**TRƯỜNG ĐẠI HỌC ĐÀ LẠT**

**KHOA CÔNG NGHỆ THÔNG TIN**

**A picture containing room

Description automatically generated**

**BÁO CÁO MÔN LẬP TRÌNH MẠNG**

**ĐỀ TÀI: XÂY DỰNG ỨNG DỤNG BẮT GÓI TIN**

GVHD: Thầy Trần Ngô Như Khánh

Sinh viên thực hiện: La Quốc Thắng,

Nguyễn Thị Linh,

Nguyễn Thị Bích Ngọc

**Đà Lạt, tháng 06 năm 2020**

# DANH SÁCH THÀNH VIÊN

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Số TT** | **Họ Và Tên** | | **MSSV** | **Lớp** |
| 1 | Nguyễn Thị Bích | Ngọc | 1610171 | CTK40 |
| 2 | La Quốc | Thắng | 1610207 | CTK40 |
| 3 | Nguyễn Thị | Linh | 1610156 | CTK40 |

# BẢNG PHÂN CÔNG CÔNG VIỆC

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **STT** | **Ngày phân công thực hiện** | **Nội dung công việc** | | | **Ghi chú** |
| La Quốc Thắng | Nguyễn Thị Linh | Nguyễn Thị Bích Ngọc |  |
| 1 | 18/04/2020 | Thảo luận chọn đề tài, thống nhất đặt tên ứng dụng là Sniffer | | | Đề tài tự chọn 3: Ứng dụng bắt gói tin |
| 2 | 20/04/2020 | Tìm hiểu thư viện Pcap.Net,  <https://github.com/PcapDotNet/Pcap.Net> | | | Nghiên cứu một số thư viện có sẵn cho C# như Pcap.Net và SharpPcap |
| 3 | 23/04/2020 | Tạo file báo cáo theo mẫu | | |  |
| 4 | 27/04/2020 | Xây dựng GUI cho ứng dụng |  |  | Nêu rõ cho mọi người hiểu được cách code chức năng cho chương trình theo hướng sự kiện |
| 5 | 11/05/2020 | Code chức năng chương trình | | | Ghi một số nội dung vào file báo cáo |
| 6 | 24/05/2020 |  | Code chức năng chương trình | |  |
| 7 | 31/05/2020 |  | Code chức năng chương trình | |  |
| 8 | 07/06/2020 | Cải thiện và sửa một số lỗi |  |  | Hoàn thiện chức năng của chương trình |
| 9 |  | Hoàn thiện báo cáo đề tài | | |  |

# MÔ HÌNH HOẠT ĐỘNG CỦA CHƯƠNG TRÌNH

Chương trình này áp dụng mô hình 3 tầng nhằm chia tách công việc thành các project khác nhau, đang rất thông dụng hiện nay.

Hình 1. Mô hình hoạt động

Trong đó:

* *Pcap.Net* là thư viện mã nguồn mở, có thể tham khảo mã nguồn tại địa chỉ <https://github.com/PcapDotNet/Pcap.Net>. Pcap.Net được xây dựng thành Nuget Package và quản lý bởi nuget.org. Chi tiết về thư việc này, xem trong phần sau của báo cáo.
* *Business* là tầng nghiệp vụ, thuộc loại library project, sử dụng thư viện Pcap.Net nhằm xây dựng các phương thức cho tầng Application sử dụng.
* *Application* là tầng ứng dụng, có thể là Windows Forms (WinForms) hoặc Windows Presentation Foundation (WPF) hoặc Mobile… sử dụng các phương thức được định nghĩa trong tầng Business mà không cần sử dụng trực tiếp Pcap.Net.

Lợi ích của mô hình này:

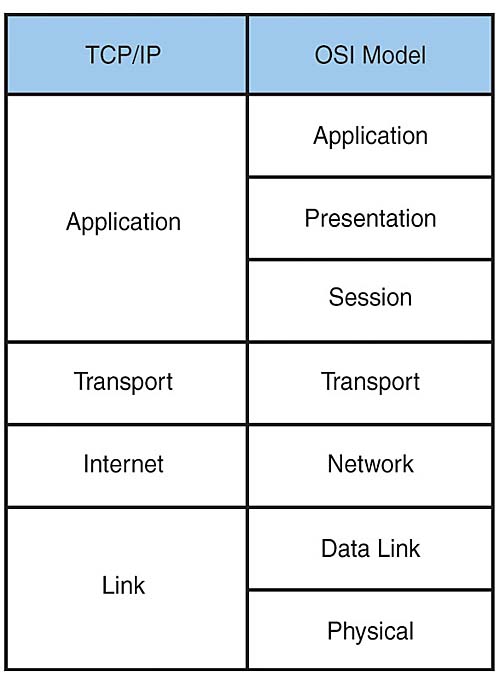
* Phát triển các thành phần/module một cách độc lập.
* Dễ bảo trì phần mềm, không ảnh hưởng đến các lớp khác thuộc tầng khác.
* Tầng cao hơn chỉ cần sử dụng các phương thức của tầng thấp hơn mà không cần quan tâm đến thực hiện những gì.
* Dễ phát triển, nâng cấp và tái sử dụng cho sau này.
* Dễ phân chia công việc cho các thành viên.

# THƯ VIỆN SỬ DỤNG

## WinPcap

### Giới thiệu

Nhiều năm nay, WinPcap được biết đến như là một công cụ chuẩn công nghiệp cho việc truy cập tầng liên kết của mạng trong mỗi trường Windows, cho phép các ứng dụng có thể bắt và truyền gói tin bỏ qua ngăn xếp giao thức.

