Token

服务端拿到用户名和密码，创建新的User，并根据User相关信息生成Token返回给客户端

客户端把Token存储起来，作为该User的唯一识别字符串

未来客户端请求User相关信息时带上Token，服务器端根据Token返回User相关数据
服务器端解析Token并对Token进行校验

客户端检测状态码，如果判断为是与Token相关的error/异常，则Token失效

如果Token失效，应该让强制用户登录，弹出登录界面

登录成功，客户端存储Token

HTTPS+有效的Token就可以获取用户的所有信息

如果Token被盗 .....?

1 调整Token过期时间，但在失效的这段时间内，仍然可以从服务器上随意获取数据

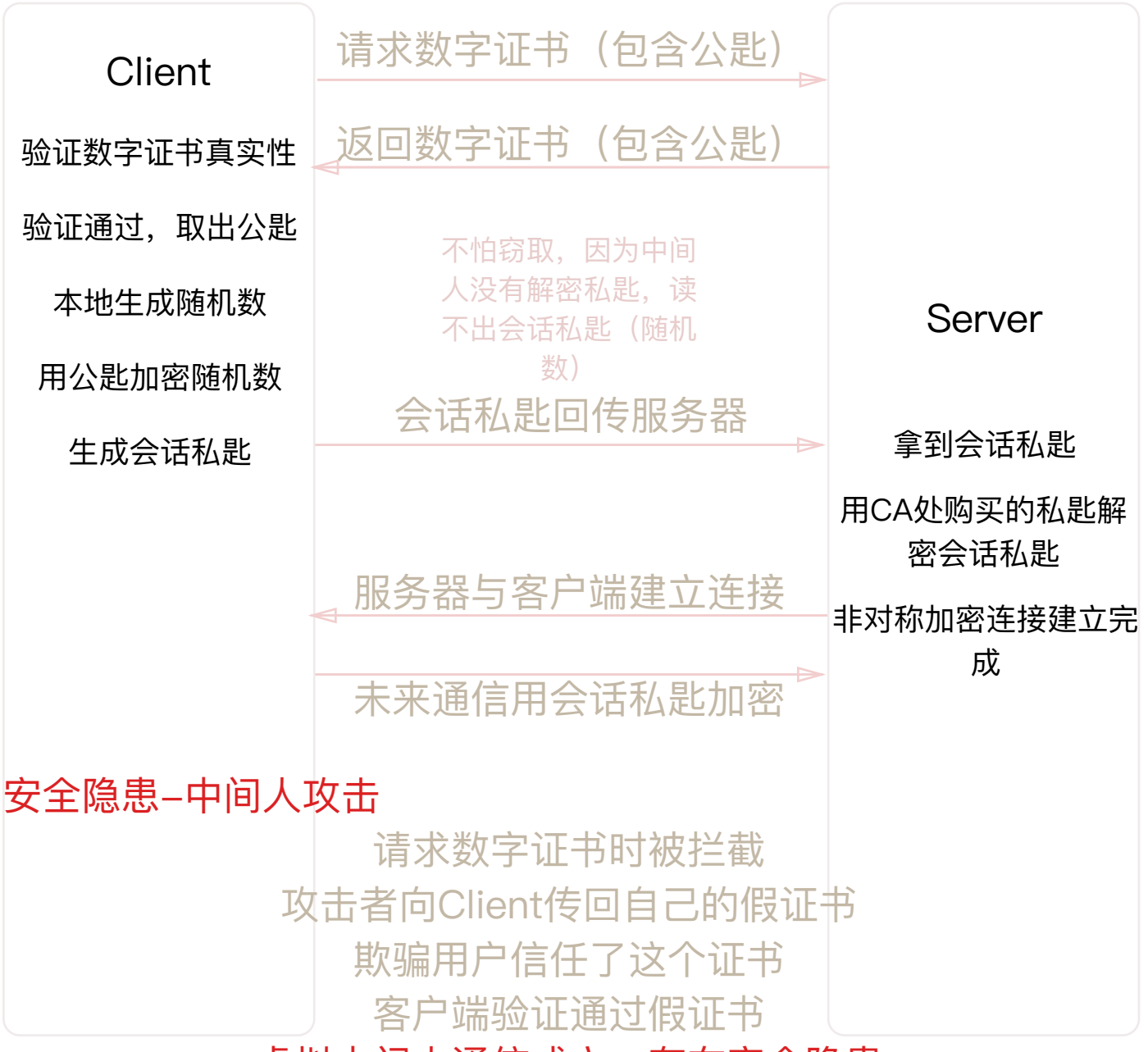
2 Request签名： 客户端和服务端统一好某种加密方法和一个密匙，密匙同时存储在服务端和客户端，客户端每发起一个请求，对请求的API和参数通过这种加密算法和密匙进行加密，从而得到加密后的字符串或数，这种方法称作Request签名。最后再把这个签名放到Request中与其他参数一起传给服务端，服务端对签名进行校验，即用同样的密匙和算法加密除签名之外的API参数，验证两个签名的真实性

用Request签名可以确保服务端所收到的Request都来自我们自己的客户端，即使有人得到Token想伪造Request，他也不知道如何计算Request签名，可以减少Token被盗盒被盗后的风险

Rrequest签名的漏洞：
Request签名的方法意味着我们必须在客户端保存好加密算法和密钥，这可以通过代码混淆，密匙存储到.so文件等来提高破解难度

HTTPS

购买数字证书，
存储在服务器上



安全隐患–中间人攻击

请求数字证书时被拦截
攻击者向Client传回自己的假证书
欺骗用户信任了这个证书
客户端验证通过假证书

虚拟中间人通信成立，存在安全隐患

解决方法：客户端直接绑定公匙，公匙存储在客户端本地

客户端收到公匙以后与存储在本地的公匙进行校验，防虚假证书