Project Report

NetCQ

Network Packet Capture and Query Tool

Athukorala A.S.P. E/11/024

Karunatissa K.P.D.H.M E/11/204

Piyasinghe W.D.Y.N E/11/305

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Contents

[Abstract 1](#_Toc406619296)

[Introduction 2](#_Toc406619297)

[Literature Review 3](#_Toc406619298)

[Requirement Analysis 4](#_Toc406619299)

[System Design and Architecture 5](#_Toc406619300)

[Methodology 6](#_Toc406619301)

[Implementation Details 9](#_Toc406619302)

[Unit Testing 10](#_Toc406619303)

[Results and Analysis 12](#_Toc406619304)

[Conclusion & Future Work 13](#_Toc406619305)

[References 14](#_Toc406619306)

# Abstract

Network monitoring is essential in modern day because there are heavy downloaders, Hackers and many other people who are affecting the Network. Not only that but also Network should be monitored to get a description of the general network behavior. For that Network packets contain a lot of information about Network activity. Network packet Capture and Query tools become a useful tool for Network Administrators. In this report, an Implementation of a network packet capture and query tool based on JpCap, a java Library for packet capturing, will be described. This can be used to capture packets which are transferring through a chosen network by user like Local Area Network, Wi-Fi Network and etc. Hash Map was used in this tool to efficiency Store and Query data as user needed. After Capturing packets and Saving, User can Query data as he or she wanted and monitor all the computers connected to the network.

# Introduction

In present, we are living in a networked world where we satisfy most of our needs through it like ordering something (food, electronic item) and also Share resources and communicating each other. Therefore computers are no longer an isolated machine. Network packets are units that move data through inside the network from one computer to one another where they carry all the important information from its source to final destination. Beside the actual data carries inside the packet the packet headers contain useful information about the network infrastructure and much more things like indication of the behavior of the network traffic. For example, the header information used to analyze the computers connected to and find who uses the most of the resources in the network and like identifying network attacks.

Therefore Network Packet Capture and Analysis has applications in Network Monitoring, Fault Diagnosis, Network Security, Intrusion & Threat Detection, Network Forensics and many other areas. This Network Packet Capture and Query tool was implemented to monitor the data traffic of the network like which are the computers connected to this network.

This tool based on commonly used free java library called JpCap. It get the interfaces where computer connected to Networks and Capture in other means monitor the network as user needs. Then these captured data is saved to efficiently created hash map set where user can query data quickly even in a Large data usages. It has a Graphical User Interface where users can easily use this tool and query the data as user needs.

This report served as a starting point for users to get to know about features and advantages of this tool. It describe the general information about the tool and the basic understand about the backend of this tool. Not only that but also use this as an help for anyone interested in Advanced Network monitoring by using this technology and querying the relevant data for each scenario network faces like Hacking, freezing, Overflowing and etc.

# Literature Review

There are few software where some of them are Open Source available in the field of packet capturing

**Wireshark** is a free and open-source packet analyzer. It is used for network troubleshooting, analysis, software and communications protocol development, and education. Wireshark is cross-platform, using the GTK+ widget toolkit in current releases, and Qt in the development version, to implement its user interface, and using pcap to capture packets; it runs on GNU/Linux, OS X, BSD, Solaris, some other Unix-like operating systems, and Microsoft Windows.

There is also a terminal-based (non-GUI) version called **TShark.**

**tcpdump** is a common packet analyzer that runs under the command line. It allows the user to display TCP/IP and other packets being transmitted or received over a network to which the computer is attached. Distributed under the BSD license. tcpdump is free software. It works on most Unix-like operating systems: Linux, Solaris, BSD, OS X, HP-UX and AIX among others. In those systems, tcpdump uses the libpcap library to capture packets. The port of tcpdump for Windows is called WinDump; it uses WinPcap, the Windows port of libpcap.

# Requirement Analysis

The tool can performs following tasks.

* Capture all types of IP or Ethernet packets from the Ethernet interface.
* Extract the relevant information from the data packets. The relevant information may include:
  + Source IP and Port
  + Destination IP and Port
  + Layer 4 protocol
  + Packet Length
* User can choose the needed query from the given list or can create a manual query and extract information he needed.

With comparing with Wireshark the tool has a GUI not like in Wireshark but it can query data by selecting given queries or creating manual queries.

# System Design and Architecture

A general description of the tool is shown in below.

It has a GUI and from that user can select following details.

* Select interface and give number of packets and start capturing.
* Those capturing data are shown in a table and they were saved to hashMap set.
* User can select a given query or go to manual mode.
* From manual mode user can create a own query by a filtering data.