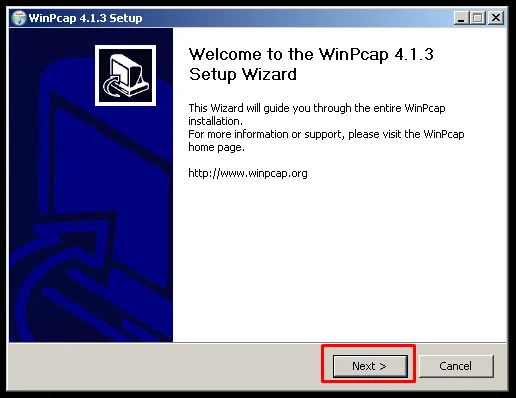


Hình . Tầng Liên kết trong mô hình TCP/IP

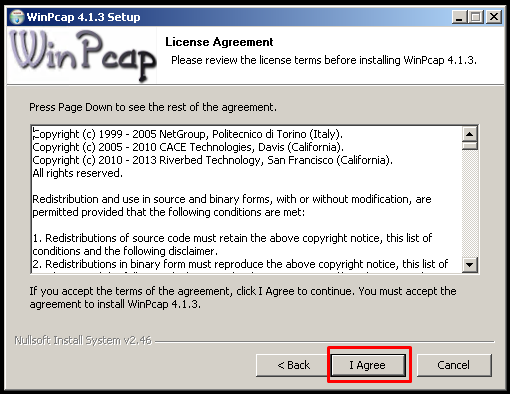
WinPcap bao gồm một trình điều khiển mở rộng khả năng của hệ điều hành trong truy cập mạng ở tầng thấp (tầng liên kết) và một thư viện để có thể sử dụng dễ dàng hơn.

### Cài đặt

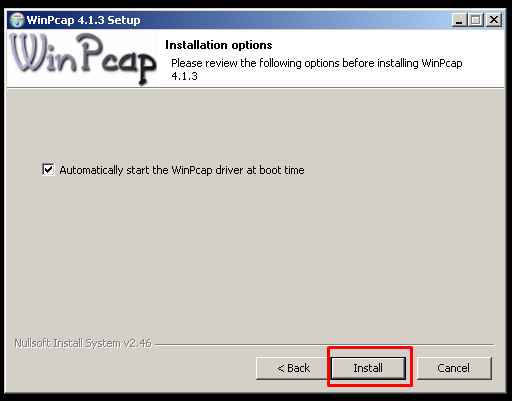
Để sử dụng được Pcap.Net thì trước hết phải tải về WinPcap tại địa chỉ sau <https://www.winpcap.org/install/default.htm> và cài đặt nó.



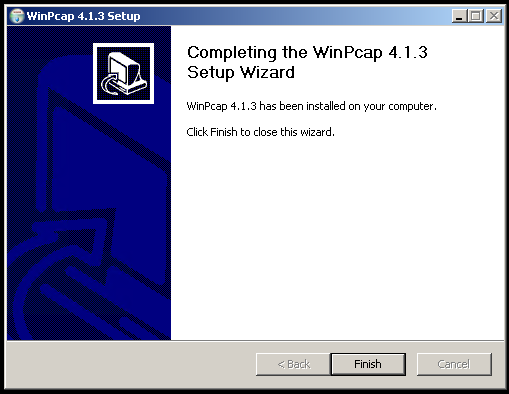
Hình . Giao diện cài đặt WinPcap



Hình . Đồng ý các điều khoản của chương trình



Hình . Chấp nhận WinPcap tự động khởi động lúc bật máy



Hình . WinPCap cài đặt thành công

## Pcap.Net

### Giới thiệu

Pcap.net thực chất là một lớp bao bọc WinPcap vốn được viết bằng ngôn ngữ C/C++, sau được viết ra nhằm hỗ trợ trong lập trình .NET. Pcap.Net có hầu hết các tính năng của WinPcap và bao gồm một framework phân tích các gói tin.

Các tính năng của Pcap.Net – gói bao bọc .Net cho WinPcap:

* Lấy danh sách các thiết bị mạng (interface hay còn gọi là adapter) ở máy tính.
* Đọc các gói tin trực tiếp từ thiết bị mạng hoặc từ tập tin.
* Nhận thống kê toàn bộ gói tin được bắt hoặc một số gói tin.
* Hỗ trợ nhiều phương thức khác nhau.
* Áp dụng bộ lọc Berkeley Packet[[1]](#footnote-2).
* Gửi các gói tin trực tiếp đến thiết bị mạng.
* Lưu các gói tin vào file.
* Sử dụng Enumerables (và LINQ) để nhận các gói tin.

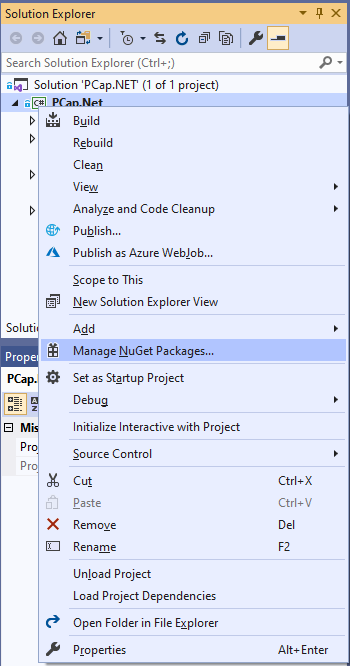
Pcap.Net có thể phân tích các gói tin sau:

* Ethernet + VLAN tagging (802.1Q)
* ARP
* IPv4
* IPv6
* GRE
* ICMP
* IGMP
* UDP
* TCP
* DNS
* HTTP

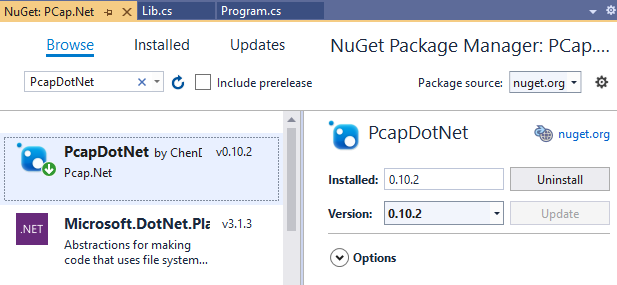
### Sử dụng Pcap.Net trong Visual Studio

Để project có thể sử dụng Pcap.Net, trong Microsoft Visual Studio, ta thực hiện các bước sau:

