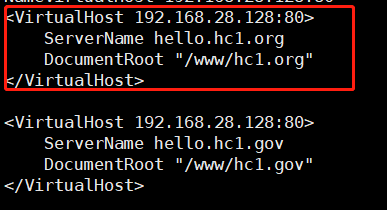
保证本机可以解析。



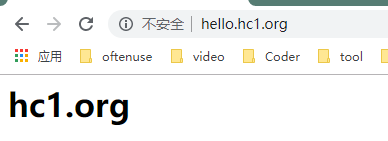


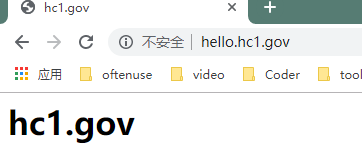
如果是基于名称的虚拟主机访问，有多个的话，用IP去访问的话，返回符合条件的配置文件中的第一个





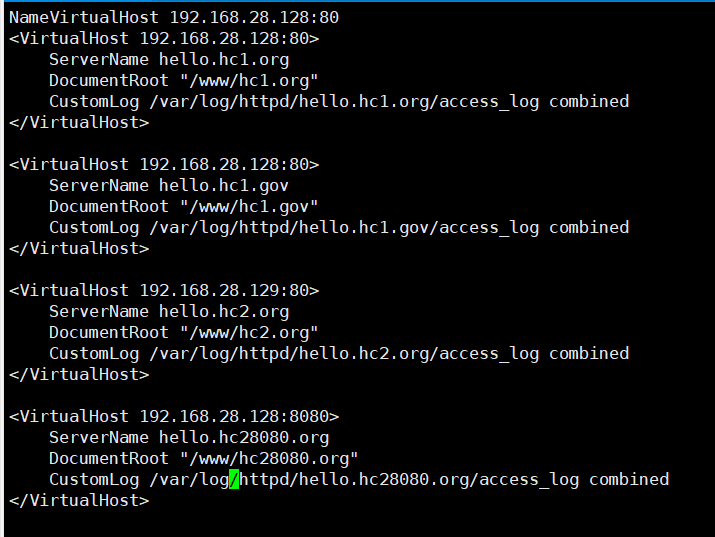
用主机名称ServerName访问的话



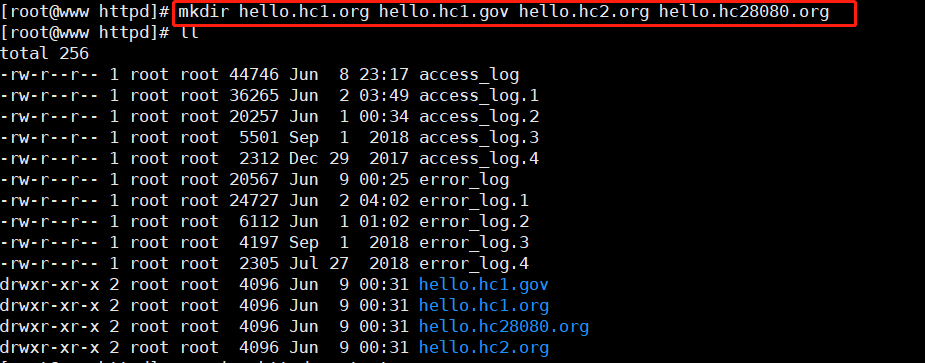


###### 日志配置

如果不配日志，那么日志都在一起，../httpd/logs目录下，所有如果多个虚拟主机，那么日志需要各自配置



创建日志目录



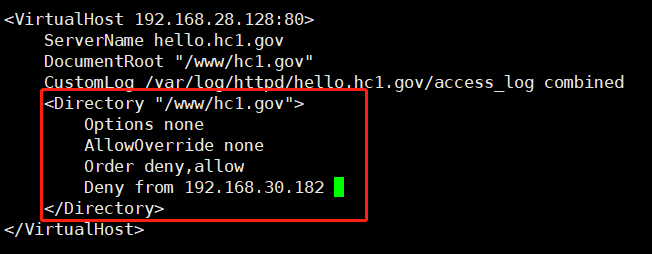
改变属主， 属组

Chown apache:apache





###### directory配置



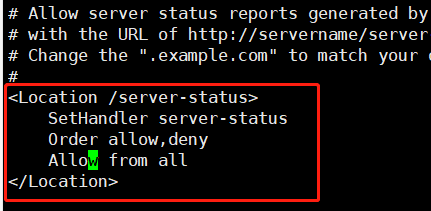
定义一个默认的虚拟主机

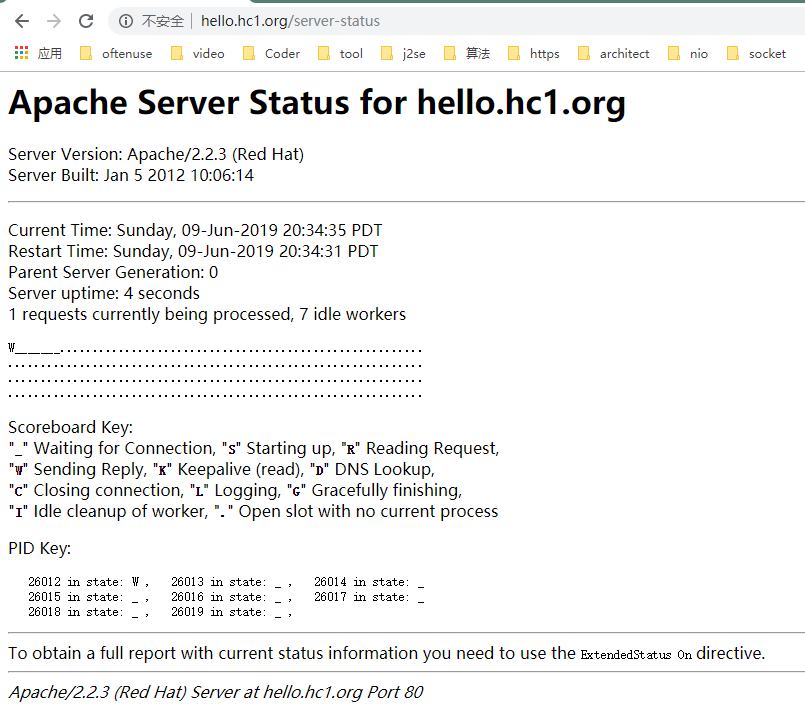


