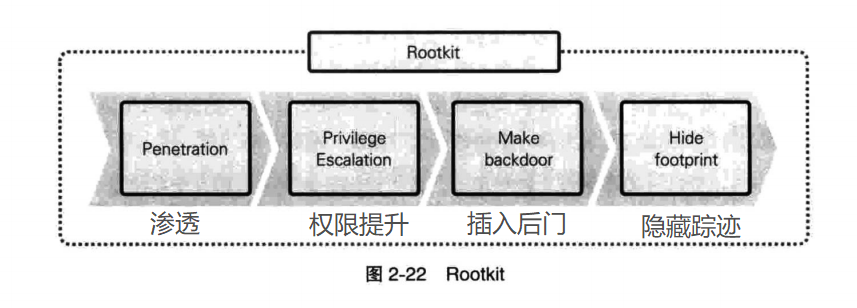
## 1. Rootkit

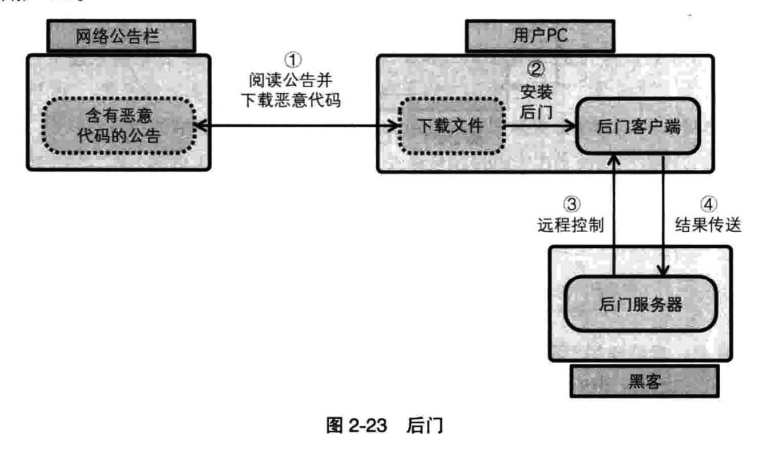
Rootkit是一种特殊的黑客攻击程序，其功能是获取目标主机的root权限，或者安装可以控制系统的后门。它具有很强的隐蔽性，不易被杀毒软件发现。Rootkit有用户模式、内核模式、引导模式三种类型。用户模式在应用程序级别工作，比较容易测出，对系统危害较低。Rootkit工作在内核模式下会向内核添加其他代码，或者直接用新代码替换原有代码。开发虽然有难度，但能够对系统造成致命损害。引导模式对MBR（Master Boot Record，主引导记录）、VBR（VolumeBoot Record，卷引导记录）、引导扇区产生影响，能够对整个文件系统加密，或者使系统陷入无法引导的困境。



## 2.后门

后门是指可以远程控制用户PC的程序。黑客通过网络公告栏、电子邮件、木马等传播含有后门的恶意代码。用户无意中将恶意代码下载到PC后，后门客户端就会被安装到用户系统。黑客运行后门服务器程序，等待客户端连接。后门客户端安装好后连接到服务器，这样黑客即可远程控制用户PC。