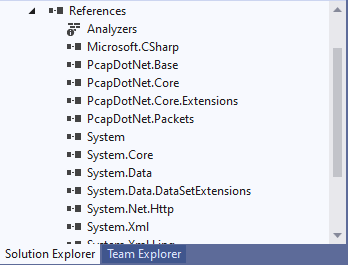
* Cài đặt phiên bản .NET Framework 4.5 trở lên;
* Tạo một dự án Console, Windows Form hay Class Library;
* Thêm thư viện PcapDotNet vào project;



Hình . Nhấp chuột phải vào project, chọn Manage Nuget Packages…



Hình . Vào Browse, gõ PcapDotNet và nhấn Install (nằm bên phải)



Hình . Các thư viện tham chiếu của PcapDotNet sẽ xuất hiện trong project

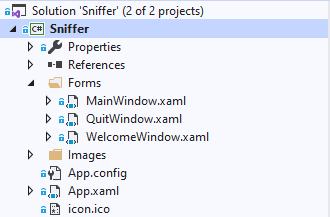
Ngoài ra, để tham khảo một số cách sử dụng Pcap.Net, ta có thể truy cập vào trang hướng dẫn này <https://github.com/PcapDotNet/Pcap.Net/wiki/Pcap.Net-Tutorial>.

# MÔ TẢ CHƯƠNG TRÌNH

## Tầng Application

### Giao diện

Tầng Application chứa dự án WPF có nhiệm vụ phát triển ứng dụng có giao diện đồ họa cho phép người dùng thao tác trực tiếp trên đó.

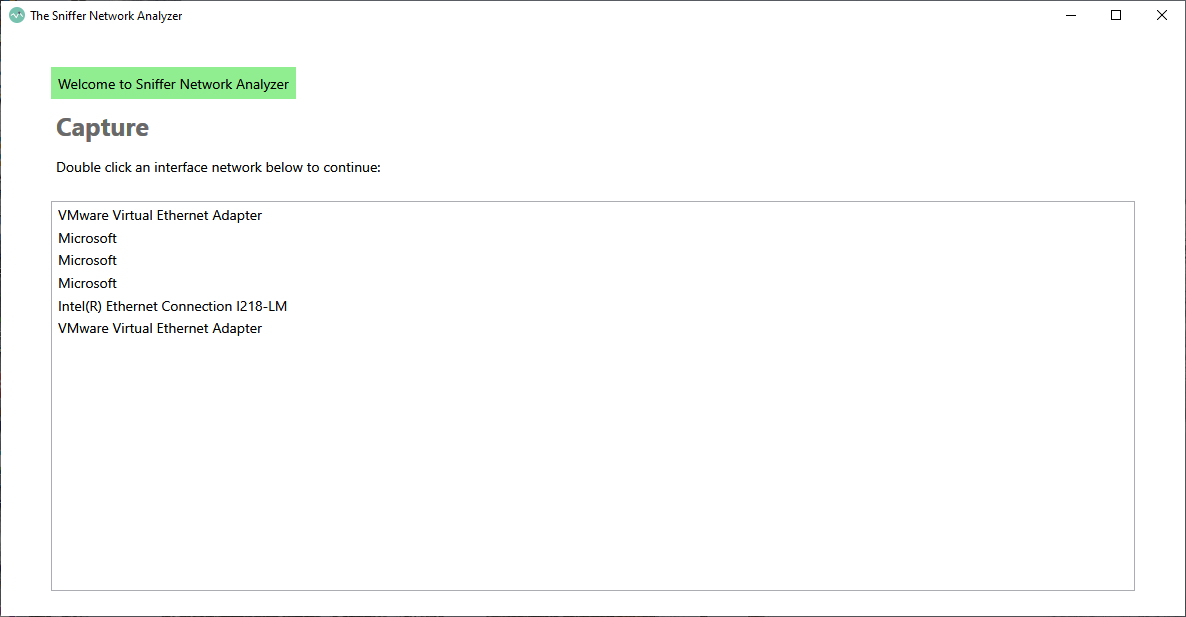


Hình . Bên trong tầng Application

Thư mục Forms tập hợp các form (các cửa sổ) có trong ứng dụng. Phần mở rộng của form trong WPF là .xaml. XAML là chữ viết tắt của *eXtensible Application Markup Language*, là thế hệ form tiếp theo của Microsoft nhằm thay thế Windows Forms truyền thống, dựa trên XML để mô tả giao diện người dùng đồ họa (GUI).

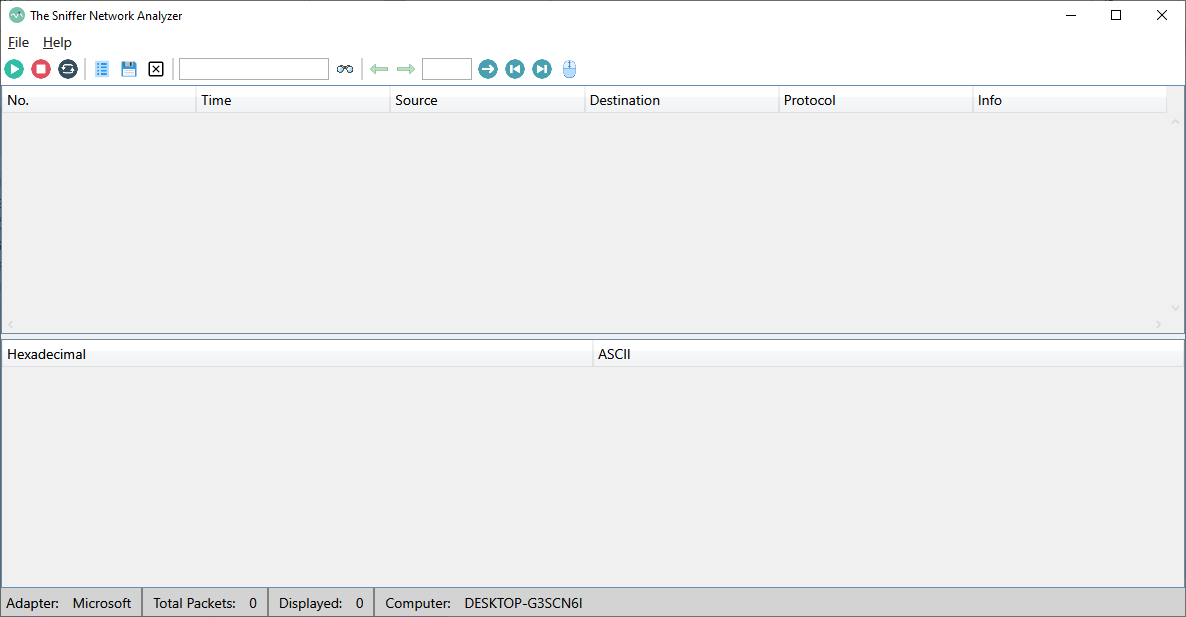
Các form có trong chương trình:

