1. **Windows下载并安装openssl后：**

Microsoft Windows [版本 10.0.18362.720]

(c) 2019 Microsoft Corporation。保留所有权利。

E:\shrimpSUDAProject\demo\express\_mysql\_rumen\_test\express\_mysql\_rumen\_test\certificate>**openssl**

OpenSSL> genrsa -out private.pem 2048

Generating RSA private key, 2048 bit long modulus

.......................+++++

..+++++

e is 65537 (0x010001)

OpenSSL> **req -new -key private.pem -out csr.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]:**CH**

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

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

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

Organizational Unit Name (eg, section) []:**CH**

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

Email Address []:**CH**

Please enter the following 'extra' attributes

to be sent with your certificate request

A challenge password []:**111111**

An optional company name []:**CH**

OpenSSL> **x509 -req -in csr.pem -signkey private.pem -out csr.crt**

Signature ok

subject=C = CH, ST = CH, L = CH, O = CH, OU = CH, CN = CH, emailAddress = CH

Getting Private key

OpenSSL>

1. **项目配置openssl：https://www.jianshu.com/p/9072dce6eb2e**

之后在项目\bin\www中设置：

……

**let https = require('https');**

// let http = require('http');

**let fs = require('fs');**

**let privateKey = fs.readFileSync('certificate/private.pem', 'utf8');**

**let certificate = fs.readFileSync('certificate/csr.crt', 'utf8');**

**let cert = {key: privateKey, cert: certificate};**

……

// var server = http.createServer(app);202003231059

// let httpServer = http.createServer(app);

**let httpsServer = https.createServer(cert, app);**

……

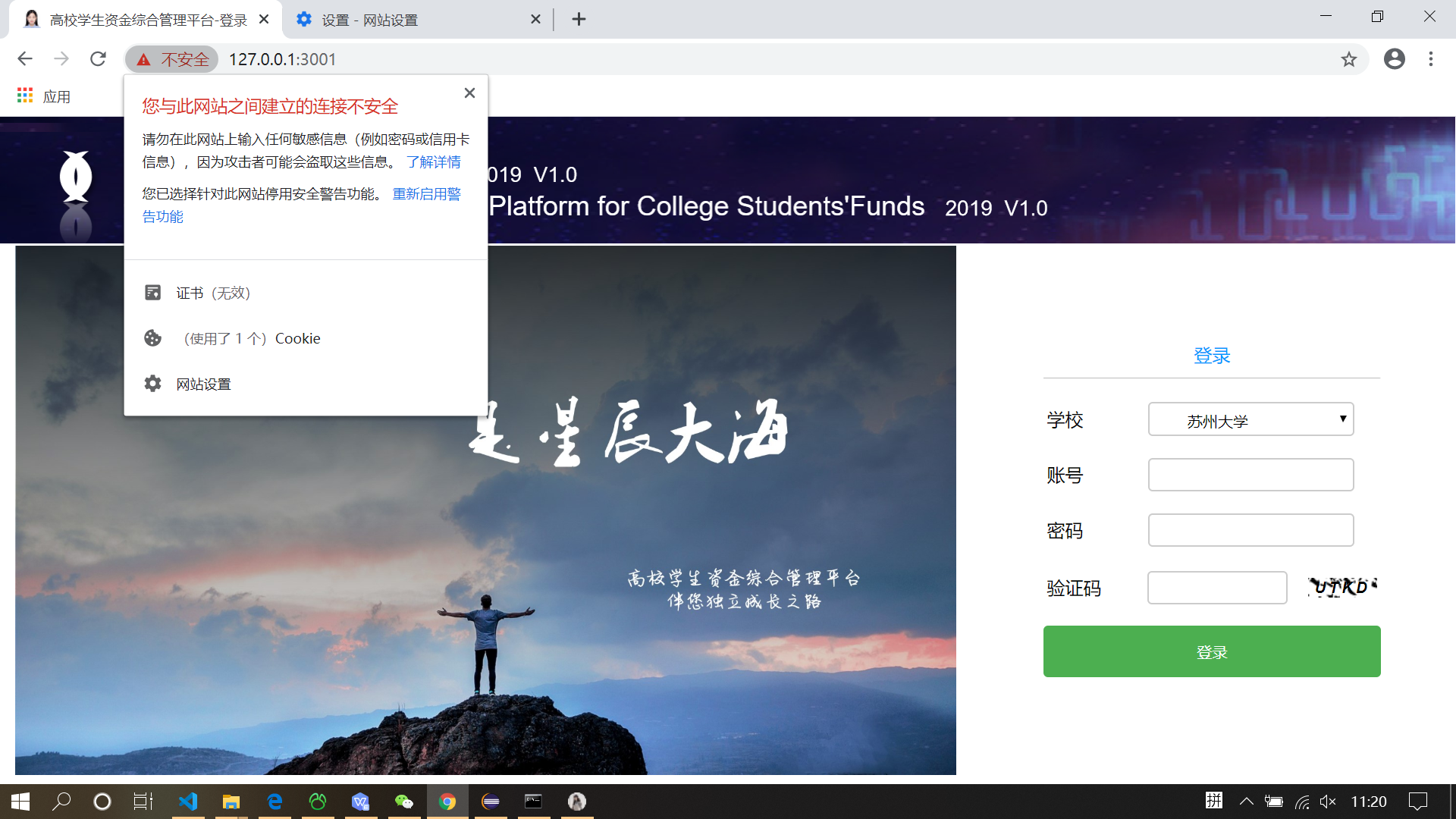
// var addr = server.address();202002231059

