HTTPS 基础

HyperText Transfer Protocol over Secure Socket Layer
即 HTTP+SSL/TLS

找靓机-黄隆

加密算法

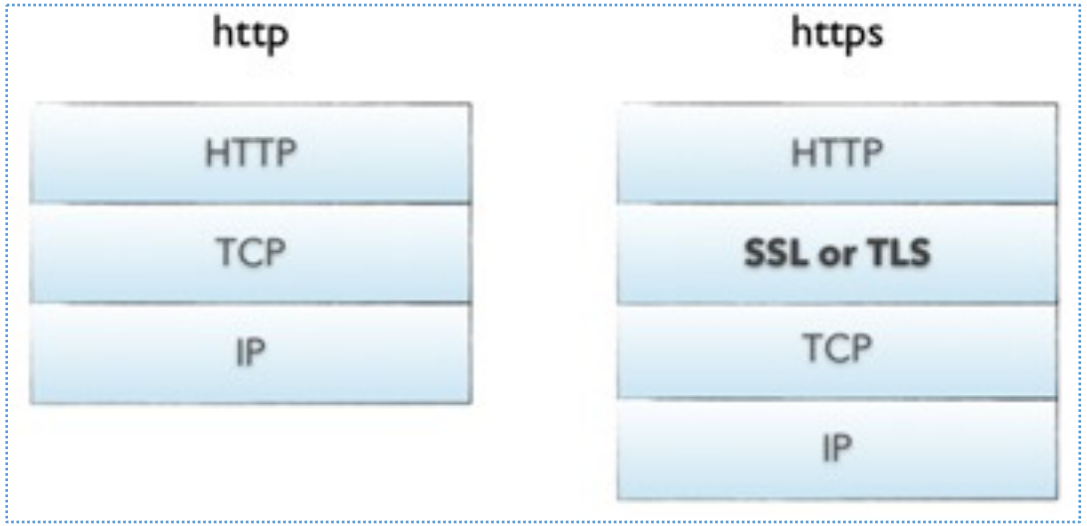
对称加密：采用这种加密方法的双方使用方式用**同样的密钥进行加密和解密**。密钥是控制加密及解密过程的指令。算法是一组规则，规定如何进行加密和解密。如DES、3DES、TDEA、Blowfish、RC2、RC4、RC5、IDEA、SKIPJACK、AES等

- **算法公开、计算量小、加密速度快、加密效率高，适合加密大量数据。**

非对称加密：需要两个密钥来进行加密和解密，这两个秘钥是公开密钥（public key，简称公钥）和私有密钥（private key，简称私钥）。公开密钥与私有密钥是一对，如果用公开密钥对数据进行加密，只有用对应的私有密钥才能解密；如果用私有密钥对数据进行加密，那么只有用对应的公开密钥才能解密。因为加密和解密使用的是两个不同的密钥。如RSA、Elgamal、背包算法、Rabin、D-H、ECC（椭圆曲线加密算法）等。

