Mã độc

Chương 5. Phân tích các chương trình độc hại trên Window

Mục tiêu

- Nhắc lại một số kiến thức về hệ điều hành Window
- Giới thiệu một số hành vi của mã độc chạy trên hệ điều hành Window

Tài liệu tham khảo

[1] Michael Sikorski, Andrew Honig, 2012, Practical Malware Analysis: The Hands-On Guide to Dissecting Malicious Software, No Starch Press, (ISBN: 978-1593272906).

[2] Sam Bowne, Slides for a college course at City College San Francisco, https://samsclass.info/126/126_S17.shtml

Nội dung

1. Windows API
2. Windows Registry
3. Các API xử lý kết nối mạng
4. Phân tích mã độc trên Windows
5. Chế độ nhân và chế độ người dùng
6. Các native API

Nội dung

1. Windows API
2. Windows Registry
3. Các API xử lý kết nối mạng
4. Phân tích mã độc trên Windows
5. Chế độ nhân và chế độ người dùng
6. Các native API

Windows API

❑ Quản lý cách các chương trình tương tác với các thư viện của Microsoft

- Handles
- File System Functions
- Special Files

Common API Types

Kiểu (Tiền tố)

- WORD (w) – 16 bits giá trị không âm
- DWORD (dw) – 32 bits giá trị không âm
- Handle (H) – Tham chiếu đến một đối tượng (Object)
- Long Pointer (LP) – Points to another type

7

Handles

- Handle được tạo bởi hệ điều hành
  - Giống như: Windows, Process, Menu, File,...
- Handles giống như con trỏ tới các đối tượng
- Điều duy nhất có thể làm với một handle là lưu trữ và sử dụng nó sau khi gọi hàm để tham chiếu đến một đối tượng

8

Handles

- Hàm CreateWindowEx trả về một HWND, một handle cho một cửa sổ
- Với handle đó có thể làm bất cứ thứ gì với cửa sổ mà nó đã tạo như: DestroyWindow...

9

File System Functions

- CreateFile, ReadFile, WriteFile
  - Nhập/xuất với file thông thường
- CreateFileMapping, MapViewOfFile
  - Thường được sử dụng bởi mã độc, tải tệp vào RAM
  - Có thể được sử dụng để thực thi một tệp tin mà không cần thông qua Windows loader

10

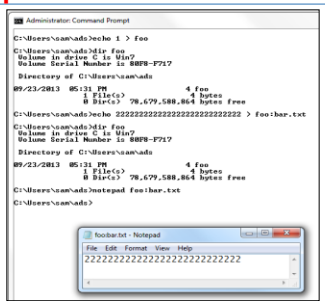
Special Files

- Các tệp được chia sẻ như \\server\share
  - Hoặc \\?\server\share
  - Ngừng phân tích cú pháp chuỗi, cho phép tên tập tin dài hơn
- Không gian tên (Namespace)
  - Các thư mục đặc biệt trên hệ thống tệp tin của windows
  - \ : Thư mục gốc chứa mọi thứ
  - \\.\ : Thiết bị được lưu trữ sử dụng cho input/output
  - Sâu Witty đã viết vào \\.\PhysicalDisk1 để làm hỏng đĩa

11

Special Files

- Điều khiển luồng dữ liệu
- Dữ liệu được đẩy vào files



12

Nội dung

1. Windows API
2. Windows Registry
3. Các API xử lý kết nối mạng
4. Phân tích mã độc trên Windows
5. Chế độ nhân và chế độ người dùng
6. Các native API

13

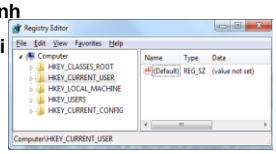
Registry

- Nơi lưu trữ cài đặt cấu hình của hệ điều hành và các ứng dụng
  - Desktop background, mouse preferences, etc.
- Mã độc thường sử dụng Registry để duy trì sự tồn tại của nó trên hệ thống
  - Làm cho mã độc tự khởi động cùng hệ thống

14

Registry

- ROOT KEYS – Có 5 khóa chính
- SUBKEY – Mỗi khóa chính lại có các khóa con bên trong
- KEY – Thư mục, có thể chứa các thư mục khác hoặc giá trị
- VALUE ENTRY – Gồm hai phần: Name và Data
- VALUE hoặc DATA – Dữ liệu được lưu trữ trong Registry entry