当其他虚拟主机不可以达将访问默认的虚拟主机

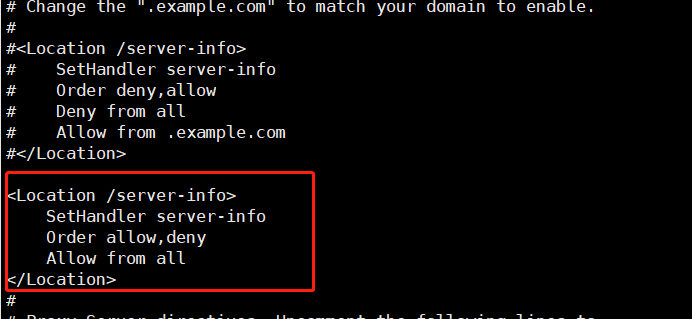
###### 配置服务器状态信息

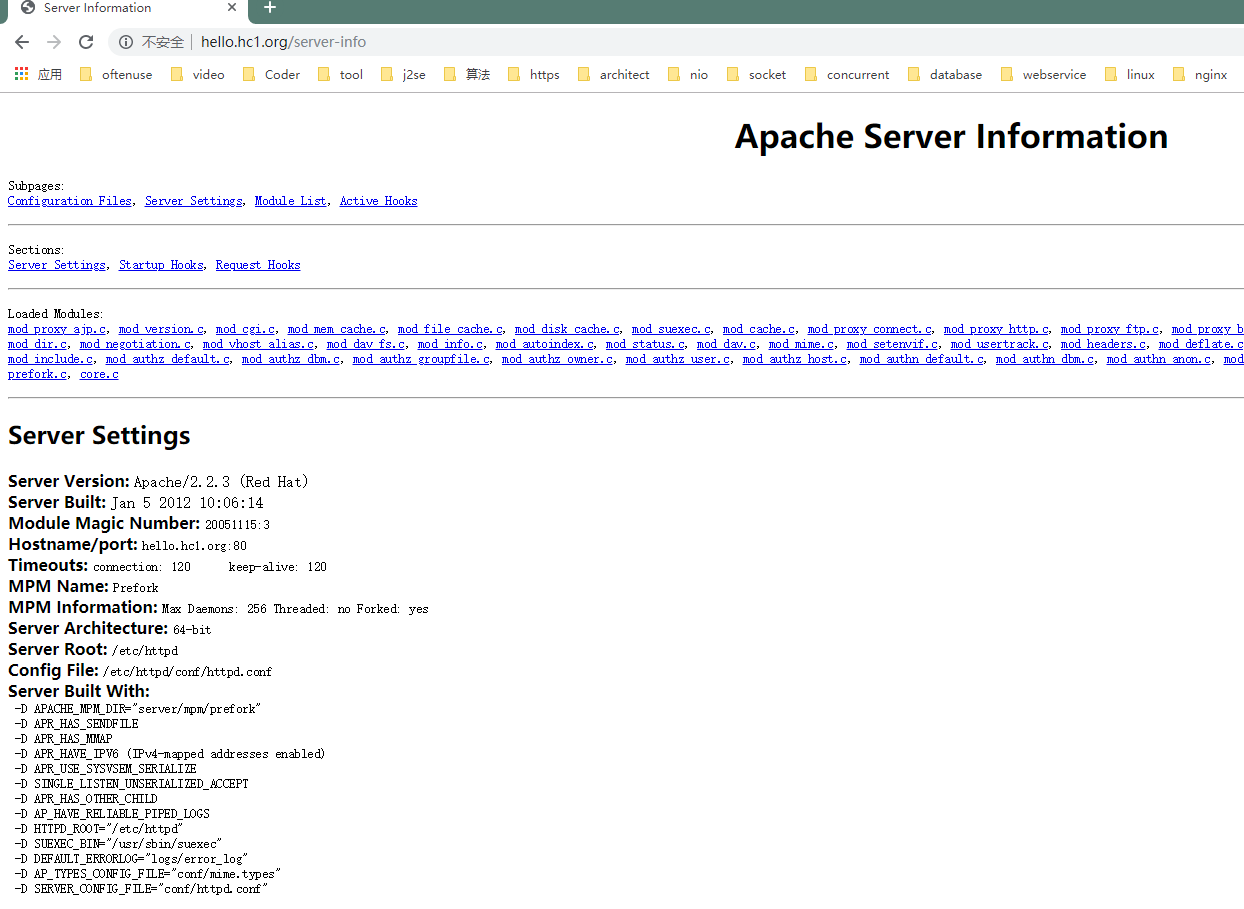






###### 配置server-info



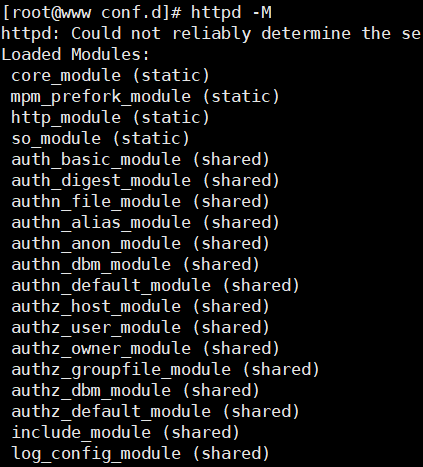


## 基于OpenSSL的https服务配置

SSL会话，只能基于IP地址进行，如果有多个基于主机的虚拟主机，只能提供其中一个虚拟主机使用。

如果要想httpd 支持SSL,首先要看有没有支持SSL的模块

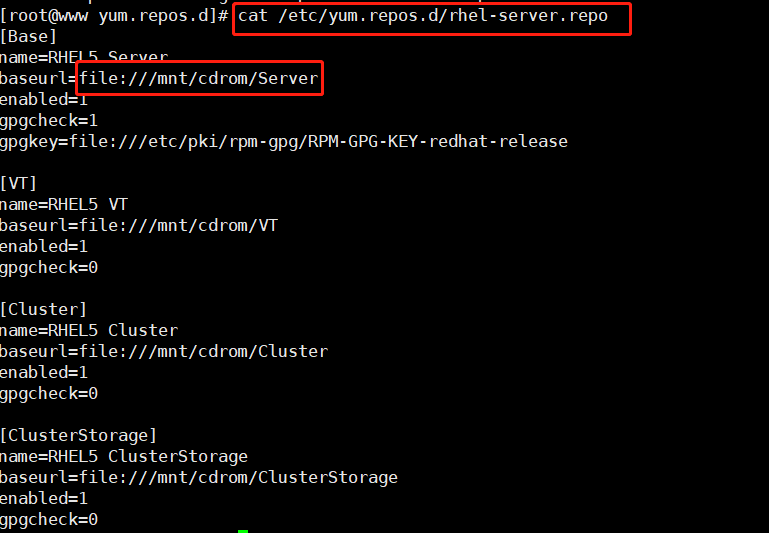
httpd –M



### 安装mod\_ssl

如果没有，那么先安装

首选确保

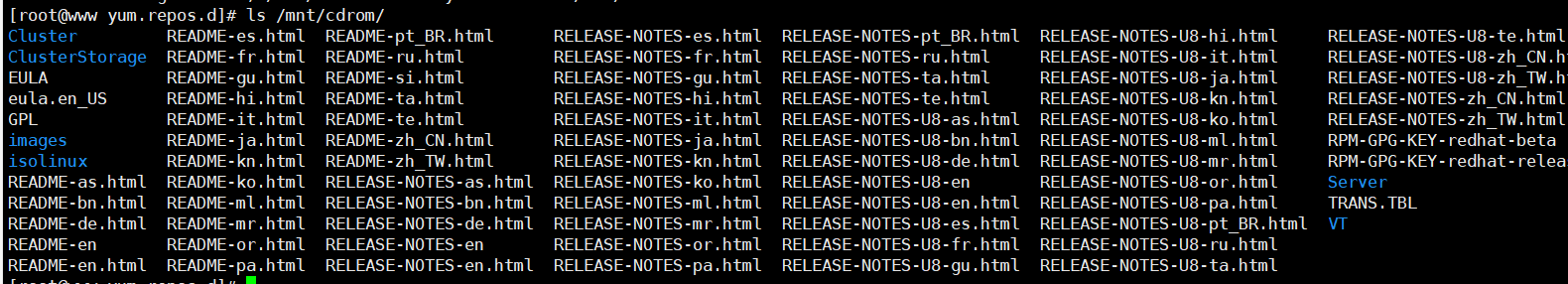


进行挂载

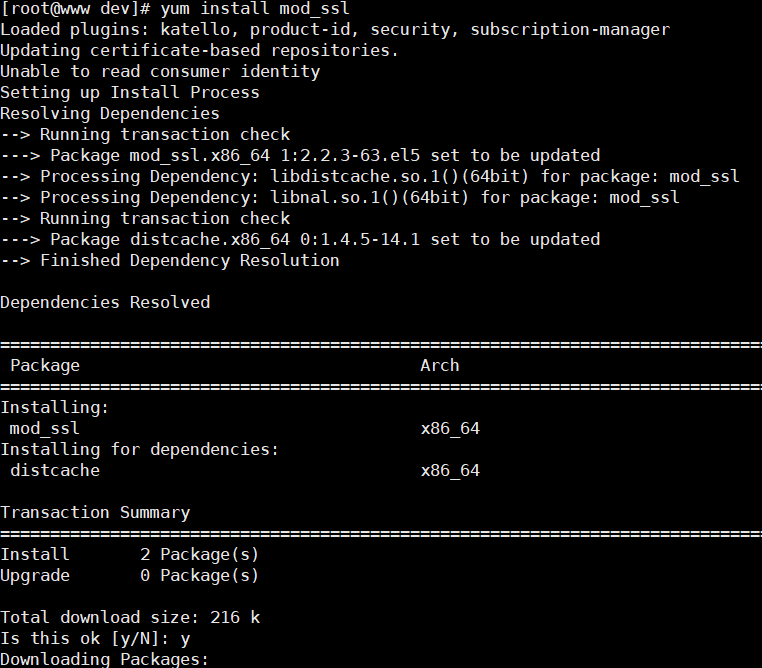
Mount –ro /dev/cdrom /mnt/cdrom 只读挂载

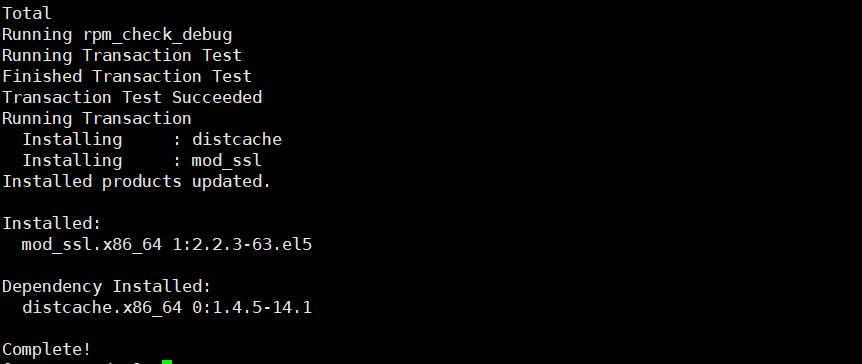


挂载成功

