

# HTTPS原理及OKHTTP对HTTPS的支持



panda\_Vicky (/u/28d9a4ba4cbc) [+ 关注](#)

2017.08.18 14:53\* 字数 1734 阅读 1996 评论 11 喜欢 57

(/u/28d9a4ba4cbc)

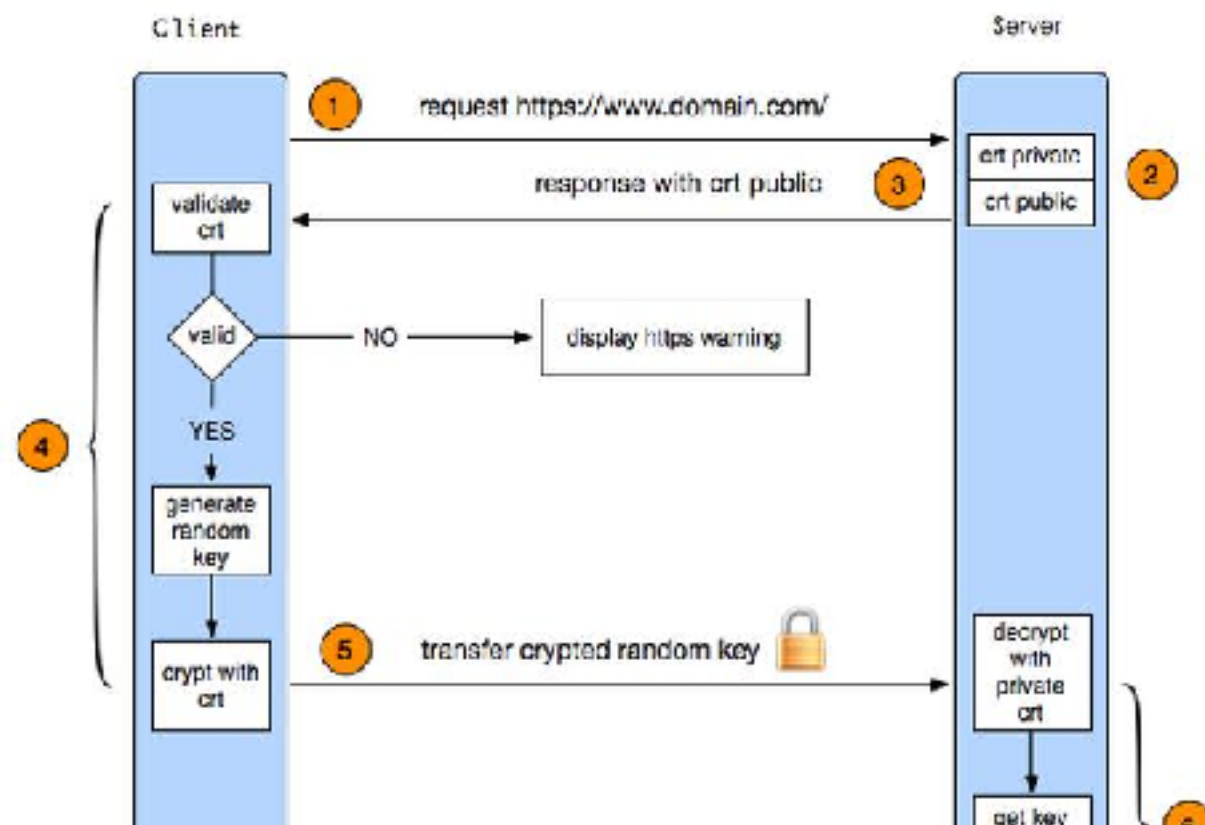
## HTTPS原理

我们先看一下定义，来自wikipedia的一个介绍：

HTTPS (also called **HTTP over Transport Layer Security (TLS)**, **HTTP over SSL**, and **HTTP Secure**) is a communications protocol for secure communication over a computer network which is widely used on the Internet. HTTPS consists of communication over Hypertext Transfer Protocol (HTTP) within a connection encrypted by Transport Layer Security, or its predecessor, Secure Sockets Layer. The main motivation for HTTPS is authentication of the visited website and protection of the privacy and integrity of the exchanged data.