User Interface

* Shows the capturing packet header information
* Show the number of packets captured sofar.

In Backend

* Save captured packet data info to a Hashmap set.

User interface

* User can select the Interface he can capture the packet.
* User can Start and stop the packet capturing

Capture

User Interface

* User can extract info from manual query.

In Backend

* Extract relevant info from the hashMap.

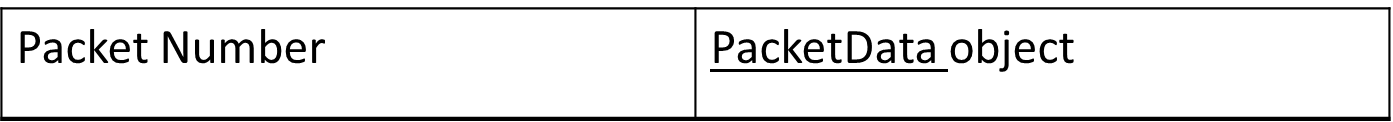
Extract Info.

# Methodology

For capturing section java JpCap Library was used and a general description of the functionalities of each of these classes is shown below in Table 1.

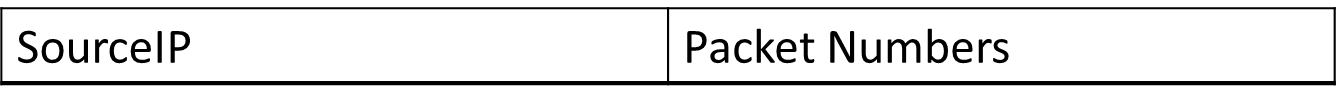
|  |  |
| --- | --- |
| Class Name | Functions |
| JpcapCaptor | In Jpcap Library this is used to get information about available interfaces and and capture packets |
| getDeviceList() | This is used for get the information about available interfaces. Use this and the information about available interfaces can be stored to a arraylist. |
| openDevice() | This is used to monitor a select interface  By using this can save information to JpcapCaptor object and get information. |
| getPacket() | If a JpcapCaptor object is created then we can use get packet to get the information of the packet. This can be used in a loop and get info about lot of packets. |

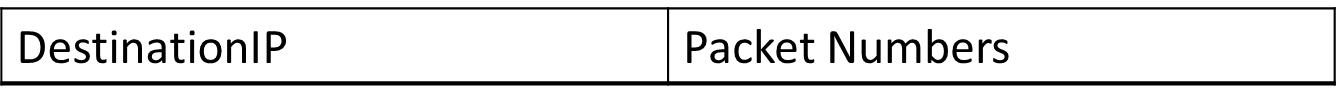
Storage structure is shown below. All packets are numbered as they caputred as packetnumber.

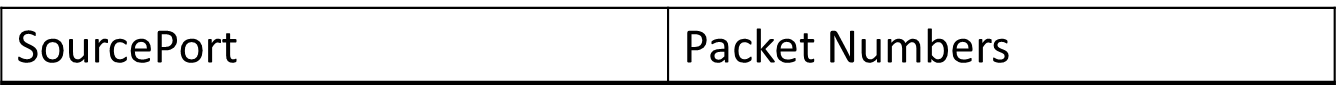
 Arralist 1(Master one)

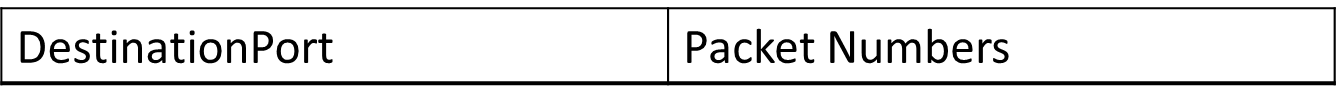
PacketData is an object which has Source IP Address, Destination IP address,

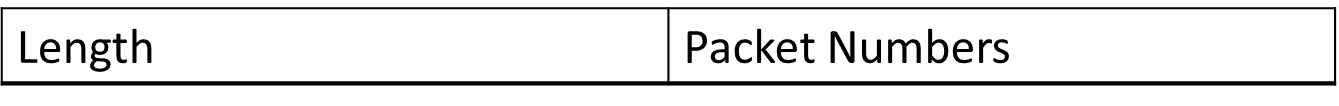
Source Port, Destination Port, packet Length and protocol.

