PASSIVE OPERATING SYSTEM FINGERPRINTING BASED ON MULTI-LAYERED SUB-SIGNATURE MATCHING SCHEME (MLSMS)

By

ASHRAF HAMDAN RASHID ALJAMMAL

Thesis submitted in fulfillment of the requirements for the degree of Doctor of Philosophy

August 2011

Acknowledgements

All the praises and thanks be to Allah, the Lord of the world for giving me the energy and the talent to finish my research; He guides me and grants me success in my life. I cannot count His bounties on me.

I would like also to deeply thank my supervisor Associate Professor Dr.Bahari Belaton for his encouragement, endless support and all the help and valuable guidance he provided to me from the beginning until the moment of coming out with this thesis.

Moreover, I would like to convey my appreciation to Universiti Sainis Malaysia (USM) for providing us such a great research environment and support. I would not forget to convey my deeply appreciation and thanks to the School of Computer Sciences for their endless help and support.

Last but not least, I would like to thank whom in my heart; my father Dr.Hamdan Aljammal whom I learned from him too much in this life. I would like to thank him for his endless and continuous encouragement and support. To the big heart, my mother, I would like to deeply thank her for the continuous prayers, inspirations and encouragement. My deepest thanks go to my dearest brothers and sisters for keeping me smiling and motivated. I would not forget my brother loai's kids Jawad and Janah whom added a smile to our life. I dedicate this humble work to all of them as without their support and understanding, this thesis would not have been completed. Thank you very much.

Ashraf Hamdan Aljammal

Table of Contents

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

ACK	Acknowledgement
BP	Back Propagation
DB	Database
DF	Do-not-Fragment Bit
FrameTL	Frame Total Length
HTTP	Hyper Text Transfer Protocol
ICMP	Internet Control Message Protocol
ICT	Information and Communication Technology
IP	Internet Protocol
IT	Information Technology
JPCAP	Java Packet Capturing
LAN	Local Area Network
MAC	Media Access Control
MLSMS	Multi-Layer Sub-Signature Matching Scheme
MSS	Maximum Segment Size
MTU	Maximum Transfer Unit
NIC	Network Interface Card
OCR	Optical Character Recognition
OS	Operating System
OSF	Operating System Fingerprinting
SigDB	Signatures Database
SNMP	Simple Network Management Protocol
SYN	Synchronize
TCP	Transmission Control Protocol

ToS	Type of Service
TTL	Time to Live
UDP	User Datagram Protocol
W/Wscale	Window scale
WINPCAP	Windows Packet Capture
WS	Window Size

PENYIDIKJARIAN SISTEM PENGENDALIAN PASIF BERDASARKAN SKEMA PEMADANAN SUB-TANDATANGAN BERBILANG LAPISAN

ABSTRAK

Rangkaian komputer merupakan dimensi yang penting dalam organisasi moden. Oleh itu, usaha memastikan rangkaian ini dapat berjalan pada prestasi puncak dianggap amat penting dalam organisasi ini. Justeru itu, usaha memastikan rangkaian adalah selamat merupakan isu yang penting untuk mencapai tahap prestasi yang baik. Namun begitu, tugas ini adalah mustahil terutamanya apabila terdapat beberapa isu persaingan yang perlu dipertimbangkan. Dalam tesis ini, kami memfokus pada sistem pengendalian (OS) yang merupakan salah satu maklumat penting untuk keselamatan dan pengurusan rangkaian. Maka maklumat yang lengkap dan tepat tentang OS yang dikendalikan di rangkaian tertentu adalah sangat penting. Teknik dan algoritma penyidikjarian OS yang sedia ada, terutamanya yang berdasarkan OS pasif tidak memberi perincian tentang OS seperti tahap tampung dan kejituan yang baik. Tujuan tesis ini adalah untuk mencadangkan dan membina satu algoritma bagi menyidikjari dengan jitunya OS yang dikendali di mesin-mesin yang dihubungkan kepada rangkaian. Tiga mekanisme dicadangkan untuk meningkatkan kejituan penyidik jarian OS. Mekanisme pertama ialah rangka kerja penyidikjarian berbilang lapisan. Ia adalah untuk mengurangkan ruang carian parameter untuk menguji dan sebagai imbalannya ini akan meningkatkan kejituan penyidikjarian. Mekanisme kedua ialah sub-tandatangan OS itu sendiri yang memainkan peranan mengumpul maklumat yang lebih menonjol atau signifikan tentang OS. Mekanisme ketiga ialah *signature DB Auto-Update* yang bertujuan untuk memastikan DB dikemaskini dengan tandatangan OS yang terkini. Algoritma yang dicadangkan ialah penyidikjarian OS pasif yang bergantung kepada lalu lintas rangkaian TCP/IP. Kami telah melaksanakan penyidikjarian OS prototaip yang dapat berfungsi penuh, dan ia telah diuji dalam pelbagai keadaan senario simulasi dan juga nyata. Keputusan awal menunjukkan bahawa kaedah kami dapat menghasilkan maklumat yang lebih jitu tentang OS berbanding dengan P0f dan NMAP.

