using DotRas;

using Microsoft.Win32;

using System;

using System.Diagnostics;

using System.Net;

using System.Net.NetworkInformation;

using System.Runtime.InteropServices;

using System.Windows.Forms;

namespace AutoVPN

{

class Program

{

static System.Threading.ManualResetEvent \_quitEvent = new System.Threading.ManualResetEvent(false);

static void Main(string[] args)

{

VPN vpn = new VPN();

vpn.Start();

PreventShutOff pso = new PreventShutOff();

pso.Start();

\_quitEvent.WaitOne(); // Pause and listen

}

public class PreventShutOff

{

[DllImport("user32.dll")]

static extern bool SetForegroundWindow(IntPtr hWnd);

System.Timers.Timer timer = new System.Timers.Timer(120000);// Every 2mins(120000) check if bat. low

bool runOnce = true;

private int Fully\_Charge;

public PreventShutOff()

{

timer.Elapsed += new System.Timers.ElapsedEventHandler(timer\_Elapsed);

SystemEvents.PowerModeChanged += new PowerModeChangedEventHandler(SystemEvents\_PowerModeChanged);

Fully\_Charge = 0;

}

private void SystemEvents\_PowerModeChanged(object sender, PowerModeChangedEventArgs e)

{

// When I plug in AC power, it triggers twice for some reason

PowerStatus p = SystemInformation.PowerStatus;

if (p.PowerLineStatus == PowerLineStatus.Online)

{

if (runOnce)

{

PauseMovie();

runOnce = false;

}

}

else

{

runOnce = true;

timer.Enabled = true;

Fully\_Charge = 0;

}

}

public void Start()

{

timer.Enabled = true;

}

private bool BatteryIsLow()

{

PowerStatus p = SystemInformation.PowerStatus;

if (p.PowerLineStatus == PowerLineStatus.Offline)

{

int batterylife = (int)(p.BatteryLifePercent \* 100);

if (batterylife < 18) // 18 Min. or less left on battery, pause movie

{

runOnce = true;

return true;

}

}

else if (p.PowerLineStatus == PowerLineStatus.Online)

{

int batterylife = (int)(p.BatteryLifePercent \* 100);

if (batterylife == 100 && Fully\_Charge == 2) // Skip it twice to ensure it is fully charged

{

if (PauseMovie()) // If watching a movie pause for 2.5 secs To alert me its ok to unplug

{

System.Threading.Thread.Sleep(2500);

PauseMovie();

}

Fully\_Charge += 1;

}

else if (batterylife == 100)

Fully\_Charge += 1;

}

return false;

}

private bool PauseMovie()

{

Process[] proc = new Process[10];

try

{

proc = Process.GetProcessesByName("vlc");

IntPtr h = proc[0].MainWindowHandle;

SetForegroundWindow(h);

SendKeys.SendWait(" ");

return true;

}

catch { }

try

{

proc = Process.GetProcessesByName("popcorn");

IntPtr h = proc[0].MainWindowHandle;

SetForegroundWindow(h);

SendKeys.SendWait(" ");

return true;

}

catch { }

return false;

}

private void timer\_Elapsed(object sender, System.Timers.ElapsedEventArgs e)

{

timer.Enabled = false;

if (BatteryIsLow())

{

timer.Enabled = false;

if (!PauseMovie())

MessageBox.Show("The computer is going to die! Plug it in to save it.", "AutoVPN", MessageBoxButtons.OK, MessageBoxIcon.Stop, MessageBoxDefaultButton.Button1, MessageBoxOptions.DefaultDesktopOnly);

}

else

timer.Enabled = true;

}

}

public class VPN

{

[DllImport("user32.dll")]

static extern bool SetForegroundWindow(IntPtr hWnd);

[DllImport("user32.dll", CharSet = CharSet.Auto, ExactSpelling = true)]

public static extern IntPtr SetFocus(HandleRef hWnd);