**var addr = httpsServer.address();**

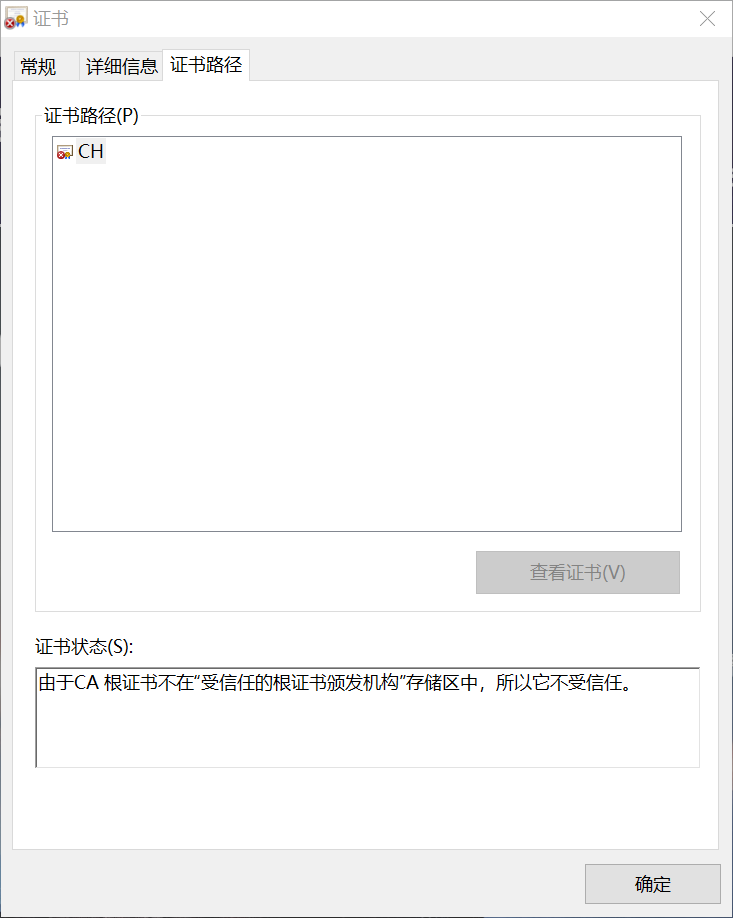
1. **启动https://127.0.0.1:3001,选择继续前往不安全网站，即可刷新出登录页面。**