- **安全性更好，加密和解密花费时间长、速度慢，只适合对少量数据进行加密。**

Http和Https



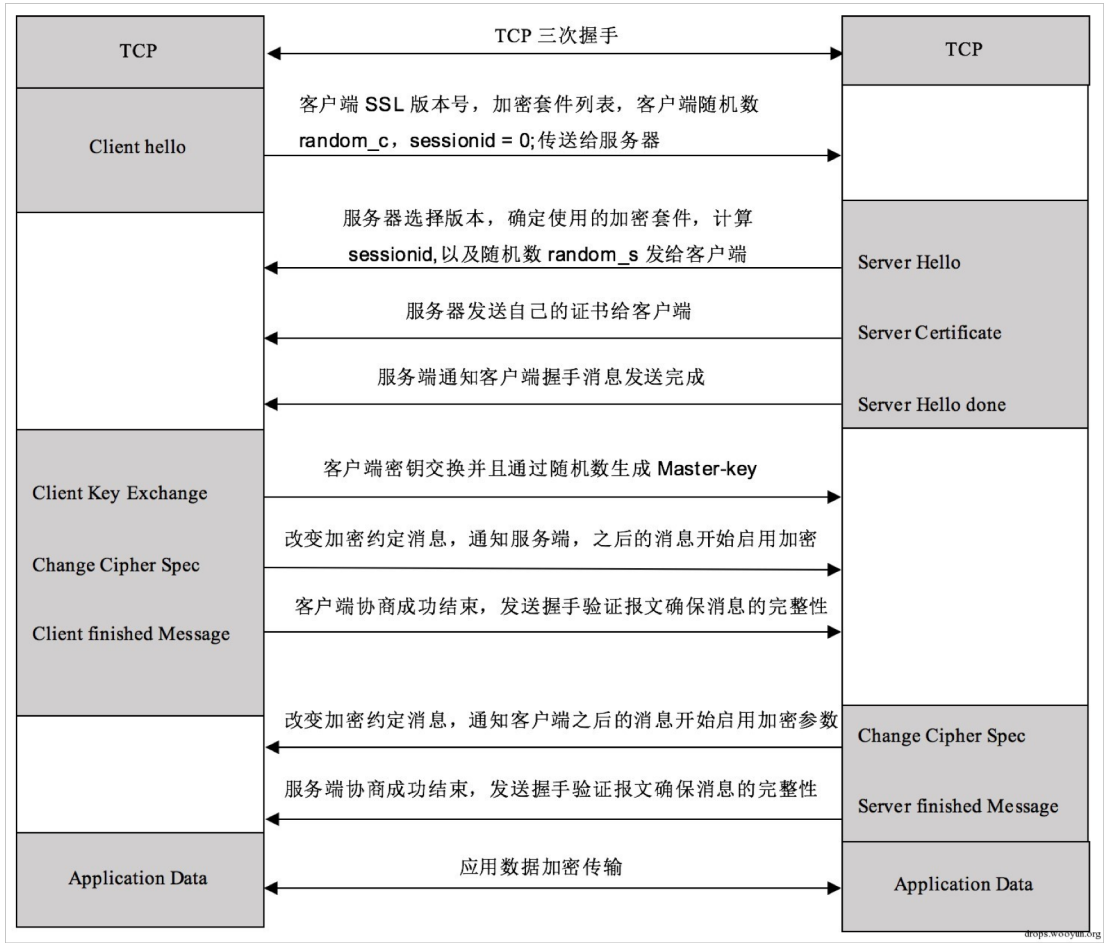
https协议的作用

1. 建立一个信息安全通道，来保证数据传输的安全；数字签名是保证数据不被篡改，加密是保证不被窃取。
2. 确认网站的真实性。

SSL和TLS

- SSL：（Secure Socket Layer，安全套接层协议），SSL协议位于TCP/IP协议与各种应用层协议之间，为数据通讯提供安全支持。SSL通过互相认证、使用数字签名确保完整性、使用加密确保机密性，以实现客户端和服务器之间的安全通讯。该协议由两层组成：SSL记录协议和SSL握手协议。
- Secure Socket Layer是Netscape于1994年开发的，目前有三个版本：SSL2.0、SSL3.0、SSL3.1，最常用的是1995年发布的第3版，已被广泛地用于Web浏览器与服务器之间的身份认证和加密数据传输。
- TLS：(Transport LayerSecurity，传输层安全协议)，是IETF(工程任务组)制定的一种新的协议，它建立在SSL 3.0协议规范之上，是SSL 3.0的后续版本，目前有TLS 1.0，TLS1.1，TLS1.2等版本。