从这个定义中我们可以看出，HTTPS是包含了HTTP协议及SSL /TLS协议这两部分内容，简单的理解就是基于SSL/TLS进行HTTP的加密传输。HTTP是一个应用层的协议，定义了很多请求和响应方通信的遵循的规则，这部分内容可以从HTTP权威指南([https://link.jianshu.com?t=https://www.amazon.cn/HTTP%E6%9D%83%E5%A8%81%E6%8C%87%E5%8D%97-%E5%90%89%E5%B0%94%E5%88%A9/dp/B008XFDQ14/ref=sr\\_1\\_1/460-1292118-1495910?s=books&ie=UTF8&qid=1502806302&sr=1-1&keywords=HTTP%E6%9D%83%E5%A8%81%E6%8C%87%E5%8D%97](https://link.jianshu.com?t=https://www.amazon.cn/HTTP%E6%9D%83%E5%A8%81%E6%8C%87%E5%8D%97-%E5%90%89%E5%B0%94%E5%88%A9/dp/B008XFDQ14/ref=sr_1_1/460-1292118-1495910?s=books&ie=UTF8&qid=1502806302&sr=1-1&keywords=HTTP%E6%9D%83%E5%A8%81%E6%8C%87%E5%8D%97))这部巨作中得到很详细的介绍，这里就不赘述了。其实主要还是想探讨一下SSL/TLS协议的一些具体细节，毕竟这是HTTPS区别于HTTP最大的地方，首先我们来看下一个SSL/TLS完整的握手过程。