15

Root Keys

`HKEY_LOCAL_MACHINE (HKLM)` Stores settings that are global to the local machine

`HKEY_CURRENT_USER (HKCU)` Stores settings specific to the current user

`HKEY_CLASSES_ROOT` Stores information defining types

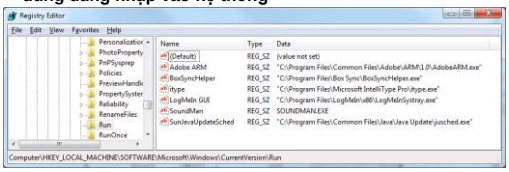
`HKEY_CURRENT_CONFIG` Stores settings about the current hardware configuration, specifically differences between the current and the standard configuration

`HKEY_USERS` Defines settings for the default user, new users, and current users

16

Run Key

- HKLM\SOFTWARE\Microsoft\Windows\ \CurrentVersion\Run
  - Các chương trình thực thi được khởi chạy khi người dùng đăng nhập vào hệ thống



17

Common Registry Functions

- RegOpenKeyEx
  - Mở một Registry key để chỉnh sửa và truy vấn
- RegSetValueEx
  - Thêm một giá trị (key) mới vào Registry và set dữ liệu cho nó
- RegGetValue
  - Trả về dữ liệu cho một entry trong Registry

18

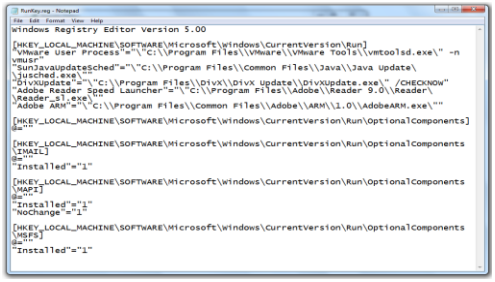
Registry Code

Code that modifies registry settings

```
0040286F   push    2                 ; samDesired
00402871   push    eax               ; ulOptions
00402872   push    offset SubKey     ;
"Software\\Microsoft\\Windows\\CurrentVersion\\Run"
00402877   push    HKEY_LOCAL_MACHINE ; hKey
```

19

.REG Files



20

Nội dung

1. Windows API
2. Windows Registry
3. Các API xử lý kết nối mạng
4. Phân tích mã độc trên Windows
5. Chế độ nhân và chế độ người dùng
6. Các native API

21

Các API xử lý kết nối mạng

- ❑ Berkeley Compatible Sockets
- ❑ The WinINet API

22

Các API xử lý kết nối mạng

- ❑ Berkeley Compatible Sockets
- ❑ The WinINet API

23

Berkeley Compatible Sockets

- ❑ Thư viện Winsock, chủ yếu là ws2_32.dll
- ❑ Hầu như là giống nhau trên cả Windows và Unix

24

Berkeley Compatible Sockets

Function	Description
socket	Creates a socket
bind	Attaches a socket to a particular port, prior to the accept call
listen	Indicates that a socket will be listening for incoming connections
accept	Opens a connection to a remote socket and accepts the connection
connect	Opens a connection to a remote socket; the remote socket must be waiting for the connection
recv	Receives data from the remote socket
send	Sends data to the remote socket

25

Server and Client Sides

- ❑ Server side
  - ▪ Duy trì và luôn mở socket đợi kết nối từ client
  - ▪ Các hàm thường được gọi theo thứ tự: socket, bind, listen, accept
  - ▪ Sau đó send và recv khi cần thiết
- ❑ Client side
  - ▪ Kết nối vào một socket đang chờ
  - ▪ Gọi các hàm theo thứ tự: socket, connect
  - ▪ Sau đó send và recv khi cần thiết

26

Simplified Server Program

```
00401041  push    ecx             ; lpWSAData
00401042  push    202h            ; wVersionRequested
00401047  mov     word ptr [esp+250h+name.sa_data], ax
0040104C  call    ds:WSAStartup
00401052  push    0               ; protocol
00401054  push    1               ; type
00401056  push    2               ; af
00401058  call    ds:socket
0040105E  push    10h             ; namelen
00401060  lea     edx, [esp+24Ch+name]
00401064  mov     ebx, eax
00401066  push    edx             ; name
00401067  push    ebx             ; s
00401068  call    ds:bind
0040106E  mov     esi, ds:listen
00401074  push    5               ; backlog
00401076  push    ebx             ; s
00401077  call    esi ; listen
00401079  lea     eax, [esp+248h+addrlen]
0040107D  push    eax             ; addrlen
0040107E  lea     ecx, [esp+24Ch+hostshort]
00401082  push    ecx             ; addr
00401083  push    ebx             ; s
00401084  call    ds:accept
```

