A Method of Identify OS Based On TCP/IP Fingerprint

Jian Jiao **and** *Wei* **wu**

Beijing University of Aeronautics & Astronautics, BeiJing China

Summary
This paper present a method that classify the fingerprint of protocol, use the frame to describe the fingerprint in order to create the frame system, get the information of host to match the system to identify the type of OS in remote host. Result from experimental appears that this method can identify the OS effectively, the action of is more secretly than other systems such as nmap and xprobe

.Key words: TCP/IP Fingerprint OS

1.Introduction

It is an important field that identify what OS in remote host. Mastering the OS can analyse and acquire some information such as memory management、the kind of CPU. These information is important for computer network attack and computer network defense.

The main way to identify is through the TCP/IP fingerprint to finish. Nearly all kind of OS customize their own's protocol stack by following the RFC. This instance cause the fact that every protocol stack has some different details during implementing. These details are known as fingerprint which make it possible to identify the OS .

Nmap、Queso[1] use the fingerprint in transport layer. They send the particular packets to the target and analyse the retured packets, matching the fingerprint in the fingerprint warehouse in order to get the result. The information in the warehouse is affected by the specified message for probing. It hardly to distinguish the similar OS (eg.windows98/2000/xp).

Xprobe[2] mainly use the ICMP which make use of five kinds of packets in ICMP to identify OS. It can give the probability of all possible situation which maybe the indeed OS. The main shortage is it excessively depend on ICMP Protocol.

SYNSCAN[3] use some typical fields' fingerprint to identify when it communicaties with target host in application protocol. The warehouse of fingerprint have limited types of field.

Ring 、 Ttbit[5][6] identify the OS using the performance character of TCP/IP. Because this kind of character is affected by network environment greatly. the result is often not exactly.

Literature[7] analysis the action in messages which are acquired through interception(eg. The number of SYN request, a closed port how to response a connection request).Although this way is availability, it only distinguish a few given OS .

Above all the kinds of system, they all be scare of a way to describe the fingerprint of OS integrallty, which cause the proceeding of identify only depend on a part of TCP/IP . This paper propose a new method to resolve the problem: it uniformly the fingerprint of OS, acquire the message by some technology, identify the OS at last.

The rest of the paper is organized as followed: Section Ⅱ we present based concept of this method. Section Ⅲ present how to describe and match the protocol fingerprint using frame technology. Section Ⅳ present an algorithm to implement the method and Section Ⅴ use experiment to validate its effectiveness and analysis the result. Finally Section Ⅵ present the concluding remark and possible future work.

2. Based Concept

The proceeding of identify is to acquire message, extract the fingerprint and match the record of fingerprint warehouse, in order to know the type. This section define the measure which are to acquire message, the action and status of communication, also classify the fingerprint. These work are all prepared for the next section which how to built a frame system describing the fingerprint.

2.1 Acquirement of Communication Message

To insert "Tables" or "Figures", please paste the data as stated below. All tables and figures must be given sequential numbers (1, 2, 3, etc.) and have a caption placed below the figure ("FigCaption") or above the table("FigTalbe") being described, using 8pt font and

Manuscript received July 5, 2006.
Manuscript revised July 25, 2006.

please make use of the specified style "caption" from the drop-down menu of style categories

The mode of acquiring message have two kinds: **probe** and **monitor**.

Define1: **Probe** is the proceeding of recording the message which is sent from the target host through sending some packets to the host, it can be defined as followed:

Probe(target, type, message) (1)

Target denote the host that is probed. Type denote the kinds of probe. Message is the packet by acquired.

Define 2: Monitor is the process of capturing the message which are from the target host when it communicate with other host, it can be defined as followed:

Monitor(target, layer, message) (2)

Layer point to the layer of networked communication (eg. link layer, network layer, transport layer)

2.2 Action and Status of Networked Communication

Define 3: Communication Action is a series of acts which host execute communication protocols. As Fig 1 showed, TCP/IP Protocol can be classed as IP Action, TCP Action, UDP Action. For example, the IP Action have several sub- action: IP_COMMUNICATION which express transmitting the common IP packets.IP_ICMP_ECHO,IP_ICMP_STAMP and others express the actions of ICMP protocol. The form can be defined as followed:

