ANALYZING OF HTTP AND HTTPS PROTOCOL AND COMPARISON USING PACKET SNIFFER TOOLS ON THREE OPERATING SYSTEMS

AZRI BIN MUKHTAR

UNIVESITI TEKNIKAL MALAYSIA MELAKA

BORANG PENGESAHAN STATUS TESIS *

JUDUL: ANALYZING OF HTTP AND HTTPS PROTOCOL AND COMPARISON USING PACKET SNIFFER TOOLS ON THREE OPERATING SYSTEMS

SESI PENGAJIAN: 2010/2011

Saya AZRI BIN MUKHTAR
(HURUF BESAR)

mengaku membenarkan tesis (PSM/ Sarjana/ Doktor Falsafah) ini disimpan di Perpustakaan Teknologi Maklumat dan Komunikasi dengan syarat-syarat kegunaan seperti berikut:

1. Tesis dan projek adalah hakmilik Universiti Teknikal Malaysia Melaka.
2. Perpustakaan Fakulti Teknologi Maklumat dan Komunikasi dibenarkan membuat salinan untuk tujuan pengajian sahaja.
3. Perpustakaan Fakulti Teknologi Maklumat dan Komunikasi dibenarkan membuat salinan tesis ini sebagai bahan pertukaran antara institusi pengajian tinggi.
4. ** Sila tandakan (/)

_____ SULIT	(Mengandungi maklumat yang berdarjah keselamatan atau kepentingan Malaysia seperti yang termaktub di dalam AKTA RAHSIA RASMI 1972)
_____ TERHAD	(Mengandungi maklumat TERHAD yang telah ditentukan oleh organisasi/ badan di mana penyelidikan dijalankan)
___/___ TIDAK TERHAD	

(TANDATANGAN PENULIS)
Alamat tetap: 181 Kg Panglima Bayu
17500 Tanah Merah,
Kelantan
Tarikh: _____

(TANDATANGAN PENYELIA)
En Ariff Bin Idris

Tarikh: 7/7/2011

CATATAN: *Tesis dimaksudkan sebagai Laporan Akhir Projek Sarjana Muda (PSM)
**Jika tesis ini SULIT atau TERHAD, sila lampirkan surat daripada pihak berkuasa.

ANALYZING OF HTTP AND HTTPS PROTOCOL AND COMPARISON USING PACKET SNIFFER TOOLS ON THREE OPERATING SYSTEMS

AZRI BIN MUKHTAR

This report is submitted in partial fulfillment of the requirements for the Bachelor of Computer Science (Networking)

FACULTY OF INFORMATION AND COMMUNICATION TECHNOLOGY
UNIVESITI TEKNIKAL MALAYSIA MELAKA
2011

DECLARATION

I hereby declare that this project report entitled

ANALYZING OF HTTP AND HTTPS PROTOCOL AND COMPARISON USING PACKET SNIFFER TOOLS ON THREE OPERATING SYSTEMS

is written by me and is my own effort and that no part has been plagiarized without citations

STUDENT : _____ DATE: 7/7/2011
(AZRI BIN MUKHTAR)

SUPERVISOR: _____ DATE: 7/7/2011
(EN. ARIFF BIN IDRIS)

DEDICATION

Thank you for supportive and motivate me to do best in my studies especially for my parents.

Thank you for guiding me in this project from start to the end especially for my supervisor

Thank you for those who helping me a lot especially my beloved lecturers and friends.

ACKNOWLEDGEMENTS

I would like to express my deepest gratitude toward my supervisor En Ariff bin Idris for his kindness, full support, understanding towards my problem and giving a lot of opinion and guidance to me through this project. I would like to express my sincere thanks and appreciation to my beloved family especially my parent for giving me support and motivation from starting of the project until it complete. I would like to thank you to my lecturer from Faculty of Information And Communication Technology (FTMK) who had help me throughout this project. Thank you for your knowledge and guidance.I would like to thank my entire course mate for being able to bear with me during my hardship accomplishes my project.