27

Các API xử lý kết nối mạng

- ❑ Berkeley Compatible Sockets
- ❑ The WinINet API

28

The WinINet API

- ❑ Thuộc dạng API mức cao hơn Winsock
- ❑ Các hàm trong Wininet.dll sử dụng với các chương trình ở mức ứng dụng: HTTP, FTP, SMTP, POP,...
- ❑ Một số hàm
  - ▪ InternetOpen – Mở kết nối internet
  - ▪ InternetOpenURL – Kết nối đến một đường dẫn
  - ▪ InternetReadFile - Đọc dữ liệu từ tập tin đã tải

29

Nội dung

1. Windows API
2. Windows Registry
3. Các API xử lý kết nối mạng
4. Phân tích mã độc trên Windows
5. Chế độ nhân và chế độ người dùng
6. Các native API

30

Phân tích mã độc trên Windows

Việc thực thi code được chuyển sang cho một đối tượng khác xử lý, có những cách để chuyển việc thực thi đó như:

- DLLs
- Processes
- Threads
- Mutexes
- Services

31

DLLs (Dynamic Link Libraries)

- Chia sẻ và dùng chung những đoạn code giữa nhiều ứng dụng
- DLLs export code có thể được sử dụng bởi các ứng dụng
- Static libraries được sử dụng trước DLLs
  - Chúng vẫn tồn tại nhưng ít phổ biến hơn
  - Static libraries thì không chia sẻ bộ nhớ giữa các tiến trình
  - Các Static libraries sử dụng nhiều RAM hơn các DLLs

32

Ưu điểm của DLL

- Sử dụng DLL có sẵn trong windows giúp cho kích thước của chương trình được nhỏ hơn
- Các công ty về phần mềm cũng có thể tạo ra những DLL tùy chỉnh

33

Cách mã độc sử dụng DLLs

- Lưu trữ những đoạn mã độc hại trong DLL
  - Đôi khi những DLL độc hại được nạp vào các tiến trình
- Sử dụng Windows DLLs
  - Các mã độc hầu hết đều sử dụng những DLL cơ bản
- Sử dụng DLL của bên thứ 3
  - Sử dụng FireFox để kết nối tới server thay vì Windows API

34

Cấu trúc DLL cơ bản

- Cấu trúc của DLL gần giống với EXEs
- Định dạng file thực thi – PE File
- Một cờ chỉ ra rằng nó là DLL chứ không phải EXE
- DLL có nhiều exports và ít imports
- DLLmain là một hàm chính, không export nhưng được chỉ định là Entry point trong PE Header
  - Được gọi khi một hàm nạp hoặc gỡ bỏ thư viện

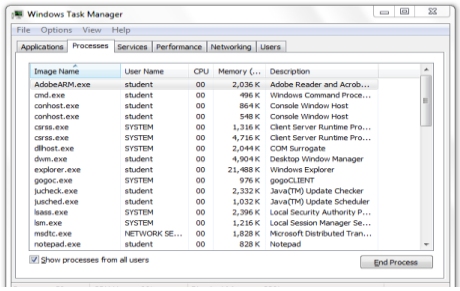
35

Processes

- Tiến trình là những chương trình đang chạy bởi windows
- Mỗi tiến trình đều có các tài nguyên riêng (Handles, bộ nhớ,...)
- Mỗi tiến trình có một hoặc nhiều luồng xử lý
- Những mã độc cũ thường chạy như một tiến trình độc lập
- Những mã độc mới thực thi mã của nó như là một phần của tiến trình

36

Processes



Quản lý bộ nhớ

- ❑ Mỗi tiến trình đều sử dụng tài nguyên hệ thống như: CPU, File System và bộ nhớ,...
- ❑ Hệ điều hành sẽ cấp phát vùng nhớ cho mỗi tiến trình
- ❑ Hai tiến trình cùng truy cập vào cùng địa chỉ bộ nhớ nhưng thực sự là truy cập vào vị trí khác nhau trên RAM
  - Virtual address space