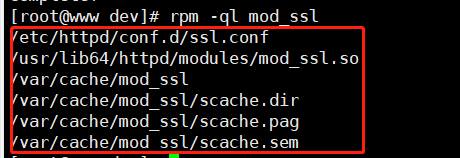


yum install mod\_ssl





rpm –ql mod\_ssl

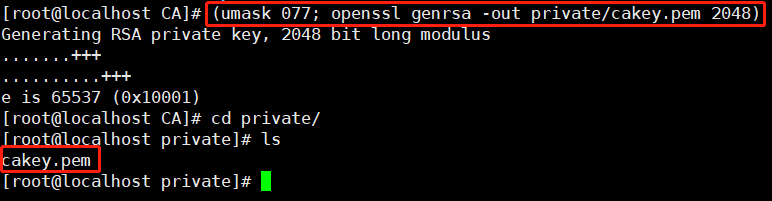


### 创建根CA(派出所)

使用另一台服务器做CA

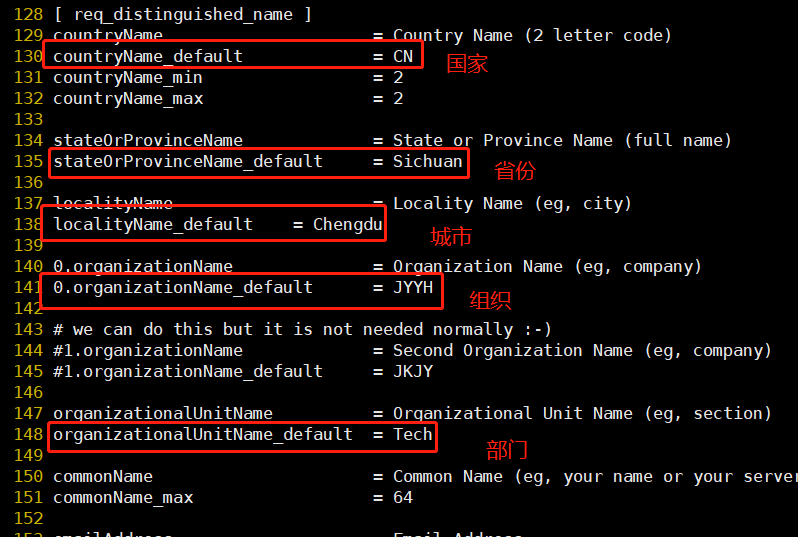
#### 生成私钥

（umask 077; openssl genrsa –out private/cakey.pem 2048）



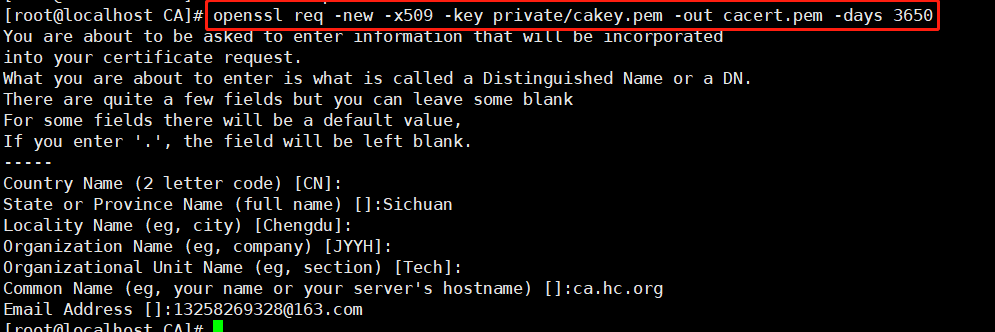
#### 生成自签证书

修改配置文件 vi ../tls/openssl.cnf 中的默认配置项，后面就可以不用手动输入这些信息



从私钥中提取公钥并生成自签证书**（自签证书就是为改体系下的证书申请进行签证，浏览器客户端也需要安装公钥证书）**

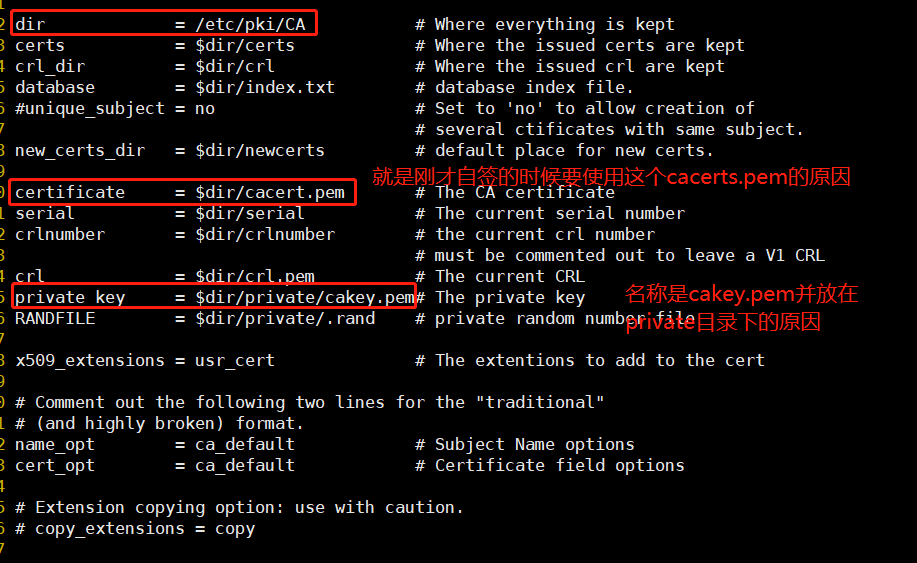
openssl req -new -x509 -key private/cakey.pem -out cacert.pem -days 3650



