南開大學

恶意代码分析与防治技术课程实验报告

实验一：R77 rootkit

学　　院　网络空间安全学院

专　　业　信息安全

学　　号　2213041

姓　　名　李雅帆

班　　级　信安班

一、实验目的

1. 运行 R77 程序，实现对指定的进程、文件、注册表、网络连接的隐藏，了解R77的工作原理。

3. 了解 R77 所利用的 Windows 的 Detours 机制。

二、实验原理

R77 是一种 Ring 3 级别的恶意软件，通过操作系统的用户模式运行，而不需要通过更高权限的内核模式。其核心目的是通过操控用户模式接口与操作系统交互，隐藏其恶意活动并避免被检测。以下是 R77 rootkit 的关键特性和技术细节：

1. 隐藏机制：R77 rootkit 利用特定的前缀（如 $77）来标记需要隐藏的对象。 为了实现这一隐藏效果，R77 会通过 修改系统 API 的行为 来拦截查询操作。当系统或安全软件试图列出文件、进程或网络连接时，R77 会对带有 $77 前缀的对象进行 过滤，使其在正常查询中不可见。这种机制大大增强了其隐蔽性，使得恶意活动不容易被发现。

2. 文件无关性（Filelessness）：R77 是一个 无文件（fileless）的 rootkit，这意味着它不依赖于硬盘上的文件来持久化或执行。相反，它通过以下方式在内存中运行和自我复制：

（1）内存驻留：R77 将自己的代码加载到系统内存中执行，而不是依赖于磁盘文件。这样可以避免被基于文件的检测工具（如传统的反病毒软件）发现。

（2）内存自复制:R77 利用内存中的各种技术来在系统内存中自行复制和传播，从而实现持久化。这使得它能够绕过文件系统监控，增加清除难度。

3. 规避技术：为了避免被现代的防病毒软件和端点检测与响应（EDR）系统检测，R77 使用了多个高级规避技术，包括但不限于：

（1）AMSI 绕过（AMSI Bypass）：R77 能够绕过 Windows 防病毒 API（AMSI），使得恶意脚本和代码不会被 Windows Defender 或其他基于 AMSI 的防护工具检测。

（2）DLL 取消挂钩（DLL Unhooking）：R77 通过取消或替换常见的 防病毒检测工具中的钩子函数，防止它们检测和分析恶意代码。这使得系统调用和 API 请求在被检查时，无法捕获到 R77 的恶意行为。

三、实验过程

1. R77 基本介绍

77 是一个 Ring 3 Rootkit，它在操作系统的用户空间（Ring 3）运行，而不是内核空间（Ring 0）。作为一个恶意工具，它能隐藏并修改系统的许多重要部分，常用于规避检测和增强隐蔽性。具体来说，R77 可以隐藏以下内容：