ABSTRACT

Analyze the packet that captured by packet sniffer at different operating system was developed as an analysis project. The main purpose of this project is to analyze packet that being transfer when a computer communicate with the server. This project used three operating system applications which are Ubuntu Linux and Windows Server 2008. There are four protocols being analyze in this project which are hypertext transfer protocol (HTTP), and hypertext transfer protocol secure (HTTPS). These protocols will be configured in the Ubuntu, Fedora and Window Server 2008 server. Results from the analysis can help us to select which operating system is suitable to become a server in terms of durability, the smooth transmission of data and the time taken for the server sending data to users.

ABSTRAK

Analisa paket yang ditangkap oleh *packet sniffer* pada sistem operasi yang berbeza telah dibangunkan sebagai projek analisis. Tujuan utama projek ini adalah untuk menganalisis paket yang sedang pemindahan apabila komputer berkomunikasi dengan pelayan. Projek ini menggunakan tiga aplikasi sistem operasi yang Ubuntu, Linux dan Windows Server 2008. Terdapat dua protokol yang menganalisis dalam projek ini yang *hypertext transfer protocol* (HTTP), dan *hypertext transfer protocol secure* (HTTPS). Protokol ini akan dikonfigurasi di server Ubuntu, Fedora dan Window Server 2008. Keputusan daripada analisis boleh membantu kita untuk memilih sistem operasi yang sesuai untuk menjadi pelayan dari segi ketahanan, penghantaran data yang lancar dan masa yang diambil untuk pelayan menghantar data kepada pengguna.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

1.1 Project Background

In this project, HTTP and HTTPS are chosen protocol that will be analyzed . There are three Operating Systems will be used in this project which are Ubuntu linux, Fedora linux, and Windows Server 2008 .Each server in this project will be install various web server for HTTP protocol and will also will be install SSL certificate for HTTPS . For the packet sniffer tools, it will be installed either on client side or server itself .

The reason of this project is to analyze the protocols using packet sniffer/analyzer tools and compare the result between 3 types of operating systems which are Fedora Linux, Ubuntu Linux and Windows Server 2008 on Wireless Network (WIFI) Each packet that captured will be analyze and compare especially in send and receive packet .The packet that captured are usually displayed in many types such as ASCII , HEX and EBCDIC. The result that will compared such as reveals of message protocols , data sizes , packet capturing duration time and others

It also will compared network traffic performance and bandwidth usage in different platform and also maybe using different tools. After all the information that need are gathered, it will be analyzed and compared from all the platform of Operating System.

1.2 Problem Statement

At the present day, computer and internet connection are can't be separated from people. Everyday, internet usage is increasing rapidly and it was most important communications medium that used by the people. It is us to surf the website, sending and receiving email, and many more. While using the internet , its occurred two or more computers are communicate of each other to send and receive data. The packets will carry the data and also have header and trailer that contains information for the destination computer. Apart from this, the data that contains in the packets is something different from the original data that send through the internet from the computers to other computers. Many user doesn't know that the information that their send through the internet it safe or not before reaching to the receiver. Layer 7 Protocol (Application Layer) that from OSI model are involved to make sure the data are correctly and securely sent to destination computer

1.3 Objective

The purposes of this project are as below :

i. To analyze packet of HTTP and HTTPS protocol on different server platform

ii. To compare the result of packet analyze using three servers. Using three Operating System can produce different result.

iii. To analysis the data that being captured by packet sniffing tools. The information that collect from all Operating System platform from packet sniffing tools will be slightly different.

1.4 Scope

Scope of this project are client and server that connected to a network using Wireles connection. By evaluating the process of accessing a web that use a HTPP and HTPPS, will be know the best packet send and receive between 3 Operating System servers in term of time. It also will prove, how HTTPS and SSH, will encrypt the private client information that tested by using packet sniffer.

1.5 Project Significant

The most importance from this project is to choose which Operating Sytem that reliable to deal with the protocol in term of security and also to deal with server application that make the server more stable and secured.Thus, data that captured by packet sniffer is the key to solved the puzzle in deciding to build a good server.

1.6 Expected Output

The result of this project will be able to choose suitable Operating Systems that will use to create a better server and running better services especially the web server, ssh server and mail server.The most importance is to secured the server properly and to keep services running smoothly that can shows a good performance overall.

1.7 Conclusion

This chapter mainly described summarize of this project and how it going to be done later. Hopefully the objectives of this project that will be achieved when it finished. In the next chapter, literature review and project methodology will be discussed. It will be about discussion of the technique used in a given field of study.