**但显示网站不安全：**

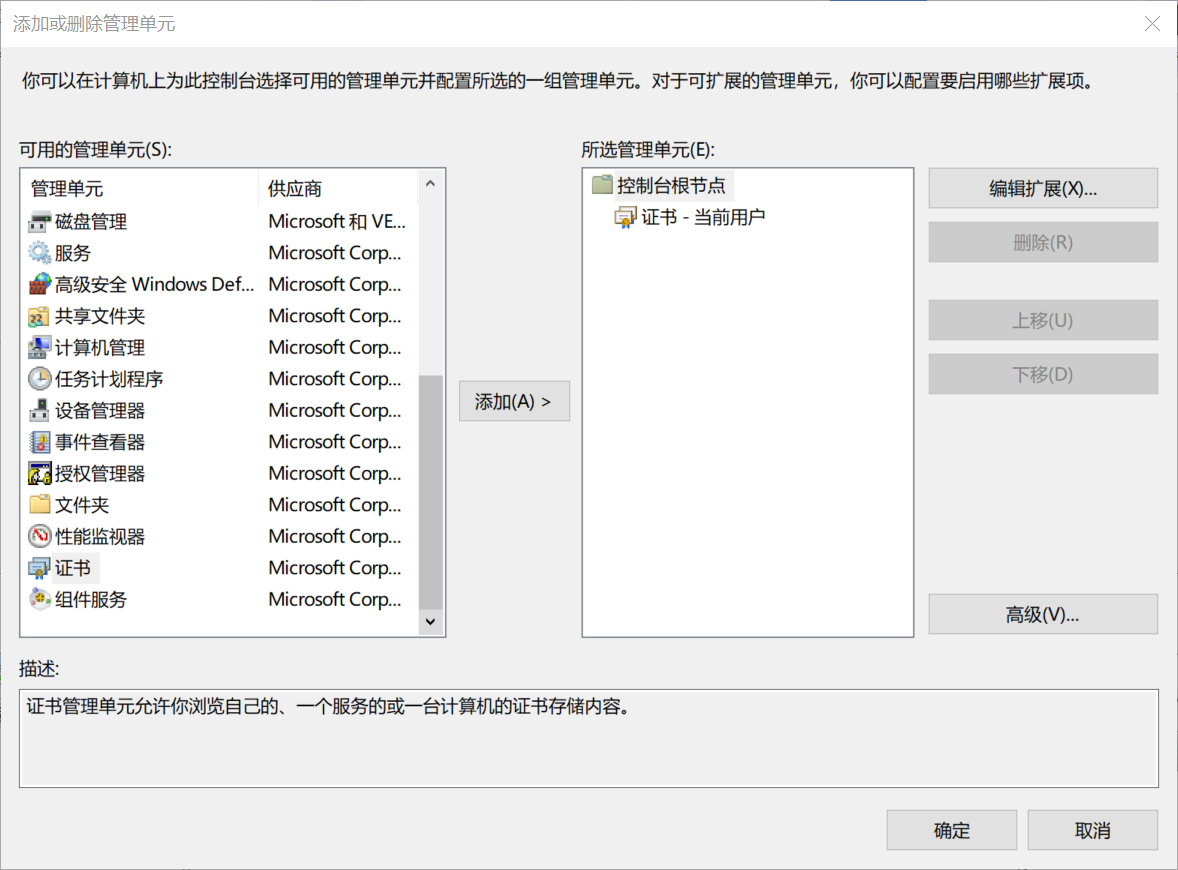
IMG_256



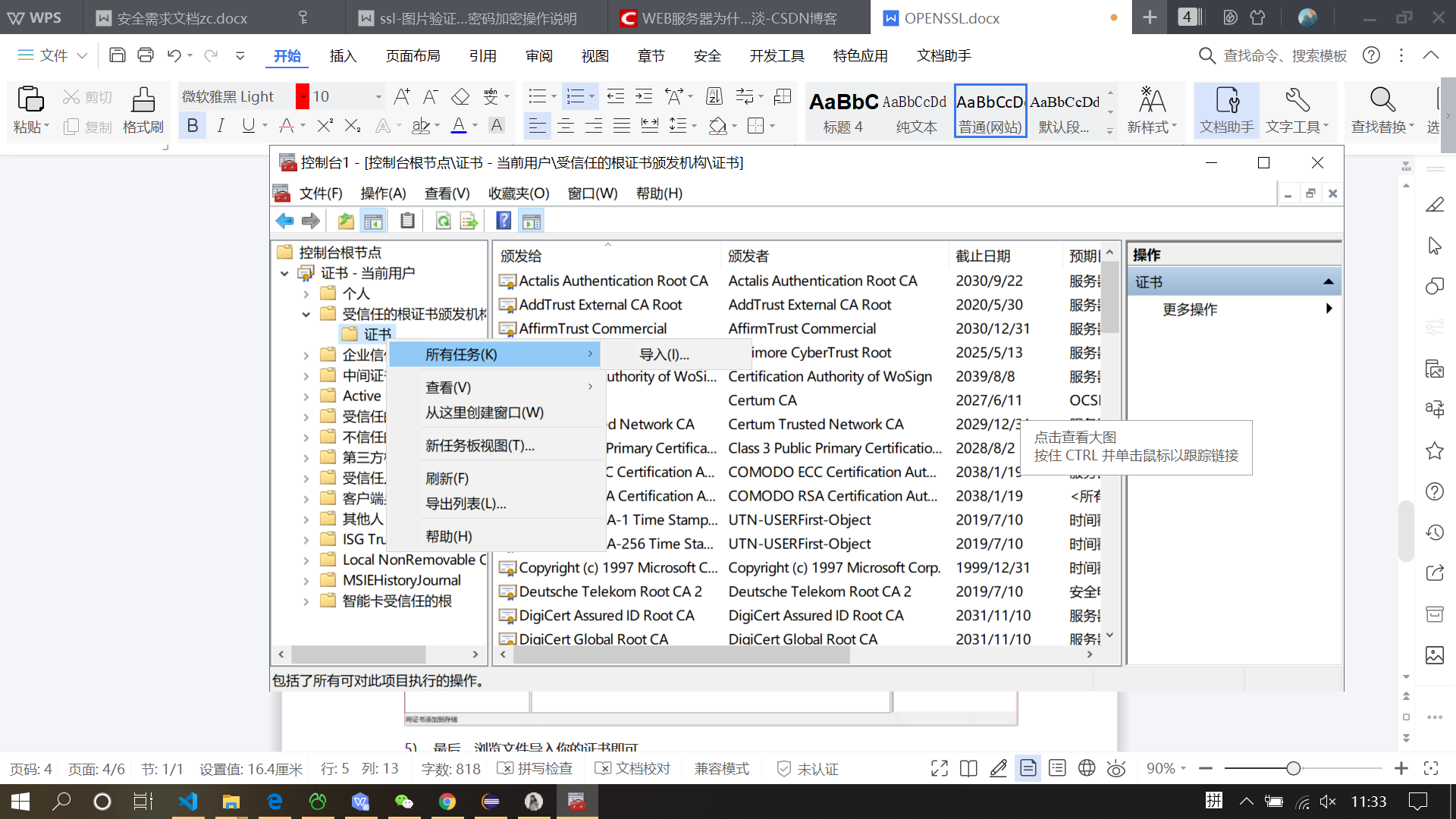
**点击证书——证书路径：得到证书状态**