这里的后门一般是指反弹shell。

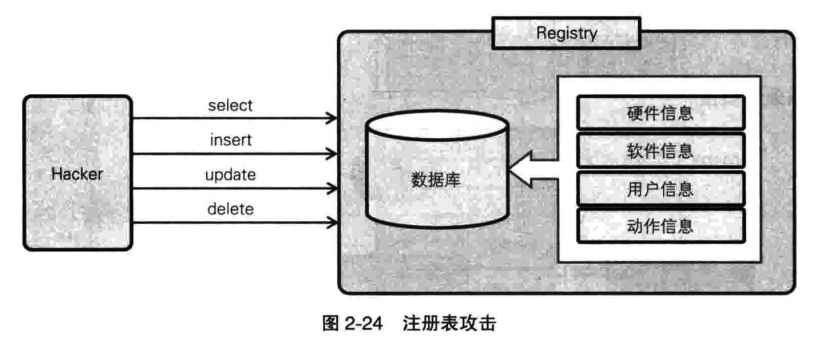


## 3.注册表攻击

### python访问注册表-获取用户账户列表

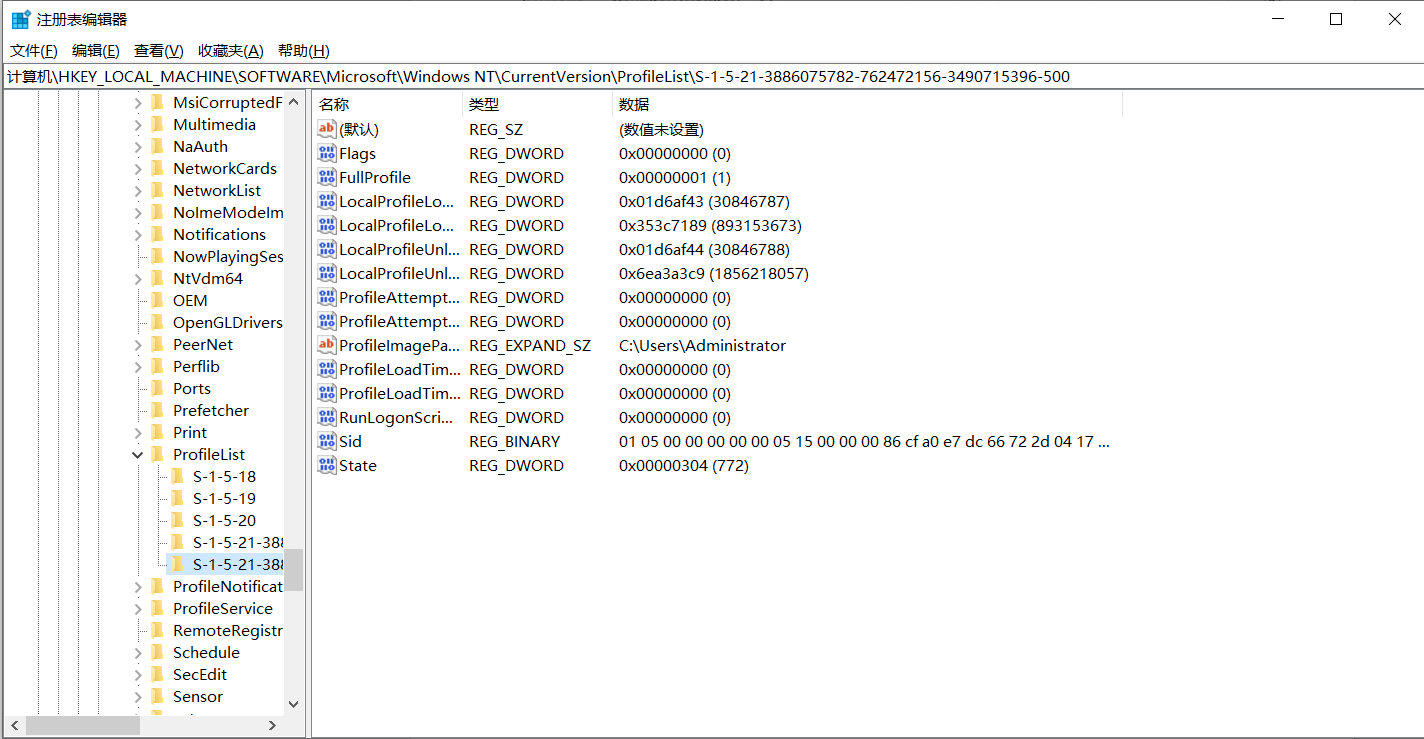
#### 什么是注册表

Windows中使用的注册表是一种数据库，它采用“键，值”的形式保存硬件信息、软件信息、用户信息，以及行为控制所需的各种信息。为了控制注册表，Windows通过接口支持与CRUD（Create Read Update Delete)相关的所有功能。入侵系统的黑客可以通过接口操作注册表，尝试初始化用户密码、修改防火墙设置、DLL注入等多种攻击。注册表也保存着用户使用网络的信息，通过这些信息，黑客也可以获知用户的生活方式。