PASSIVE OPERATING SYSTEM FINGERPRINTING BASED ON MULTI-LAYERED SUB-SIGNATURE MATCHING SCHEME

ABSTRACT

Computer networks become an important dimension of the modern organizations. Thus, keeping the computer networks running at the peak performance is considered as a crucial part for these organizations. To achieve this goal, networks must be secure since it is considered as the key issue to reach a good performance level. However, this task is next to impossible especially when there are many competing issues that need to be considered.

This thesis is focusing on Operating system (OS), which is one of the crucial information for network security and management as well. Therefore, complete and accurate information about the OS running on the machines connected to particular network is of utmost importance. Existing OS fingerprinting techniques and algorithms, particularly passive-based do not provide details of the OS such as patch level with certainty and good accuracy.

The intention of this thesis is to propose and develop an algorithm to accurately fingerprint OSs running on the machines that are connected to the network. In this thesis, three mechanisms have been proposed to improve the OS fingerprinting accuracy. The first mechanism is the multi-layer fingerprinting framework. This basically to make it possible to have unique signatures of the existing OSs and could reduce searching space of possible parameters to test and in return this will improve the fingerprinting results accuracy. The second mechanism is the OS sub-signature itself, which plays the role of capturing a more salient or significant information about OS. The third mechanism is the signature DB Auto-Update which aims to keep the DB updated with the latest OSs signatures. The proposed algorithm is passive OS fingerprinting that relies on the TCP/IP network traffic. A fully functional prototype of OS fingerprinting has been implemented, and tested over both simulated and real time in various scenarios. Results showed that our method is able to produce more accurate information about OS than P0f and NMAP.

CHAPTER ONE

INTRODUCTION

1.0 General Overview

Operating System or OS is the soul of the digital devices, it is existed in various forms, and for instance OS is embedded into the chips of the device often known as firmware. More commonly it appears in every single computer as exemplified by big names like Microsoft, IBM, Debian, RedHat, and many others. OS drives and manages the essential functions of the device; it also acts as a middleware between users, programs and the hardware. Therefore, the OS health conditions have to be maintained all the times as one of the key success factors of an organization that relies heavily on ICT assets to deliver and support their main business/operations. On the other hand, maintaining and managing ICT assets so as to keep every single machine updated with the latest security patches and running the latest features and bugs free version is a very challenging task. Different types of OSs running on different devices, some of them are very device or vendor specific hence depending totally on the vendor/supplier, while others are open for the public (like open source OS – linux variations) where changes/updates happen on non-regular or cyclic patterns. Furthermore, the current trends are merging toward cloud computing and virtualization with the promise that ICT resources are ubiquitously available for us to tap. In such new scenario there is no clear demarcation between what ICT assets are running which OSs; the task of managing this new phenomenon of cloud is very challenging.

Securing computer network is not a simple task that can be performed once. It is a continuous effort that should address various security factors and motivations, such as worm infection, botnet attack, etc. Those are motivated by several factors that vary from challenges to financial gains. These kinds of security breaches require an entry point to the victim machine, regardless of their propagation means. These entry points are mainly acquired through the operating system that is running and operating the victim machine (Foundstone.Inc, 2003).

Unfortunately, each operating system has its own gaps and weaknesses (vulnerabilities). Hence, the operating system must be updated and patched frequently through the system user or network administrator, to check for the updates and send the patches to the different machines to ensure that the network is running smoothly (Adams & Erickson, 2000).

However, ensuring the integrity, reliability and security of the operating system itself is the first step to protect them (Foundstone.Inc, 2003). Thinking the same way the black-hat users think should enable users, systems and network administrators to understand the various means the black-hat community employs to gain access to any machine (victim) operating system. According to (McClure, Scambray, & Kurtz, 2009) this will help them to guess little vulnerability that may not be patched or identified yet, where black-hat guys can initiate their infection propagation.

On the other hand, once a new update is released by the OS vendor, the network administrator needs to find out the computers that should be updated. In addition, he will be able to make sure that the network respects the company's policy regarding to the allowed operating systems to be used in the company/organization.

Another important field is the network reconnaissance that can be active or passive based (Chuvakin & Peikari, 2004). Network reconnaissance is considered as the first step the attacker takes to compromise the target network and could be used by the network administrators to collect information about the target network as well.

Since most exploitable vulnerabilities are based on the operating system, OS fingerprinting is considered as a crucial element in this field. Thus, knowing the operating system that is running on the target machine would exploit the vulnerabilities of that machine which makes it easier to the hacker to access and compromise it as well (i.e. identifying the OS of the web server is considered as an entry point to the web server and the other components of the target network). This in turn, decreases the detected attacks and increases the percentage of successful attack (Graves, 2007; Millican, 2003; Montigny-Leboeuf & Massicotte, 2005). As the active reconnaissance techniques may face some obstacles such as the firewalls and NATs, passive reconnaissance techniques could help to evade these obstacles based on the fact that it could rely on the packets which bypass the firewalls and NATs to perform the reconnaissance process , thus, it would not be affected by them (Chuvakin & Peikari, 2004).