* *WelcomeWindow*: Form xuất hiện lần đầu khi mở chương trình, cho phép người dùng chọn một card mạng (interface) từ danh sách. Tất cả các interface có trong máy sẽ được hiển thị ở đây. Tuy nhiên, một số interface chỉ lấy được tên chung là “Microsoft”.



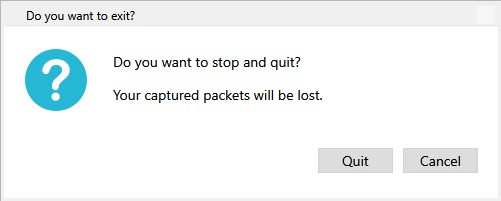
Hình . Form WelcomeWindow

* *MainWindow*: Sau khi người dùng chọn một interface mạng ở WelcomeWindow, một form mới mở ra chứa các chức năng chính để người dùng có thể thao tác với chương trình.



Hình . Form MainWindow

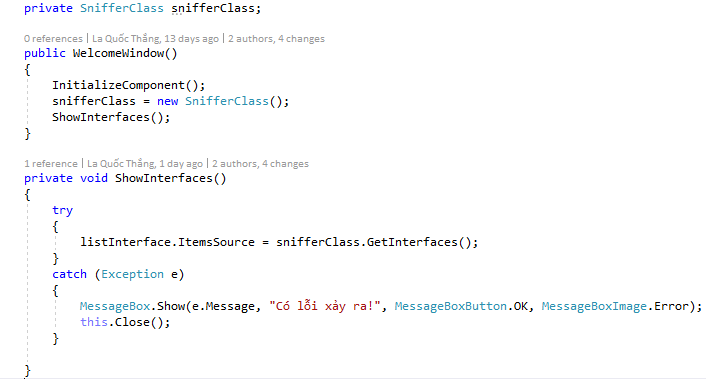
* *QuitWindow*: Sau khi thao tác xong ở form MainWindow, nếu người dùng nhấn nút **X** để thoát, một form mới xuất hiện, yêu cầu xác thực của người dùng có muốn thoát hay không.