Action=<Protocol> <actionname> (3)

<ProtocolType>=TCP|UDP|IP|ICMP|…

<actionname>=IP_ICMP_ECHO|TCP_Listen|TCP_Connect_SYN|…

Define 4: Communication Status express the status before communication action is executing. For example in UDP Protocol sort open/close can be expressed:

UDP SortOpen

UDP SortClose

So the expression of communication:

Status=<ProtocolType><statusname> (4)

2.3 category of character

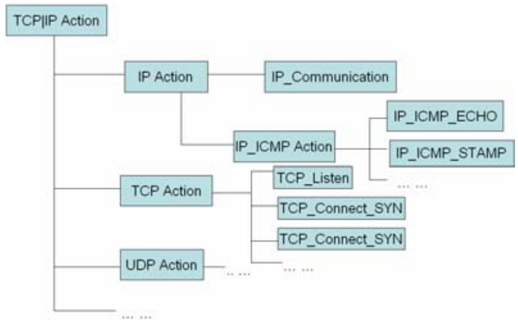

Fig1 Classify the TCP/IP Protocol Suits Communication Actions

Although RFC[8] describe the implement of TCP/IP particularly, it does not define many details of protocol. When the TCP/IP protocol is come into true as a part of OS kernel, the different may be seen as fingerprinting. This paper make the character into two classes: communication content fingerprinting and communication action fingerprinting.

Defined 5 : Communication content fingerprint is the special attribute in some field of protocol when the protocol is implemented. The example are the SYN initialization value and increased value, the order of option and default value, the number of TTL. It can be defined as followed:

Content_Fingerprint=<Action><Contents> (5)

<Action> defined in (3)

<Contents>=<value>|<value><Contents>

<value>::=bool|int|null|f(x)|…

f(x) is function which is calculate the value of field that is often changed.

Give an example: The "option" field in TCP have MSS,NOP,TimeStamp and others sub-fields. When the solaris is received a SYN segment, it response SYN+ACK, and the option in segment should be followed:

Content_Option_Fingerprint=TCP SYN_ACK Nop Nop SackPermit MSS(1460) When the OS is windowsXP, the character should be :

Content_Option_ Fingerprint=TCP SYN_ACK MSS(1460) Nop Nop SackPermit

Define 6: Communication action fingerprint is action's different that would be expressed when it be faced with same precondition during the protocol is executed.

Action_Fingerptint=<Status>
<Action><Actions>

<Action>、<Status> defined in (3) and (4)

<Actions>::=<Action>|<action><Actions>

For Example: When a open sort receive a FIN request. In RFC793, it does not response anything. But CISCO and HP/UX response a RST response. This would be:

Action_FIN_Fingerprint=TCP SortOpen TCP_FIN TCP_RST

3. Frame System for Character

Frame Expression technology is a kind of way for knowledge expression. It is fit for describing the entity's knowledge which has steady properties. Because frame can be inherited, every kinds of frame can be joined with Is-a and Ako into a frame system for expressing Os fingerprint. Fig 2 is architecture of the frame system for OS fingerprint. TCP,UDP and ICMP all inherited the Frame OSName(Fingerprint), constitute a whole system to identify the OS.

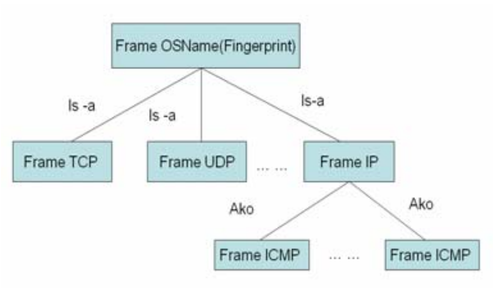

Fig 2 Frame System for OS Fingerprint

3.1 Description of Fingerprint

Using the way of expression in section Ⅱ, it construct a frame as followed to express fingerprint which some protocol have:

```
(Frame OSName
   (ProtocolName: If_ added))
   (Frame ProtocolFingerPrint
      (isa  OSName)
     (Slot1  value
           if_needed : Content_Fingerprint )
     (Slot2  if_neeed: Action_Fingerptint)
      … …
   )
```