（1）文件和目录：R77 能够隐藏特定的文件和目录，防止它们被检测或删除。

（2）进程和 CPU 使用：它能隐藏运行的进程以及占用的 CPU 资源，使得恶意进程不易被发现。

（3）注册表键和值：R77 可以修改或隐藏注册表中的键和值，阻止安全软件或管理员发现恶意的修改。

（4）服务：R77 可以隐藏或篡改服务的状态，防止其被关闭或检测。

（5）TCP 和 UDP 连接：R77 能够隐藏恶意的网络连接，使得网络流量的异常行为不容易被监控和识别。

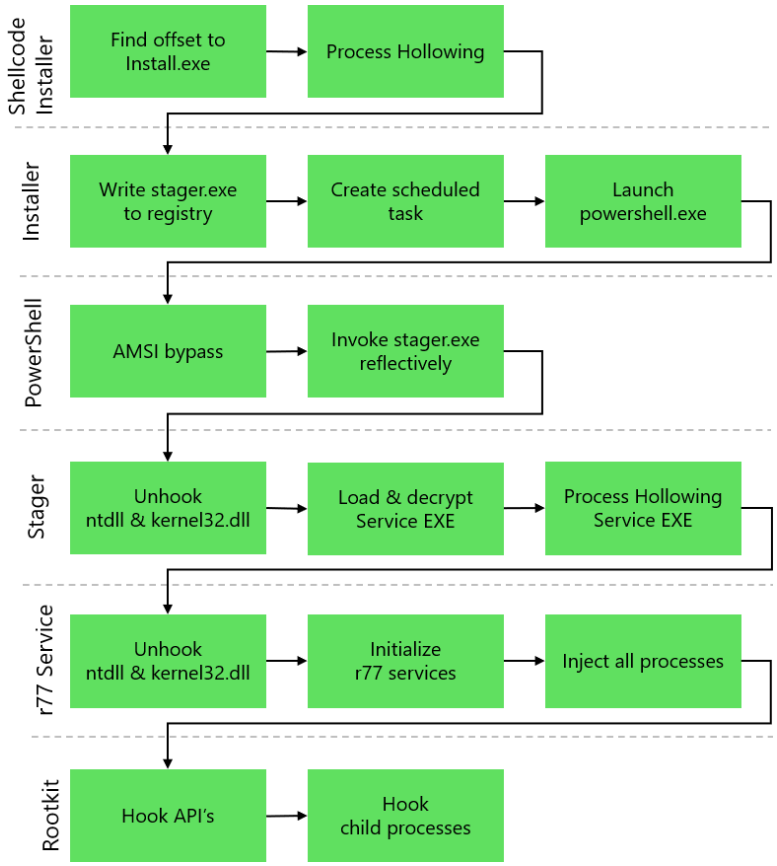
（6）联接、命名管道、计划任务：R77 还可以隐藏系统中的连接、管道或计划任务，进一步增强它的隐蔽性。

2.R77 配置与安装

配置位于 HKEY_LOCAL_MACHINE\SOFTWARE\$77config，任何进程都可以无需提升权限就对其进行写操作。此键的 DACL 被设置为允许任何用户完全访问。此外，$77config 键也被根工具包隐藏。

部署 r77 只需要：Install.exe。执行后，r77 将在系统中持久化并注入所有运行中的进程。Uninstall.exe 可以完全从系统中移除 r77。

3.R77 执行流程



（1）安装阶段

①Shellcode 和 Installer

Find offset to Install.exe：首先找到 Install.exe 的偏移量。这一步可能是为了定位系统中的安装文件，以便后续操作。

Write stagerex.exe to registry：将 stagerex.exe 写入注册表。这一步是为了在系统启动时能够自动运行 stagerex.exe。

Create scheduled task：创建一个计划任务，用于定时执行某些操作。

Launch powershell.exe：启动 PowerShell，这通常是为了利用 PowerShell 的强大功能来执行后续操作。

Process Hollowing：进行进程挖空操作，这是一种常见的恶意软件隐藏技术，通过替换合法进程的内存空间来隐藏恶意代码。

（2）PowerShell 阶段

①AMSI bypass：在 PowerShell 中绕过 AMSI（反恶意软件扫描接口），以避免被安全软件检测到。

②Invoke stagerex.exe reflectively：以反射方式调用 stagerex.exe，这是一种在内存中加载和执行可执行文件的技术，能够绕过一些基于文件系统的检测。

（3）Stager 阶段

①Unhook ntdll & kernel32.dll：解除对 ntdll.dll 和 kernel32.dll 的钩子。钩子通常是安全软件用来监控系统调用的，解除钩子可以避免被检测到。

