DOI:10.16515/j.cnki.32-1722/n.2018.04.001

浅析软件安全中的注入技术

程 远,李文珏,柳亚男,邱 硕

(金陵科技学院软件工程学院,江苏 南京 211169)

摘 要:结合目前软件安全现状,详细介绍了DLL注入技术和代码注入技术的基本原理和实现方法;比较了不同注入技术之间的差别和优缺点;归纳了针对DLL注入技术可采用的防御措施及业界的研究成果;指出了未来Android系统安全的可研究方向。

关键词:软件漏洞;恶意代码;代码注入;DLL注入;消息钩取

中图分类号:TP311 **文献标识码**:A **文章编号**:1672-755X(2018)04-0001-04

Analysis of Injection Technology in Software Security

CHENG Yuan, LI Wen-jue, LIU Ya-nan, QIU Shuo

(Jinling Institute of Technology, Nanjing 211169, China)

Abstract: Considering the current situation of software security, the basic principles and implementation methods of DLL injection technology and code injection technology are introduced in details. The differences, advantages and disadvantages among different injection technologies are compared. The defensive measures and industry research results of DLL injection are summarized. Finally, future research field of Android system security is pointed out.

Key words: software vulnerabilities; malicious code; code injection; DLL injection; message hooking

近年来,随着云计算、大数据技术的快速发展,数字化转型正对国民经济各个领域的发展和变革产生着深远的影响。目前,信息技术已经全面渗透到各行各业中,应用软件与产业技术紧密结合。以大数据、云计算和人工智能为代表的新型信息技术革命将成为提高生产力、优化企业管理的重要手段,也是工业5.0时代的显著特征。

软件作为构建计算机网络和移动互联网的基础,与操作系统以及通信网络构成了整个信息化系统。计算机软件中缺陷和漏洞会使得整个信息系统存在巨大的隐患。黑客采用信息技术手段进行信息欺诈和攻击,窃取、篡改和盗用用户数据,日益增多的安全漏洞不仅影响了普通用户和企业的正常生产生活,也威胁到了全社会的信息安全[1]。软件在编程过程中难免会存在不完善的漏洞,尽管软件企业都有严格的质量控制和管理过程,但很难完全杜绝漏洞的产生。卡耐基梅隆大学的Cy Lab实验室进行的一项研究显示,具有代表性的商业性非开源程序每1 000行语句平均带有20~30个漏洞。Windows 98有1 800万行代码,Windows XP有3 500万行代码,Vista系统的代码则达到了5 000万行,比Windows XP多出了40%[2]。像Windows这样一个被海量用户使用的操作系统,存在任何漏洞都是相当危险的。

收稿日期:2018-12-08

基金项目:金陵科技学院高层次人才启动基金(jit-b-201726,jit-b-201639);江苏省高等学校自然科学研究面上项目(17KJD520003);网络安全专项项目(2017YFB0802800)

作者简介:程远(1981—),男,江苏南京人,高级工程师,博士,主要从事网络空间安全技术、NB-IoT技术研究及应用、基于SDN的数据中心网络与广域网技术等研究。

1 注入技术

在软件安全领域，注入技术是黑客广泛应用的主流渗透技术之一。恶意代码一旦被成功注入到目标进程，就可以截获、修改用户数据，获得操作系统权限并执行任意操作。因此，注入技术研究和发展始终受到广大研究学者和软件工程师的关注。注入技术的基本原理是利用 Windows 系统提供的库以及 API 函数，将恶意代码就是俗称的 shellcode 植入到当前运行的进程中并执行。根据实现的方式不同，注入技术大体上可以分为代码注入和 DLL(Dynamic Link Library)注入两大类。