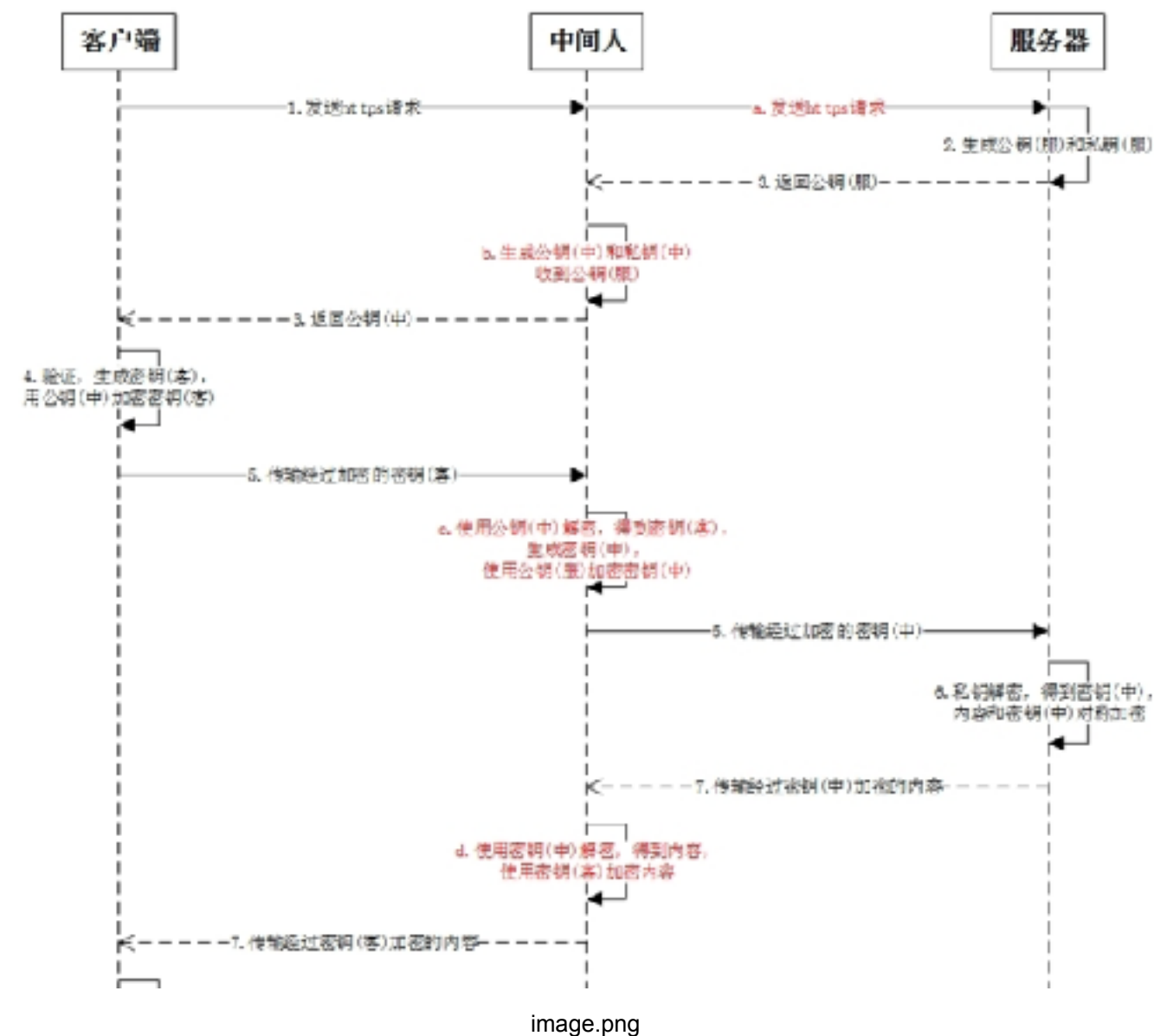
SSL/TLS握手过程

很复杂的交互过程，但是理解下来就是用非对称加密的手段传递密钥，然后用密钥进行对称加密传递数据。在这个握手过程中最重要的就是证书校验，其他就是正常的数据交互过程。如何校验一个证书合法有很大的文章，处理不好就会让你的网络失去了安全性。一个证书的校验，主要包括以下几个方面：

- 第一，校证书是否是由客户端中“受信任的根证书颁发机构”颁发；
- 第二，校证书是否在上级证书的吊销列表；
- 第三，校证书是否过期；
- 第四，校证书域名是否一致。

一天我们的QA妹子气愤愤的找到我说，为啥别人的APP可以用Charles抓到HTTPS的包，为啥我们的不能，我心中窃喜的告诉她只能说明我们技高一筹了。具体如何做到的后面我会分享一下我们的做法，先讨论一下Charles如何实现https的抓包的，这里面涉及到一个中间人攻击的问题。

一个针对SSL的中间人攻击过程如下：



中间人其实是做了一个偷梁换柱的动作，核心是如何欺骗客户端，从而让客户端能够放心的与中间人进行数据交互而没有任何察觉。我们来看Charles如何做到HTTPS抓包的，网上有很多Charles如何抓HTTPS包的教程，几步就搞定了，其中最核心的就是：

将私有CA签发的数字证书安装到手机中并且作为受信任证书保存

自签发一个证书实现上述二、三、四条校验规则很简单，要把这个证书安装到手机端信任列表必须得到用户的许可，这里不好绕过，但是鉴于大部分用户的网络安全意识比较差，有时也会稀里糊涂的信任了，那我们作为APP的开发人员，能否避免这种情况的发生呢？

其实也很简单，我们把服务端的证书内置在我们的APP里，我们在做服务端证书校验的时候只比对是否和这个证书完全相同，不同就直接抛错，那中间人便没有办法绕过证书进行攻击。但是这里面也有一个问题就是服务端的证书可能会过期或者升级，而且服务