HTTPS 握手



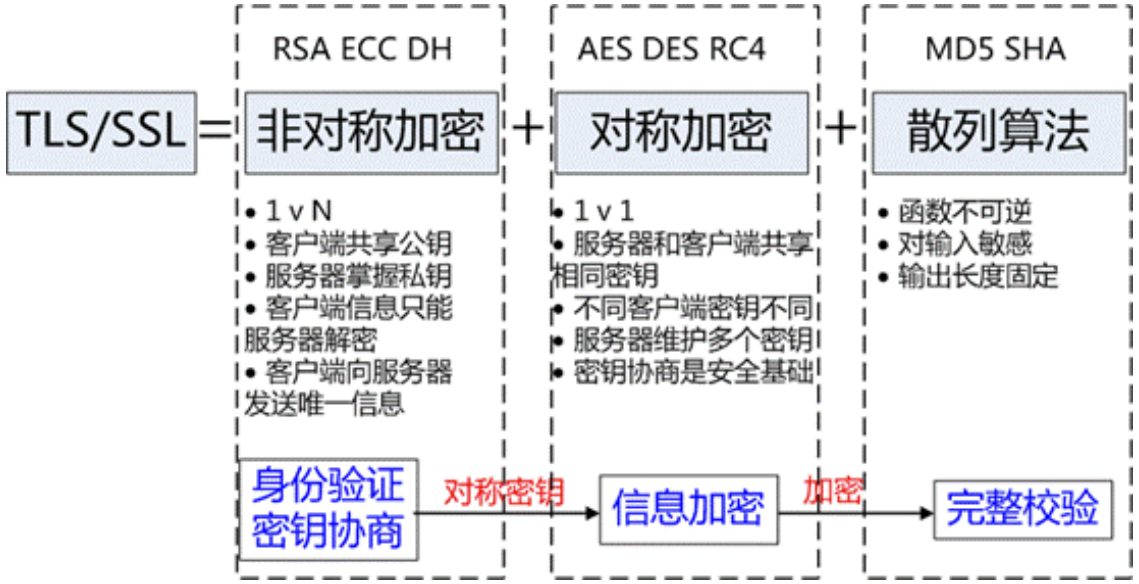
1. 客户端发起握手请求，携带随机数、支持算法列表等参数。
2. 服务端收到请求，选择合适的算法，下发公钥证书和随机数。
3. 客户端对服务端证书进行校验，并发送随机数信息，该信息使用公钥加密。
4. 服务端通过私钥获取随机数信息。
5. 双方根据以上交互的信息生成session ticket，用作该连接后续数据传输的加密密钥。

相比HTTP，额外开销的产生

1. 非对称算法的加解密在握手过程中的计算量很大，占据了握手过程中绝大多数的计算量。 非对称加密算法不能在加密超过自己公钥长度的内容。
2. RSA协商密钥2个RTT（ Round-Trip Time ），四次握手，RSA算法主要作用是加密premaster_secret。
3. 最后由PRF(pseudo-random function)伪随机函数，生成master_secret。
4. ECDHE密钥协商算法，对于ECDHE密钥协商算法比RSA多一次握手，ECDHE在服务器端发送一个含有ECDH_PARA的参数的握手信息。

握手过程相关算法

- 服务端身份验证：数字签名（RSA、ECDSA）
- 密钥交换：RSA/密钥交换算法（ECDH）
- 加密/解密：流加密（RC4）和分组加密（3DES/AES/AESGCM）
- 生成消息认证码：SHA/AEAD



Https的优点

1、https协议是**密码学**的应用结晶。对称加密算法和非对称加密算法相结合。**公钥加密的内容只能通过私钥解密，私钥加密的内容只能通过公钥解密。**非对称加密算法主要用于**数字签名检验**和**交换密匙**

2、https证书由第三方权威机构签发和管理，如Symantec、Comodo、GoDaddy、GlobalSign。

3、客户端会校验证书的真实性，访问的域名和证书信息一致性等。

4、每次连接都会创建随机会话密匙。

5、使用HTTPS协议可认证用户和服务器，确保数据发送到正确的客户机和服务器。

6、HTTPS协议是由SSL+HTTP协议构建的可进行加密传输、身份认证的网络协议，要比http协议安全，可防止数据在传输过程中不被窃取、改变，确保数据的完整性。如**防止运营商劫持，弹广告，盗取隐私信息等。**

