setReachabilityStatusChangeBlock：

设置网络转态改变的回调，监听网络改变的回调有两种方式

1.使用上边的这个方法。

2.监听AFNetworkingReachabilityDidChangeNotification通知。

接受的通知中会有一个userinfo 是一个NSDictionary 其中key就是

AFNetworkingReachabilityNotificationStatusItem

FOUNDATION\_EXPORT 和#define 都能定义常量。FOUNDATION\_EXPORT 能够使用==进行判断，效率略高。而且能够隐藏定义细节(就是实现部分不在.中)

注册键值依赖

+ (NSSet<NSString \*> \*)keyPathsForValuesAffectingValueForKey:(NSString \*)key ;

告知服务器意图的HTTP方法

①GET： 直接访问URI识别的资源，也就是说根据URI来获取资源。

②POST： 用来传输实体的主体。

③PUT： 用来传输文件。

④HEAD： 用来获取报文首部，和GET方法差不多，只是响应部分不会返回主体内容。

⑤DELETE： 删除文件，和PUT恰恰相反。按照请求的URI来删除指定位置的资源。

⑥OPTIONS： 询问支持的方法，用来查询针对请求URI指定的资源支持的方法。

⑦TRACE： 追踪路径，返回服务器端之前的请求通信环信息。

⑧CONNECT： 要求用隧道协议连接代理，要求在与代理服务器通信时简历隧道，实现用隧道协议进行TCP通信。SSL(Secure Sockets Layer)和TLS(Transport Layer Security)就是把通信内容加密后进行隧道传输的。

https 原理



CF和OC相互转化  
我们可以通过**bridge,**bridge\_transfer,\_\_bridge\_retained 来进行内存管理

1.\_\_bridge  
CF和OC对象转化时只涉及对象类型不涉及对象所有权的转化

2.\_\_bridge\_transfer  
常用在CF对象转化成OC对象时，将CF对象的所有权交给OC对象，此时ARC就能自动管理该内存,作用同CFBridgingRelease()

3.**bridge\_retained  
与**bridge\_transfer 相反，常用在将OC对象转化成CF对象，且OC对象的所有权也交给CF对象来管理，即OC对象转化成CF对象时，涉及到对象类型和对象所有权的转化，作用同CFBridgingRetain()

<http://www.jianshu.com/p/11c3bc21f56e>

**\_\_builtin\_expect 说明**

这个指令是gcc引入的，作用是**允许程序员将最有可能执行的分支告诉编译器**。这个指令的写法为：\_\_builtin\_expect(EXP, N)。  
意思是：EXP==N的概率很大。

#define likely(x) \_\_builtin\_expect(!!(x), 1) //x很可能为真 #define unlikely(x) \_\_builtin\_expect(!!(x), 0) //x很可能为假

\_\_builtin\_expect((x),1)表示 x 的值为真的可能性更大；  
\_\_builtin\_expect((x),0)表示 x 的值为假的可能性更大。

