常见注册表&DLL&函数

一、常见注册表

- `HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Run`：开机自启
- `HKLM\SYSTEM\CurrentControlSet\Services`：服务
- `HKLM\SOFTWARE\Classes\CLSID\`，`HKCU\SOFTWARE\Classes\CLSID`：COM类的CLSID
- `HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows NT\CurrentVersion\AeDebug`：检查调试器

二、常见DLL

DLL	介绍
Kernel32.dll	包含核心功能，例如访问和控制内存、文件、硬件
Advapi32.dll	提供访问高级核心Windows组件的功能，例如Service Manager以及注册表
User32.dll	UI组件，例如按钮、滚动条以及控制和响应用户操作的功能
Gdi32.dll	显示及控制图形的功能
Ntdll.dll	内核接口，一般只由kernel32.dll间接导入。恶意软件直接导入该文件实现功能隐藏或进程控制等功能
WSock32.dll, Ws2_32.dll	网络功能
Wininet.dll	更高级别的网络功能，实现更多的协议，例如FTP, HTTP, NTP

二、常见函数定义

1. socket

```
SOCKET WSAAPI socket(
  int af,       // 2 AF_INET
  int type,     // 1 SOCK_STREAM
  int protocol  // 6 IPPROTO_TCP; 17 IPPROTO_UDP
);
```

2. CreateProcessA

```
BOOL CreateProcessA(
  LPCSTR                lpApplicationName,
  LPSTR                 lpCommandLine,      // 要执行的命令（进程名）
  LPSECURITY_ATTRIBUTES lpProcessAttributes,
  LPSECURITY_ATTRIBUTES lpThreadAttributes,
  BOOL                  bInheritHandles,
  DWORD                 dwCreationFlags,    // CREATE_SUSPENDED: 0x4
  LPVOID                lpEnvironment,
  LPCSTR                lpCurrentDirectory,
  LPSTARTUPINFOA        lpStartupInfo,      // 查看【常用结构体】
  LPPROCESS_INFORMATION lpProcessInformation
);
```

3. CreateThread

```
HANDLE CreateThread(
  LPSECURITY_ATTRIBUTES   lpThreadAttributes,
  SIZE_T                  dwStackSize,
  LPTHREAD_START_ROUTINE  lpStartAddress,       // 线程要执行的函数
  __drv_aliasesMem LPVOID lpParameter,          // 函数的参数
  DWORD                   dwCreationFlags,      // CREATE_SUSPENDED:  0x4
  LPDWORD                 lpThreadId
);
```

恶意软件可以使用该函数加载库文件，其中`lpStartAddress`是`LoadLibrary`的地址，`lpParameter`是要加载的库文件名；或者创建两个线程分别用于输入和输出。

4. CreateServiceA

```
SC_HANDLE CreateServiceA(
  SC_HANDLE hSCManager,
  LPCSTR    lpServiceName,      // 服务名
  LPCSTR    lpDisplayName,      // 在界面上显示的名称
  DWORD     dwDesiredAccess,
  DWORD     dwServiceType,      // SERVICE_WIN32_OWN_PROCESS: 0x10 代码在EXE文件中，作为进程单独执行
                                // SERVICE_WIN32_SHARE_PROCESS: 0x20 代码在DLL中，多个服务组合在一个形成共享进程，例如svchost.exe
                                // SERVICE_KERNEL_DRIVER: 0x1 将代码载入内核
  DWORD     dwStartType,        // SERVICE_AUTO_START: 0x2
  DWORD     dwErrorControl,
  LPCSTR    lpBinaryPathName,   // 二进制文件的完整路径
  LPCSTR    lpLoadOrderGroup,
  LPDWORD   lpdwTagId,
  LPCSTR    lpDependencies,
  LPCSTR    lpServiceStartName,
  LPCSTR    lpPassword
);
```

5. CoCreateInstance

```
HRESULT CoCreateInstance(
  REFCLSID  rclsid,         // CLSID
  LPUNKNOWN pUnkOuter,
  DWORD     dwClsContext,
  REFIID    riid,           // IID
  LPVOID    *ppv            // 指向riid请求的接口指针的指针地址
                            // 接口的结构体名称为【InterfaceName】Vtbl
);
```

`CLSID`可以在注册表`HKLM\SOFTWARE\Classes\CLSID\`，`HKCU\SOFTWARE\Classes\CLSID`查找

确定`IID`代表的接口名称后，可以在IDA的`Structures`子窗口中插入结构体 *InterfaceName*Vtbl，确定代码调用的具体是哪个函数；或者直接在网上搜索接口的头文件

6. SetWindowsHookEx

```
HHOOK SetWindowsHookExA(
  int       idHook,     // WH_CBT: 5; WH_KEYBOARD: 2; WH_KEYBOARD_LL: 13
  HOOKPROC  lpfn,       // hook后执行的函数
  HINSTANCE hmod,       // 包含了lpfn的DLL句柄，若dwThreadId指向的是当前进程线程，则为NULL
  DWORD     dwThreadId  // 若为0，则关联所有线程
);
```

7. KeInitializeApc

```
VOID KeInitializeApc(
  IN  PRKAPC Apc,            // 要返回的初始化KAPC结构
  IN  PRKTHREAD Thread,      // 执行该APC的线程
  IN  KAPC_ENVIRONMENT Environment,
  IN  PKKERNEL_ROUTINE KernelRoutine,
  IN  PKRUNDOWN_ROUTINE RundownRoutine OPTIONAL,
  IN  PKNORMAL_ROUTINE NormalRoutine OPTIONAL,  // 非0
  IN  KPROCESSOR_MODE ApcMode OPTIONAL,         // KERNELMODE: 0 USERMODE:1
  IN  PVOID NormalContext OPTIONAL
);
```

三、常见函数组合

1. `FindFirstFile`、`FindNextFile`说明程序在文件系统中进行遍历搜索
2. `LoadResource`，`FindResource`，`SizeOfResource`：需要查看程序的`rsrc`段，可能隐藏了其他信息
3. `CreateFile`，`WriteFile`，`WinExec`：创建了文件并执行
4. `CreateProcess`+`Sleep`+`exec`：可能是后门
5. `CreateToolhelp32Snapshot`：生成进程列表
6. `OpenService`，`DeleteService`，`OpenSCManager`，`CreateService`：服务相关功能
7. `CreateFileMapping`，`MapViewOfFile`：将文件导入内存并可对其进行任意修改，可以模仿windows loader的功能
8. `OpenMutex`，`CreateMutex`，`WaitForSingleObject`，`ReleaseMutex`：使用互斥锁获取共享资源。由于互斥锁的名称一般是硬编码在程序中的，所以是一个很好的host-based indicator。

9. `DllCanUnloadNow`，`DllGetClassObject`，`DllInstall`，`DllRegisterServer`，`DllUnregisterServer`：输出函数中有这些，说明实现了COM服务器功能
10. `CoCreateInstance`，`OleInitialize`：使用了COM功能
11. `OpenProcessToken`、`LookupPrivilegeValueA`、`AdjustTokenPrivileges`：权限提升，看到这些函数之后进行标记，不需要仔细分析
12. `VirtualAllocEx`，`WriteProcessMemory`：进程注入
13. `QueueUserAPC`，`KeInitializeApc`，`KeInsertQueueApc`：APC注入
14. `IsDebuggerPresent`，`CheckRemoteDebuggerPresent`，`NtQueryInformationProcess`，`OutputDebugString`：反调试技术
15. `QueryPerformanceCounter`、`GetTickCount`：可能有反调试技术