CHAPTER II

LITERATURE REVIEW AND PROJECT METHODOLOGY

2.1 Introduction

In this chapter we did a literature review and segment critical analysis of the published body of knowledge through summary, classification, and comparison of prior research studies, reviews of literature, and theoretical articles. We also did both summary and explaination of the complete and current state of knowledge on a limited topic journal articles.

This chapter will explain about methodology that will help to describe the detail activities in each stage of the project. And the methodology that been use commonly choose accordingly from the previous research. It will determine the requirement, performance, and behavior during the project will be held.

2.2 Literature Review

Literature review is a body of text that aims to review the critical points of current knowledge on a particular topic. This section is use to describe the detail about the project.

2.2.1 Domain

The domain in this project is about to capture and analyze packets. All the information that came cross on the network is sent by packets. The sniffer captures each packet and decodes the packet's raw data when it flow across the network.

The captured information is decoded from raw digital form into a human-readable format that permits users of the protocol analyzer so can review the exchanged information easily. From that data, we can analyze the information based on the protocols that we used when doing the packet capture.

2.2.2 Keyword

2.2.2.1 HTTP

HTTP functions as a request-response protocol in the client-server computing model. In HTTP, a web browser, for example, acts as a client, while an application running on a computer hosting a web site functions as a server. The client submits an HTTP request message to the server. The server, which stores content, or provides resources, such as HTML files, or performs other functions on behalf of the client, returns a response message to the client.

a) HTTPS

HTTPS is a combination of the Hypertext Transfer Protocol with the SSL/TLS protocol to provide encrypted communication and secure identification of a network web

server. HTTPS connections are often used for payment transactions on the World Wide Web and for sensitive transactions in corporate information systems. The main idea of HTTPS is to create a secure channel over an insecure network.

b) Packet Sniffing

Packet sniffing is a form of wire-tap applied to computer networks. It came into vogue with Ethernet, which is known as a "shared medium" network. This means that traffic on a segment passes by all hosts attached to that segment. Ethernet cards have a filter that prevents the host machine from seeing traffic addressed to other stations. Sniffing programs turn off the filter, and thus see everyones traffic.

c) Packet Sniffer Tools

These tools facilitate the capture and visualization of network traffic. Passive recording and decoding of network packets is extremely useful in network and security troubleshooting and research. These free tools accomplish network visibility to varying degrees, and each has its own unique way of presenting the packet data.

d) Operating Systems

Operating systems (OS) is software, consisting of programs and data, that runs on computers, manages computer hardware resources, and provides common services for execution of various application software. The examples of Operating systems that used widely is Microsoft Windows, Apple Mac OS X, GNU/Linux, and Unix,

2.2.3 Previous Research

a) Identifying Slow Server Response at Packet Level

First, it can be hard to dig through thousands of packets to find a solid example of a slow response. Once a slow response is isolated, identifying the root cause can also present a challenge. In this tip, we will show how to isolate a slow response from a server, filter on it, and determine if the root cause is the network or the server itself.