Tạo một Process mới

- ❑ Tạo ra một remote shell với một lời gọi hàm
- ❑ Tham số STARTUPINFO chứa các handle cho nhập/xuất chuẩn và standard error streams.
  - Có thể được thiết lập một socket, tạo một remote shell

Tạo một Shell

Sample code using the CreateProcess call

```
004010DA  mov     eax, dword ptr [esp+58h+SocketHandle]
004010DE  lea     edx, [esp+58h+StartupInfo]
004010E2  push    ecx             ; lpProcessInformation
004010E3  push    edx             ; lpStartupInfo
004010E4  mov     [esp+60h+StartupInfo.hStdError], eax
004010E8  mov     [esp+60h+StartupInfo.hStdOutput], eax
004010EC  mov     [esp+60h+StartupInfo.hStdInput], eax
004010F0  mov     eax, dword_403098
004010F5  push    0               ; lpCurrentDirectory
004010F7  push    0               ; lpEnvironment
004010F9  push    0               ; dwCreationFlags
004010FB  mov     dword ptr [esp+6Ch+CommandLine], eax
```

Nạp socket handle, StdError, StdOutput và StdInput thành lpProcessInformation.

Tạo một Shell

```
004010FF  push    1               ; bInheritHandles
00401101  push    0               ; lpThreadAttributes
00401103  lea     eax, [esp+74h+CommandLine]
00401107  push    0               ; lpProcessAttributes
00401109  push    eax             ; lpCommandLine
0040110A  push    0               ; lpApplicationName
0040110C  mov     [esp+80h+StartupInfo.dwFlags], 101h
00401114  call    ds:CreateProcessA
```

- CommandLine contains
- Nó được thực thi khi hàm CreateProcess được gọi

Threads

- ❑Các tiến trình chính là các containers
  - Mỗi tiến trình chứa một hoặc nhiều Thread
- ❑Thread là những gì Windows thực sự thực thi
- ❑Thead (luồng)
  - Chuỗi các lệnh độc lập
  - Được thực thi bởi CPU mà không cần phải đợi các thread khác
  - Các thread trong cùng một tiến trình sẽ dùng chung vùng nhớ
  - Mỗi thread có thanh ghi và Stack riêng

Thread Context

- ❑ Khi một thread đang chạy nó có toàn quyền kiểm soát CPU
- ❑ Các thread khác không làm ảnh hưởng đến trạng thái của CPU
- ❑ Khi một thread thay đổi một thanh ghi, nó không ảnh hưởng đến thread khác
- ❑ Khi hệ điều hành chuyển qua thread khác, nó lưu lại tất cả các giá trị của CPU trong một cấu trúc (struct) được gọi là thread context

43

Tạo một Thread

- ❑ CreateThread
- ❑ Hàm gọi đã chỉ định một địa chỉ bắt đầu, còn được gọi là một hàm bắt đầu

44

Cách mã độc sử dụng Thread

- ❑ Sử dụng CreateThread để tải một DLL độc hại vào một tiến trình
- ❑ Tạo hai thread cho input và output
  - ▪ Sử dụng để giao tiếp với một ứng dụng đang chạy

45

Mutexes

- ❑ Mutexes là các đối tượng toàn cục, điều phối các tiến trình và thread
- ❑ Ở kernel, chúng được gọi là mutants
- ❑ Mutexes thường sử dụng hard-coded để xác định mã độc

46

Functions for Mutexes

- ❑ WaitForSingleObject
  - ▪ Cung cấp một thread truy cập vào mutex
  - ▪ Bất kỳ các thread con khác truy cập vào nó phải đợi
- ❑ ReleaseMutex
  - ▪ Được gọi khi một thread hoàn tất sử dụng mutex
- ❑ CreateMutex
- ❑ OpenMutexs
  - ▪ Get một handle để xử lý những tiến trình mutex khác