[DllImport("user32.dll")]

static extern bool PostMessage(IntPtr hWnd, UInt32 Msg, int wParam, int lParam);

System.Timers.Timer timer = new System.Timers.Timer(30000);

public VPN()

{

// Use this because isnetworkavailable wont fire if internet is still on and just drop VPN

NetworkChange.NetworkAddressChanged += new NetworkAddressChangedEventHandler(AddressChangedCallback);

timer.Elapsed += new System.Timers.ElapsedEventHandler(timer\_Elapsed);

isConnecting = false;

}

public void Start()

{

if (!isConnected())

Connect();

}

void timer\_Elapsed(object sender, System.Timers.ElapsedEventArgs e)

{

timer.Enabled = false;

if (!isConnected())

Connect();

else

timer.Enabled = true;

}

void AddressChangedCallback(object sender, EventArgs e)

{

if (!isConnected())

Connect();

}

public void Connect()

{

if (NetworkInterface.GetIsNetworkAvailable() && !isConnected() && !this.isConnecting)

{

isConnecting = true;

using (RasPhoneBook pb = new RasPhoneBook())

{

pb.Open(); // Using obsolete method, because the suggested method doesn't work

RasEntryCollection entries = pb.Entries;

RasDialer rd = new RasDialer();

rd.EntryName = "US TX"; // The name of my specific VPN connection

rd.PhoneBookPath = pb.Path;

rd.Credentials = new NetworkCredential("######", "######");

System.Threading.Thread.Sleep(20000);

while (!isConnected())

{

if (!rd.IsBusy) // Still tries connecting if a connection is already in progress.

{

try

{

rd.Dial();

System.Threading.Thread.Sleep(20000); // Increase time if still seeing warning about already connecting

// Hacky way of getting around the warning message

// of multiple connection attempts

// VPN is potentially unconnected for 20 secs

}

catch (Exception e)

{

// Don't break the program just cause its having trouble connecting

// If a warning appears try to close it

CloseWarning();

}

}

}

isConnecting = false;

timer.Enabled = true;

/\* TODO: Allow the user to enter this information so its more universal

Manually add L2TP with preshared key

string l2tpConName = "US-TX";

string ip = "";

string username = "#######";

string password = "#######";

string sharedKey = "mysafety";

RasEntry entryL2TP = RasEntry.CreateVpnEntry(l2tpConName, ip, RasVpnStrategy.L2tpOnly, RasDevice.GetDeviceByName("(L2TP)", RasDeviceType.Vpn));

pb.Entries.Add(entryL2TP);

entryL2TP.UpdateCredentials(new NetworkCredential(username, password));

entryL2TP.Update();

entryL2TP.Options.UsePreSharedKey = true;

entryL2TP.UpdateCredentials(RasPreSharedKey.Client, sharedKey);

entryL2TP.Update();\*/

}

}

else

Connect();

}

// When I try to reconnect while a connection is in progress a warning will appear (VERY annoying)

// I've tried to figure out how to bring the window to the front and hit ok/cancel with no luck 100% of the time

// Checking if RAS is busy doesn't work

private void CloseWarning()

{

Process[] proc = new Process[10];

proc = Process.GetProcessesByName("rasautou");

if (proc.Length > 0)

{

IntPtr h = proc[0].MainWindowHandle;

// Similar to pressing ALT+TAB to bring window forward and press space for ok/cancel

uint WM\_SYSCOMMAND = 0x0112;

int SC\_PREVWINDOW = 0xF050;

PostMessage(proc[0].MainWindowHandle, WM\_SYSCOMMAND, SC\_PREVWINDOW, 0);

SetForegroundWindow(h);

SetFocus(new HandleRef(null, proc[0].MainWindowHandle));

SendKeys.SendWait(" ");

}

}

public bool isConnected()

{

foreach (NetworkInterface nic in NetworkInterface.GetAllNetworkInterfaces())

{

if (nic.Name == "VPN US-TX")

return true;

}

return false;

}

protected bool isConnecting { get; private set; }

}

}

}