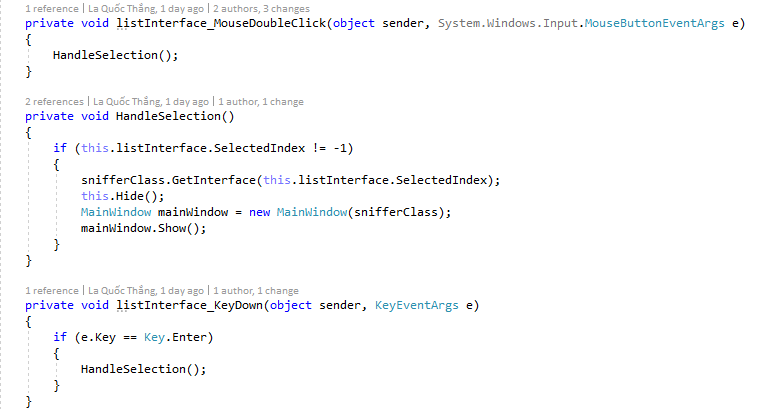
Hình . Form QuitWindow

### Xử lý sự kiện

1. Form WelcomeWindow



Hình . Lấy và hiển thị danh sách interface ngay sau khi form xuất hiện

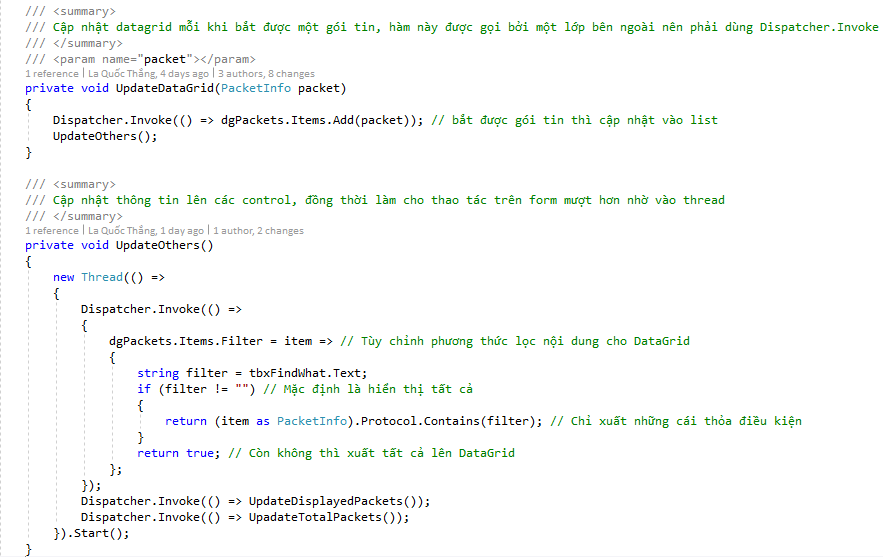


Hình . Sự kiện nhấn đúp chuột hoặc nhấn phím Enter sẽ chọn interface

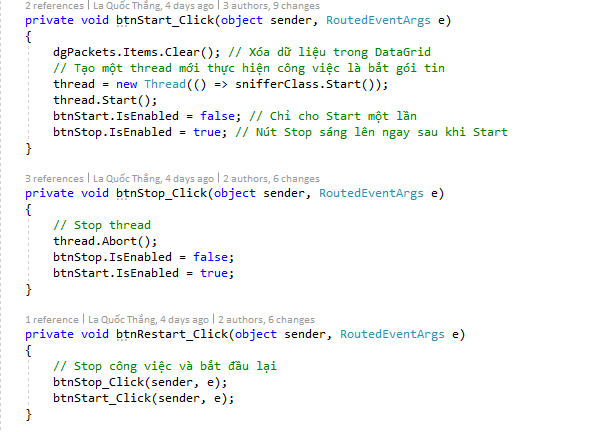
1. Form MainWindow



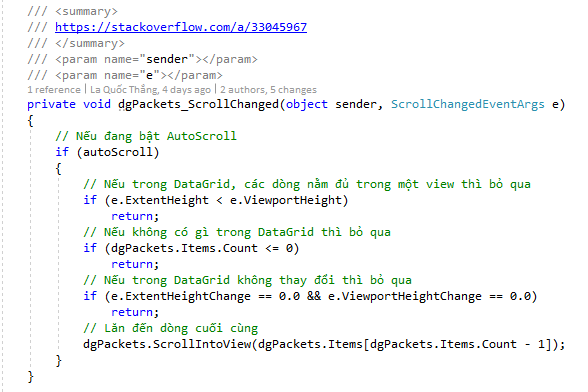
Hình . Thực hiện một số công việc ngay sau khi form hiện trên màn hình



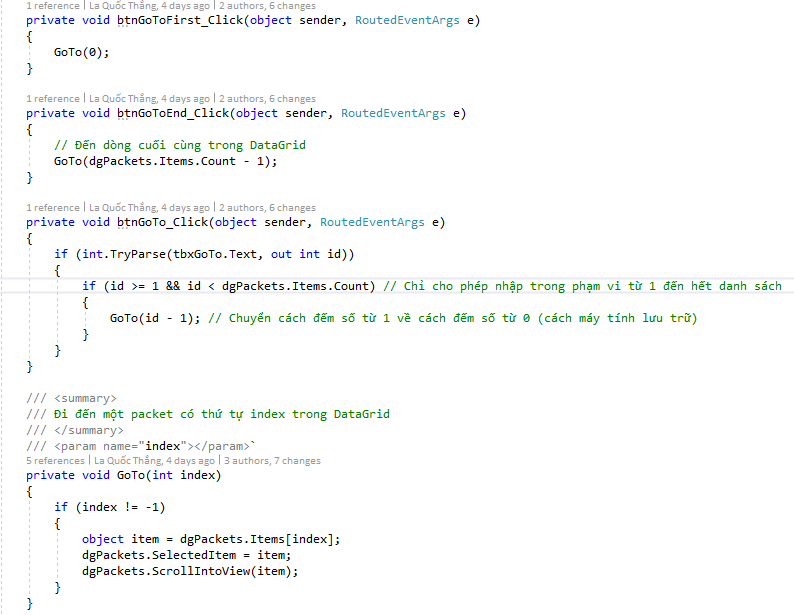
Hình . Các thao tác xảy ra khi một gói tin bắt được



Hình . Chức năng Start, Stop và Restart bắt gói tin



Hình . Chức năng tự động lăn xuống dòng cuối cùng trong DataGrid



Hình . Đi đến gói tin nằm đầu, cuối và bất kỳ trong DataGrid

1. Form QuitWindow

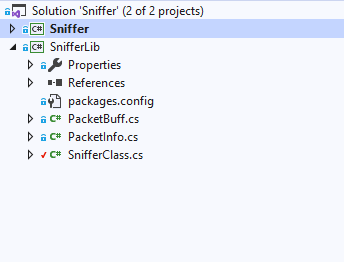


Hình . Sự kiện bên trong QuitWindow

## Tầng Business

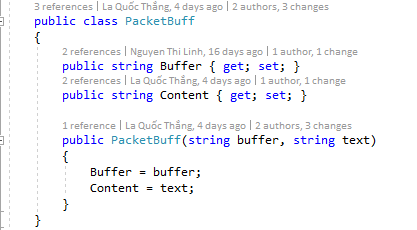
Tầng Business trong đề tài có tên là SnifferLib, nó thuộc dạng Class Library (thư viện lớp). Bên cạnh các project có dạng Windows Application, Console Application thì Class Library tập hợp các mẫu (template) trong lập trình hướng đối tượng. Class Library chứa các mã được định nghĩa trước, các project khác có thể tái sử dụng các mã này trong project của mình, sử dụng như một tham chiếu (reference). Bên trong Class Library có thể chứa mã dành cho xử lý các thành phần giao diện (nút, biểu tượng, cửa sổ, datagrid,…) và phi giao diện (các chức năng khác). Mọi ngôn ngữ lập trình hướng đối tượng đều có sẵn các thư viện như thế này, được đi kèm trong ngôn ngữ đó và người lập trình có thể định nghĩa riêng cho mình.

Khi build chương trình thì các dự án dạng Class Library này sẽ trở thành các tập tin .dll ( Dynamic Link Library).



Hình . Tầng Business của ứng dụng này chính là SnifferLib

### PacketBuff



Hình . Lớp PacketBuff

Lớp PacketBuff có hai thuộc tính là Buffer chứa nội dung của gói tin ở dạng thập lục phân (hexadecimal), còn Content là thông tin đã được chuyển thành chuỗi Unicode từ Buffer.