注册表是保存硬件、软件、用户、操作系统与程序的一般信息和设置信息的数据库。以前使 用ini文件保存相关信息，但由于每个程序都对应一个ini文件，所以很难开展有效管理与维护。

为解决这一问题，系统工程师引入了具有数据库形态的注册表。Windows操作系统与程序会自动向注册表输入信息，并进行更新。当然，用户也可以使用regedit等工具修改注册表。为防止程序启动错误，用户只能修改注册表的部分内容。注册表错误会给系统造成严重影响，请尽量不要随意修改。



在Windows命令行窗口输入regedit命令，即可打开“注册表编辑器”，如上图所示。注册表大致由4部分组成，最左侧是键区域，最上层的键称为根键，其下层键称为子键。选择某个键，就会看到对应的键值，由数据类型与数据成对组成。注册表以“配置单元”（Hive)为单位进行逻辑管理，以文件形式进行备份。注册表以根键为中心划分“配置单元”，最终保存到以“配置单元”为单位管理的文件。根键及其作用如图表所示：

|  |  |
| --- | --- |
| HKEY—CLASSES—ROOT | 该根键根据windows操作系统中所安装的应用程序的扩展名，来指定文件类型。 |
| HKEY—CURRENT—USER | 该根键包含了本地工作站中存放的当前登录的用户信息，包括用户登录名和存放的口令（注意，这个密码在输入的时候是隐藏的。）用户登录windos操作系统的时候，其信息从HKEY USERS 中相应的项复制到HKEY—CURRENT—USER中。 |
| HKEY—LOCAIMACHINE | 该根键存放本地计算机的硬件信息，该根键下的子关键字包含在SYSTEM.DAT中用来提供HKEY—LOCAL—MACHINE所需要的信息，或者在远程计算机中可访问的一组键盘中。该根键下的许多与System.ini中的设置项类似。 |
| HKEY-USERS | 该根键保存了存放在本地计算机口令列表中的用户标识和密码列表。每个用户的预配置信息都存储在HKEY-USERS根键中。HKEY—USERS是远程计算机中访问的根键之一。 |
| HKEY—CURRENT—CONFIG | 该根键存放着当前用户桌面配置的数据、最后使用的文档列表和其他有关的当前用户的windows版本的安装信息等。 |

#### 代码实现

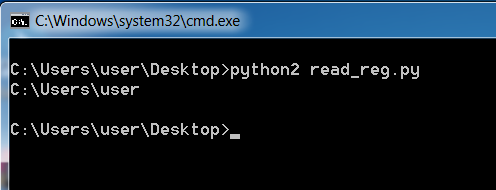
Python提供\_winreg模块，用于访问注册表信息。\_winreg模块像一座桥梁，借助它，可以在Python中使用Windows提供的注册表API，且方法非常简单。先将根键设置给变量，再调用ConnectRegistry（）函数，连接注册表句柄。OpenKey（）函数以字符串形式设置下层注册表名称，返回控制句柄。然后，通过EnumValue（）函数获取注册表值。处理完成后，使用CloseKey（）函数关闭句柄。

使用regedit程序打开注册表，然后在HKEY\_LOCAL MACHINE部分查看SOFTWARE\Microsoft\Windows NT\CurrentVersion\ProfileList项目，子目录中存在用户账户SID项，从中可以看到每个项目的ProfilelmagePath变量。系统将根据用户账户名分配的目录列表保存到相应变量。