### 编译器在编译过程中，会将可能性更大的代码紧跟着前面的代码，从而减少指令跳转。

<http://www.jianshu.com/p/2684613a300f>

Certificate, Key, and Trust Services为管理证书、公钥私钥和信任策略提供了C语言接口。您可以在您的应用程序中使用这些服务：

1. 通过私钥确定身份；
2. 创建和请求证书；
3. 导入证书、密钥和身份；
4. 创建公钥-私钥对；
5. 代表信任策略。

<http://blog.csdn.net/y550918116j/article/details/52080499>

http://www.cnblogs.com/polobymulberry/p/5174298.html

SecCertificateCreateWithData 创建证书

SecPolicyCreateBasicX509

SecPolicyCreateSSL 创建验证策略

SecTrustCreateWithCertificates 基于给定的证书和政策创建一个信任对象

SecTrustSetPolicies 设置信任对象策略

SecTrustSetAnchorCertificates 设置证书

SecTrustEvaluate 同步验证证书

**函数定义：**  
OSStatus SecTrustEvaluate(SecTrustRef trust, SecTrustResultType \* \_\_nullable result)  
   
**返回值：**  
OSStatus     定义在SecBase.h，0是无错误。

**参数：**  
Trust           信任管理对象，里面包含了待验证的证书，和自定义的策略。  
Result          验证结果，有以下情况：

* kSecTrustResultUnspecified 证书验证成功，但是用户没有明确指出信任此证书。这是最常见的返回值。
* kSecTrustResultProceed 用户选择信任此证书。
* kSecTrustResultDeny 用户选择不信任次证书。
* kSecTrustResultRecoverableTrustFailure 证书不可信，但是经过较小的改动可以修复问题，例如忽略过期证书、增加信任链节点等。在iOS中通常会拒绝该证书，但是收到信息时证书未过期可以通过验证，因此可以验证证书在收到信息时是否过期。
* kSecTrustResultConfirm 用户预先选择了证书链中得某一个证书在每次使用前询问允许。这个返回值已经不再使用，只在老版本的OS X中使用。
* 该函数评估证书的有效性用于在数字签名、建立SSL连接等情况。在调用函数前，可以调用SecTrustSetVerifyDate来设置用于验证证书的时间，默认是当前的时间。如果证书是自定义的，则需要在调用SecTrustEvaluate之前，自己构造证书链等信息。  
       
      如果信任管理对象（trust management object）中缺少验证页证书的上一级证书，它会在以下位置搜索证书：
* 1.调用者自定义的证书链（SecTrustSetKeychains）  
      2.通过SecTrustSetAnchorCertificates设置的证书  
      3.系统的证书链  
      4.从网络获取
* 注意：该函数在调用时可能涉及到网络请求，因此不能在主线程调用，可以调用它的异步方法SecTrustEvaluateAsync

SecTrustGetCertificateCount

获取证书数量

SecTrustGetCertificateAtIndex

获取信任对象指定位置证书

NSURLAuthenticationChallenge是一个认证挑战类，服务器向客户端发起挑战，要求客户端提供一个挑战凭证NSURLCredential（用户名密码、客户端证书等信息）来接受挑战。如函数：[NSURLCredential credentialForTrust:challenge.protectionSpace.serverTrust]就是用SecTrustRef类创建一个挑战凭证。NSURLAuthenticationChallenge中的protectionSpace中保存了挑战相关的信息（如服务器提供的证书，主机名，端口号，协议等），由于willSendRequestForAuthenticationChallenge回调时不止HTTPS服务器身份鉴别，如Basic、Digest、HTML Form等方式的身份鉴别也会调用该方法，我们关注的是HTTPS证书验证，因此首先判断一下身份鉴别的类型。通过challenge.protectionSpace.authenticationMethod可以获取，可能获取到以下值：

NSURLAuthenticationMethodDefault // 使用默认的身份鉴别方法,默认为Basic方式  
NSURLAuthenticationMethodHTTPBasic // 同上  
NSURLAuthenticationMethodHTTPDigest // 使用Digest鉴别方式  
NSURLAuthenticationMethodHTMLForm // HTML表单方式  
NSURLAuthenticationMethodNegotiate  
NSURLAuthenticationMethodNTLM  
NSURLAuthenticationMethodClientCertificate // 用客户端提供的证书来完成身份鉴别  
NSURLAuthenticationMethodServerTrust // 验证服务器证书，该方法主要用于SLL和TLS连接的建立。

http://blog.csdn.net/myzlhh/article/details/50255805

自签名证书配置

http://www.cnblogs.com/lihuang/articles/4205540.html

<http://www.cocoachina.com/ios/20160928/17663.html>

<http://www.cnblogs.com/fuunnyy/p/6114086.html>

mac下配置本地域名

<http://www.jianshu.com/p/b2a9655fe687>

http://blog.csdn.net/zhu410289616/article/details/46566073

用sudo apachectl configtest命令检查配置时，出现下面的提示：

Could not reliably determine the server's fully qualified domain name

是因为httpd.conf文件中的ServerName没有配置，处于缺省状态。

只需要在apache安装目录/etc/apache2/httpd.conf文件中启用ServerName配置指令即可。

加上：ServerName localhost:80

php脚本

<http://www.jb51.net/article/91391.htm>

http://blog.csdn.net/jdj\_1027/article/details/7732578