DLL 注入是向一个正在运行的进程插入/注入代码的过程，注入的代码通常以动态链接库的形式存在。利用 DLL 注入，能够实现功能更加强大的 API 钩取。按照工作原理的不同，DLL 注入大致可以分为消息钩取、创建远程线程以及修改注册表注入等。DLL 注入的核心思想是要让被注入的目标进程运行 API 函数 Load Library。一方面，DLL 注入技术可以被用于合法正当的用途，如改善功能、修复漏洞。比如，反病毒软件使用注入技术来将代码嵌入系统中“所有”正在运行的进程，从而能够监控保护程序。另一方面，DLL 注入技术也能够实现恶意代码植入。例如，将恶意线程隐藏到正常运行的用户进程或者系统进程中，打开后门，尝试从外部连接或者键盘偷录，盗取用户个人信息。

接下来，给出 DLL 注入技术的 4 种常用实现方法：

1.1 利用全局消息钩子实现 DLL 注入

Win32 应用程序一般都要收发消息，驱动 Windows 操作系统。操作系统消息队列通过将不同应用程序产生的消息分发到相应的进程消息队列中，完成消息的传递。在消息从系统队列发往进程消息队列的过程中，利用消息钩取函数，能够在应用程序收到消息之前，对消息进行监听、拦截甚至篡改。消息钩取的基本原理如图 1 所示。

图 2 给出了安装钩子钩取向指定进程传递的所有键盘输入消息的流程。首先，注入程序先加载 KeyHook. dll 文件，利用 KeyHook. dll 中的导出函数，安装钩取键盘消息的钩子；接下来，当操作系统向某个进程传递键盘输入消息时，钩子程序先于目标进程监听到该消息，并向相应的目标进程加载 KeyHook. dll 的入口点，同时调用钩子函数对该键盘输入消息进程处理(拦截)。早期的 DLL 注入大部分是通过这个原理实现的。

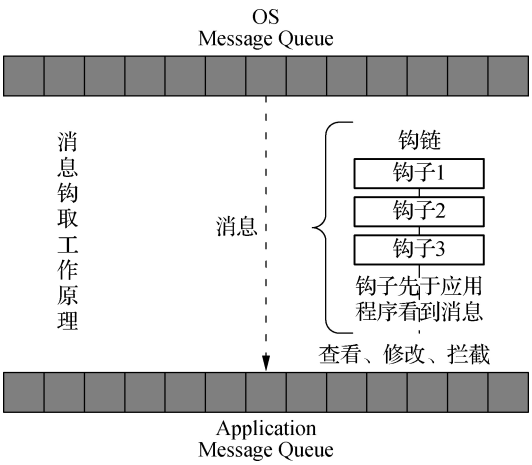

图 1 消息钩取的基本原理

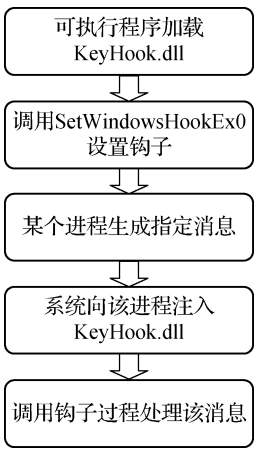

图2 键盘消息钩取流程

Windows 系统提供了 14 种不同类型的钩子来处理不同类型的消息。例如，WH_MOUSE HOOK 用来监视鼠标消息。Windows 为这些钩子维护着各自的钩链。钩链是一串由应用程序定义的回调函数队列，当产生某种类型消息时，操作系统向此类型钩链的第一个函数(钩链的顶部)发送该消息，第一个函数处理完该消息后，由该函数向链表中的下一个函数传递消息，依次向下。钩子过程可以监视消息、修改消息、阻止消息传递到下一个钩子过程或者目的进程窗口。最近安装的钩子放在链的前端，即后加入钩链的