下面使用Python编写程序，用于自动访问用户账户列表。先设置前面提及的注册表子目录，添加一些程序代码即可提取我们感兴趣的系统中使用的用户账户列表。

*#!/usr/bin/python3  
# -\*- encoding: utf-8 -\*-  
# @Time : 2020/12/24 11:18   
# @Author : ordar  
# @File : read\_reg.py   
# @Project : dll\_injection2.py  
# @Python : 2.7.1  
import* sys  
*from* \_winreg *import* \*  
  
*# 1. 设置子健目录：用户账号信息的子健目录*varSubKey = r"SOFTWARE\Microsoft\Windows NT\CurrentVersion\ProfileList"  
*# 2. 获取根注册表句柄对象：使用\_winreg模块提供的HKEY\_LOCAL\_MACHINE，设置要搜索的根键*varReg = ConnectRegistry(*None*, HKEY\_LOCAL\_MACHINE)  
*# 3. 使用OpenKey函数获取句柄对象，可以打开注册表项*varKey = OpenKey(varReg, varSubKey)  
*for* i *in range*(1024):  
 *try*:  
 *# 4. 使用EnumKey访问子健目录* keyname = EnumKey(varKey, i)  
 varSubKey2 = "%s\\%s" % (varSubKey, keyname)  
 *# 5. 获取最终要访问的键的句柄* varKey2 = OpenKey(varReg, varSubKey2)  
 *try*:  
 *for* j *in range*(1024):  
 *# 6. 访问键中键值项，包含键值名、键值、数据类型* n, v, t = EnumValue(varKey2, j)  
 *if* n:  
 *# 7. 匹配账户信息并输出  
 if*("ProfileImagePath" *in* n *and* "Users" *in* v):  
 *print*(v)  
 *else*:  
 *break  
 except*:  
 errMsg = "Exception Inner:", sys.exc\_info()[0]  
 *# print(errMsg)* CloseKey(varKey2)  
 *except*:  
 errMsg = "Exception Outter:", sys.exc\_info()[0]  
 *# print(errMsg)  
 break  
# 8. 关闭句柄对象*CloseKey(varKey)  
CloseKey(varReg)

#### 运行结果



### python更新注册表-关闭防火墙

与访问信息类似，使用\_winreg模块提供的函数可以获取注册表句柄。使用CreateKey（）函数可以创建键，并输入数据。若注册表中已存在相同键，则更新其数据。SetValue（）函数提供数据输入功能。句柄使用完毕后，要调用CloseKey（）函数将其返还系统。

与Windows防火墙相关的设置信息也保存于注册表，包括防火墙启用/禁用信息、防火墙状态通知信息、是否添加启动程序、防火墙规则设置信息、注册程序信息等。

关闭防火墙：

HKEY\_LOCAL\_MACHINE\SYSTEM\CurrentControlSet\services\SharedAccess\Parameters\FirewallPolicy\StandardProfile\EnableFirewall

EnableFirewall = 1表示防火墙当前状态是开的，为0时表示防火墙已被禁用。

还可以关闭Windows defende：

HKEY\_LOCAL\_MACHINE\SYSTEM\CurrentControlSet\Services\SecurityHealthService

Start=4表示关闭Start=2表示开启

开启3389远程桌面：

HKEY\_LOCAL\_MACHINE\SYSTEM\CurrentControlSet\Control\Terminal Server

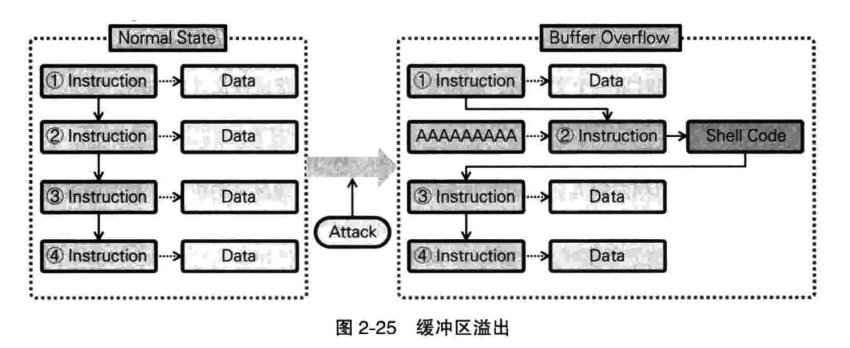
下的fDenyTSConnections=0表示开启了3389远程桌面，改为1表示关闭。