端往往为了提高网络的安全性，证书的有效时间不会设置太长，这样APP就会因为这个证书的事情频繁发版，也很痛苦。（前段时间我司IOS的APP就是因为授权企业用户的证书没有及时更新，导致大家无法正常打开APP，血的教训导致我们不想重走这条路）可能你又想到了，我们可以把证书配置在后端，有更新的时候直接去下载不就完了，那我们的证书下载没有没拦截的风险吗，一旦拦截，我们所有的证书校验都会失效，比直接信任手机内置的证书更可怕。我们既不想只信任我们服务器的证书，又不想信任手机上所有的 CA 证书。有个不错的信任方式是把签发我们服务器的证书的根证书导出打包到APP中，这样虽然不能做到百分之百的证书无漏洞，但是相比于信任手机中几百个证书，我们只信任一个风险会小很多，这也就是我们的QA妹子用Charles抓不了我们的包的原因。~~~

## OKHTTP

作为一个Android开发者，我们来看一下牛逼闪闪的网络库OKHTTP对于HTTPS的支持。下面这段话摘自OKHTTP对于HTTPS的介绍中（地址请戳 (<https://link.jianshu.com?t=https://github.com/square/okhttp/wiki/HTTPS>)，中文地址 (<https://www.jianshu.com/p/2cdb81bb0c87>)）：

OkHttp attempts to balance two competing concerns:

- **Connectivity** to as many hosts as possible. That includes advanced hosts that run the latest versions of boringssl (<https://link.jianshu.com?t=https://boringssl.googlesource.com/boringssl/>) and less out of date hosts running older versions of OpenSSL (<https://link.jianshu.com?t=https://www.openssl.org/>).
- **Security** of the connection. This includes verification of the remote webserver with certificates and the privacy of data exchanged with strong ciphers.

几个与HTTPS相关的API：

### SSLSocketFactory:

安全套接层工厂，用于创建SSLSocket。默认的SSLSocket是信任手机内置信任的证书列表，我们可以通过OkHttpClient.Builder的sslSocketFactory方法定义我们自己的信任策略，比如实现上面提到的我们只信任服务端证书的根证书，代码实现如下：



```

/**
 * 载入证书
 */
public static SSLSocketFactory getSSLSocketFactory(InputStream... certificates) {
    try {
        //用我们的证书创建一个keystore
        CertificateFactory certificateFactory = CertificateFactory.getInstance("X.509");
        KeyStore keyStore = KeyStore.getInstance(KeyStore.getDefaultType());
        keyStore.load(null);
        int index = 0;
        for (InputStream certificate : certificates) {
            String certificateAlias = "server"+Integer.toString(index++);
            keyStore.setCertificateEntry(certificateAlias, certificateFactory.generateCertificate(certificate));
            try {
                if (certificate != null) {
                    certificate.close();
                }
            } catch (IOException e) {
                e.printStackTrace();
            }
        }
        //创建一个trustmanager, 只信任我们创建的keystore
        SSLContext sslContext = SSLContext.getInstance("TLS");
        TrustManagerFactory trustManagerFactory =
            TrustManagerFactory.getInstance(TrustManagerFactory.getDefaultAlgorithm());
        trustManagerFactory.init(keyStore);
        sslContext.init(
            null,
            trustManagerFactory.getTrustManagers(),
            new SecureRandom()
        );
        return sslContext.getSocketFactory();
    } catch (Exception e) {
        e.printStackTrace();
        return null;
    }
}

```

## X509TrustManager:

```

public interface X509TrustManager extends TrustManager {
    void checkClientTrusted(X509Certificate[] var1, String var2) throws CertificateException;

    void checkServerTrusted(X509Certificate[] var1, String var2) throws CertificateException;

    X509Certificate[] getAcceptedIssuers();
}

```

checkServerTrusted方式实现了对于服务端校验，这里一般使用系统默认的实现，有些教程讲到这样配置ssl



```
private static synchronized SSLSocketFactory getDefaultSSLSocketFactory() {
    try {
        SSLContext sslContext = SSLContext.getInstance("TLS");
        sslContext.init(null, new TrustManager[]{
            new X509TrustManager() {
                public void checkClientTrusted(X509Certificate[] x509Certificates, String s) {}

                public void checkServerTrusted(X509Certificate[] x509Certificates, String s) {}

                public X509Certificate[] getAcceptedIssuers() {
                    return new X509Certificate[0];
                }
            }, null);
        return sslContext.getSocketFactory();
    } catch (GeneralSecurityException e) {
        throw new AssertionError();
    }
}
```