钩子先获得控制权。

为了拦截和处理特定的消息，使用 SetWindowsHookEx 函数安装钩子函数，即回调函数。当所安装钩子的消息事件发生时，就会调用钩子函数。例如，钩子是 WH_MOUSE 类型，只要有鼠标操作的事件发生，相应的钩子函数就会被调用。钩子函数的语法如下：

```
Public function MyHook(ByVal nCode, ByVal wParam, ByVal lParam)
{处理代码}
End function
```

nCode 指定钩子传入的消息类型，参数值依赖于钩子类型；wParam 参数表示键盘按键的虚拟键值；iParam 的各个位可以作为状态标志，用于指定扩展键、扫描码、按键次数等信息；wParam、iParam 取值随 iCode 不同而不同，代表了某类钩子的特定动作。

函数 SetWindowsHookEx 用于将一个应用程序定义的回调函数安装到钩链中，并利用回调函数对系统某类事件进行监控。若回调函数执行成功，则返回值就是该钩子处理过程的句柄。API 函数 SetWindowsHookEx 的语法及参数说明如下：

```
HHOOK SetWindowsHookEx(
int idHook, //被安装的钩子所钩取的事件类型
HOOKPROC lpfn, //指向相应的钩子处理过程，即回调函数地址
HINSTANCE  hMod, //指示了一个 DLL 句柄
DWORD  dwThreadId //此参数为 0，安装的钩子为全局钩子
);
```

1.2　修改注册表实现 DLL 注入

修改注册表项 HKEY_LOCAL_MAHINE\Software\Microsoft\WindowsNT\CurrentVersion\Windows\AppInit_DLLs 也能够实现 DLL 注入。但这种方法只适用于 NT 系统，且注入的目标进程必须是调用 user32.dll 的程序，局限性较大。通过修改注册表后，所有调用 user32.dll 的程序在开机后会自动加载注入的 DLL，且所加载 DLL 不能被卸载。

1.3　创建远程线程实现 DLL 注入

创建远程线程方式的基本原理是先调用 API 函数 LoadLibrary 加载指定 DLL，再使用 CreateRemoteThread 函数在目标进程中运行插入的线程函数。创建远程线程完成 DLL 注入的六个主要步骤如图 3 所示。

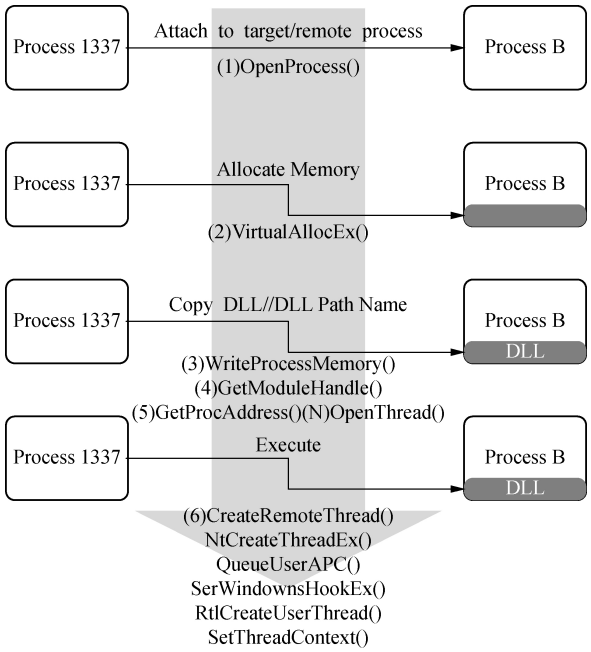

图 3　创建远程线程进行 DLL 注入的步骤

第一步:打开要写入的进程,赋予权限。用 API 函数 OpenProcess 实现;

第二步:在目标进程的内存中分配一块内存空间。使用 API 函数 VirtualAllocEx 实现;

第三步:将 dll 写入内存区域。使用 API 函数 WriteProcessMemory 实现;

第四步:创建远程线程。使用 API 函数 CreateRemoteThread 实现;

第五步:等待加载完成。使用 API 函数 WaitForSingleObject 实现;