#### 代码实现

## 4.缓冲区溢出

缓冲区溢出攻击指通过向进程输入非正常数据，将黑客精心准备的数据保存到内存，并使之运行。进程运行时，处理流相应的数据会进入其所用的内存区域，如栈、堆以及寄存器。修改此类数据后，进程的处理顺序就会改变或者停止运行。黑客通过不断修改输入值，观察哪些数据会引发错误，以及在输入值的哪一部分植入shell code可以运行，最终编写攻击代码。

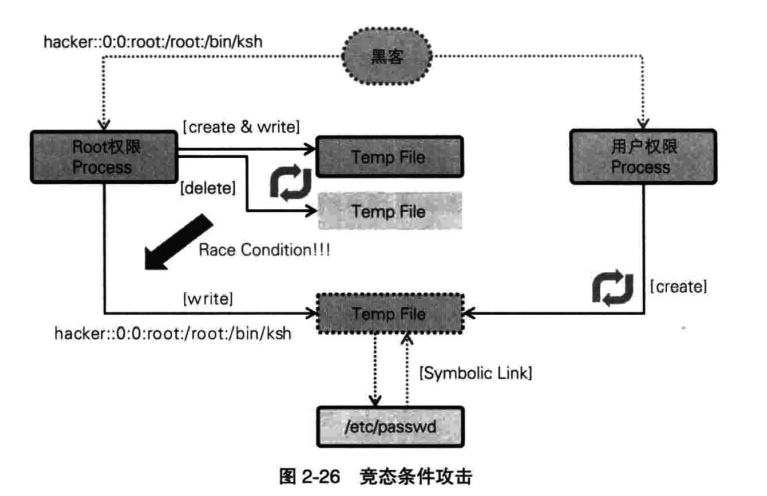
缓冲区溢出攻击代码不是独立运行的程序，它是随视频、音乐、文档一起运行的程序文件。假设要对视频播放器发起缓冲区溢出攻击。首先，黑客将含有错误代码的视频发布到网络，用户下载该视频后，在播放器中将其打开的瞬间，其内部含有的shell code会使内存陷入异常，从而得以运行。



## 5.竞态条件攻击

竞态条件是指两个进程为了使用一种资源而彼此竞争的状况。比如，写文件时，必须先获得文件句柄。多个进程同时写一个文件时，就需要相互竞争，以获得该文件的句柄。竞态条件攻击利用这一过程中出现的安全漏洞发动攻击。

最常用的方法是利用/etc/passwd文件的符号链接，/etc/passwd文件保存着用户账户信息。首先，使用root权限获得用户输入，创建临时文件，查找用于处理逻辑的进程。黑客向该进程反复输入代表添加用户的值（hacker：0:0:root:/root:/bin/ksh）。另一方面，运行程序，反复创建/etc/passwd文件的符号链接（与临时文件同名）。两个进程争夺文件句柄的过程中，进程会向符号文件链接保存用户输入。此时，hacker::0:0:root:/root:/bin/ksh会被写入etc/passwd文件，最终黑客账户获得root权限。



## 6．格式字符串攻击

格式字符串用于设置printf语句输出的字符串格式。比如，输出字符串的printf("printstring: %s",strName)语句中，%s为格式字符串。若要输出的参数（strName)正常出现在格式字符串之后，则不会有什么问题，否则就会取出栈中的值。利用这一原理，黑客将%s、%d、%x等多种格式字符串用作输入值，即可操作栈。尤其是使用类似于%n的格式字符串，它将printf()输出的字节数保存为整型指针。这使得黑客可以将想运行的shell code的返回地址输入栈。