如果输入信息输入错误，想要删除你会发现backspace也会输入，这时Ctrl+backspace 你值得拥有。

确保dir = /etc/pki/CA，并且CA目录下有 certs，crl，newcerts, private目录

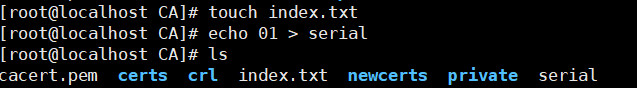
还有index.txt, serial 等



mkdir certs newcerts crl



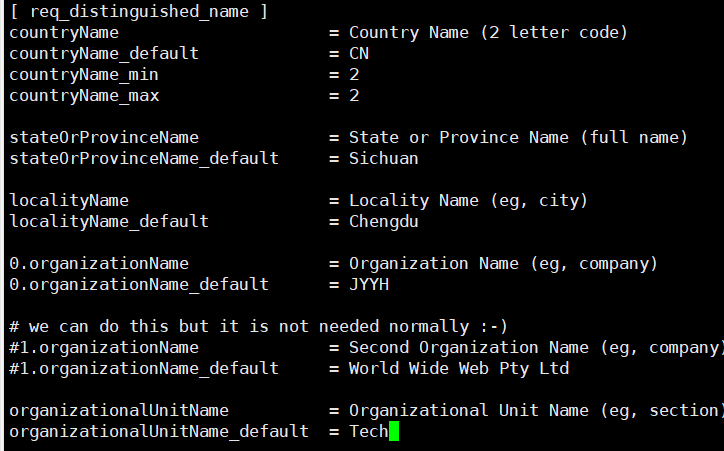
创建文件index.txt 写入序列号



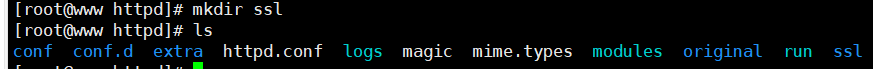
### 机构生成私钥并生成证书请求

#### 机构OpenSSL.cnf配置

与CA服务器的openssl.cnf默认配置保存一致，并且输入的信息一定要保持一致



#### /etc/httpd/目录下创建ssl目录



#### 机构生成秘钥

(umask 077; openssl genrsa 1204 > httpd.key) 和

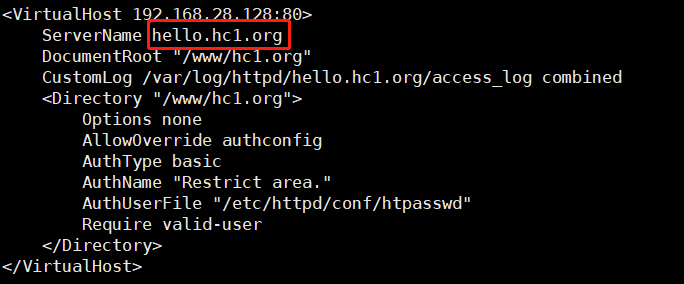
(umask 077;openssl genrsa 1024 –out httpd.key）用法相同