When we want to construct a frame system about FreeBSD4.2, at fist make a father-frame to express OS, construct other child-frames using Is-a and Ako to connect with father-frame.

```
(Frame FreeBSD4.2
  (ProtocolName: If_ added))
  (Frame: IP
         (ProtocolName:IP)
         (isa : FreeBSD4.2)
         (TTL: 64)
         ( DF :1)
         (ID:IP_ICMP_ECHO f(previous_id) )
         … …
  )
         (Frame: TCP)
         (isa: FreeBSD4.2)
         (windowsize : SYN_ACK 5B4|403D|C0B7)
         (Option:SYN_ACK  MSS(1024)  Nop Windows(4091) Nop Nop TimeStamp)
         (Action_SYN :SortOpen SYN(9)
  )
```

f(x)=previous_id +1 Formula (**1**)

Formula (1) express that the value of 'ID' field would be increased one with the number of packets.

3.2 Matching of Fingerprint

Define 7: Matching of Fingerprint is the process of make the message into a frame which expressed fingerprint, and to judge whether the frame is consistent with known frame system. It should be followed:

Match (message, Frame) (7)

The detailed approaches are:

1. Using Probe and Monitor (Defined1,2)to acquire data, make a new frame to express a protocol:

```
(Frame ProtocolFingerPrint1
   (isa  OSName)
   (Slot1  : Action_Fingerptint)
```

(Slot2 : Action_Fingerptint)
… …
)

2. Compare this frame with the known frame system, the principles are:
   1) If the slot value of the new frame is equal with one frame system in a known frame system, the new frame is consistent with the Frame system.
   2) If the new frame is consistent, with the is-a and ako the new frame would find its own father frame.
3. The type of OS expressed father-frame is maybe the truly OS which the host that is identified

4. Methodological Implementation

Implementing the OS needs two parts: 1.Building the warehouse to store the frame information;2.A algorithm makes use of ways above all to finish identifying the OS.

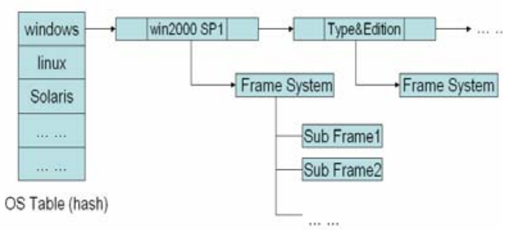

Fig 3 The Fingerprint Warehouse

As fig 3 showed ,every kind of OS suit have their own record in a hash table (eg. Windows, Solaries,FreeBSB).Each record has its linktable which is made up some nodes which is represented every edition OS of this suit, and format of the node is a frame system. All of this is the fingerprint warehouse.

After building the warehouse. An algorithm was represented to finish identifying the OS. It is pseudocode of this algorithm as followed:

```
Initialize(warehouse)
flag=false
While(flag not true )
{
    Probe(target,type,message)
    Mointor(target,layer,messge)
    If (Match(message,warehouse.Frame) is succeed ) {
    count<< OS
    flag=true
        }
    else {
    delete warehouse.Frame
    Probe(target,type,message)
    Mointor(target,layer,messge)
}
    if(warehouse is empty){
        exit
    }
}
```

In this algorithm, a flag is set up to judge whether identifying is successful. It use **Probe** and **Monitor** to get data, **Match** every node (warehouse.Frame). If matching is failed, delete the node from the warehouse. Other character should be identified on the next process until finding the node whose type of OS is exactly the target's OS.

5. Experiment and Analysis

In the network environment, we use the algorithm to construct IOS(Identify Operation System). At first, using IOS、nmap and xprobe to identify a host which os is window xp. As known, nmap uses eight times to use TCP/UDP and xprobe uses five icmp packet to identify(As **Table 1** shown),they get the response of these packets and match the os fingerprint database to get the result step by step. Based on this theory, list every action of these systems by times.Recording each precision of result of the identification.

Table 1 IOS Nmap and Xprobe orderTable for probing

NO	IOS	Nmap	Xprobe
1	Monitor(TCP)	SYN_Option	ICMP ECHO
2	Monitor(IP)	NULL_Option	ICMP Timestamp
3	Probe(target,SYN,Mes)	SYN|FIN|URG|PSH	ICMP Address Mask Request
4	Probe(target,FIN,Mes)	ACK	ICMP Information Request
5	… …	… …	… …

As fig.4 shown. IOS only use about 5 packets to identify the OS which precisionis more than eighty percents, nmap use more than 8 packets to identify the OS nearly, xprobe only guess the OS is win2000 with the probability is 60%,and the number of packet is maximum.