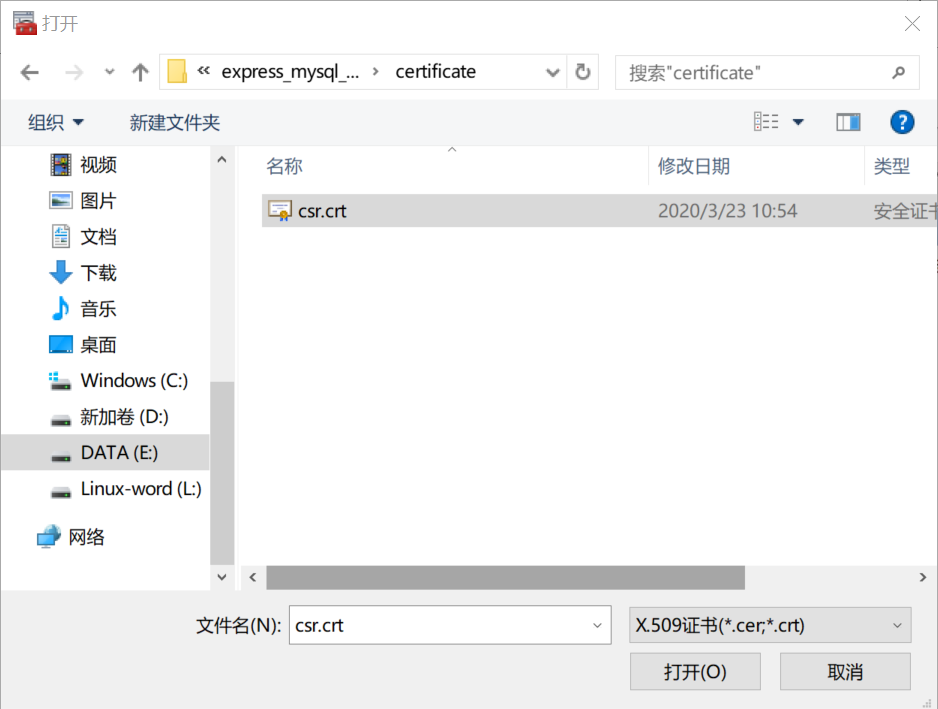
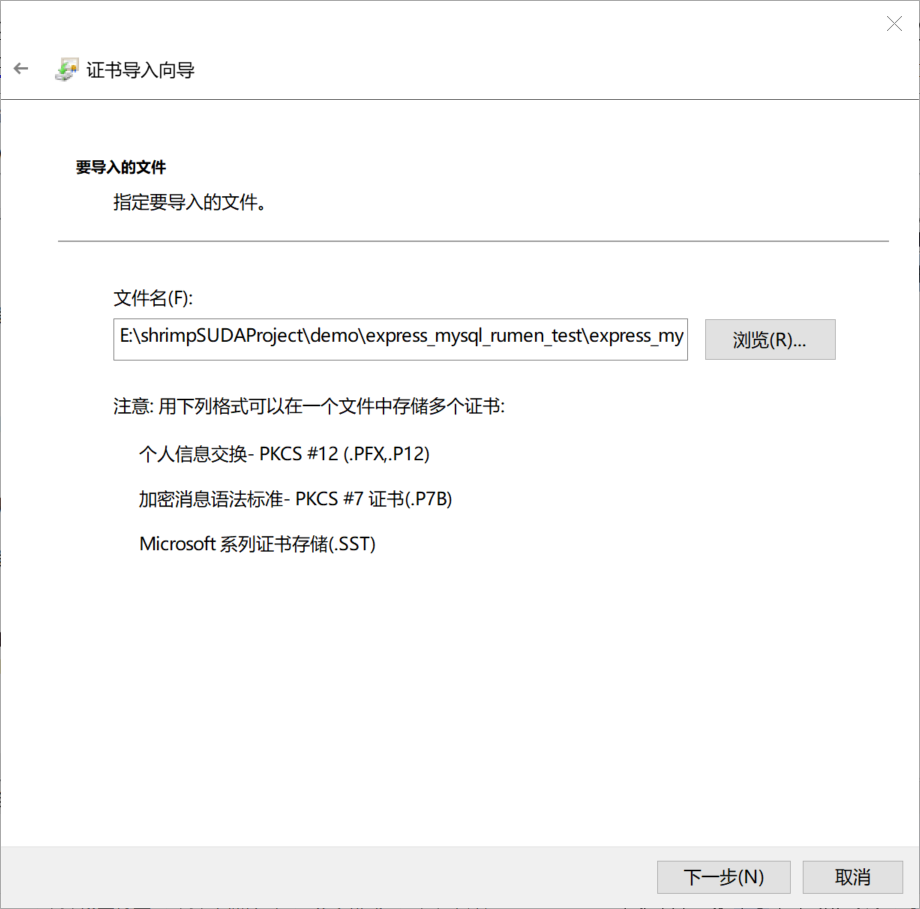
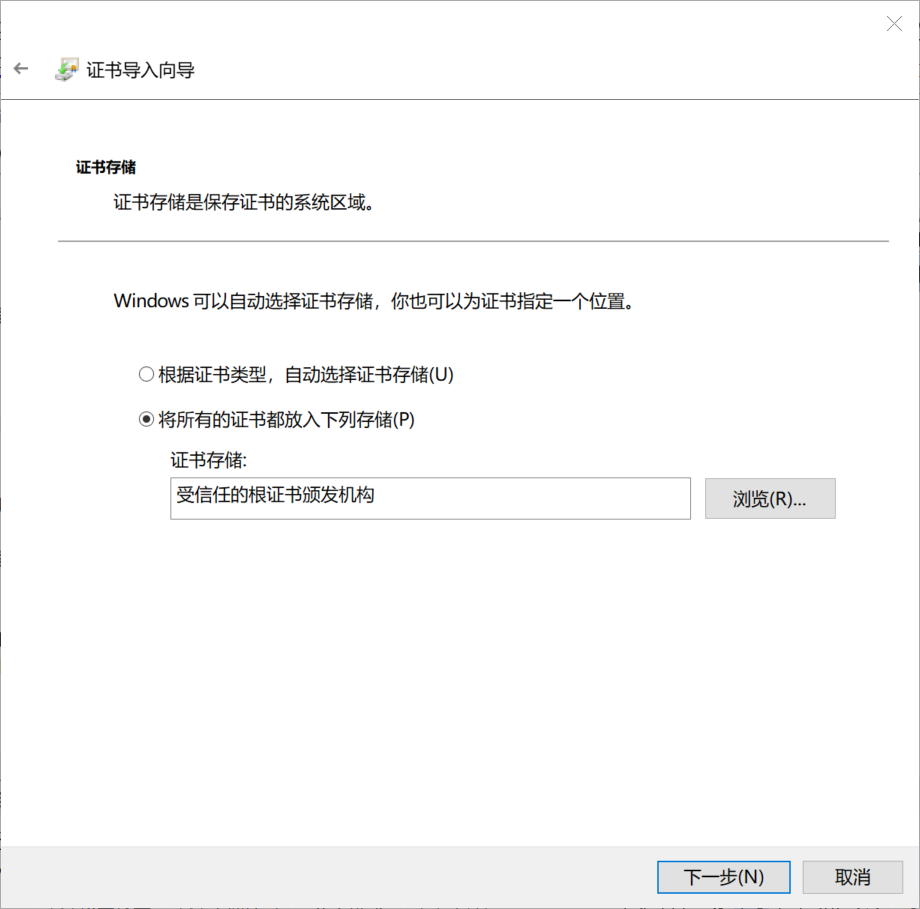
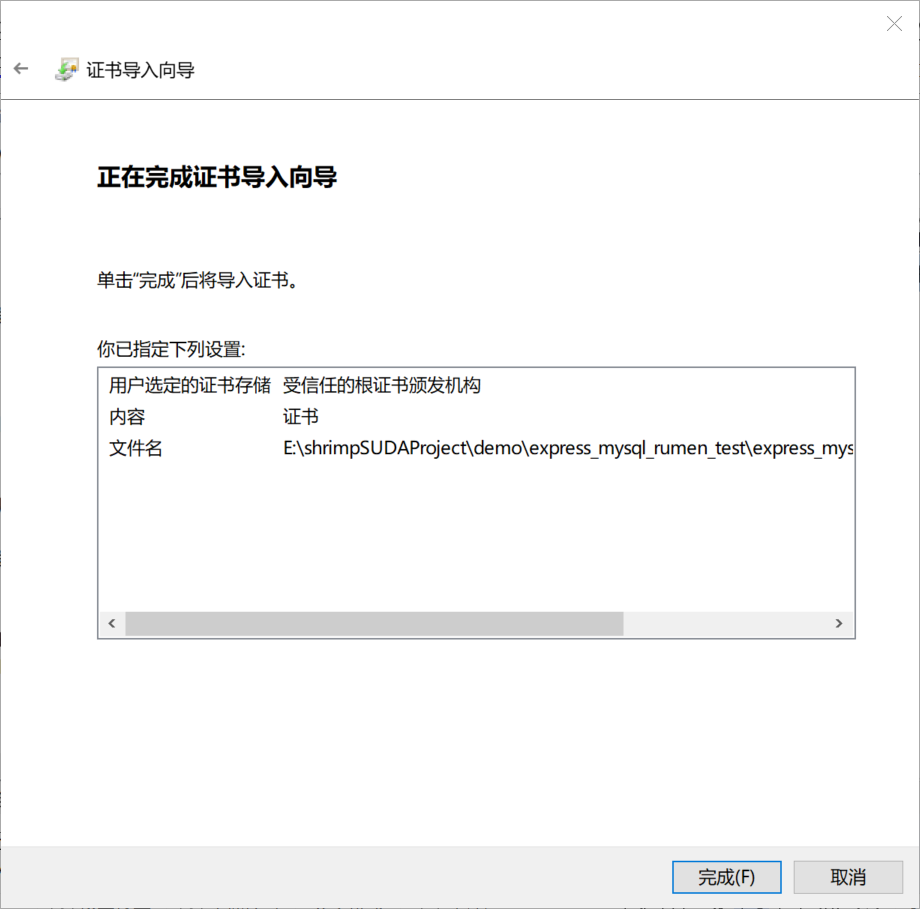