### PacketInfo

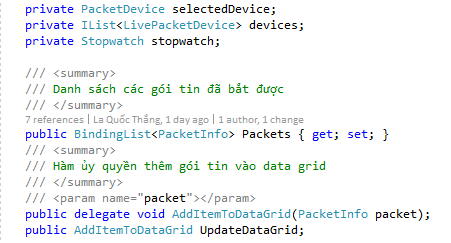


Hình . Lớp PacketInfo chứa một số thuộc tính cần thiết để hiển thị trên DataGrid

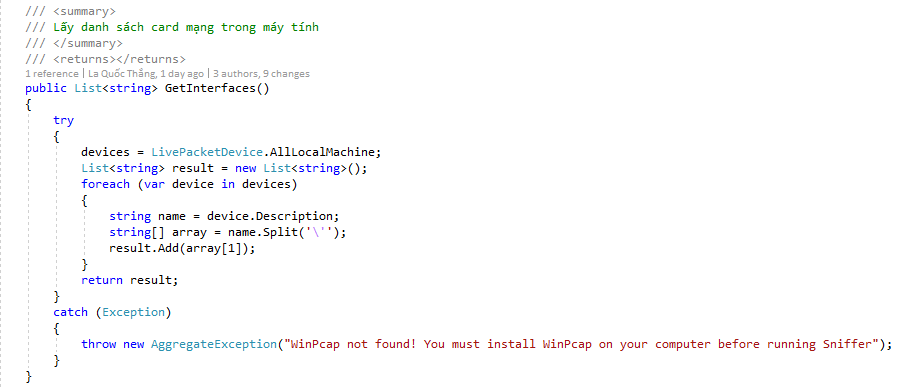
Bên trong lớp PacketInfo có định nghĩa một số thuộc tính của gói tin, trong đó có sử dụng PacketBuffer. Vì bên trong gói tin có chứa nhiều thông tin, nên trước khi đưa vào đây thì các thông tin đó phải được xử lý. Công việc xử lý là của lớp SnifferClass.

### SnifferClass

SnifferClass là lớp trọng yếu của tầng Business vì nó trực tiếp xử lý các gói tin bắt được thông qua một số hàm từ thư viện Pcap.Net. Như vậy, SnifferClass có chức năng liên kết giữa tầng Pcap.Net với tầng Application, giúp đảm bảo cho các form ở tầng Ứng dụng hoạt động được.



Hình . Một số trường (fields) và thuộc tính (properties) của SnifferClass



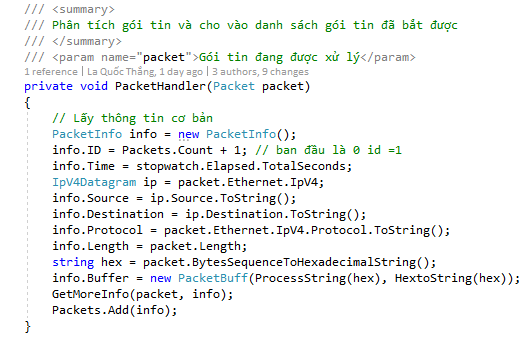
Hình . Phương thức lấy các interface có trong máy tính



Hình . Phương thức khởi động chức năng bắt gói tin

Open() mở một adapter ghi nhận lại lưu lượng đang truyền trong mạng với các tham số:

* **snapshotLength**: Chỉ định phần của gói sẽ được bắt lấy. Ở một số hệ điều hành, trình điều khiển gói tin có thể được cấu hình để bắt một phần gói tin, làm giảm bộ nhớ dữ liệu của ứng dụng, cải thiện hiệu suất của phương thức. Tuy nhiên, giá trị 65536 được sử dụng trong đây là kích thước tối đa của gói tin có thể có, tương đương 64KB. Điều này đảm bảo sẽ nhận được toàn bộ gói tin.
* **attributes**: Ở chế độ bình thường, adapter chỉ nhận các gói tin từ mạng được gửi đến cho nó, các gói tin trao đổi khác sẽ bị bỏ qua. Còn trong chế độ ngẫu nhiên (promiscuous), adapter sẽ bắt tất cả các gói tin cho dù nó có dự định gửi đến chính thiết bị đó hay không.
* **readTimeOut**: Thời gian chờ để đọc, tính bằng mili-giây. Mỗi lần adapter đọc, nó sẽ luôn trả về cái gì đó sau mỗi readTimeOut, dù có hay không gói tin. Nếu đặt giá trị 0, adapter sẽ không trả về nếu không có gói tin đến, -1 nghĩa là luôn trả về ngay lập tức.



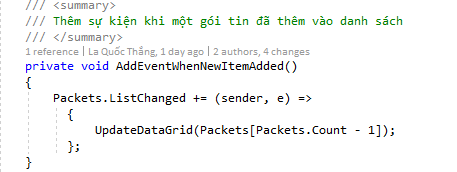
Hình . Phân tích gói tin đã bắt được, chuyển nó thành PacketInfo, thêm vào danh sách

Để chi tiết hơn thông tin của gói tin, ngoài các thông tin như IP nguồn, IP đích, độ dài gói tin, giao thức sử dụng thì thông tin mô tả cho nó cũng khá là quan trọng. Nếu gói tin là ARP thì cần phải thể hiện thông tin là nó đang đi tìm địa chỉ MAC của một máy có IP này, chẳng hạn vậy.



