

بسم الله الرحمن الرحيم

توابع و کامپوننت های متن باز برای دسترسی و مانیتورینگ API های سیستم عامل ویندوز

گردآوری توسط :

فرهنگ رشیدی

دانشجوی کارشناسی ارشد مهندسی نرم افزار

زیر نظر استاد محترم :

دکتر پارسا

دانشگاه آزاد واحد شبستر 1392

پیشگفتار

با رشد و پیشرفت سریع علم رایانه و گسترش هر چه بیشتر آن در زیر ساخت های علوم دیگر ، مبحث مهندسی معکوس اهمیت خود را هر چه بیشتر به معرض نمایش قرار داده است.

با توجه به این مهم و برای پیشبرد این هدف در علم رایانه تلاش های زیادی در سال های اخیر صورت گرفته است که در زیر به یکی از این ابزار ها یعنی مانیتورنگ ، دسترسی به توابع کاربردی در مد هسته می پردازیم.

لیست توابع و برنامه های موجود و متن باز در این زمینه که به ما امکان استفاده آنها در برنامه خود و ایجاد برنامه هایی در سطح برنامه های مانیتورینگ حال حاضر را قرار می دهد ارائه می دهیم :

نام توابع	زبان توابع	32bit	64bit	قابلیت زبان های برنامه نویسی	متن باز بودن	سازنده توابع
Paladin	C++	32bit	No	C++ - VC++	open source	Paladin
ProcViewer	C++	32bit	No	C++ - VC++	open source	ProcViewer
Deviare	C++	32bit	64bit	C - C# - VB – VBScript - Python	open source	Deviare
WinAPIOverride32	C++	32bit	No	C++ - VC++	open source	WinAPIOverride
madCollection	C++	32bit	No	C++ - VC++	Semi open source	Mad
EasyHook	C++	32bit	64bit	C++ - VC++ Console C# - VC#	open source	Microsoft Detoris

جدول : لیست توابع و کامپوننت های کاربردی برای مانیتورینگ توابع سیستم عامل ویندوز

از توابع نامبرده در جدول بالا ، کامپوننتی را که قابلیت 64 بیتی بودن و قابلیت استفاده از آن در زبان های برنامه نویسی سطح بالا را دارد بررسی می نماییم :



در واقع این تابع محصول شرکت مایکروسافت و زیر گروه برنامه دیتریس می باشد . و بروز شده برنامه دیتریس نوشته شده دسامبر 2006 می باشد که تمامی توابع به زبان سی نوشته شده است و این قابلیت را دارد که در محیط سی شارپ نیز مورد استفاده قرار گیرد .

با کمک توابع و کلاسهای این کامپوننت می توانیم برنامه هایی در حد

RegMOn , FileMon

آن هم با تعداد خطوط بسیار کم برنامه نویسی تولید نماییم .

لازم به ذکر است که این تابع قابلیت 32 بیتی و 64 بیتی برای کنترل مد هسته را دارا می باشد.



EasyHook64.dll



EasyHook32.dll

32bit and 64bit DLLs of EasyHook

نمونه ای عملی و کاربردی از تابع ارائه شده در قالب مانیتورینگ توابع سیستمی که شبیه ساز برنامه

FileMon می باشد.

```

using System;
using System.Collections.Generic;
using System.Runtime.Remoting;
using System.Text;
using EasyHook;

namespace FileMon
{
    public class FileMonInterface : MarshalByRefObject
    {
        public void IsInstalled(Int32 InClientPID)
        {
            Console.WriteLine("FileMon has been installed in target
{0}.\r\n", InClientPID);
        }

        public void OnCreateFile(Int32 InClientPID, String[] InFileNames)
        {
            for (int i = 0; i < InFileNames.Length; i++)
            {
                Console.WriteLine(InFileNames[i]);
            }
        }

        public void ReportException(Exception InInfo)
        {
            Console.WriteLine("The target process has reported an error:\r\n"
+ InInfo.ToString());
        }

        public void Ping()
        {
        }
    }

    class Program
    {
        static String ChannelName = null;

        static void Main(string[] args)
        {
            try
            {
                Config.Register(
                    "A FileMon like demo application.",
                    "FileMon.exe",
                    "FileMonInject.dll");

                RemoteHooking.IpcCreateServer<FileMonInterface>(ref
ChannelName, WellKnownObjectMode.SingleCall);

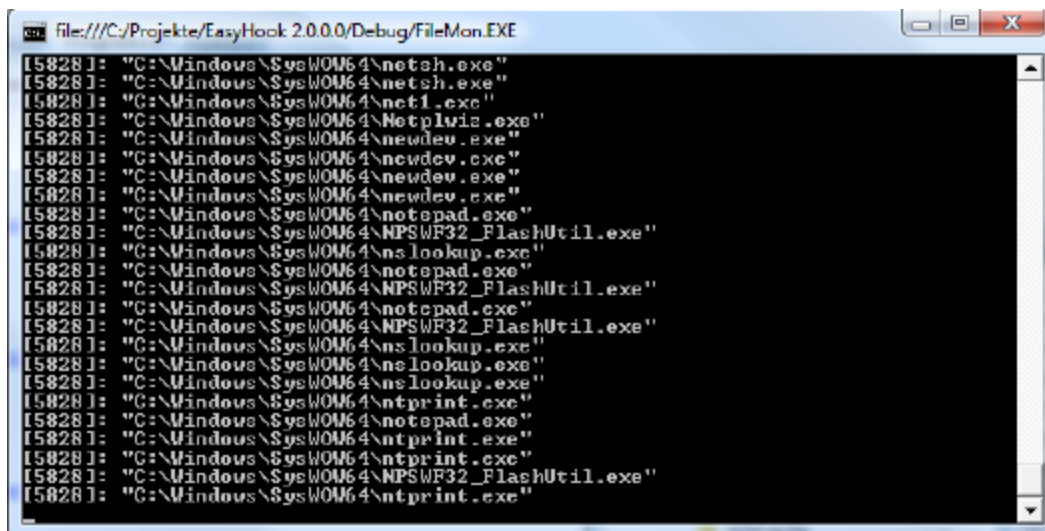
                RemoteHooking.Inject(
                    Int32.Parse(args[0]),
                    "FileMonInject.dll",
                    "FileMonInject.dll",
                    ChannelName);
            }
        }
    }
}