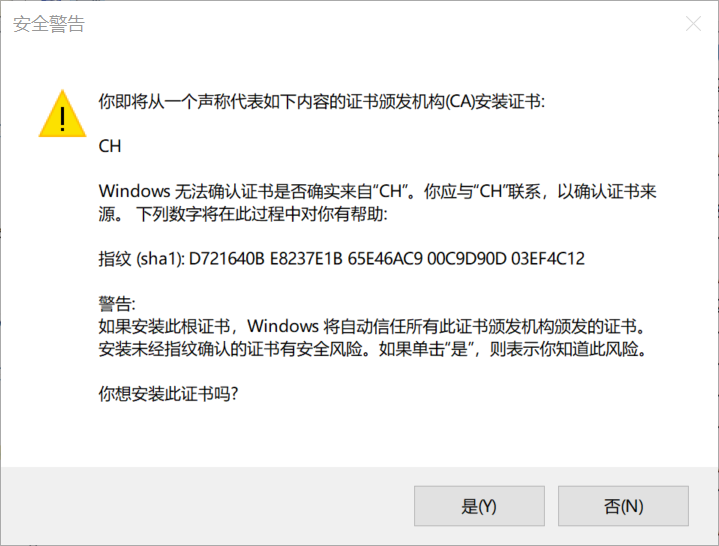
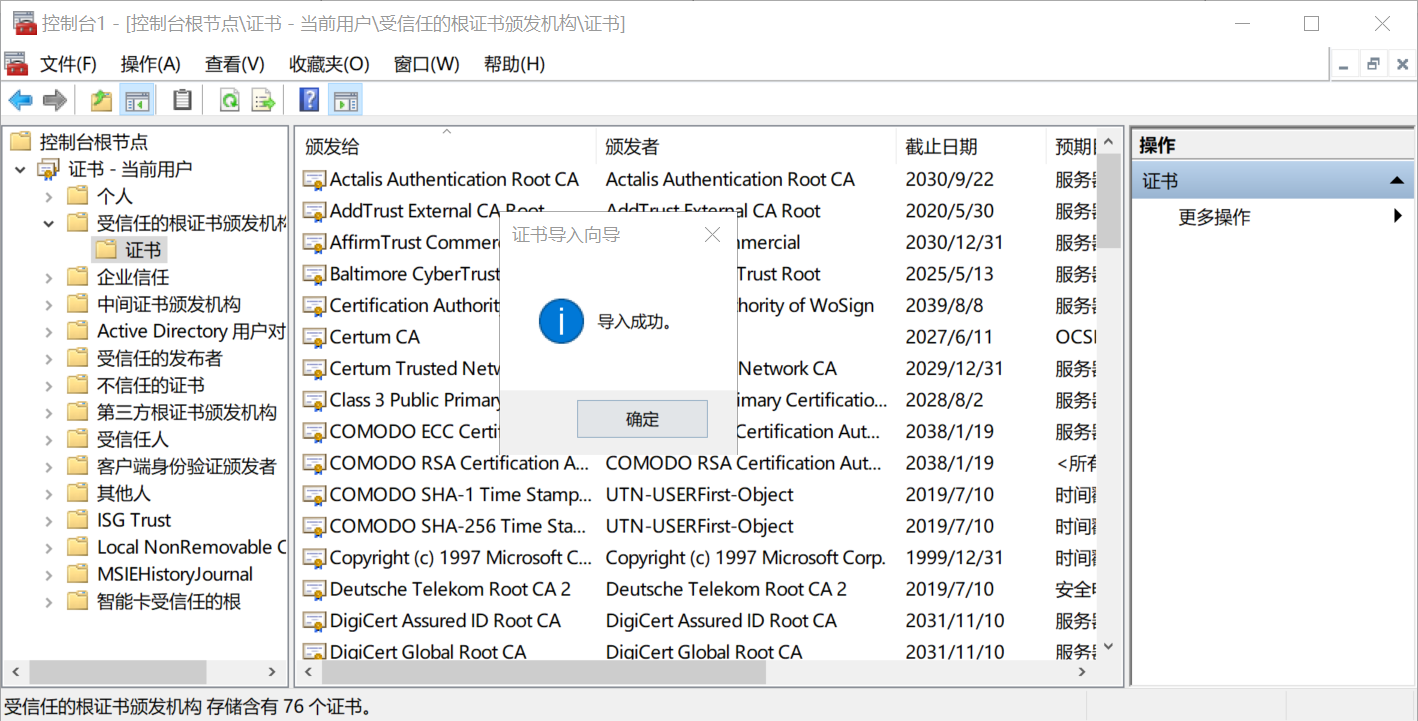
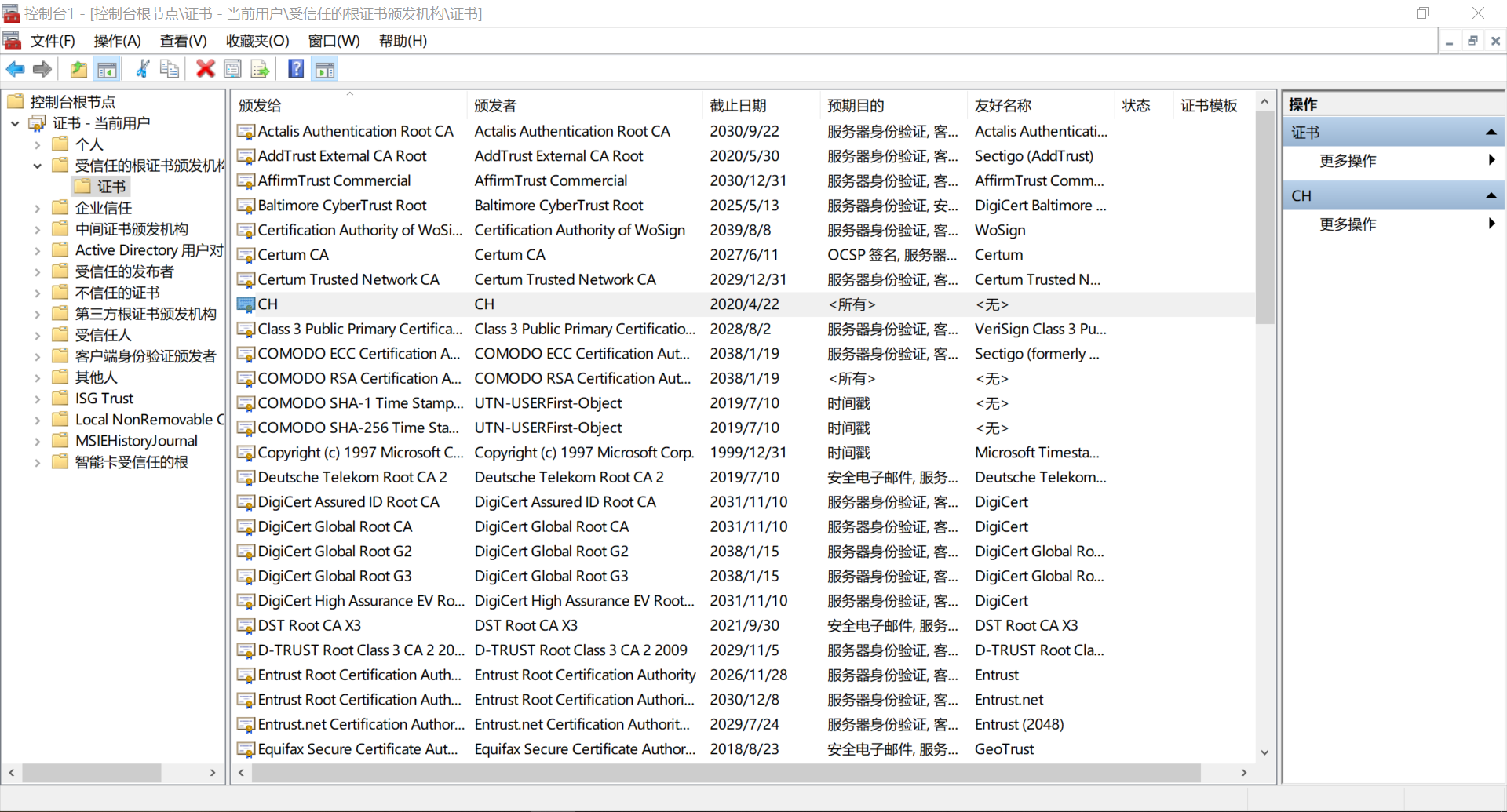
1. **可以通过将证书导入到“受信任的根证书颁发机构”存储区中解决该问题：<https://zhidao.baidu.com/question/580017187.html>**
2. **win+r 运行mmc;**
3. **文件>添加删除管理单元；**
4. **在可用的管理单元中选择”证书“，点击添加-->确定；**