#### 机构生成证书签署请求

就像办理身份证一样，姓名，性别，住址，年龄，头像 等需要自己填写，填写好了以后，公安机关给你盖个戳。

CSR - Cerificate Signing Request 证书签署请求



openssl req -new -key httpd.key -out httpd.csr

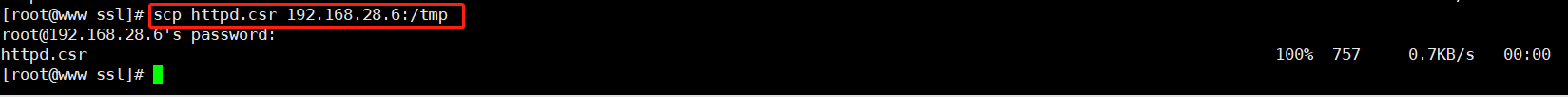
SSL基于IP的，证书只能给其中一个虚拟主机使用，这里做如下配置



#### 把证书请求发送给CA服务器进行签证

使用scp 命令将证书发到CA服务器上的tmp目录下

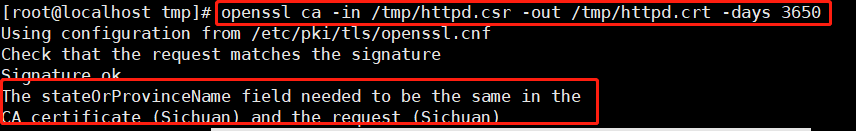
scp httpd.csr 192.168.28.6:/tmp



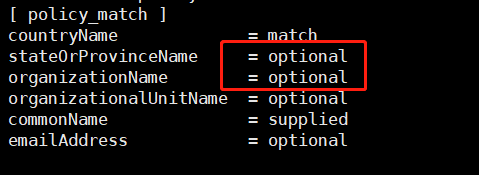
#### CA服务对证书申请进行签署

签署并放在CA服务器的tmp目录下，签署10年

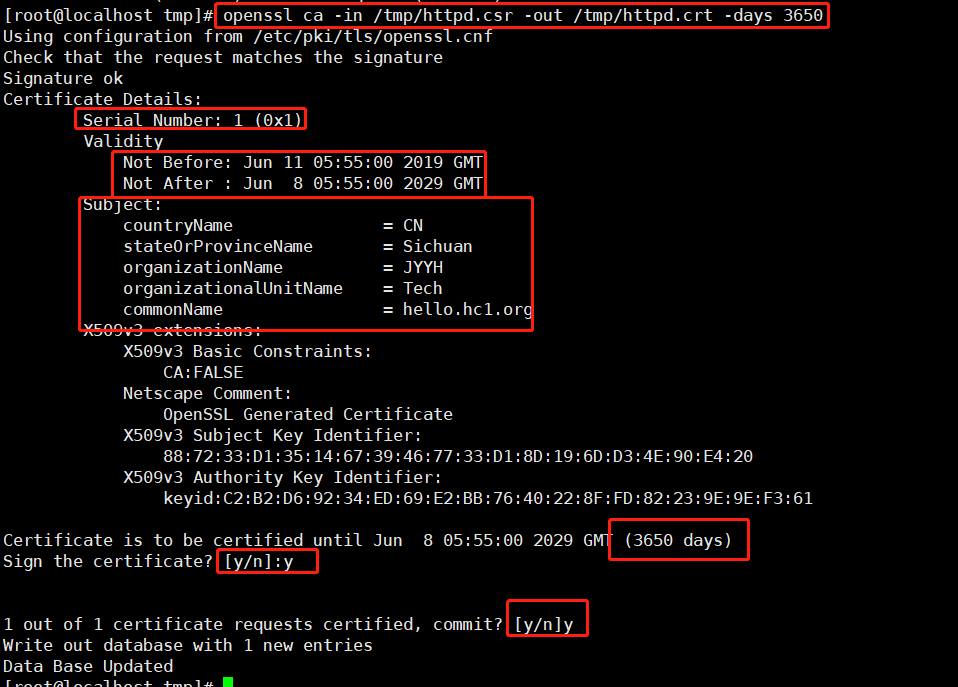
openssl ca -in /tmp/httpd.csr -out /tmp/httpd.crt -days 3650



如果这儿有问题，stateOrProvinceName field needed to be same in the CA certificat(Sichuan) and the request(Sichuan) 省份不匹配，明明我的配置文件中都是Sichuan,为什么呢？ 其实我们在可以将CA服务器上的openssl.cnf验证策略上改一下



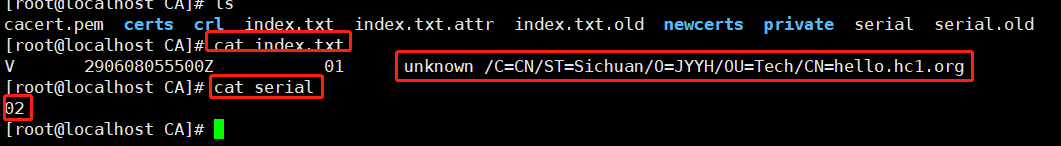
将stateOrProvinceName 改成可选 optional在进行签署，就OK