Network vulnerability test is regularly conducted by the security specialist or network administrators to detect and evaluate the security vulnerabilities existed at any network. As the first step is to collect information about the target network; OSs running on the machines connected to this network is one of the crucial information could be collected as previously discussed (Chuvakin & Peikari, 2004; Stopforth, Vorster, & Erwin, 2007).

Furthermore, once the network administrator is tempting to find the operating system that is running on some machines. This information will be useful to build a network inventory by having accurate and up-to-date information which is crucial for the network administrator in term of decision making process.

As aforementioned, having a maintained and completed database about ICT assets in the organization is a must. Therefore, the OS fingerprinting tool should be able to deduce accurate information about the OS, having automated solutions with less intervention by the users and less intervention to the network.

To identify the operating systems accurately with details; collecting useful and unique information is needed about it. In addition, a proper processing and analysis need to be done for this information to come out with accurate and detailed results about the OS. The identification accuracy means to correctly identify the operating system which runs on some machines i.e if the target machine runs Win 7 ultimate, the OS fingerprinting tool should identify the OS as Win 7 ultimate otherwise, it will not be considered as an accurate tool. On the other hand, details about the operating system are to show the OS family name, version and the service pack (i.e Windows XP SP3).

1.1 Problem Statement

Based on the current researches in the domain of the operating system fingerprinting, existing methods in OS identification still lacking in providing the details and accurate information about OS (Li, Zhang, & Yang, 2005; Greg Taleck, 2004; Veysset, Courtary, & Heen, 2002). Crucial information about the host OS such as OS major version, OS minor version, OS service pack major version, patch level, and other are essential for many purposes as explained earlier. Active OS fingerprinting approach with its flexibility of crafting and probing the target host has some potential to address this problem. However, in the long run this method is not reliable as more security conscious organizations are tightening their ICT defenses prohibiting probed packets from reaching the target hosts. On the other hand passive based method is not affected by such security mechanism, and its stealth and network's friendly nature is quite attractive for us to investigate further on addressing the stated research problems. Two points which are quite significant to be addressed in this research: first is the selection of the reliable parameters, and the second is the matching procedure to deduce the identity of OS.

The existing operating system fingerprinting techniques relies on a certain parameters such as TTL, WS, MSS, FrameTL, DF, timestamp, ToS, and TCPoptions (Berrueta, 2003; Höfler, 2004; Jiao & Wu, 2006; Li, et al., 2005). However, some of these parameters are not reliable for the process of operating system fingerprinting, because, some of these parameters might have the same values even for different operating systems, versions and patches. Therefore, the usage of these parameters might lead to uncertain and inaccurate results.

In this thesis a passive OS fingerprinting framework with a multi-layer sub-signature matching scheme is proposed. The ingenuity of the proposed framework is based on the segregation of the OS parameters matching into three hierarchical layers; where in each layer a specific OS signature database is prepared and compared. While matching process of the existing operating system fingerprinting tools depends on a single layer in fingerprinting process. In other words, it uses the whole parameters at once (not hierarchical), to be used in the matching process. This will include some parameters which should not be combined with some other parameters at the same level of matching. This in turn will affect the fingerprinting tool/method to come out with an unreliable and hard-to-guess operating system identification results.

Based on the earlier discussions and our researches in the domain of the passive operating system fingerprinting, we have been prompted to answer the following questions:

1. Is it possible to further improve the accuracy and provide detail information about operating system of a remote host with a high accuracy?
2. Is it possible to achieve the above improvements using passive OS fingerprinting method and how?

1.2 Research Goals and Objectives

The main goal of this thesis is to propose and design a new Passive Operating System fingerprinting method based on Multi-Layered Sub-Signature Matching Scheme that is capable of fingerprinting the operating system with higher accuracy.

Therefore the objectives of this thesis are as follow:

1. To propose a new multi-layer OS fingerprinting framework, this is basically to reduce searching space of possible parameters to test and in return this will improve the fingerprinting accuracy.

2. To propose a unique OS sub-signature for each OS family, version and sub-version.
3. To propose a simple signature DB Auto-Update technique which aims to keep the DB updated with the latest OS signatures.

1.3 Research Scope and limitations

The research scope is the passive operating system fingerprinting based on TCP/IP SYN packets. Therefore, this approach is limited to the information availability via the network. In other words, sometimes it takes long time to capture the needed information about some machines. Furthermore, the machine is considered absent unless its traffic appears on the network.