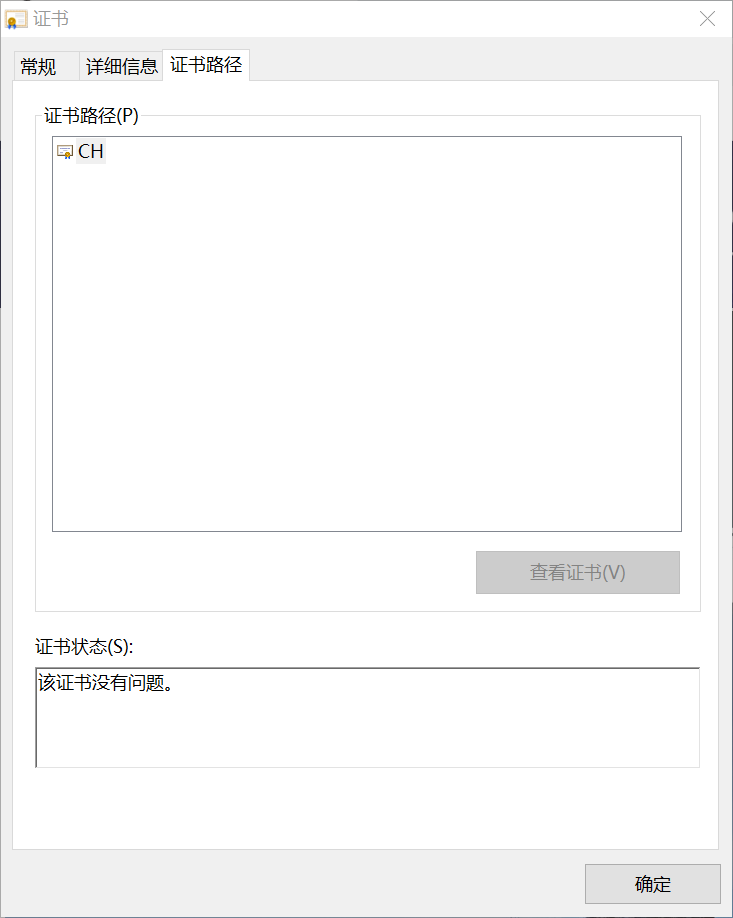
1. **在控制节点中展开证书-->受信任的证书颁发机构-->证书，右击所有任务-->导入.**