查看CA服务器上的信息

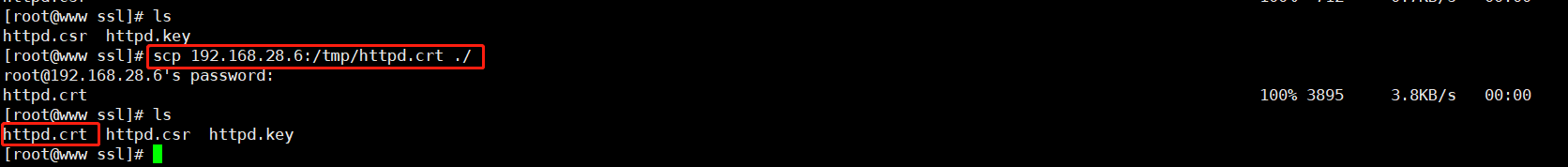




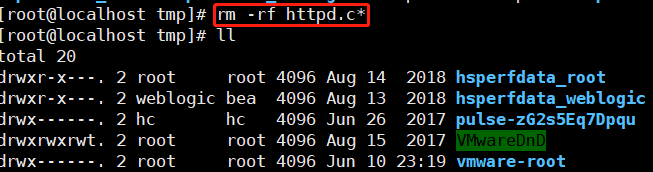
#### 机构将签署好的证书拷贝回来

拷贝到当前ssl目录

scp 192.168.28.6:/tmp/httpd.crt ./

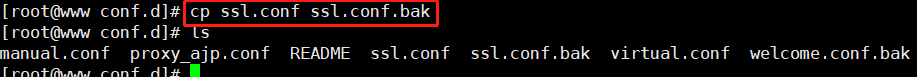


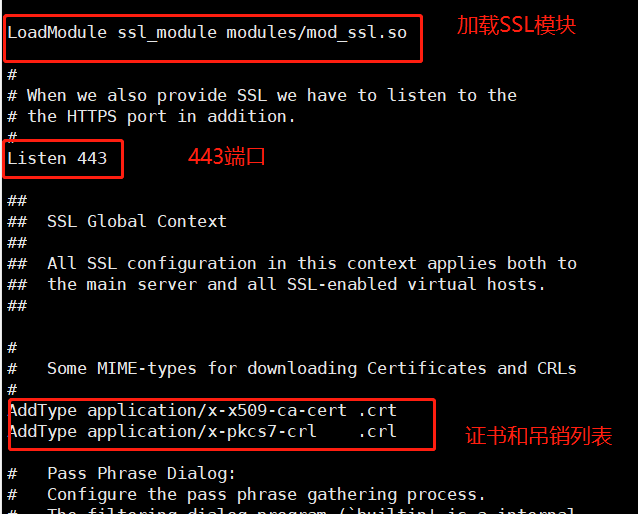
#### 将CA服务器上tmp目录下的删掉，防止泄露

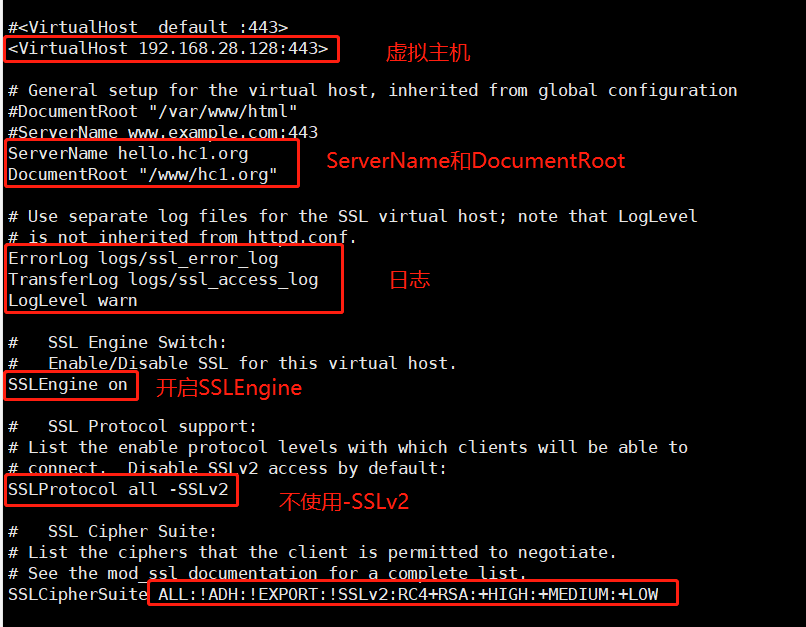


#### 配置机构的/etc/httpd/conf.d/ssl.conf

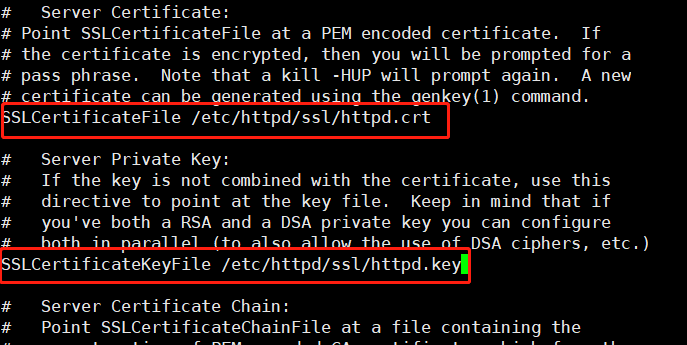
先备份

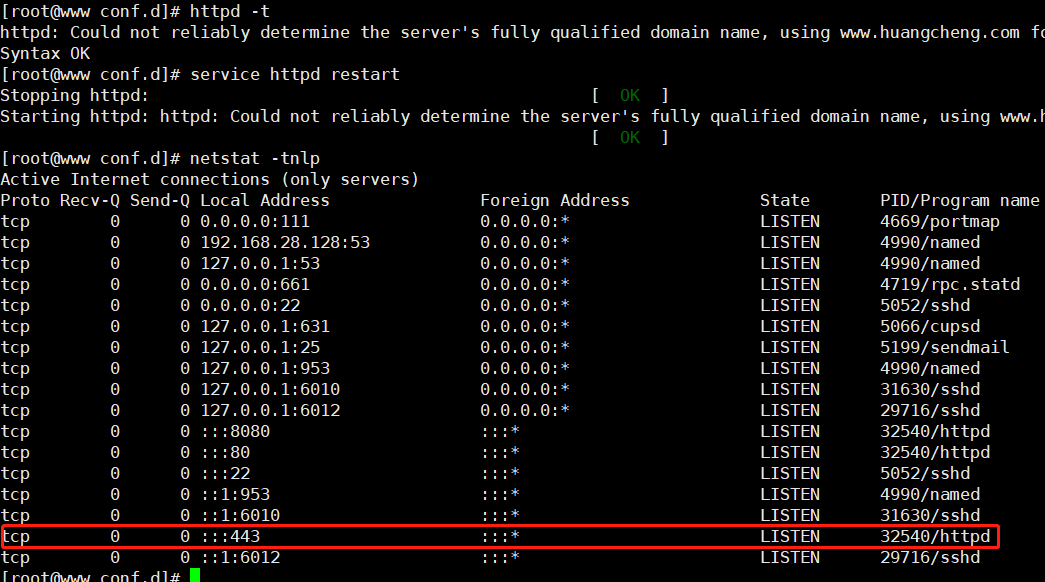