7、HTTPS是现行架构下最安全的解决方案，虽然不是绝对安全，但它大幅增加了中间人攻击的成本。

Https的缺点

1）HTTPS协议握手阶段比较费时，会使页面的加载时间延长近50%，增加10%到20%的耗电。现在有些机器提供了硬件加密解密，性能会有所提升。

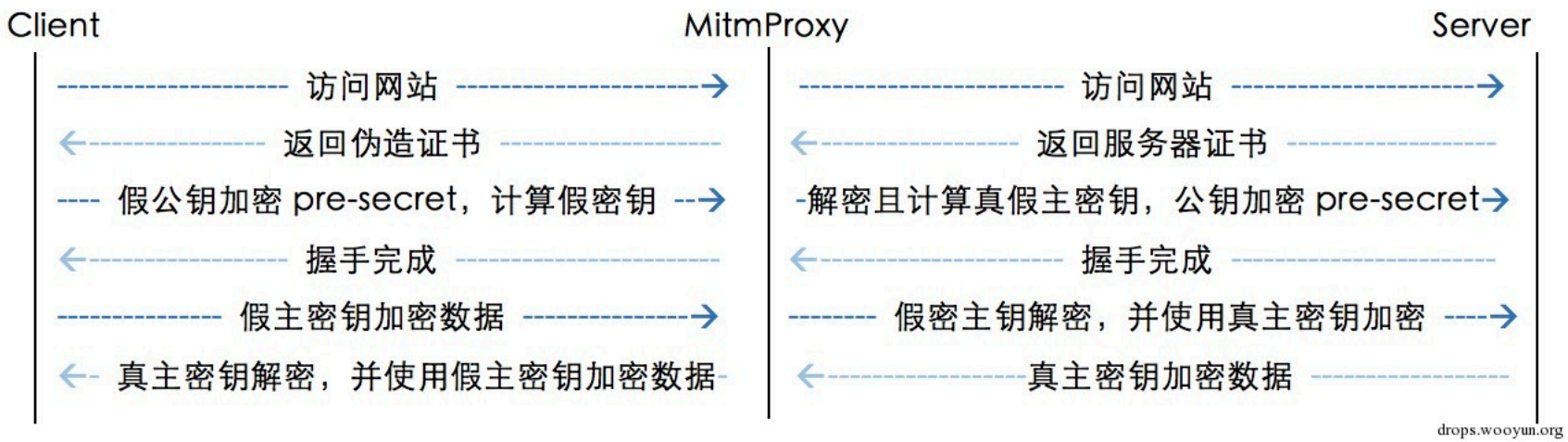
2）HTTPS连接缓存不如HTTP高效，会增加数据开销和功耗，甚至已有的安全措施也会因此而受到影响。如一个网站使用混合内容http+https，因为跨域会导致不安全，缓存失效等，如 Android-21 webview 默认不允许加载混合内容，需要配置 WebSettings.MIXED_CONTENT_ALWAYS_ALLOW进行兼容。

3）SSL证书需要钱，功能越强大的证书费用越高。

4）SSL证书通常需要绑定IP，不能在同一IP上绑定多个域名，IPv4资源不可能支撑这个消耗。 SSL证书通常是颁发给域名的，但有很多客户需要为IP申请SSL证书。

5）HTTPS协议的加密范围也比较有限，在黑客攻击、拒绝服务攻击、服务器劫持等方面几乎起不到什么作用。最关键的，SSL证书的信用链体系并不安全，特别是在某些国家可以控制CA根证书的情况下，中间人攻击一样可行。

中间人攻击原理



代理抓包原理

DNS劫持

由于SSL证书通常是颁发给域名的，而排除用户个人行为修改，
如下两个地方可能会被黑客攻击。

1. 用户路由器DNS服务设置
2. ISP运营商DNS服务设置

怎样解决问题？

HTTPDNS解决DNS攻击劫持

HTTPDNS使用HTTP协议进行域名解析，代替现有基于UDP的DNS协议，域名解析请求直接发送到HTTPDNS服务器，从而绕过运营商的Local DNS，能够避免Local DNS造成的域名劫持问题和调度不精准问题。