1. **最后，浏览文件导入你的证书即可。**

注意：截止日期为4.22



为开发方便起见（防止ssl证书失效），同时设置http路由接口，开发时适用（上线时关闭）：bin\www

let http = require('http');

……………………

var portHTTP = normalizePort(process.env.PORT || '3001');// 开发时适用

……………………

let httpServer = http.createServer(app);// 开发时适用

……………………

httpServer.listen(portHTTP);// 开发时适用

httpServer.on('error', onError);// 开发时适用

httpServer.on('listening', onListeningHTTP);// 开发时适用

……………………

function onListeningHTTP() {// 开发时适用

  // var addr = server.address();202003231059

  var addr = httpServer.address();

  var bind = typeof addr === 'string'

    ? 'pipe ' + addr

    : 'port ' + addr.port;

  debug('Listening on ' + bind);

}

至此：

测试http网页：127.0.0.1：3001

测试https网页：127.0.0.1：3002

1. ****扩展资料：****

从技术上讲，证书其实包含三部分，用户的信息，用户的公钥，还有CA中心对该证书里面的信息的签名。验证一份证书的真伪（即验证CA中心对该证书信息的签名是否有效），需要用CA 中心的公钥验证，而CA中心的公钥存在于对这份证书进行签名的证书内，故需要下载该证书，但使用该证书验证又需先验证该证书本身的真伪，故又要用签发该证书的证书来验证，这样一来就构成一条证书链的关系，这条证书链在哪里终结呢？

答案就是根证书，根证书是一份特殊的证书，它的签发者是它本身，下载根证书就表明您对该根证书以下所签发的证书都表示信任，而技术上则是建立起一个验证证书信息的链条，证书的验证追溯至根证书即为结束。所以说用户在使用自己的数字证书之前必须先下载根证书。

1. **但还可以：如果你有自己的域名，你看去网上申请免费试用的证书。举个网站 https://www.startssl.com**