Another limitation is the packet parameters which the proposed method depends on for the process of operating system fingerprinting and that are WS, MSS, FrameTL and TCPoptions including W. But in case these parameters have been changed or modified with some ratio, our method will not be able to identify the correct operating system. Therefore, as a future work, more researches should be conducted to find additional reliable information to be used in the OS fingerprinting process.

1.4 Research Contributions

The main contribution of this thesis is the Passive Operating System Fingerprinting based on Multi-Layered Sub-Signature Matching Scheme. The above contribution is divided into; new matching mechanism (Multi-layering) and the sub-signature structure for reliable and more accurate operating system fingerprinting in addition to the automated technique for DB update.

1. New multi-layer Operating System fingerprinting framework.
2. Unique OS sub-signature. By, having different signatures for each of the matching process layer.
3. Simple technique for automated signature DB update.

1.5 Research Methodology

The research methodology is divided into five stages, the first stage is the Study and Test OS dependant parameters (parameters selection), which aims to select the reliable and significant parameters to be used in the process of operating system fingerprinting. The second and the third stages are proposing the multi-layering and the sub-signature approaches, which will make it possible to come out with unique sub-signatures for the OS family names, OS versions and sub-versions in addition to reducing the searching space in the signature DBs. The fourth is the matching approach which is divided into exact matching and approximate matching which will provide the ability to match the original signatures in addition to the manipulated signatures as well. And the fifth stage is the signature DB Auto-update which aims to keep the signatures DB updated with the latest released OSs signatures. Figure 1.1 illustrates the five stages that will be discussed in details in the following sections in chapter 3.

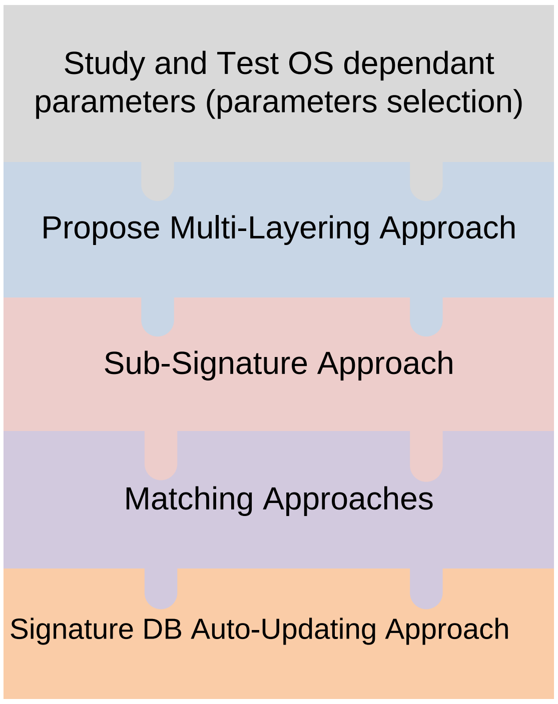
Study and Test OS dependant parameters (parameters selection)
Propose Multi-Layering Approach
Sub-Signature Approach
Matching Approaches
Signature DB Auto-Updating Approach

Figure 1.1 MLSMS Stages

1.6 Thesis outline

This thesis is organized into six chapters. Each chapter provides a basic idea to further proceed to the next chapter. Firstly, this chapter (**Chapter 1**) introduces the background of the content distribution of our research along with our objectives and contributions.

In **chapter 2**, a review of the literature and fundamental concepts related to our work. The other operating system fingerprinting techniques and methods in the latest researches have been discussed in details too.

Chapter 3 covers the methodology and how the proposed method was designed. The Multi-Layer technique and the Sub-Signature structure are described in details.

In addition, the exact matching, approximate matching method and the signature DB Auto-Update are discussed too in this chapter.

Chapter 4 covers the implementation details of the Passive Operating System Fingerprinting based on Multi-Layered Sub-Signature Matching Scheme.

Chapter 5 covers in-depth the conducted experiments and the results discussion. Four experiments are conducted to test the results accuracy of the MLSMS against well known tools (NMAP and P0f).

Finally, **Chapter 6** summarizes this thesis. This chapter revises the research contributions with regard to the proposed method in **Chapter 3** and its results in **Chapter 5**. Finally, a discussion and suggestions for future work pertaining to this thesis is discussed.

CHAPTER TWO

LITERATURE REVIEW ON OPERATING SYSTEM FINGERPRINTING

This chapter provides an overview of the current state of the art in Operating System (OS) fingerprinting methods and tools.

2.0 Introduction

Getting accurate and details information about operating systems of the hosts that are connect to the network is useful and beneficial for many purposes. A network engineer may use such information to deploy software updates or to patch serious security holes more efficiently. Maintaining complete and accurate database of OS for ICT's assets (Montigny-Leboeuf & Massicotte, 2004) such as computers, network gears, and printers helps system administrator to manage the software and system installation, applying updates, tuning and customizing network's parameters etc. Despite of its importance, many existing OS fingerprinting methods are still lacking identifying and more importantly in capturing details (and accurate) information of OS. In this chapter, a quick literature review on OS fingerprinting methods will be presented. The focus of the review will be on the more recent techniques and tools that are relevant to the approach proposed in this research study.