```

نمونه سورس کد بالا که به زبان سی شارپ می باشد امکان کنترل و مانیتورینگ تمام توابع و برنامه های در حال اجرا در مد کاربر و مد هسته سیستم عامل ویندوز را به ما می دهد و تعداد و زمان دسترسی هر برنامه به توابع سیستمی قابل مشاهده و بررسی می باشد.

با استفاده از امکانات این برنامه هر لحظه می توانیم از دستیابی های برنامه های ما به توابع سیستم عامل آگاه شویم و این امر در تشخیص و آنالیز رفتار برنامه های در حال اجرا در مبحث مهندسی معکوس بسیار کار آمد است مثلاً از ماهیت یک برنامه در حال اجرا و دسترسی به توابع سیستمی در زمان و تعداد کنترل شده از بدافزار بودن آن یقین حاصل نماییم.

تصویر برنامه که به حالت کنسول می باشد :

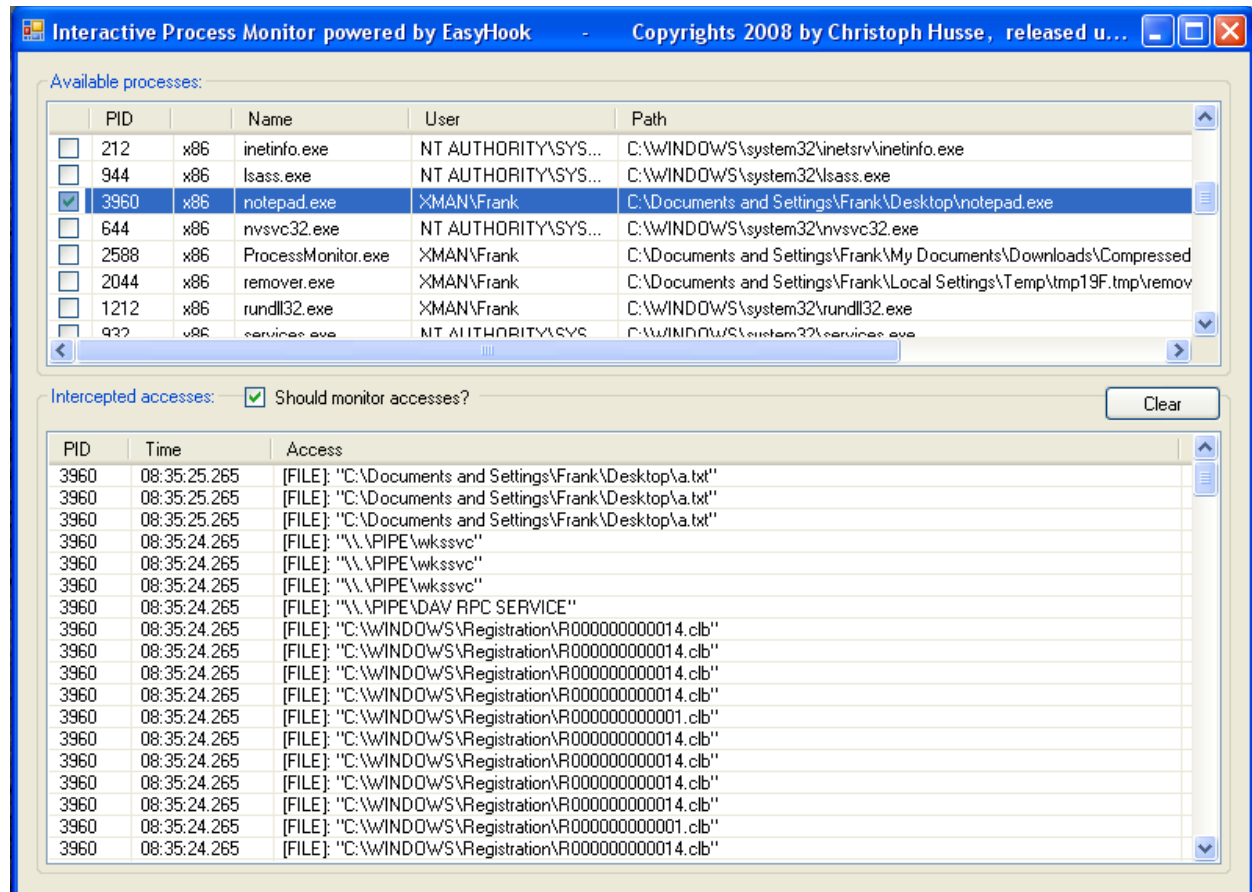


```
File:///C:/Projekte/EasyHook 2.0.0.0/Debug/FileMon.EXE
[5828]: "C:\Windows\SysWOW64\netsh.exe"
[5828]: "C:\Windows\SysWOW64\netsh.exe"
[5828]: "C:\Windows\SysWOW64\net1.exe"
[5828]: "C:\Windows\SysWOW64\Netplwiz.exe"
[5828]: "C:\Windows\SysWOW64\newdev.exe"
[5828]: "C:\Windows\SysWOW64\newdev.exe"
[5828]: "C:\Windows\SysWOW64\newdev.exe"
[5828]: "C:\Windows\SysWOW64\notepad.exe"
[5828]: "C:\Windows\SysWOW64\NPSWF32_FlashUtil.exe"
[5828]: "C:\Windows\SysWOW64\nslookup.exe"
[5828]: "C:\Windows\SysWOW64\notepad.exe"
[5828]: "C:\Windows\SysWOW64\NPSWF32_FlashUtil.exe"
[5828]: "C:\Windows\SysWOW64\notepad.exe"
[5828]: "C:\Windows\SysWOW64\NPSWF32_FlashUtil.exe"
[5828]: "C:\Windows\SysWOW64\nslookup.exe"
[5828]: "C:\Windows\SysWOW64\nslookup.exe"
[5828]: "C:\Windows\SysWOW64\nslookup.exe"
[5828]: "C:\Windows\SysWOW64\ntprint.exe"
[5828]: "C:\Windows\SysWOW64\notepad.exe"
[5828]: "C:\Windows\SysWOW64\ntprint.exe"
[5828]: "C:\Windows\SysWOW64\ntprint.exe"
[5828]: "C:\Windows\SysWOW64\NPSWF32_FlashUtil.exe"
[5828]: "C:\Windows\SysWOW64\ntprint.exe"
