1. 【问题：】没有被验证的输入

【测试方法：】

数据类型（字符串，整型，实数，等）

允许的字符集

最小和最大的长度

是否允许空输入

参数是否是必须的

重复是否允许

数值范围

特定的值（枚举型）

特定的模式（正则表达式）

1. 【问题：】有问题的访问控制

【测试方法：】

主要用于需要验证用户身份以及权限的页面，复制该页面的URL地址，关闭该页面以后，查看是否可以直接进入该复制好的地址

【例：】从一个页面链到另一个页面的间隙可以看到URL地址

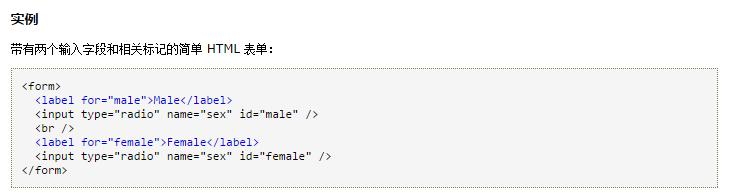
直接输入该地址，可以看到自己没有权限的页面信息

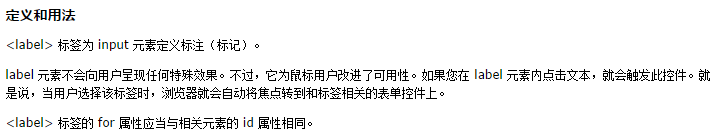
1. 【问题：】错误的认证和会话管理

【例：】对Grid、Label、Tree view类的输入框未作验证，输入的内容会按照html语法解析出来

Grid：网格标签，以网格形式来展示内容

Label：





Tree view：用来显示信息的分级视图，如同win中资源管理器目录

1. 【问题：】缓冲区溢出

没有加密关键数据

【例：】view－source：http地址可以查看源代码

在页面输入密码，页面显示的是 \*\*\*\*\*,  右键，查看源文件就可以看见刚才输入的密码

1. 【问题：】拒绝服务

分析：攻击者可以从一个主机产生足够多的流量来耗尽很多应用程序，最终使程序陷入瘫痪。需要做负载均衡来对付。

1. 【问题：】不安全的配置管理

分析： Config中的链接字符串以及用户信息，邮件，数据存储信息都需要加以保护

程序员应该作的：配置所有的安全机制，关掉所有不使用的服务，设置角色权限帐号，使用日志和警报。

分析：用户使用缓冲区溢出来破坏web应用程序的栈，通过发送特别编写的代码到web程序中，攻击者可以让web应用程序来执行任意代码。

1. 【问题：】注入式漏洞。

【例：】一个验证用户登陆的页面，

如果使用的SQL语句为：

Select \*  from  table A where  username＝”+ username+” and pass word …..

SQL 输入  `or 1＝1 ――  就可以不输入任何password进行攻击

1. 【问题：】不恰当的异常处理

分析：

程序在抛出异常的时候给出了比较详细的内部错误信息，暴露了不应该显示的执行细节，网站存在潜在漏洞

1. 【问题：】不安全的存储

分析：

帐号列表：系统不应该允许用户浏览到网站所有的帐号，如果必须要一个用户列表，推荐使用某种形式的假名（屏幕名）来指向实际的帐号。

浏览器缓存：

认证和会话数据不应该作为GET的一部分来发送，应该使用POST

1. 【问题：】跨站脚本（XSS）

分析：攻击者使用跨站脚本来发送恶意代码给没有发觉的用户，窃取他机器上的任意资料

【测试方法：】

* HTML标签：<…>…</…>
* 转义字符：&(&)；<(<)；>(>)； (空格) ；
* 脚本语言：

      <script language=‘javascript’>

       …Alert(‘’)

       </script>

* 特殊字符：‘  ’ <  >  /
* 最小和最大的长度
* 是否允许空输入