Here a brief discussion of some of the "old" operating system fingerprinting methods and techniques. Banner grabbing is one of these methods; it is considered the most basic and easiest method of operating system fingerprinting. Operating system could be fingerprinted by using Telnet which is a standard tool that can be found on most of the operating systems. By connecting to the remote host, Telnet will show some information about the machine including the operating system running on it. Unfortunately, this technique is not reliable – the operator of the remote machine may configure their hosts to hide or even fake or provide limited information about the operating system. In additional, due to its insecure nature (clear text protocol), nowadays the usage of Telnet is prohibited on public network. Another popular technique is to inspect operating system information embedded in some of the common network traffics such as those typically found in HTTP's header. Some popular browsers (the client), as part of their headers, happily included the type of operating system and this is mainly done to identify to the server the type of client it is serving – perhaps for compatibly purposes. The information encoded in this way once again is not reliable, as the browser can be configured to provide inaccurate information or to hide the information all together. Other indirect approaches such as relying on secondary information or relevant data may be used to deduce the type of operating system. For instance, it is well known that certain applications can only be deployed on specific operating system; hence the traces of network traffic from those applications may be used to identify the operating system of the host. Once again, such methods at best can identify the type of operating system, the details but without accurate information about operating system which still the main issue.

In this thesis, the focus of our work is on operating system fingerprinting solely based on the most commonly used network traffic or network protocols. Hence, this chapter reviews related works on operating system fingerprinting based on network protocol such as ICMP, TCP/IP, UDP and the like. The literature review is divided based on the type of OS fingerprinting general approach i.e. active vs.

passive approach. The type of protocols is highlighted for each reviewed method (some approaches may use multiple protocols) together with the parameters used, the matching mechanism and strengths & weaknesses of each approach.

2.0.1 Operating System

"Operating system is a set of system software programs in a computer that regulate the ways application software programs use the computer hardware and the ways that users control the computer" (Madison, 2010; Tanenbaum, 2001). Figure 2.1 shows the role of operating system in the machine as an intermediate between the hardware and software.

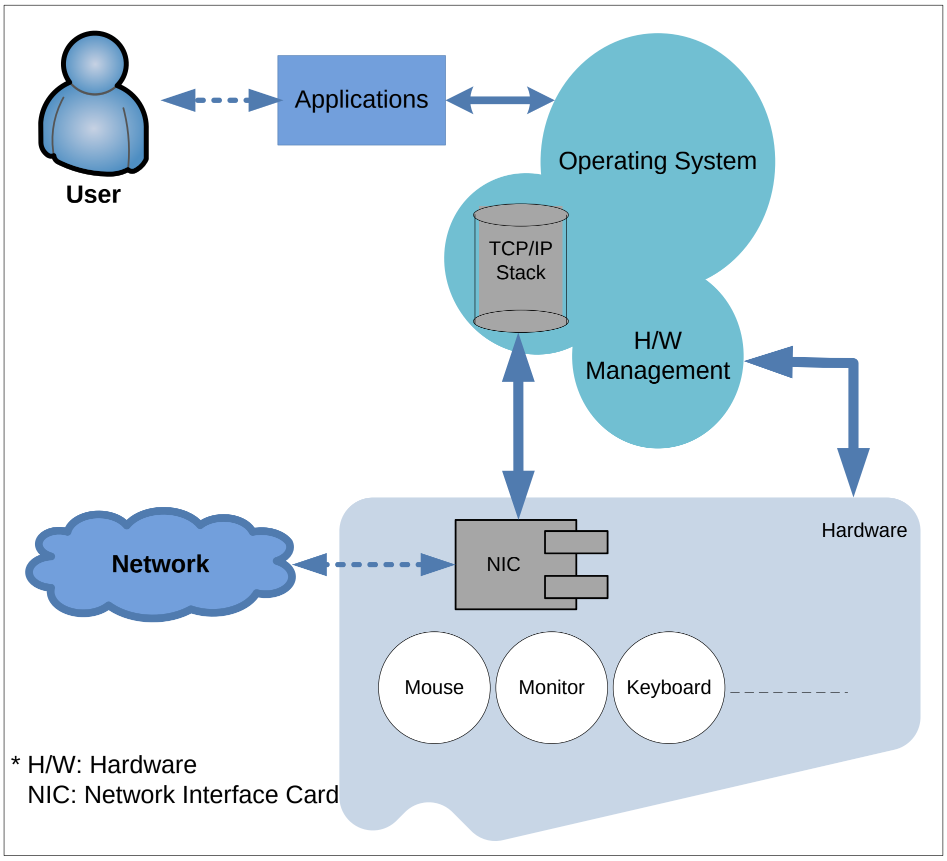

Figure 2.1 Operating System Roles in the computers

Operating system is considered as the backbone and the most important program running on the machine. It manages both the software and hardware resources, through controlling and allocating the memory as well as managing the inputs from the external devices and transmitting outputs to the computer displays. The critical role of the operating system in term of networking is the control and management of the peripherals (McHoes & Flynn, 2007).

