【学习笔记】通过样本分析之一CVE-2010-2883

diffway / 2017-06-20 06:32:00 / 浏览数 5825 新手 入门资料 顶(0) 踩(0)

---

1 分析环境

操作系统：window xp sp3
软件 ：Adobe Reader 9.3.4

2 基本信息

漏洞类型：栈溢出

影响范围：基于Window和Mac OS X的Adobe Reader和Acrobat 9.4之前的9.x版本，8.2.5之前的8.x版本

3 样本分析

我分析的是漏洞战争中的样本

样本MD5 ： 3f41dc8e22deca8302db1207e5cdc11c

样本名称 ： 名企面试自助手册

在我们拿到样本的时候，一般是不知道漏洞CVE编号，我这个主要展示的是通过一个攻击样本去定位漏洞，我们首先进行行为分析，可以看到在临时文件中释放了一个文件，

这个时候我们打开Windbg，并且在WinExec下了断点，并运行，可以看到我们断到的地方正是执行svrhost的地方。

我们开始执行，并且执行到这个函数返回，可以看到返回的地址在下面从039a0001开始的内存中，

我们看下这个地址，可以发现这个地址，是从桌面的文件读取进来的。

这个文件也是从样本中释放出来的，其实这是段shellcode,被读入并执行起来。

既然这样我们继续下断点在 kernel32!CreateFileW 下断点,你会发现整个会断下来很多，这个时候，我们需要条件断点，我们将a.txt变为要下断点的文件名

断到断点后回到后就会发现ROP代码

打印下esp ,可以看到黑客作者构造的ROP链

这个时候我们在堆喷的时候下断点，这个时候我们下个内存写断点，在0c0c0c50地址被写入时候下断到后，我们看下内存布局从0c0c0c0c 开始时ROP链

我们近一步分析可以发现，里面在执行js脚本进行堆布局

这个时候我们利用工具将里面的js代码提取出来，我们使用的是pdfStreamDumper这款工具

可以清楚的看到里面的js代码，确实堆喷的代码，并且看看黑客精心构造的代码

我们拷贝出来，并进行了简单的替换，这样可以好看一些

我们可以看到里面的ROP链式硬编码到里面的，用windbg的插件mona看看模块中ASLR开启的情况，可以看到很多未开启ASLR，其中黑客选用的是icucnv36.dll这个模块。

我们知道这个肯定会跳转到0c0c0c0c,我们在这个下内存读断点

我们r一下，会发现这个时候的esp不是栈地址，而是变成了之前堆喷的地址了，这也是开始绕过DEP，毕竟这块地址在win7上就没有执行权限了。我们可以看看其他的寄存器

可以看看eax和ecx里面的地址都是栈地址。经过查看，可以看到ecx里面的地址更像是之前栈的地址。

我们找个返回地址看一下，可以看到是CoolType,我们在这个地方下断点

下完断点后，执行后，发现就是这个模块出的问题，我们将这个模块放到IDA中，重点看一下。

我们重点跟一下，可以发现在0808b308这个位置出现的问题，这个eax是个栈地址，可以看到这个栈地址被覆盖

然后将0c0c0c0c 这个地址pop给esp,然后开始

我们来分析下漏洞的成因，我们可以发现是在解析SING这个字段出现的问题

我们将TTF文件拷贝出来，看一下

我们在其中找到了ROP链,跳到0c0c0c0c

我们这个时候开始使用Immunity Debugger定位漏洞

我们首先看到开辟了104H大小的栈空间

然后再来看看strcat函数连接的TTF文件的字符串长度明显超过104H的大小，造成了栈溢出。

这个时候我们需要了解下TTF文件的对于Sing的定义和布局，好知道到底是什么字段导致了栈溢出，主要是下面这个字段的解释

可以查看官方文档对TableEntry的解释，可以知道SING表相对于文件的偏移为0x0000011c

我们通过代码可以知道是将偏移10h字节的拷贝过来通过查看官方文档可以知道是这个字段是uniqueName

4总结

通过我们的分析我们可以知道，主要是对SING表的uniqueName字段在拼接的时候对大小没有进行严格的控制，通过官方补丁也可以知道在修复中对大小进行了检查，防止