47

Kiểm tra tiến trình đang chạy

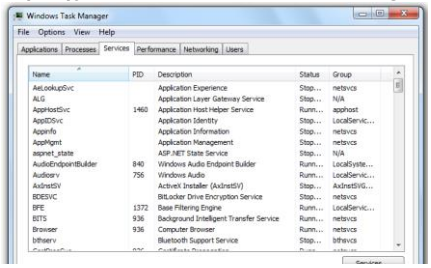
- ❑ OpenMutex kiểm tra có HGL345 tồn tại hay không
- ❑ Nếu không, nó sẽ được tạo ra với CreateMutex
- ❑ test eax, eax – Set cờ Z nếu eax bằng 0

```
00401007   push  1F0001h         ; dwDesiredAccess
0040100C  Ecall  ds:__imp__OpenMutexW@12 ;
OpenMutexW(x,x,x)
00401012  Etest  eax, eax
00401014  Ejz    short loc_40101E
00401016   push  0               ; int
00401018  Ecall  ds:__imp__exit
0040101E   push  offset Name     ; "HGL345"
00401023   push  0               ; bInitialOwner
00401025   push  0               ; lpMutexAttributes
00401027  Ecall  ds:__imp__CreateMutexW@12 ;
CreateMutexW(x,x,x)
```

48

Services

❑ Dịch vụ chạy nền mà không có đầu vào của người dùng



SYSTEM Account

❑ Dịch vụ thường chạy ở mức hệ thống, thậm chí còn mạnh hơn cả Administrator

❑ Dịch vụ có thể tự khởi chạy khi windows khởi động

- Đây là một cơ chế để mã độc có thể tận dụng để duy trì sự tồn tại của nó trên hệ thống
- Mã độc vẫn tồn tại khi hệ thống khởi động lại

Service API Functions

❑ OpenSCManager

- Trả về một handle cho Service Control Manager

❑ CreateService

- Thêm một service mới vào Service Control Manager
- Có thể xác định dịch vụ sẽ tự khởi động khi khởi động

❑ StartService

- Chỉ được sử dụng nếu service được thiết lập khởi động một cách thủ công

Svchost.exe

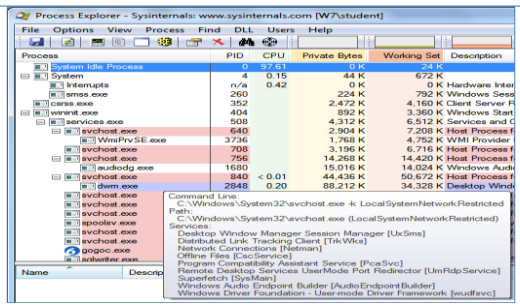
WIN32_SHARE_PROCESS

❑ Là một loại service phổ biến nhất, mục tiêu của mã độc

❑ Lưu trữ code cho service trong một DLL

❑ Kết hợp một số service vào một tiến trình chia sẻ duy nhất có tên svchost.exe

Svchost.exe



Một số service khác

❑ WIN32_OWN_PROCESS

- Chạy như một EXEs trong một tiến trình độc lập

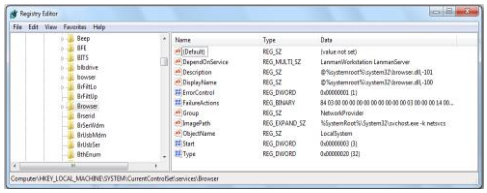
❑ KERNEL_DRIVER

- Được sử dụng để load code vào kernel

Thông tin về Service trong Registry

HKLM\System\CurrentControlSet\Services

- Bắt đầu giá trị = 0x03 cho "Load on Demand"
- Type = 0x20 cho WIN32_SHARE_PROCESS



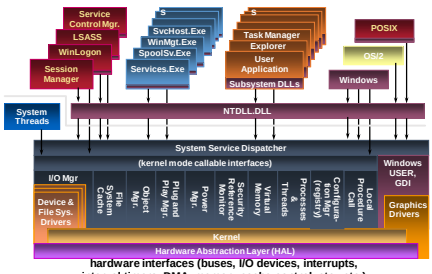
55

Nội dung

1. Windows API
2. Windows Registry
3. Các API xử lý kết nối mạng
4. Phân tích mã độc trên Windows
5. Chế độ nhân và chế độ người dùng
6. Các native API

56

System Architecture

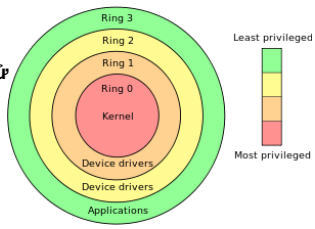


hardware interfaces (buses, I/O devices, interrupts, interval timers, DMA, memory cache control, etc., etc.)