On the other hand, operating system has even more work to do. Such operating systems monitor different programs and users, making sure everything runs smoothly, despite the fact that numerous devices and programs are used simultaneously (Hollander & Agostini, 2000). An operating system also has an important role to play in security. Its job includes preventing unauthorized users from accessing the computer system (INS, 2004; Post & Kagan, 2003).

2.0.2 Operating System Fingerprinting

OS Fingerprinting is the process of determining the operating system that is running on the remote machine (Spangler, 2003; Ttrowbridge, 2003). Chapter 2 will explain in details the various approaches of OS fingerprinting, while in this chapter we provide a general overview of OS fingerprinting approaches. OS may be identified via various means, the most direct approach is to manually inspect the host – this of course will require physical access to the host. Another direct and more automated approach is to install a software agent on the host, whereby the agent main task is to find information related to OS and update the remote host or server as requested or on regular basis. Similarly, the server may also query the remote host running the agent software (client) to get the information about OS.

Example of agent based approach is the SNMP’s agent (Simple Network Management Protocol).

The majority of OS fingerprinting methods however is focusing their approach based on the network traffics/traces exhibited by the OS running on the host. This group of methods is termed in the thesis as Network-based OS fingerprinting method. These methods based their identification on some of the unique signatures (mainly the parameters of popular protocols such as TCP/IP, ICMP, UDP, etc.) produced by certain OS. The ingenuity of the approach varies in several aspects such as the type and the number of parameters the method used, the matching mechanism or rule used and finally the signature database – the true parameters value exhibited by each OS. Figure 2.2 shows the structure of the TCP/IP packet whose parameters can be used for operating system fingerprinting, since each OS has a different TCP/IP stack configuration (Berrueta, 2003). Some of the more reliable and typical parameters used for this purpose are those as highlighted with the green color in the figure 1.2 e.g. (Time to Live (TTL), Window Size (WS), Maximum Segment Size (MSS), Frame Total Length (TL), Do not Fragment Bit (DF), and TCPoptions).