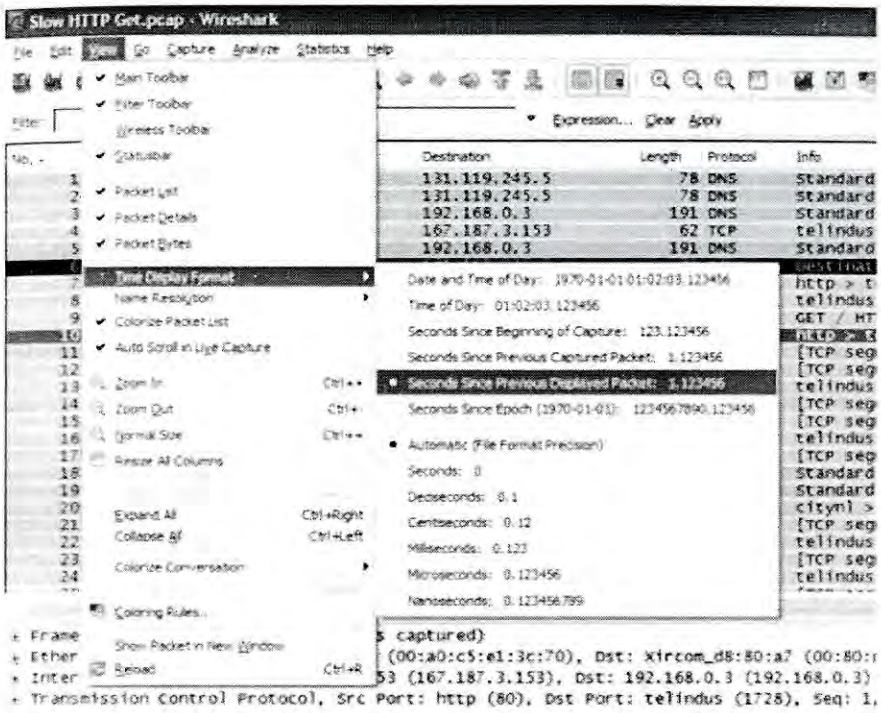
• Start at the client end

Often, when first analyzing a slow application, it is easiest to start at the client end. Although the problem may not be fully understood until a capture is taken at the server end, the trace file will be much simpler and easier to read when only one client experience is captured. Make sure that while capturing, the user is able to reproduce the performance problem.

• Look for client connecting to server

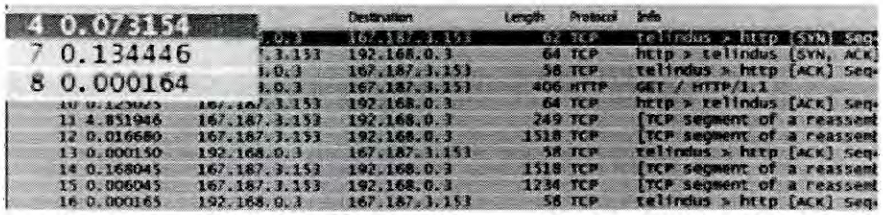
Look through the trace file to find where the client initiates a DNS query for the slow application server. It may be that they already have this server in their DNS cache, in which case the client may simply send a TCP SYN directly to the application server. If DNS is used, make sure that the DNS response time is low using the time column your packet analyzer.

-Note: When measuring application response, be sure to use a delta timer that shows the amount of time between packets. This can be accessed in Wireshark from the View drop-down menu.



If the DNS response time is quick (it should not be longer than 150ms or so), the client will next send a connection request to the application server. This will be a TCP SYN packet, the first in the TCP three-way handshake. Use a TCP Stream filter to isolate this connection (right click on any packet in the TCP connection, select TCP Stream Filter). The goal in isolating this connection is to compare the network roundtrip time to the server response time.

Once this connection is isolated, look at the delta time between the TCP SYN sent by the client and the TCP SYN-ACK sent back from the server. This can be used as a benchmark connection setup time. In the picture below, the response from the server is displayed in packet 7. It took 134msec to hear back from the server.



• Measure application response time – compare to connection setup time

Next, after the TCP connection has been established, the client will request data from the server. In the web-based application above, the client performs an HTTP GET. Use the delta time column to see how long it takes the server to respond to this request. In our example above, the server responds after 125msec with a TCP ACK. This indicates that the server received the request, but has not yet responded with actual data. After waiting 4.85 SECONDS, the server finally sends a packet with application data. After this, packets are flying by at wire-speed. Comparing 4.85 seconds to the connection setup time, 134msec, we see that the server response time is very slow.

• Server, client or network delay?

From this information, it is simple to determine where to troubleshoot next. If the server response time is significantly higher than the connection setup time, and there are no TCP retransmissions, the problem is on the server end. In the case above, the server responded to the request with an ACK, showing it received the request and was busy processing it. The network is not to blame for this delay.
If any retransmissions are observed, the network is dropping packets somewhere. The server may not be to blame for slow performance, especially if it isn't getting requests in the first place.

• If no delay is observed in this transaction ...

Move to the next request, keeping an eye on the amount of time it is taking for the server to respond to requests. Always use the connection setup time as a benchmark network roundtrip timer. This may take some time to do packet by packet, but since the capture was taken client-end, this is an excellent way to get familiar with the application behavior and look for patterns in client requests.