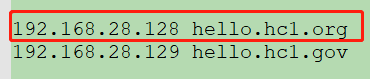


证书的位置

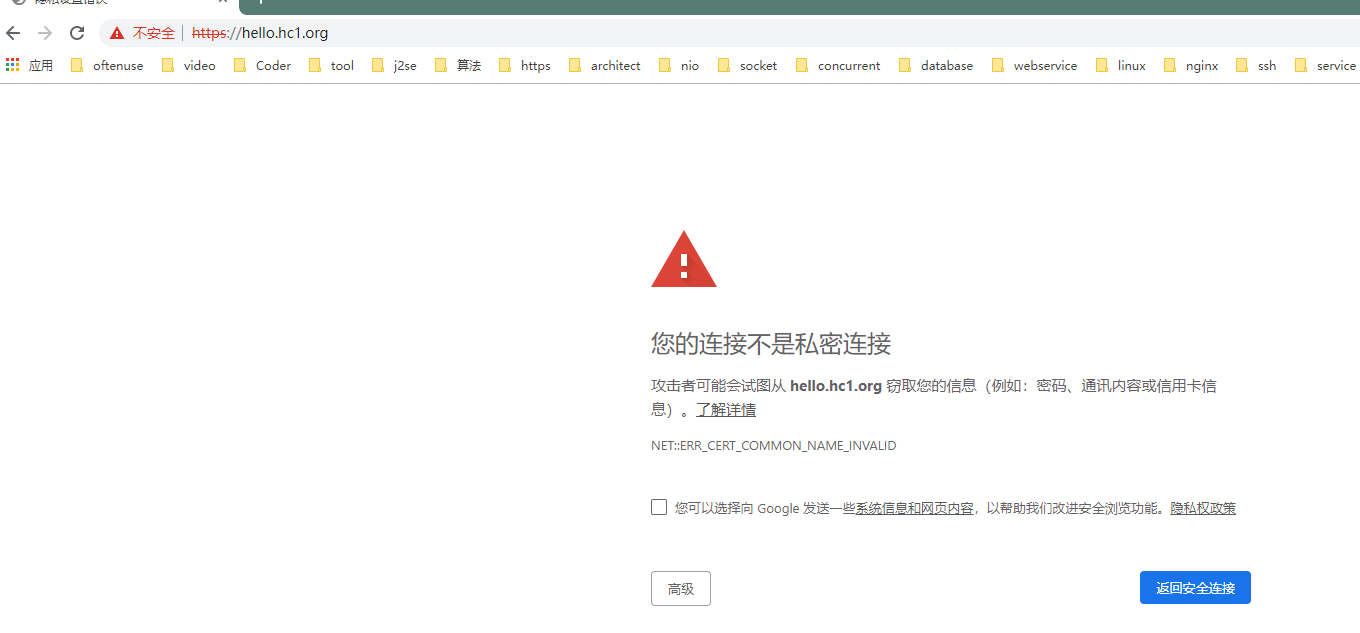




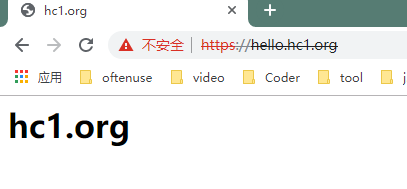
保证hosts文件中



https://hello.hc1.org

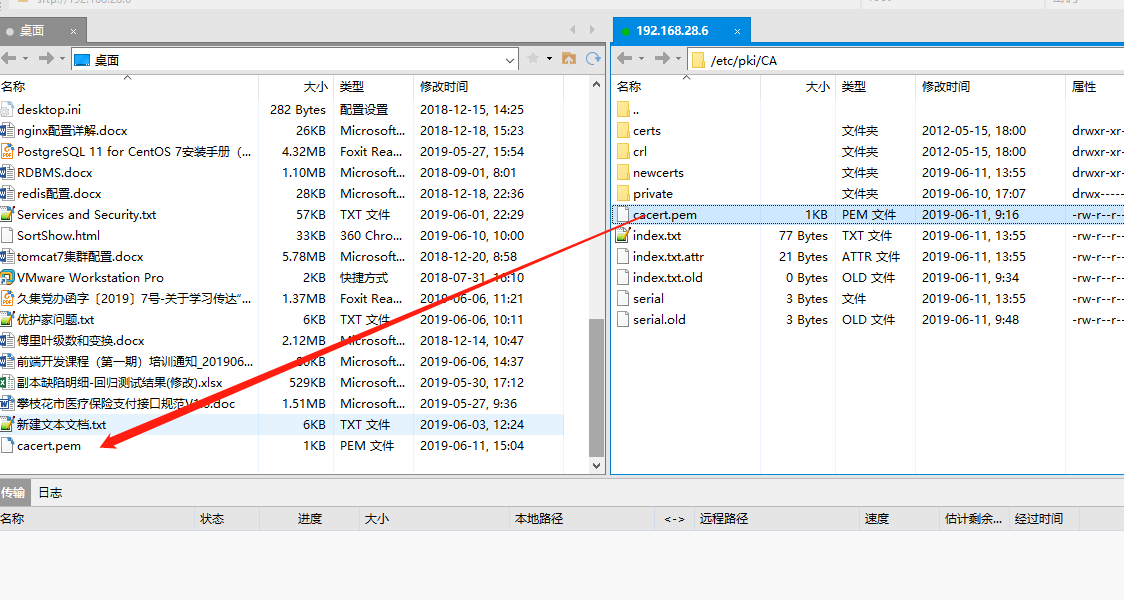


高级-继续



**提示不安全，为什么呢？ 因为浏览器上还没有安装CA服务器上生成的根证书。浏览器还无法对机构的服务器上证书进行身份验证。**

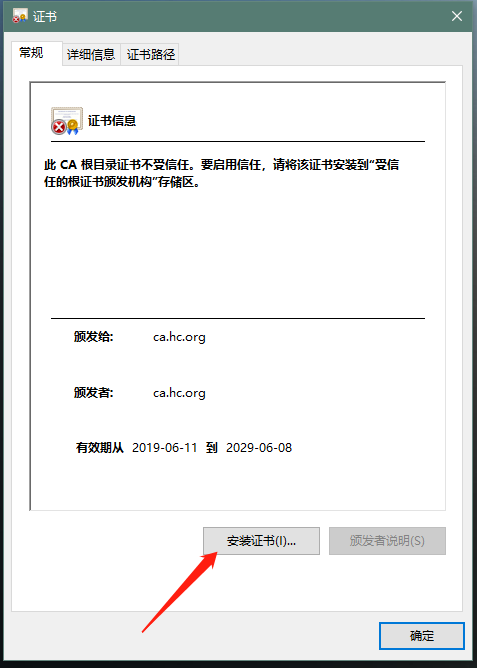
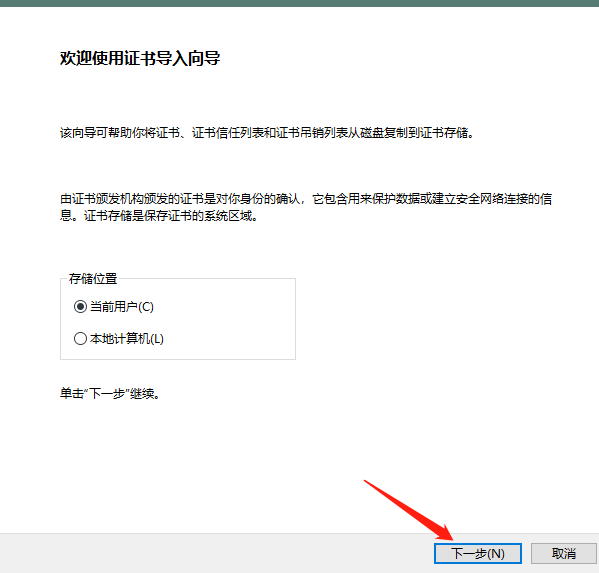
#### 浏览器上安装CA服务器上的根证书

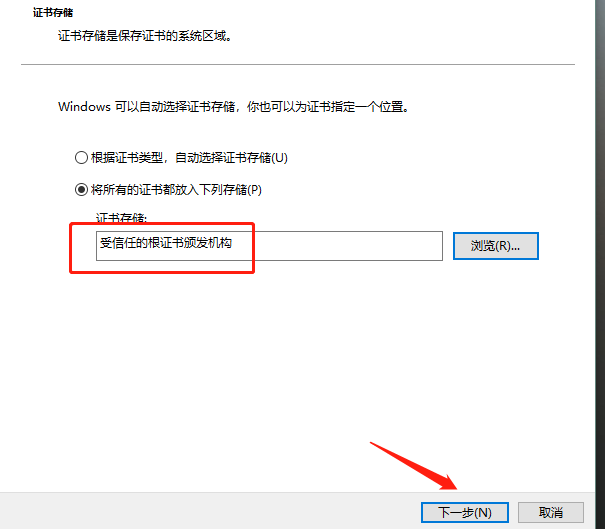
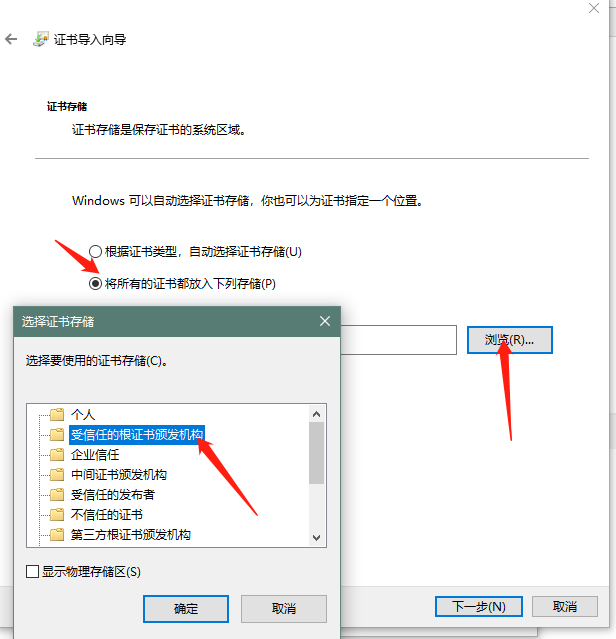