Hình . Xử lý riêng chờ từ gói tin thuộc các giao thức khác nhau

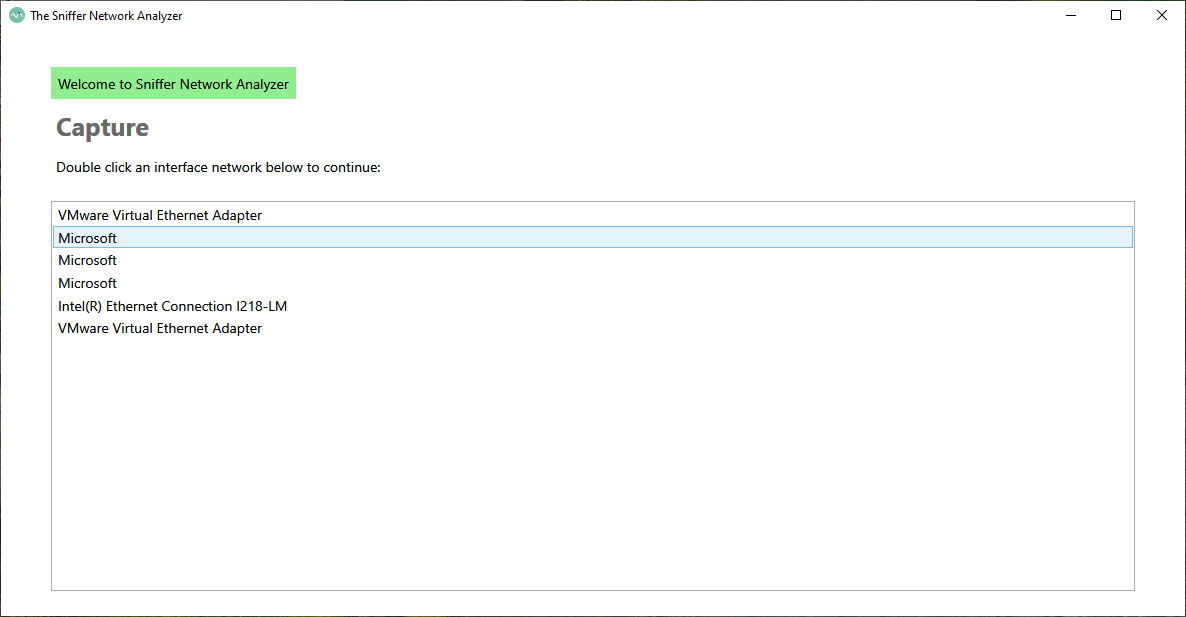
Để có thể chủ động mỗi khi bắt được một gói tin và cập nhật vào DataGrid thì cần phải định nghĩa sự kiện danh sách thay đổi (list changed event). Khi một PacketInfo thêm vào danh sách, nó sẽ gọi UpdateDataGrid (hàm mà bên form đã ủy quyền cho nó) và thêm cái PacketInfo cuối cùng của danh sách vào DataGrid.

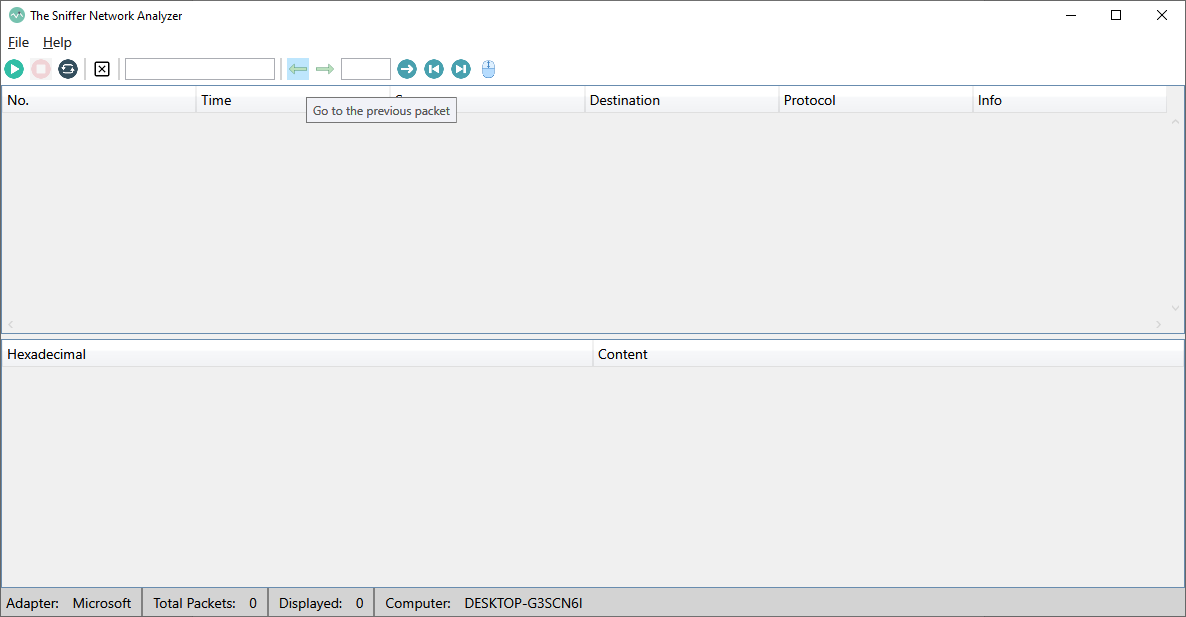


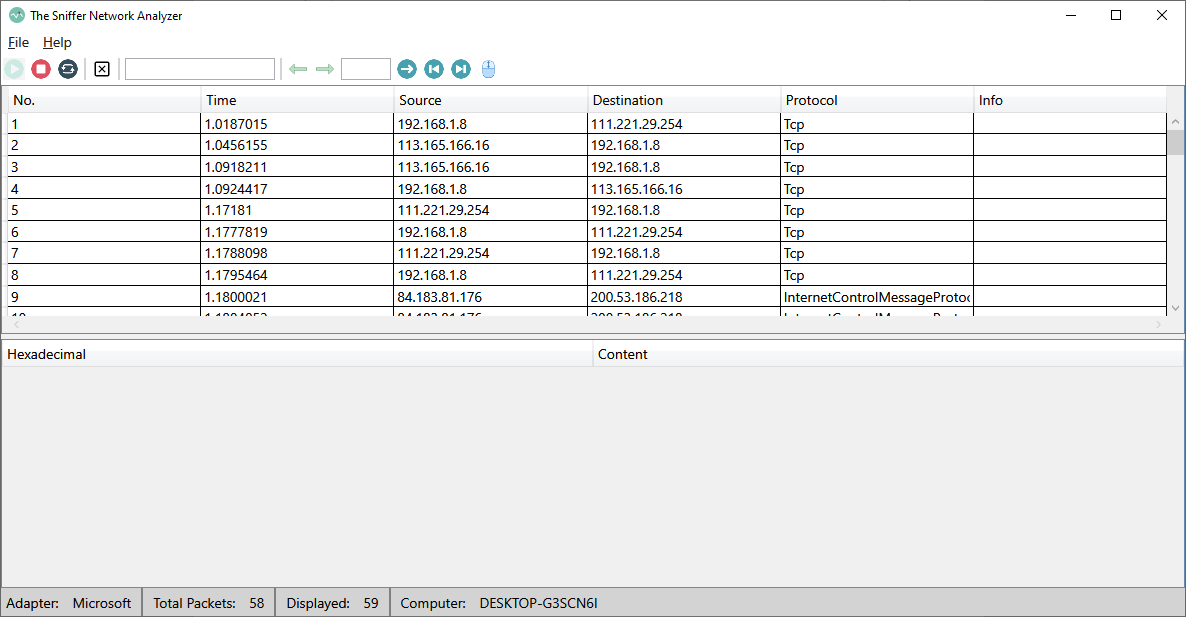
Hình . Sự kiện cập nhật PacketInfo tron DataGrid thêm một gói tin bắt được