千万不能这么做，这样将你是没有做任何校验的，这里推荐使用系统默认的，他会在校验过程中发现有异常直接抛出。

## HostnameVerifier:

```
public interface HostnameVerifier {
    boolean verify(String var1, SSLSession var2);
}
```

这个接口主要实现对于域名的校验，OKHTTP实现了一个OkHostnameVerifier，对于证书中的IP及Host做了各种正则匹配，默认情况下使用的是这个策略。有些你遇到了一些奇怪的校验问题，大部分教程会教你这样：

```
OKHttpClient.Builder.hostnameVerifier(new HostnameVerifier() {
    @Override
    public boolean verify(String hostname, SSLSession session) {
        return true;
    }
})
```

其实这样你是完全放弃了hostname的校验，这也是相当不安全的。

小礼物走一走，来简书关注我





panda\_Vicky (/u/28d9a4ba4cbc)

写了 7772 字，被 64 人关注，获得了 150 个喜欢 (/u/28d9a4ba4cbc)

+ 关注

快要起飞了。。。 Read the fucking source code 个人主页 <https://pandavickey.github.io/>

♡ 喜欢 (/sign\_in?utm\_source=desktop&utm\_medium=not-signed-in-like-button)

57




更多分享

([http://cwb.assets.jianshu.io/notes/images/15763342/weibo/image\\_02d110188c45.j](http://cwb.assets.jianshu.io/notes/images/15763342/weibo/image_02d110188c45.j)



下载简书 App ▶

随时随地发现和创作内容



(/apps/download?utm\_source=nbc)




登录后发表评论 (/sign\_in?utm\_source=desktop&utm\_medium=not-signed-in-comment-form)

11条评论

只看作者

按喜欢排序 按时间正序 按时间倒序




涂图云 (/u/d6baed7b002a)

2楼 · 2017.08.23 15:52

(/u/d6baed7b002a)

👍

👍 赞    💬 回复




听见海的声音 (/u/a03909d4d11d)

3楼 · 2017.08.23 20:37

(/u/a03909d4d11d)


👍


👍 赞    💬 回复

 **ttdevs** (/u/1c66a2a7fc6f)  
4楼 · 2017.08.23 23:35  
(/u/1c66a2a7fc6f)  
后面操作代码部分分析的更详细点就更好了

👍 赞    💬 回复

panda\_Vicky (/u/28d9a4ba4cbc) : @ttdevs (/users/1c66a2a7fc6f) 嗯，我再整理一下  
2017.08.24 09:14    💬 回复

 添加新评论


 **雷栋** (/u/aebfac77e3b7)  
5楼 · 2017.08.24 09:35  
(/u/aebfac77e3b7)  
传递的是公钥吧，私钥自己保存不能传的


👍 赞    💬 回复

panda\_Vicky (/u/28d9a4ba4cbc) : @雷栋 (/users/aebfac77e3b7) 没有说传递私钥啊，服务器下发证书，证书中包含公钥，客户端利用公钥加密传递对称加密的密钥  
2017.08.24 09:56    💬 回复

今晚不吃菜 (/u/f225db8d3884) : @panda\_Vicky (/users/28d9a4ba4cbc) 你的原文：很复杂的交互过程，但是理解下来就是用非对称加密的手段传递私钥，然后用私钥进行对称加密传递数据。这句话容易误导人，非对称加密传递的是密钥（供传输过程对称加密使用），对称加密没有公钥私钥之说，应该是密钥。  
2017.08.25 12:51    💬 回复

今晚不吃菜 (/u/f225db8d3884) : @今晚不吃菜 (/users/f225db8d3884) 我也说错了，密钥 => 密钥  
2017.08.25 12:54    💬 回复