```

FileMon شبیه ساز برنامه

برنامه دیگر با کمک این توابع و کلاس های این برنامه

باز هم به زبان سی شارپ ولی این دفعه به صورت ویژوال :



همان طور که مشاهده می شود از این برنامه برای کنترل و مانیتورینگ برنامه نوت پد مورد استفاده قرار دادیم و لیستی از دسترسی ها ، زمان دسترسی ، تعداد دسترسی ها و مکان توابع به کار رفته سیستمی قابل مشاهده و بررسی می باشد.

در زیر برای بررسی هر چه بیشتر زیر تابع مشاهده و دستیابی به تعداد مراجعات برنامه ما به توابع سیستمی را قرار می دهیم.

```

private static void OnProcessUpdate(Object InCallback)
{
    ProcessTimer.Change(-1, -1);
    try
    {
        ProcessInfo[] array = (ProcessInfo[])RemoteHooking.ExecuteAsService<Form1>("EnumProcesses", new object[0]);
        SortedDictionary<string, ProcessInfo> dictionary = new SortedDictionary<string, ProcessInfo>();
        lock (ProcessList)
        {
            ActivePIDList.Clear();
            for (int i = 0; i < array.Length; i++)
            {
                dictionary.Add(System.IO.Path.GetFileName(array[i].FileName) + " ____" + i, array[i]);
                ActivePIDList.Add(array[i].Id);
            }
            dictionary.Values.CopyTo(array, 0);
            ProcessList.Clear();
            ProcessList.AddRange(array);
        }
    }
    catch (AccessViolationException)
    {
        MessageBox.Show("This is an administrative task!", "Permission denied...", MessageBoxButtons.OK);
        Process.GetCurrentProcess().Kill();
    }
    finally
    {
        ProcessTimer.Change(0x1388, 0x1388);
    }
}

```

این زیر تابع با گوش دادن و در واقع با عمل قلاب اندازی دسترسی های انجام شده به توابع سیستمی را دریافت و در جدول داده نمایش می دهد.

با سپاس و تشکر فراوان

رشیدی 92/1/27