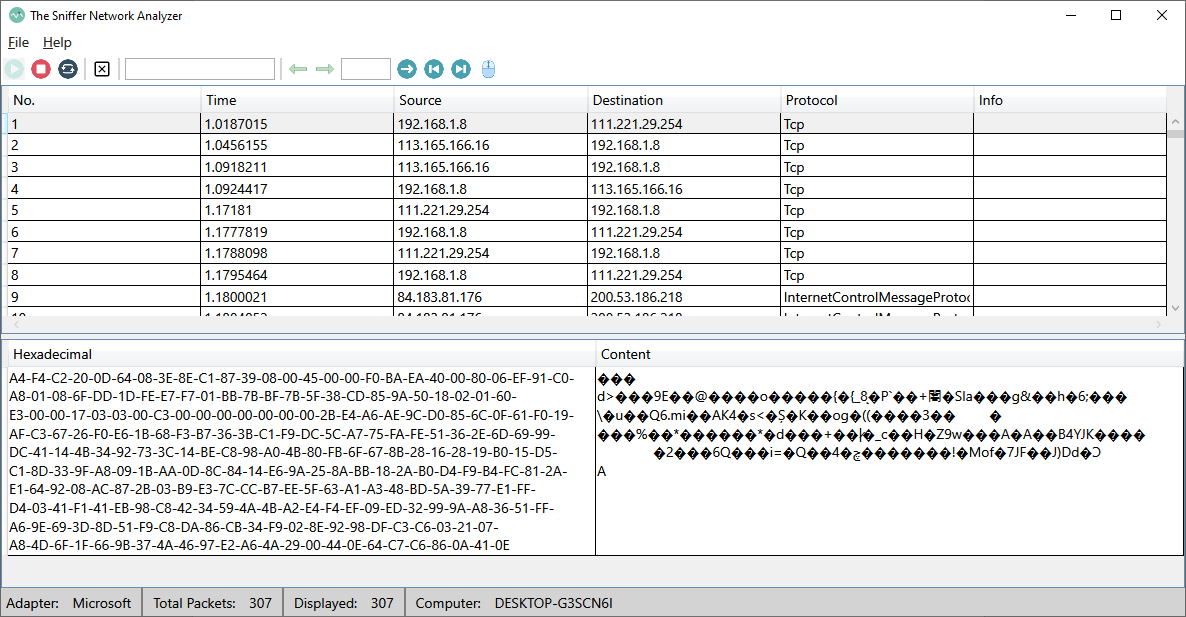
## Tầng Pcap.Net

# KẾT QUẢ THỰC HIỆN (DEMO)









# TÀI LIỆU THAM KHẢO

[1] *Pcap.Net Tutorial*. <https://github.com/PcapDotNet/Pcap.Net/wiki/Pcap.Net-Tutorial>

[2] *Trang chủ WinPcap*. <https://www.winpcap.org/>

[2] *PcapDotNet/Pcap.Net Repo*. <https://github.com/PcapDotNet/Pcap.Net>

[3] *Mô hình 3 lớp (3-layer) có gì hay?* <https://techtalk.vn/mo-hinh-3-lop-co-gi-hay.html>

[4] *XAML là gì?* <https://www.wpf-tutorial.com/vi/5/xaml/xaml-la-gi/>

[5] *XAML overview in WPF.* <https://docs.microsoft.com/en-us/dotnet/desktop-wpf/fundamentals/xaml>

[6] C*lass Library*. <https://www.techopedia.com/definition/3764/class-library#:~:text=A%20class%20library%20is%20a%20pre%2Dcoded%20object%2Doriented%20programming,as%20other%20non%2DGUI%20components.>

# MỤC LỤC

[DANH SÁCH THÀNH VIÊN 1](#_Toc42619496)

[BẢNG PHÂN CÔNG CÔNG VIỆC 2](#_Toc42619497)

[MÔ HÌNH HOẠT ĐỘNG CỦA CHƯƠNG TRÌNH 3](#_Toc42619498)

[THƯ VIỆN SỬ DỤNG 4](#_Toc42619499)

[I. WinPcap 4](#_Toc42619500)

[1. Giới thiệu 4](#_Toc42619501)

[2. Cài đặt 4](#_Toc42619502)

[II. Pcap.Net 6](#_Toc42619503)

[1. Giới thiệu 7](#_Toc42619504)

[2. Sử dụng Pcap.Net trong Visual Studio 7](#_Toc42619505)

[MÔ TẢ CHƯƠNG TRÌNH 10](#_Toc42619506)

[I. Tầng Application 10](#_Toc42619507)

[1. Giao diện 10](#_Toc42619508)

[2. Xử lý sự kiện 12](#_Toc42619509)

[II. Tầng Business 17](#_Toc42619510)

[1. PacketBuff 18](#_Toc42619511)

[2. PacketInfo 18](#_Toc42619512)

[3. SnifferClass 19](#_Toc42619513)

[III. Tầng Pcap.Net 23](#_Toc42619514)

[KẾT QUẢ THỰC HIỆN (DEMO) 24](#_Toc42619515)

[TÀI LIỆU THAM KHẢO 26](#_Toc42619516)

[MỤC LỤC 27](#_Toc42619517)

1. Tham khảo thêm tại <https://biot.com/capstats/bpf.html> [↑](#footnote-ref-2)