 添加新评论    还有1条评论， 展开查看

 **相互交流** (/u/6acf80da51eb)  
6楼 · 2017.08.25 07:53  
(/u/6acf80da51eb)  
服务器的根证书打包，你们服务器有几个证书？而且根证书，不会替换？

👍 赞    💬 回复





panda\_Vicky (/u/28d9a4ba4cbc)：服务器一般在一个CA机构去申请证书


2017.08.25 09:30 回复


添加新评论


被以下专题收入，发现更多相似内容

 Android... (/c/58b4c20abf2f?utm\_source=desktop&utm\_medium=notes-included-collection)


 Android... (/c/27d03e4a4bd2?utm\_source=desktop&utm\_medium=notes-included-collection)

 Android开发 (/c/0dc880a2c73c?utm\_source=desktop&utm\_medium=notes-included-collection)

 Net (/c/d24df793adb8?utm\_source=desktop&utm\_medium=notes-included-collection)

 Android... (/c/5139d555c94d?utm\_source=desktop&utm\_medium=notes-included-collection)

 程序员 (/c/NEt52a?utm\_source=desktop&utm\_medium=notes-included-collection)


 程序员首页投稿 (/c/89995286335f?utm\_source=desktop&utm\_medium=notes-included-collection)

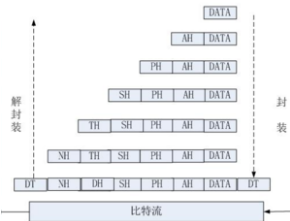
展开更多

(/p/116ebf3034d9?  
utm\_campaign=maleskine&utm\_content=note&utm\_medium=seo\_notes&utm\_source=recommendation)

OKHttp源码解析(二)：“前戏”——HTTP的那些事 (/p/116ebf3034d9?utm\_...

1.OkHttp源码解析(一):OKHttp初阶2 OkHttp源码解析(二):OkHttp连接的"前戏"——HTTP的那些事3 OkHttp源码解析(三):OkHttp中阶之线程池和消息队...

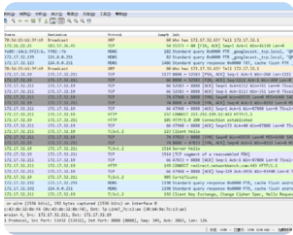
 隔壁老李头 (/u/8b9c629f69dd?)



utm\_campaign=maleskine&utm\_content=user&utm\_medium=seo\_notes&utm\_source=recommendation)




(/p/cf8c2f2cd18a?



utm\_campaign=maleskine&utm\_content=note&utm\_medium=seo\_notes&utm\_source=recommendation)

Wireshark 抓包理解 HTTPS 请求流程 (/p/cf8c2f2cd18a?utm\_campaign=...


准备 分析2.1. 三次握手2.2. 创建 HTTP 代理（非必要）2.3. TLS/SSL 握手2.4. 数据传输2.5. 四次挥手 扩展  
3.1. 摘要、MAC、数字签名到 CA 证书3.2. Session ID 和 Session Ticket3.3. SNI (Ser...

 李狄青 (/u/5e87e5d9ab47?

utm\_campaign=maleskine&utm\_content=user&utm\_medium=seo\_notes&utm\_source=recommendation)

ios中用AFN做https (/p/c9f980786f87?utm\_campaign=maleskine&utm\_...


原文地址 <http://blog.csdn.net/u012409247/article/details/49852919> SSL/TLS协议运行机制的概述  
( [http://www.ruanyifeng.com/blog/2014/02/ssl\\_tls.html](http://www.ruanyifeng.com/blog/2014/02/ssl_tls.html) ) 一、作用 ...

 123321123 (/u/0fbf551ff6fb?

utm\_campaign=maleskine&utm\_content=user&utm\_medium=seo\_notes&utm\_source=recommendation)

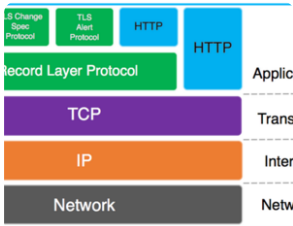
OpenSSL (/p/2e69d56e470b?utm\_campaign=maleskine&utm\_content...