用Https协议（IP代替域名）向HttpDns集群（权威DNS）的443端口进行请求，代替传统的DNS协议向DNS服务器的53端口进行请求。也就是使用Https协议去进行dns解析请求，将服务器返回的解析结果，也就是域名对应的服务器ip获得，直接向该ip发起对应的api服务请求，代替使用域名。备选情况下（HttpsDns解析失败），再走传统的LocalDNS解析方式。

防止中间人攻击

1、将证书导入app本地，本地检验证书签名，iOS AFN提供了此机制。换证书麻烦，维护成本高。

2、双向认证，金融级方案都要求使用https双向认证。

何为SSL/TLS单向认证，双向认证?
单向认证指的是只有一个对象校验对端的证书合法性。
通常都是client来校验服务器的合法性。那么client需要一个ca.crt,服务器需要server.crt,server.key
双向认证指的是相互校验，服务器需要校验每个client,client也需要校验服务器。
server 需要 server.key 、server.crt 、ca.crt
client 需要 client.key 、client.crt 、ca.crt

SSLPinningMode 定义了https连接时，如何去校验服务器端给予的证书。

```
typedef NS_ENUM(NSUInteger, AFSSLPinningMode) {
    AFSSLPinningModeNone,
    AFSSLPinningModePublicKey,
    AFSSLPinningModeCertificate,
};
```

AFSSLPinningModeNone: 代表客户端无条件地信任服务器端返回的证书。
AFSSLPinningModePublicKey: 代表客户端会将服务器端返回的证书与本地保存的证书中，PublicKey的部分进行校验；如果正确，才继续进行。
AFSSLPinningModeCertificate: 代表客户端会将服务器端返回的证书和本地保存的证书中的所有内容，包括PublicKey和证书部分，全部进行校验；如果正确，才继续进行。

为什么https之后还要数字签名？

- 虽然https在传输层已经保证了数据传输的完整性（篡改、窃取和冒充）即端到端数据传输的安全性，但是用户网络环境总是复杂的，避免不了一些在传输前、传输后或者中间人被篡改的可能性，所以业务层需要摘要算法进行数字签名，如MD5、SHA。
- Api用MD5(secret+参数+时间戳) 防止重放，避免不必要的攻击。理论上https是可以防止重放的。TLS通过为每个连接到处一组新秘钥并为每个记录分配唯一的序列号来保证加密流是不可重放的。这个防止攻击者复制这些记录并在另一个连接上重放这些记录，因为加密密钥不匹配。在同一个连接重放这些记录也不起作用，因为序列号不匹配，并且记录将被拒绝。
- MD5(secret+参数+时间戳)提高恶意调用的成本
- 万一证书被猫腻了或者dns被劫持，进行业务层的数据校验显得更安全和严谨。
- Client---https---Load Balance---http---http sever，链路复杂，业务层做数字签名更安全和严谨。

安全和代价

- 没有绝对安全，所有的安全都是有价格的。成本和产生差距大，自然容易成被攻击的对象，成本大于产出，自然就没有人攻击，也就安全了。
- 使用https也有利提高用户体验，如Google Chrome会将https站点标记为安全，搜索引擎排名会靠前，杜绝运营商劫持弹广告，防止运营商抓取用户隐私信息。
- 内网通讯不建议使用https，外部系统对接建议使用https，不使用废弃的版本，如SSL3.0。
- https主要用于传输通道的防护，即保证数据传输完整性（不被窃取和不被篡改）。但是用户的网络环境是复杂的，并不能保证传输前，传输后不被篡改，所以支付宝开发api要求用https之余还在业务层用MD5，sha等做数字签名。另外用户请求到达负载均衡之后一般就会转成http在内网传输。