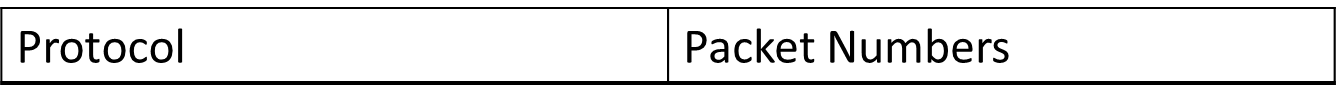
 Hash Map 2

HashMap 3

HashMap 4

HashMap 5

HashMap 6

HashMap 7

When user gives an input as Source IP address, the packet number list is taken from HashMap 2 and Using those packet numbers the info extract from the Master Arraylist.

In manual query how many key areas (Source IP , Destinaion IP and etc) are selected or given by user is selected and for each key a packet number list is taken from hashMaps.

Then a get common method is used. In here if two packet number lists are entered then a common packet number list is returned. By using this method if three packet number lists are given then run get common method twice and get the common number list for all three lists.

By this for any key combinations the common packet number list can be extracted from hashmaps and using this list the relevant packet data is extracted from master arraylist.

# Implementation Details

Iteration 1- first two weeks

* Capturing TCP and UDP packets from ethernet Interface
* Storing those header information
* Some queries for extract data.

Iteration 2 - Next two weeks

* User interface
* Switch to manual query
* Capture not only TCP and UDP other types
* Capture

# Unit Testing

Following versions are displayed in git repository in following link

<https://github.com/Nicksol/NetCQ>

V1.0.0: version 1.0

This is the first implementation of this project and in this version the capturing part was working using Jpcap library and java. This was work on command prompt user have to type the answers to select the interface where packets needes to be captured. After that packet header information was stored to HashMaps.

There was a one method that convert common packet to TCP/UDP and One method to store those data to hashMaps. HashMaps had String or Integer for each key and Integer for value.

In this If one or many packets comes from same ip address the there was a confliction and only one packet number was saved for each key in Hashmaps not the all the packets numbers for each key.

V1.0.1: version 1.1

Due to previously described problem then the storing architecture was modified that other than Major hashMap others stored integer and it changed to integer list. Not only that a method for storing packet numbers for each source and Destination IP address , source and Destination port number, packet length and packet protocol was created and by that the saving packet number list for each key was implemented.

V1.0.2: version 1.2

In this version querying the stored data was available. Previously defined number of queries was given to user and user can select one of query and give needed key and the info of the packets will be displayed. There were few limited number of queries were given.

V1.0.3: version 1.3

In this version, other than TCP and UDP packets HTTP,FTP,Telnet,DNS,NBMS and etc. Were captured. Not only that but also they were stored in HashMaps.

Version 1.4

This version had Graphical user interface and a manual query input. From graphical user interface (GUI) user can select the interface which he want to monitor and capture the packets and goto the manual query part and selet the manual query. In manual query user can select the key areas he want to search by radio buttons and select the key value from the drop down list and give command to display data by clicking button.

V1.1.0: version 1.0

All the features of version 1.4 is included. This version has a native **NetCQ** installer for windows 64 bit machines. Also benchmark testing results which have done using “YourKit Java Profiler” also included.

# Results and Analysis

Benchmark Testing results of the final product are included in the <https://github.com/Nicksol/NetCQ> from version 1.4.1 onwards.

NetCQ can capture maximum of nearly 600,000 packets in 20 minutes. After onwards crashing is occurring because allocated heap memory for JVM(Java virtual machine) is 256MB and it is not sufficient for the software.

# Conclusion & Future Work

This project successfully completed creating basic packet capture and query tool that is useful for further development or information analysis. Data storing architecture and query system can be used in any other systems because it provides the effective and correct results. The graphical user interface will help user to select interface, start capturing and in meanwhile see the number of packets captured. Therefore user can select when to stop capturing.

Manual query system will give user to create a unlimited number of self-created queries and search through packet data which will create user to easy to find needed relevant data. That will user time to search through all the packets captured.

Future developments in this project may include the offline saving those packets captured and use them for later time. Also not only capturing packets headers but also capturing the data part and save it.

# References

|  |  |  |  |
| --- | --- | --- | --- |
| <http://en.wikipedia.org/wiki/Packet_analyzer>  <http://searchnetworking.techtarget.com/definition/network-analyzer>  <http://jpcap.sourceforge.net/>  <http://jpcap.sourceforge.net/javadoc/>  <http://en.wikipedia.org/wiki/Wireshark>  http://en.wikipedia.org/wiki/Tcpdump |  |  |  |
|  |  |  |  |
|  |  |  |  |