Windows user accounts

System

Administrator

Users
Architectural notes

- System calls are all dynamically linked
- Windowing and many other functions provided as libraries in Unix are part of the system
- “Kitchen sink” design philosophy
Process / memory APIs

- CreateProcess
- OpenProcess
- TerminateProcess
- CloseHandle
- ReadProcessMemory
- WriteProcessMemory
- VirtualProtect
BOOL CreateProcess(
    LPCWSTR pszImageName,
    LPCWSTR pszCmdLine,
    LPSECURITY_ATTRIBUTES psaProcess,
    LPSECURITY_ATTRIBUTES psaThread,
    BOOL fInheritHandles,
    DWORD fdwCreate,
    LPVOID pvEnvironment,
    LPWSTR pszCurDir,
    LPSTARTUPINFOW psiStartInfo,
    LPPROCESS_INFORMATION pProcInfo
);
OpenProcess

HANDLE OpenProcess(
    DWORD fdwAccess,
    BOOL fInherit,
    DWORD IDProcess
);

TerminateProcess

// Kill -9 for windows ;)

BOOL TerminateProcess(
    HANDLE hProcess,
    DWORD uExitCode
);

BOOL CloseHandle(HANDLE hObject);
BOOL ReadProcessMemory(
    HANDLE hProcess,
    LPCVOID lpBaseAddress,
    LPVOID lpBuffer,
    DWORD nSize,
    LPDWORD lpNumberOfBytesRead
);
WriteProcessMemory

BOOL WriteProcessMemory(
    HANDLE hProcess,
    LPVOID lpBaseAddress,
    LPVOID lpBuffer,
    DWORD nSize,
    LPDWORD lpNumberOfBytesWritten
);

VirtualProtect

// must be called from target process
BOOL VirtualProtect(
    LPVOID lpAddress,
    DWORD dwSize,
    DWORD flNewProtect,
    PDWORD lpflOldProtect
);
Uses of process APIs
- Spawn a process and observe its execution
- Patch code / data in memory
Messaging system

- Windows uses messages, instead of signals, to send data to a process
- Messages must be explicitly asked for using GetMessage() and DispatchMessage()
- Messages can be intercepted by hooks
Hooks

HHOOK SetWindowsHookEx(
    int idHook,
    HOOKPROC lpfn,
    HINSTANCE hMod,
    DWORD dwThreadId
);

Uses of hooks

- Stop messages from getting to a process
- Keyloggers
- Hit tracing
- DLL injection
- Scary stories ;(
Dynamic linking

- Entire Windows API is dynamically linked
- Compiling a library creates .lib and .dll
- COMPILe TIME:
  Link with .lib, need .dll at runtime
- RUN TIME:
  Ignore .lib, load .dll at run time
- Run time linking – plugins, libraries which might not be available, etc
LoadLibrary

// equivalent of linux dlopen()
HINSTANCE LoadLibrary(
    LPCTSTR lpLibFileName
);

GetProcAddress

// equivalent of linux dlsym()
FARPROC GetProcAddress(
    HMODULE hModule,
    LPCWSTR lpProcName
);

FreeLibrary

// equivalent of linux dlcose()
BOOL FreeLibrary(
    HMODULE hLibModule
);

Abusing dynamic linking

- Call DLL functions directly without going through the application
- Replace a DLL with a shim layer
Pathname gotchas - Whitespace

CreateProcess(
    NULL,
    "c:\program files\olly beta\olly 2k",
    NULL,
    NULL,
    FALSE,
    NULL,
    NULL,
    NULL,
    NULL,
    NULL
);

Pathname gotchas - Whitespace
“c:\program files\olly beta\olly 2k”
Pathname gotchas - Whitespace

“c:\program files\olly beta\olly 2k”

c:\program.exe files\olly beta\olly 2k
Pathname gotchas - Whitespace
“\c:\program files\olly beta\olly 2k”
\c:\\program.exe files\olly beta\olly 2k
\c:\\program files\olly.exe beta\olly 2k
Pathname gotchas - Whitespace
“c:\program files\olly beta\olly 2k”
c:\program.exe files\olly beta\olly 2k
\program\files\olly.exe beta\olly 2k
\program\files\olly beta\olly.exe 2k
Pathname gotchas - Whitespace

“c:\program files\olly beta\olly 2k”
c:\program.exe files\olly beta\olly 2k
c:\program files\olly.exe beta\olly 2k
c:\program files\olly beta\olly.exe 2k
c:\program files\olly beta\olly 2k.exe
Alternate data streams (ADS)

- Microsoft's implementation of filesystem forks
  - Began with NTFS
- Windows Explorer and the `dir` utility ignore ADS when displaying file information
- Can hide data in previously existing files and directories without changing the file's size
  - `C:\Users\jay\mainstream.txt:alternatestream.txt`
Offline fun

- Change/crack passwords, add user accounts, edit/dump registry etc.
  - http://pogostick.net/~pnh/ntpasswd/