## JupyterHub的SSL连接  
  
### 生成证书文件  
  
需要在JupyterHub运行的服务器上生成CA证书，优先对系统生成根证书，随后生成服务器证书，完成后即可使用SSL访问。  
  
注意的是，jupyterhub的配置项internal\_ssl（双向认证）不能打开为true，否则将要求所有连接方具有证书。  
  
使用openssl进行linux的生成证书。  
  
#### 创建系统根证书CA  
  
先检查openssl的配置  
  
    vim /etc/pki/tls/openssl.cnf  
  
进入CA目录，并检查相关目录及文件  
  
    cd /etc/pki/CA  
      
    #若没有对应目录则执行命令  
    mkdir -pv {certs,crl,newcerts,private}  
    touch {serial, index.txt}  
  
指定证书开始编号  
  
    echo 01 >> serial  
  
生成根证书私钥  
  
    (umask 077; openssl genrsa -out private/cakey.pem 2048)  
  
参数：genrsa（产生rsa密钥），-out（输出，后跟路径），2048（密钥长度）  
  
生成根证书CA，位置需要与配置文件匹配  
  
    openssl req -new -x509 -key /path/to/private/cakey.pem -out cacert.pem -days 365  
  
参数：-new（生成新证书请求），-x509（自签证书专用选项），-key（指定私钥文件），-out（证书输出路径），-days（证书有效期限）  
  
完成上述操作后，CA根证书生成完毕  
  
#### 颁发证书（注册）  
  
在根证书服务器上，生成私钥及证书  
  
    (umask 077; openssl genrsa -out my.key 2048)  
    openssl req -new -key my.key -out my.csr -days 365  
  
接着颁发证书  
  
    openssl ca -in /path/to/my.csr -out /path/to/my.cert -days 365  
  
会得到一系列信息，若证书信息正确就直接确认通过即可。完成后检查证书信息  
  
    openssl x509 -in /path/to/my.crt -noout -serial -subject  
  
至此，CA认证步骤完成。  
  
#### 配置指向生成的证书key和cert文件  
  
jupyterhub中的配置项，要指向颁发的证书文件。  
  
    c.JupyterHub.ssl\_key = '/path/to/ssl.key'  
    c.JupyterHub.ssl\_cert = '/path/to/ssl.cert'   
  
其中的cert文件，可以在生成颁发证书时将.crt更改为.cert即可。  
  
完成上述所有操作，jupyterhub的访问协议变更为HTTPS。  
  
#### 有关JupyterHub的internal\_ssl  
  
internal\_ssl选项会使得Hub关联的所有组件之间采用HTTPS，开启后本地服务无法正常启动的原因要再研究。