57

Privilege Levels

- Ring 0: Kernel Mode
- Ring 3: User Mode
- Ring 1 & 2: không sử Dụng bởi Windows



58

User mode

- Hầu như tất cả code đều chạy ở User-mode
  - Ngoại trừ hệ điều hành và các driver phần cứng, chúng chạy ở Kernel-mode
- Chế độ User-mode không thể truy cập phần cứng trực tiếp
- Ở User-mode một số lệnh CPU bị hạn chế
- Chỉ có thể thao tác với phần cứng không qua Windows API

59

User mode processes

- Mỗi tiến trình có vùng nhớ của riêng nó, quyền hạn nhất định và tài nguyên riêng
- Các tiến trình ở User-mode không can thiệp được vào vùng nhớ của nhau
- Nếu một chương trình ở User-mode thực thi những lệnh không hợp lệ và gây Crash chương trình thì hệ điều hành Windows có thể khôi phục lại các tài nguyên và chấm dứt chương trình.

60

Calling the Kernel

- Không thể nhảy trực tiếp từ User-mode sang Kernel
- SYSENTER, SYSCALL hoặc lệnh INT 0x2E sử dụng bảng tra cứu để xác định trước các hàm.

61

Kernel Processes

- Tất cả các tiến trình ở Kernel-mode đều chia sẻ tài nguyên và địa chỉ bộ nhớ cho nhau
- Các tiến trình ở kernel-mode có quyền cao và được ưu tiên hơn các tiến trình ở user-mode
- Kiểm tra bảo mật ít hơn

62

Kernel Processes

- Ở Kernel-mode nếu thực thi một lệnh không hợp lệ, hệ điều hành sẽ gặp sự cố với màn hình xanh chết chóc
- Các phần mềm anti-virus và Firewall chạy ở chế độ Kernel-mode

63

Malware in Kernel Mode

- Loại mã độc này nguy hiểm hơn mã độc chạy ở chế độ User-mode
- Auditing không áp dụng được với kernel
- Hầu hết các Rootkits đều sử dụng kernel code
- Tuy nhiên phần lớn mã độc không bắt gặp nhiều ở kernel mode mà chủ yếu gặp user mode

64

Nội dung

1. Windows API
2. Windows Registry
3. Các API xử lý kết nối mạng
4. Phân tích mã độc trên Windows
5. Chế độ nhân và chế độ người dùng
6. Các native API

65

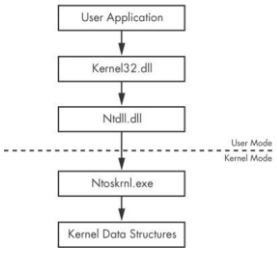
Các native API

- Là những API mức thấp để tương tác với windows mức sâu: tiến trình, bộ nhớ,...
- Các chương trình thông thường thì hiếm khi sử dụng
- Thường gặp phổ biến ở mã độc

66

Ntdll.dll

- Ntdll.dll quản lý tương tác giữa user-mode và kernel-mode
- Các hàm của Ntdll tạo thành các Native API



User mode and kernel mode

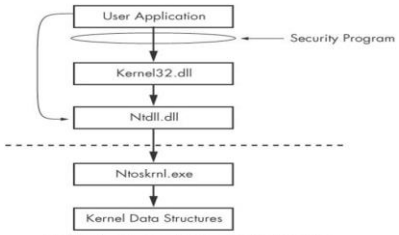
67

Các native API

- Không có tài liệu đề cập đến dạng API này
- Được dành cho Windows sử dụng nội bộ
- Native API có thể "mạnh" hơn và được gọi trong các mã độc.

68

Sử dụng native API



Using the Native API to avoid detection

69

Các native API thường có trong mã độc

- NtQuerySystemInformation
- NtQueryInformationProcess
- NtQueryInformationThread
- NtQueryInformationFile
- NtQueryInformationKey

70

Các native API thường có trong mã độc

NtContinue

- Trả về một ngoại lệ
- Có thể được dùng để chuyển thực thi theo những cách phức tạp
- Được sử dụng để gây sự nhầm lẫn cho các nhà phân tích và làm cho một chương trình khó debug hơn.

71

Nội dung

1. Windows API
2. Windows Registry
3. Các API xử lý kết nối mạng
4. Phân tích mã độc trên Windows
5. Chế độ nhân và chế độ người dùng
6. Các native API

72