**·账户密码加密**

参考：

1. 密码加密
2. DES（对称加密）+RES（非对称加密）
3. 将账号和密码通过一定的规则组合成一个字符串，然后生成一个随机密钥，以该随机密钥为KEY用DES加密成A1；
4. 随机密钥以约定的公钥用RES加密成A2；
5. 将A1和A2同时发送给后台，后台根据RES私钥解析A2获取随机密钥，在通过随机密钥DES解密A1得到账户和密码。
6. MD5（非对称加密）
7. 将密码通过公钥用MD5加密；
8. 将账户和加密的密码发送给后台；
9. 后台获取账户，并通过私钥解密得到密码；
10. cookie持久化
11. 将cookie名和cookie值存储在同一个sharedpreference里；
12. 将所有cookie名以特定规则拼接成一个字符串，以同一KEY（如：cookietName）保存在sharedpreference；
13. 分别以cookie名为key，cookie值为value的键值对保存在sharedpreference；
14. 通过账户KEY获取所有cookie名的字符串，并解析得到cookie名数组；遍历cookie名数组，以cookie名为KEY，获取对应的cookie值；
15. Token持久化

用户登陆后，后台给客户端返回一个持久化的token，客户端将token通过sharedpreference保存在本地。后台校验token的过期时间，过期则重新登陆；否则通过token获取cookie值，然后通过cookie访问后台。

1. Https请求服务端证书和客户端证书双向校验

参考：<https://www.cnblogs.com/lechance/p/4373173.html>

<http://www.droidsec.cn/%E6%B5%85%E6%9E%90https%E4%B8%AD%E9%97%B4%E4%BA%BA%E6%94%BB%E5%87%BB%E4%B8%8E%E8%AF%81%E4%B9%A6%E6%A0%A1%E9%AA%8C/>

1. 客户端校验服务端证书
2. 如何确保证书可信任
3. 自签名证书
4. 客户端内置服务端证书（证书失效时更新不方便）

发送https请求时，后台将证书下发给客户端，客户端拿到服务端证书后，跟内置证书比较是否一致，一致则为可信任证书；

1. 服务端证书动态下发（效率低，方便更新证书）
2. 生成一对RSA的公私钥，公钥可硬编码在app，私钥放服务器;
3. https握手前可通过服务器下发证书信息，例如公钥、办法机构、签名等，该下发的信息使用服务器里的私钥进行签名；
4. 客户端拿到加密的证书后，用内置的公钥解密
5. CA认证证书（android系统校验）
6. 证书有效性校验
7. 证书过期时间
8. 证书上的域名是否和服务器的实际域名相匹配
9. 校验证书链
10. 服务端校验客户端证书

将客户端证书放在assets或raw文件中，客户端发送网络请求时，读取客户端证书通过KeyManagerFactory 加载到SSLSocket中，并将其携带在网络请求中，发送给服务端。服务端通过通过白名单的形式，校验该客户端证书是在可信。