②Load & decrypt Service EXE：加载并解密服务可执行文件，这一步是为了准备后续的服务初始化。

③Process Hollowing EXE：再次进行进程挖空操作，这次是针对服务可执行文件，以隐藏其真实活动。

（4）R77 Service 阶段

①Unhook ntdll & kernel32.dll：再次解除对 ntdll.dll 和 kernel32.dll 的钩子，确保后续操作不被监控。

②Initialize r77 services：初始化 R77 服务，这是根工具包在系统中建立自身的关键步骤。

③Inject all processes：将恶意代码注入到所有进程中，这一步可以确保根工具包能够监控和控制整个系统。

（5）R77 Kit 阶段

①Hook API's：钩子 API（应用程序编程接口），通过钩子 API 可以拦截和篡改系统调用，从而实现对系统的控制。

②Hook child processes：钩子子进程，确保所有新创建的子进程也受到根工具包的控制。

4.R77 隐藏测试

（1）进程隐藏

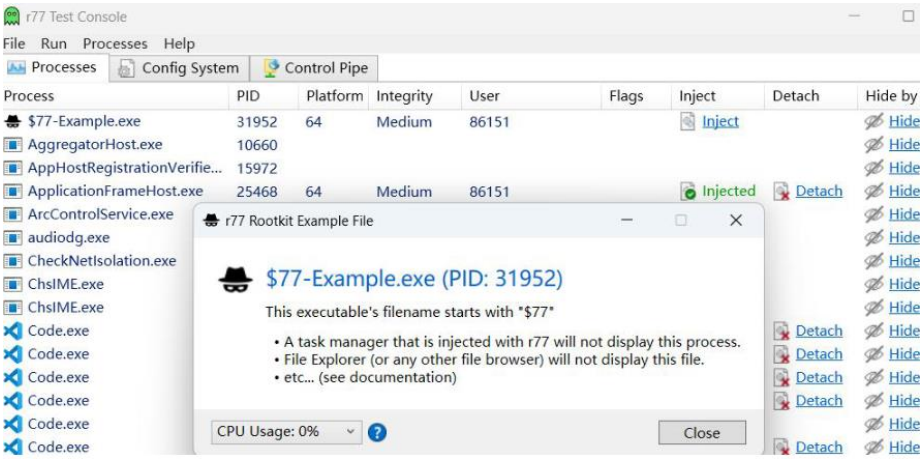
R77 使用 进程注入 和 API 挂钩 技术来隐藏进程。具体来说，R77 可以通过以下方式隐藏进程：

钩住进程创建 API：R77 会钩住操作系统中的关键进程创建 API（如 CreateProcess 或 NtCreateProcess），从而拦截新进程的创建。当一个新进程被创

建时，R77 会注入自己的恶意代码并隐藏进程，使得被注入的恶意进程对用户和系统不可见。

修改进程列表：R77 还可能直接修改系统进程列表，使得恶意进程不显示在如任务管理器、Process Explorer 或其他进程监控工具中。

空洞化技术（Process Hollowing）：R77 可能使用 进程空洞化（Process Hollowing）技术，将自身的恶意代码注入到其他合法进程的内存空间中。这些合法进程的名称和其他特征保持不变，而恶意代码在这些进程的内存中运行，避免被检测。



（2）文件隐藏

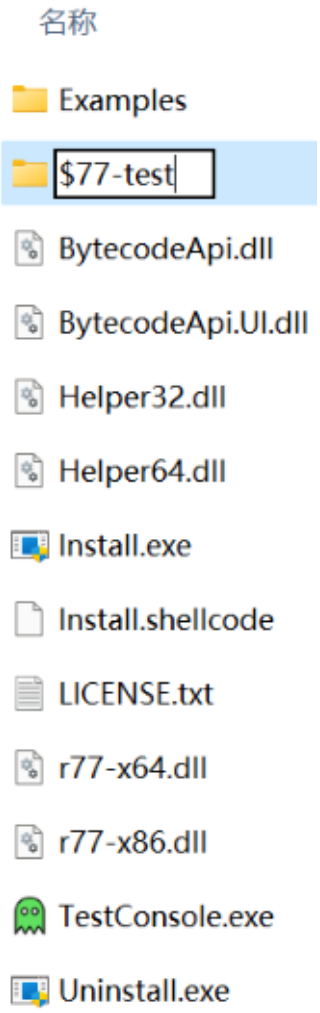
R77 使用类似的 钩住文件系统 API 技术来隐藏文件。具体操作如下：

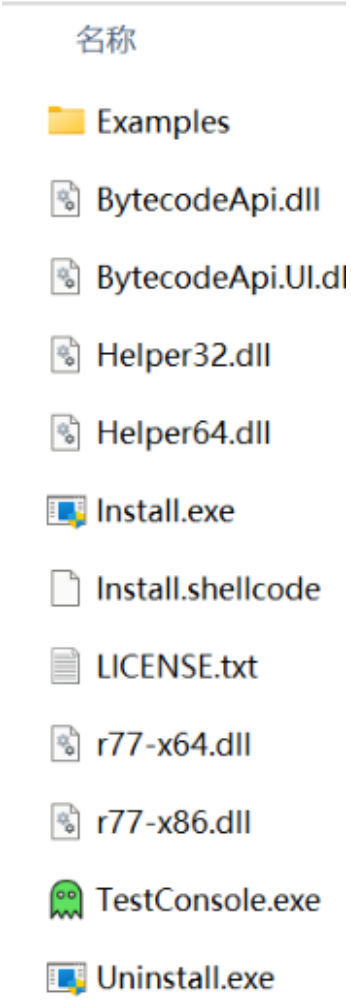
钩住文件操作 API：R77 会钩住文件相关的操作函数，如 CreateFile、ReadFile、WriteFile 等。当有文件操作请求时，R77 会拦截这些请求，并确保它自己的恶意文件不被显示或访问。