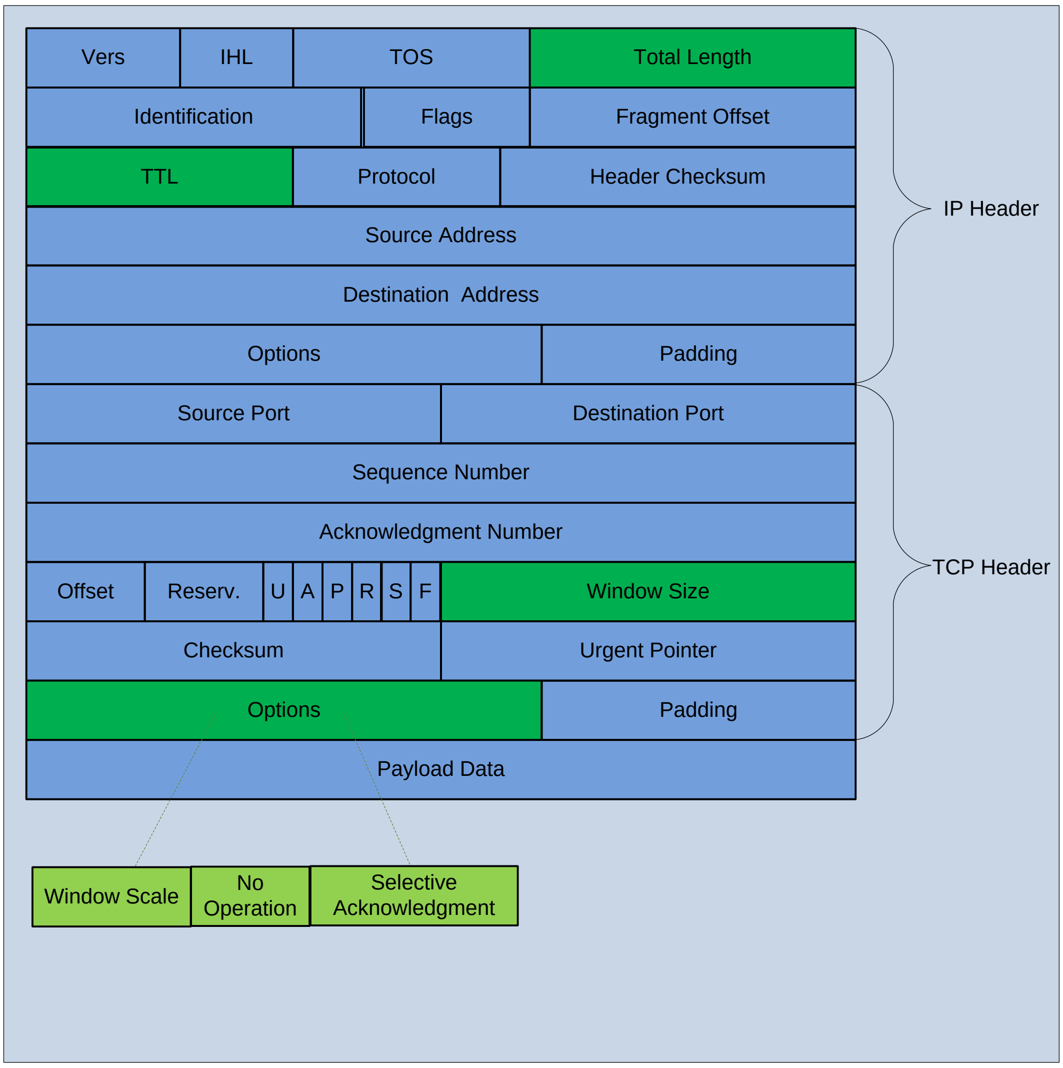

Figure 2.2 TCP/IP packet structure and its parameters

Network-based OS fingerprinting methods may be further classified into two categories on the basis of how intrusive they are in probing/collecting the OS of a remote host. Figure 2.3 depicts the approaches of operating system fingerprinting.

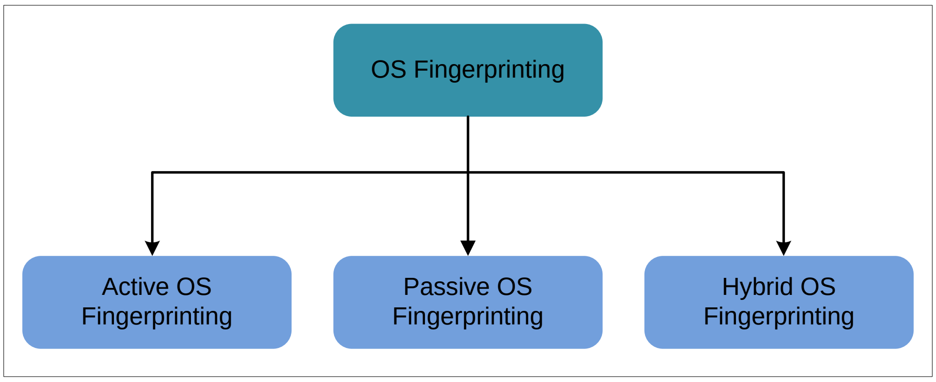

Figure 2.3 OS fingerprinting approaches

In the *active* OS fingerprinting approach, the remote host is actively being probed with the specifically crafted network packets; the responses from the probed host are then analyzed to deduce the identity of the host (Spangler, 2003). In contrast, in the *passive* OS fingerprinting approach (Honeynet-Project, 2002; Nazario, 2001; Ttrowbridge, 2003) the target hosts are unaware that their OS are been probed, instead the identification of the OS is deduced from the normal network traffics exhibited by the hosts e.g. stealthier (Giovanni, 2000). A combination of active and passive (hybrid) approach is also possible to claim the benefits from the two general approaches.

2.0.2.1 Passive Operating System Fingerprinting

Figure 2.4 illustrates passive OS fingerprinting technique in action, in this case host ***X*** passively tapping on the hub to stealthy listen and capture traffics passing though the hub. Network packets from both host ***A*** and ***B*** are accessible to host ***X*** to analyze and identify the operating systems running on both machines.

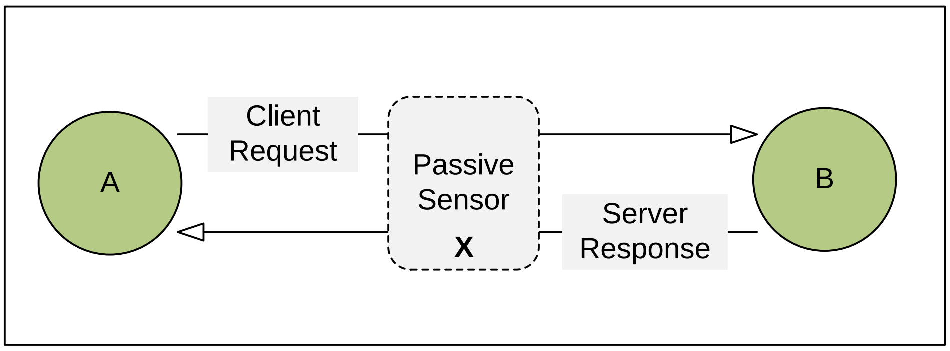

Figure 2.4 Passive operating system fingerprinting

The main advantage of the passive OS fingerprinting technique is that it does not send network traffic to the target host to be able to identify the OS. Instead, it listens to the traffic that sent from the target host to be able to identify the OS that is running on it. Thus it is not affected by security appliances such as firewall or IDS/IPS protection mechanism (Chown, 2006; Kollmann & Xnih, 2005; Nazario, 2001; Zalewski & Michal, 2005). There is no specific network probe from the passive monitoring host for a firewall to filter & block or for IDS/IPS to detect and trigger alerts. In additional, since no network traffic is injected into the monitored network, passive approach does not unnecessarily clogs the network with additional traffics e.g. the bandwidth of the monitored network is not affected.

