**问题：**

以常见的web应用程序开发中的登录认证界面的设计为例，通常的做法如下1-3。下面的作法有哪些安全问题？如何改进？你的web开发中d的登录界面通常还会使用哪种其他的方法？

1. web页面上用户输入账号、密码，点击登录
2. 请求提交前，web端首先通过客户端脚本如javascript对密码原文进行md5处理
3. 请求提交到后台，验证账号与密码是否与数据库中的一致，一致则认为登录成功，反之失败

**回答：**

1. 针对做法1：在Web页面上用户输入账号、密码，然后选择GET方式登陆后，便会在URL地址栏中看到登陆的用户名和密码，如此一来，便毫无安全性可言。
2. 针对做法2：即使使用了JS对密码进行MD5的加密，但这种方式仍然避免不了别人的恶意拦截请求，获取到MD5加密前的密码（如进入[www.cmd5.com](http://www.cmd5.com)网站，输入密文，即刻获得加密前的密码值）。
3. 针对做法3：通过POST请求，将用户名和密码提交到后台，但是可以利用诸如Firefox中的Firebug就能轻松拦截到数据体，得到用户名与密码，这种方法仍不安全。

在Web开发中的登陆界面还有以下处理方法：

1. 每次传输均随机加密：这种方式是每次请求均附加一个随机数（一般取时间戳），然后把用户输入的密码进行md5加密，加密后的值与随机数拼接，再取md5值，具体算法如下：Key = Md5(Md5(password)+ts)；然后登录接口接收三个参数：用户名，随机值（时间戳），加密后的Key，登录逻辑按照加密方式进行判断，服务端也可以根据时间戳来防止暴力登录
2. 使用RSA的非对称加密算法：在登陆界面，加入一段Java代码，生成公钥和私钥，私钥对象存储在session中，公钥在页面上输出给JS的机密函数。在登陆提交之前，先利用公钥对密码进行加密，然后再发送给服务器端；在登陆处理的URL中，从session获得私钥对象，对密码进行解密即可。