文件隐藏：通过修改文件的属性（如隐藏文件的 FILE_ATTRIBUTE_HIDDEN 属性），R77 可以使其恶意文件在普通文件浏览中不可见。此外，R77 还可以直

接修改操作系统中文件的显示方式，使得恶意文件不会出现在 Windows 资源管理器 或其他文件管理工具中。

隐藏文件路径：R77 可能将恶意文件放置在系统的隐藏路径下（如系统目录），或者将文件名和路径伪装成合法文件，从而避开安全软件的监测。





（3）注册表隐藏

R77 根工具包还会通过注册表钩子来隐藏其相关的注册表项。根工具包的配置文件通常存储在注册表中，R77 会通过以下手段隐藏关键的注册表项：

修改注册表访问权限：R77 会修改其注册表项的 访问控制列表（DACL），确保其配置文件（如 HKEY_LOCAL_MACHINE\SOFTWARE\$77config）对用户不可见，或者仅允许恶意进程访问。通过这种方式，管理员或安全软件无法直接查看和修改 R77 的配置。

注册表隐藏：R77 会隐藏其注册表项，避免被用户、管理员或安全工具发现。通过修改注册表的可见性，R77 可以确保其持久化机制不被轻易检测到。

钩住注册表操作 API：类似于文件隐藏，R77 也会钩住注册表相关的 API（如 RegQueryValue 和 RegOpenKey），当用户或程序访问注册表时，根工具包可以确保其隐藏的项不会出现在正常的查询结果中。

（4）网络连接隐藏

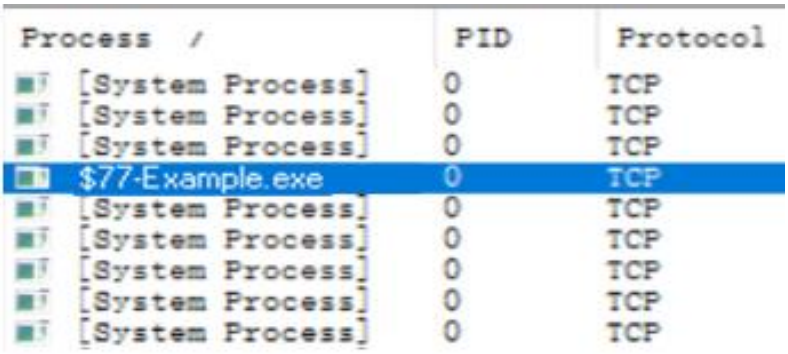
R77 根工具包通过以下方式隐藏特定端口的 TCP 和 UDP 网络连接：

钩住和修改网络 API，拦截与网络连接相关的调用。

隐藏连接信息，不允许特定端口上的网络连接显示在网络监控工具中。

动态配置，允许用户灵活指定需要隐藏的端口，以便于绕过端口扫描和防火墙检测。

防止被检测，通过隐藏通信端口来绕过防火墙、IDS/IPS 和其他网络监控系统，确保恶意活动的持续性和隐蔽性。



5.R77 所利用的 Windows 的 Detours 机制。

（1）钩住关键 API：R77 会选择性地钩住操作系统中与进程管理、文件系统和网络通信相关的 API。例如，CreateProcess、NtQuerySystemInformation 和 WriteFile 等函数被钩住，以便它能够隐藏进程、隐藏文件、或者更改系统行为。

（2）隐藏恶意进程：通过钩住 CreateProcess，R77 可以使得新启动的进程被注入恶意代码后依然保持隐蔽。恶意代码通过 Detours 机制跳过常规的进程监视，避免被安全软件检测。

（3）动态注入代码：R77 会使用 Detours 动态注入恶意代码到目标进程中。通过修改目标进程的内存空间（如栈、堆或者代码段），R77 可以确保它的恶意代码被执行。

（4）绕过安全防护：R77 还可以利用 Detours 机制绕过一些防病毒软件、端点检测响应（EDR）和其他安全检测技术。例如，它可能会钩住防病毒软件的扫描函数，使其无法检测到恶意活动。

四、实验结论及心得体会

通过本次 R77 rootkit 实验，我有很多新的收获。一方面深入理解了恶意软件的隐藏机制，如 R77 通过多种手段实现进程、文件、注册表和网络连接的隐藏，其无文件化持久化技术独特。二是认识到它对 Windows Detours 机制的巧妙利用，包括钩住关键 API 来实现恶意操作和绕过安全防护。