第六步:释放目标进程中申请的空间。使用 API 函数 VirtualFreeEx 实现。

上述几种 DLL 注入技术使用了各种 API 函数进行内存写入并操纵其他进程。微软 Windows API 函数都包含于 DLL 文件之中。

1.4 代码注入

代码注入与 DLL 注入的最显著区别就是无需额外的 DLL 文件。代码注入需要同时注入代码和数据。代码注入与 DLL 注入都使用了 CreateRemoteThread 这个重要的 API 函数,但在 DLL 注入中,其线程函数通常为 LoadLibraryA,传递给线程函数的参数通常为被映射到内存空间的 DLL 文件地址,插入的线程函数是通过 DLLMain 函数调用的。而代码注入直接将线程函数通过 CreateRemoteThread 插入目标进程,不需要使用 LoadLibraryA 函数。

代码注入技术较 DLL 注入有以下几点优势:1)代码注入技术占用的内存小;2)代码注入技术难以查找痕迹;3)无需另外的 DLL 文件。因此,代码注入技术通常适用于代码量小且功能简单的情况,DLL 注入适用于代码量大且功能复杂的情况。

2 DLL 注入的防御措施

文献[3]分析了 Windows 系统下代码注入攻击技术,提出了基于 Detours 技术挂钩装载动态链接库函数的防御方案。文献[4]提出一套通过识别 2 类编码错误发现 Web 应用中代码注入漏洞的测试方法。实验结果表明,该方法可减少测试工作量,能全面有效地测试 Web 应用中的代码注入漏洞和潜在的风险点。文献[5]中提出了一种 DLL 注入的检测方法:当 PE 文件扫描时,枚举当前 PE 文件的 IDT 表,并与合法模块列表进行对比,将没有在合法模块列表中的模块初步确定为可疑模块,并通过进一步分析来验证可疑模块是否为非法模块。文献[6]针对远程线程注入实现木马隐藏的技术,提出了一种进程被远程注入动态链接库的检测方法及相应动态链接库的卸载方法。文献[7]研究了针对远程线程注入的线程"白名单"防护技术和保证应用软件可靠运行的线程安全监控技术,能够防止木马程序利用远程线程注入对应用软件进行破坏和窃密,建立"主动防御"的管控系统。

3 结 语

除了 Windows 系统,Android 操作系统面临的安全问题也日益严峻。移动恶意代码注入主要是由于 Android 应用安全防护机制的不足以及 Html5 移动 APP 开发技术存在安全漏洞[7-8]。因此,进一步完善 Android 应用的安全保障机制,研究针对 Android 平台的注入攻击防御措施,努力消除安全漏洞和隐患,对保护移动终端用户的合法权益有着重大意义。

参考文献:

[1] 彭赓. Windows 平台下软件安全漏洞研究[D]. 合肥:电子科技大学,2010
[2] 张晛譞. 基于 Windows 平台的软件安全漏洞发掘技术研究[D]. 合肥:电子科技大学,2010
[3] 徐小玲,赵振熹. 代码注入攻击及防御技术研究[J]. 浙江教育学院学报, 2009(4):102-106
[4] 朱辉,沈明星,李善平. Web 应用中代码注入漏洞的测试方法[J]. 计算机工程,2010,36(10):173-178
[5] 陈庄,王津梁. 手工 DLL 注入的检测方法研究与实现[J]. 信息安全研究,2017(3):246-253
[6] 王佩红,赵尔敦,张瑜. 远程线程注入 DLL 的检测与卸载方法研究[J]. 计算机与数字工程,2010,38(3):106-108
[7] 乔娜. Windows 环境下线程安全防护与管控技术研究[J]. 硅谷,2014(11):49-50
[8] 暴文莹. 基于 Android 平台的恶意代码注入攻击方法研究与实现[D]. 北京:北京邮电大学,2017

(责任编辑:谭彩霞)