This invisibility and simple approach of capturing data about OS and then identifying the target OS does have its limitations. Firstly, this technique suffers from the hosts' coverage or scope limitation issue. Host ***A*** in Figure 2.4 is only able to see and analyze network traffics from the hosts connected to same network or LAN where it is physically connected (including ***A*** and ***B***). Putting this differently, passive OS fingerprinting approach is not able to detect OSs of the remote hosts on separate network/LAN. Secondly, passive approach will have to rely on more common and general network packets to analyze and determine the OS i.e. limited set of data, common and typical data. Thus with such limited information about the target host, most passive approaches are not able to provide details and accurate information about the OS of the target hosts. In this thesis, a passive based approach is proposed relying totally on the most common network traffic to identify as details as possible the identity of target OS. The details of our approach are discussed in Chapter 3.

2.0.2.2 Active Operating System Fingerprinting

Figure 2.5 shows an example of active OS fingerprinting method, where host ***X*** probes the machines which connected to the network (host ***A_1*** till host ***A_n***) in order to identify their respective OS. The active OSF (Operating System Fingerprinting) software sends a crafted packets (probes) to these machines and waiting for the replies (responses) to be analyzed later to identify the operating systems that are running on both machines (Arkin, Yarochkin, & Kydyraliev, 2003a, 2003b; Lyon, 2008; Spangler, 2003).

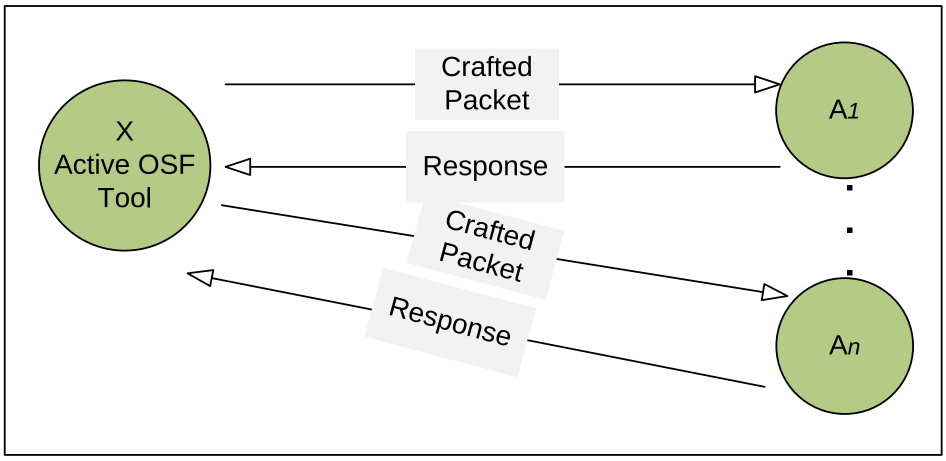

Figure 2.5 Active operating system fingerprinting

As illustrated in the diagram, active OS fingerprinting generates extra network traffic which directly affects the network performance of the hosts being probed. Sometimes the target host's performance may also be affected if the crafted probes violates standard protocol format or if too many probes are being sent to the target. This performance aspect is directly proportional to first, the number of probe being send and second, to the number of host to probe. A rough calculation on the effect of probing or scanning class B network for all 65,536 standard ports per host has been illustrated to create upwards traffics of 170 gigabyte as explained in (Chown, 2006). Packets probed by an active scanning host are subjected to security appliances defenses as well as other protection mechanisms configured at the host level e.g. personal firewall. Hence, there are clear risks here that the probing packets may not reach the target host or the responses from the target host either being dropped by firewall or being altered in such as way to give incorrect data about the hosts. Another limitation of active approach is that the result its get is as good as when the test was run, e.g. the information obtained is considered as a snapshot of the network hosts information. According to (Chown, 2006) this snapshot can be outdated in a short period of time depending on the pace at which the network's hosts are updated. In another words, any changes to the network that happens after the scan is run will not be valid until another scan is conducted.

Active OS fingerprinting however does have a few advantages compared to its passive counterpart (Spangler, 2003). First, with active technique ones can probe any hosts accessible through internet, i.e. the choice of hosts are wider and not limited to local LAN for instance. Also, given that a packet can be crafted in such a way to induce certain behavior of the target host or to reveal certain types of responses, an active method has the flexibility and freedom to craft or manipulate