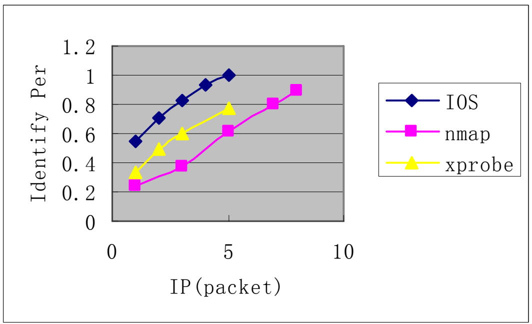

Fig 4 Comparing with the percent of identify and numbers of packet

In a LAN, using IOS and nmap to identify server hosts which OS are windows2000/XP, solaries,linux and so on. IOS and nmap use their own best method to identify the OS utmost. Comparing their result in fig 5,IOS is better in identification precision than nmap taking it with nmap. Escipally to windows OS ,it can successfully distinguish

win2000/XP,which the nmap can only give the guess probibality .

All of these tests make it clearly that IOS can effectually identify type of OS. Comparing with other system, it use only less packets and the concealment ability is more.

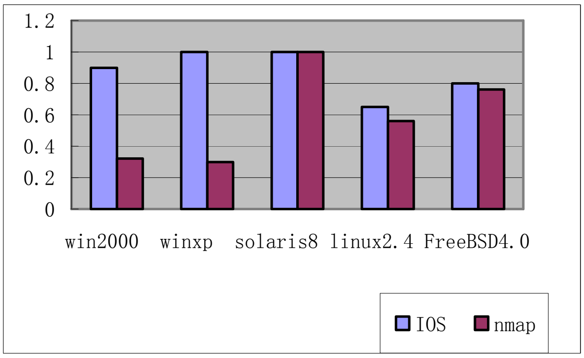

Fig 5 IOS and nmap identification to some OS

6. Conclusion

In this paper, we have presented a method for identifying OS of remote host. The method use frame technology to express the fingerprint, make up of **Probe** and **Monitor** to get message and abstract the information from the message to match the warehouse of fingerprint, identify the OS at last. Through experiment, this method can exactly identify the OS of remote hose with more secretly and less number of packets comparing with nmap and xprobe.

In the future, we plan to collect more fingerprint for each kind of OS, make the algorithm to be more intelligent, in order to improve the precision of identify.

References

[1] Fyodor.Remote OS detection via TCP/IP Stack FingerPrinting. http://www.insecure.org/. 2002,6

[2] Ofir Arkin, Fyodor Yarochkin. A “Fuzzy” Approach to Remote Active Operating System Fingerprinting. http://www.sys-.security.com/archive/papers/Xprobe2.pdf. 2002,8

[3]Greg Taleck. SYNSCAN: Towards Complete TCP/IP Fingerprinting. http://synscan.sourceforge.net/taleck-synscan-2004.pdf. 2004,6

[4]Wang YIJUN,Xue Zhi,Li Jianhua. OS Detection Based On TCP/IP Stack FingetPring. Computer Engineering. 2004,30(18): 7-9

[5]Franck Veysset, Olivier Courtay, Olivier Heen. New Tool and Technique For Remote Operating System Fingerprinting Intranode Software Technologies.2004, 4

[6]Jitendra Padhye, Sally Floyd. On Inferring TCP Behavior. June 2001, In Proceedings of SIGCOMM 2001.2001,8

[7]SANS Institute .Passive OS fingerprinting: Details and Techniques. Information Security Reading Room.2003.

[8]DARPA INTERNET PROGRAM. RFC793 http://rfc.net/rfc793.html

JiaoJian received M.S. degrees in Electrical Power University of north China in 2001 and 2004, respectively. He stayed in Beijing University of Aeronautics & Astronautics study for information security.

WuWei Professor, Doctor tutor. Main study fields are information Security, Virtual reality and IPV6.