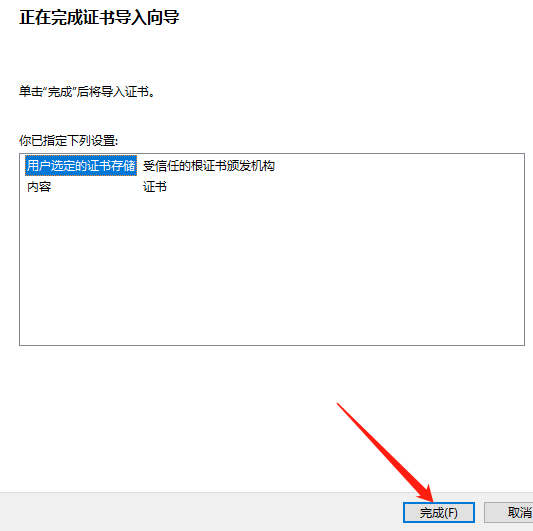
改名为cacert.crt

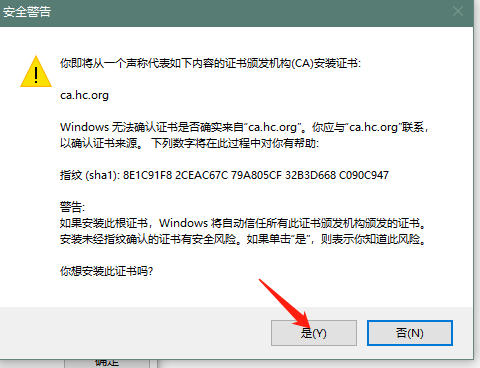


**安装证书**

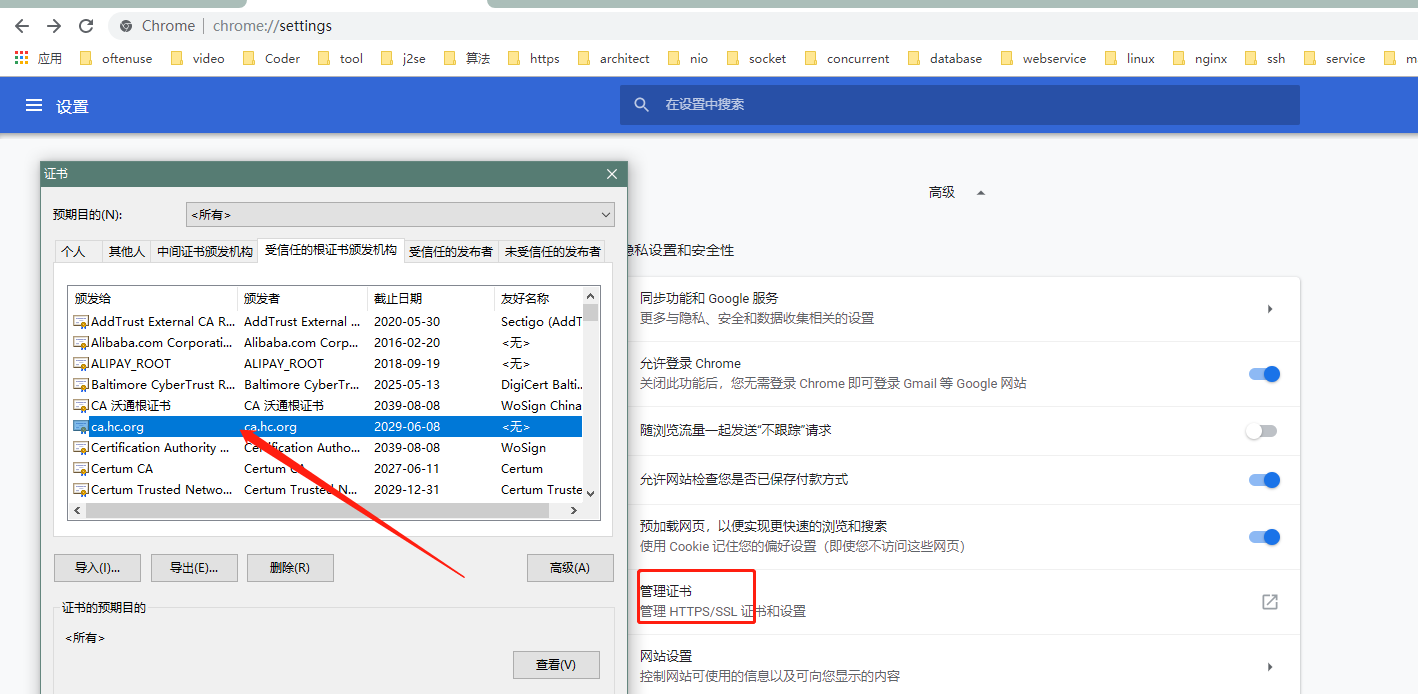
 

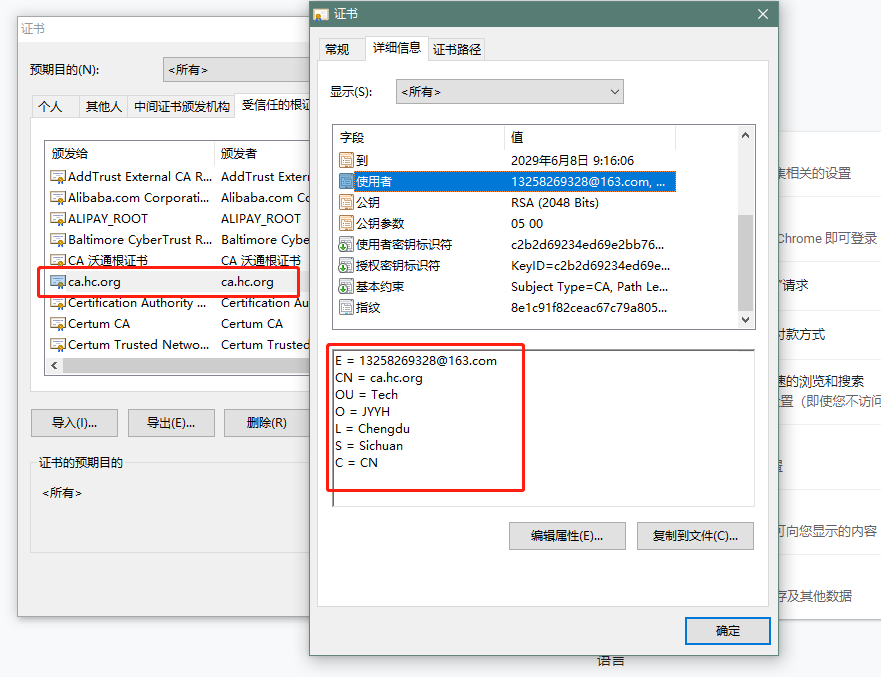




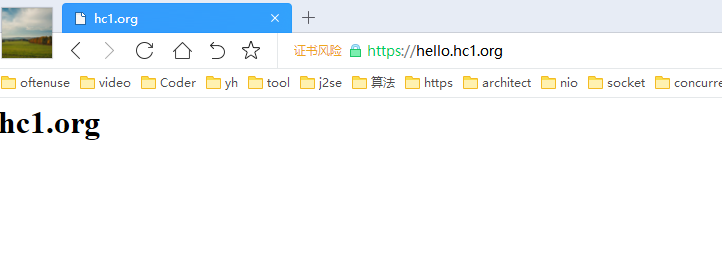


**浏览器中查看**





**换个浏览器就行了**



**整个站点都是加密访问了，**

**注意点： 机构生成证书请求的时候，server‘s hosts一定要是虚拟主机的Servername.**

**SSL基于IP的，证书只能给其中一个虚拟主机使用**

## 过程描述

**CA服务自签证书会提取机构申请证书的特征码（指纹）用自己的私钥加密（自签证书公钥可解开得到机构申请证书的指纹），放到机构签章证书后面。浏览器安装CA的自签证书为根证书，服务器会使用签证后的申请证书。当浏览器进行https访问的时候,经过TCP三次握手后SSL协商的时候，机构服务器会将证书信息发给浏览器,浏览器如果已经安装了CA自签证书,那么CA自签证书会使用公钥进行解密，能解开身份得到验证。然后协商加密算法和随机生成秘钥，进行公钥加密传递，一来一回，各自得到对方的秘钥后就使用对称加密，进行通信。**