原版：<http://blog.csdn.net/jun2ran/article/details/6491375> 第一章 前言第二章 证书第三章 加密算法第四章  
协议第五章 入门第六章 指令 verify第七章 指令asn1parse第八章 指令CA（一）第九章 指令CA（二...

 依忆依意壹懿 (/u/c80bc26f12ed?

utm\_campaign=maleskine&utm\_content=user&utm\_medium=seo\_notes&utm\_source=recommendation)


(/p/c55d52b2c811?



utm\_campaign=maleskine&utm\_content=note&utm\_medium=seo\_notes&utm\_source=recommendation)

Android 项目https网络请求封装及遇到的一些问题 (/p/c55d52b2c811?ut...

因为最近ios需要用到https，所以公司的项目都从http的请求转成了https的双向认证，这里我关于安卓端https  
相关的知识点以及在请求过程中遇到的一些问题。 一、https的介绍以及相关的专有名词 https 简单来说，...

 黄海佳 (/u/05402282c2d9?

utm\_campaign=maleskine&utm\_content=user&utm\_medium=seo\_notes&utm\_source=recommendation)

(/p/5aba0caa192f?





utm\_campaign=maleskine&utm\_content=note&utm\_medium=seo\_notes&utm\_source=recommendation)

### 手机是用来捕捉我人生的相机 (/p/5aba0caa192f?utm\_campaign=maleski...

奥斯汀·曼 ( John Austin Mann ) 用 iPhone 照了很多照片，将其所有图片发表在个人博客中，她坚持了 10 年，她认为 iPhone 是可以「用来捕捉我人生的相机」。我认为这是一个非常有力的一句名言，并不是最...



豆浆果子是我晚餐 (/u/2759c2e928b4?

utm\_campaign=maleskine&utm\_content=user&utm\_medium=seo\_notes&utm\_source=recommendation)

### 重回焦点团体 (/p/5a9de3205119?utm\_campaign=maleskine&utm\_conte...

周艳艳，焦点三期，平顶山，坚持分享第1天 大家好，我是周艳艳，以前也是咱们三期的成员。可是在一起成长的过程中，我掉队了，现在特别想念我们的团队，想回归集体，可是内心又充满了惶恐，感觉和大家...



潼娃妈妈 (/u/d32d85718da5?

utm\_campaign=maleskine&utm\_content=user&utm\_medium=seo\_notes&utm\_source=recommendation)

### 2016.9.23 上午 《随遇而安-孟非著》读后感 (/p/02cd6451f272?utm\_camp...

这本来是昨天的任务，但是昨天因为太累了，实在不想写，就留到了今天早上。起码能写，这我就很满足了，本来写文也是为了培养自己的一个习惯，想写东西，有精力，有时间的时候就写，当成任务来做...似...



樟Yi (/u/9f386bc9cf51?

utm\_campaign=maleskine&utm\_content=user&utm\_medium=seo\_notes&utm\_source=recommendation)

### 我想告诉你一个秘密【真人故事】 (/p/5cf850b7ec66?utm\_campaign=ma...

我是一名中学生，一星期前我华文试卷出了个这么样的一个题“告诉你一个秘密”。毫不犹豫的我选中了这题，开始下笔默默地把我感受写了出来。 以下内容是这样的：我不再是单纯自由的小孩了，我现在已是面临...



夜之声 (/u/d69991757efe?

utm\_campaign=maleskine&utm\_content=user&utm\_medium=seo\_notes&utm\_source=recommendation)

### 穿越时空的爱恋 (/p/de7d0c32b6ab?utm\_campaign=maleskine&utm\_co...

我知道他爱我，也因此我更渴望他在我的时空里守护我。 01 "宝贝，早，起床上班啦~"早上7点，我睡眼惺忪的看了一下手机微信，这是上个情人节他送我的魅族手机，这好像就是他的陪伴，我们还共同拥有了一...



麦乐迪 (/u/5f6cb5c84be0?

utm\_campaign=maleskine&utm\_content=user&utm\_medium=seo\_notes&utm\_source=recommendation)

