

# On the Accuracy of Remote Active Operating System Fingerprinting Tools

**Ofir Arkin**  
CTO & Co-Founder



# What this talk is about?

- **This talk examines different aspects of remote active operating system fingerprinting**
  - Examines different active OS fingerprinting methods & techniques
  - Discusses their limitations and advantages
  - Explains the state of the current used technology
  - Deals with the question of what can and cannot be accomplished using remote active OS fingerprinting
  - Looks at what should be done in the future
  - Analyzes the accuracy aspects of remote active OS fingerprinting and of several active OS fingerprinting tools

# Ofir Arkin

- **CTO & Co-Founder, Insightix**  
<http://www.insightix.com>
  
- **Founder, The Sys-Security Group**  
<http://www.sys-security.com>
  
- **Computer Security researcher**
  - Infrastructure Discovery
    - ICMP Usage In Scanning
    - Xprobe2
  - VoIP Security
  - Information Warfare



# What is Remote Active OS Fingerprinting?

# What is Remote Active OS Fingerprinting?

- Remote active operating system fingerprinting is a technology, which uses stimulus (sends packets) in order to provoke a reaction from network elements
- The implementations of active operating system fingerprinting will monitor the network for a response to be, or not, received from probed targeted network elements, and according to the type of response, and the conclusions following (part of an implementation's intelligence), knowledge will be gathered about the underlying operating system

# What is Remote Active OS Fingerprinting?

- Remote active OS fingerprinting tools **uses the differences between different TCP/IP stack implementations** in order to fingerprint an underlying OS
- **'Stack-based'** active OS fingerprinting tools usually examines the differences with the implementation of the TCP, UDP, and ICMP protocols
- The way of operation of an active OS fingerprinting tool, **the OS fingerprinting tests it uses** (i.e. what does it check for), **the type of packets it sends** (i.e. RFC compliant, crafted), **the number of packets it sends**, **varies** from one remote active OS fingerprinting tool to another

# Main Advantages

- **Control over the parameters to scan for (i.e. the stimulus)**
- **Control over the pace of the scan and its initiation**
- **Provides with fast results**
- **Can be used from a single point to scan multi-points**
- **Can be used from multi points to scan multi-points**
- **Can cover entire IP ranges**

# Main Disadvantages (Highlights)

- There is **no control** over the **quality of the scan**
- The **quality of the scan** is **directly affected by the environment**:
  - The location of the scanning system and the target system and what is between them (local network, remote network, over the Internet)
    - The path between the scanning element to the target element (firewalls, load balancers, scrubbers, etc.)
    - The target element itself (i.e. personal firewall, tunable parameters, etc.)
- **Lack of intelligence** (i.e. to determine the topology the scanning system is facing, switching scanning tactics, ‘understand’ results)

# Main Disadvantages (Highlights)

- Some active OS fingerprinting tools **consumes a lot of network bandwidth**
- The bandwidth consumption affects the scan time...
- Which is usually time-consuming as is...
- The **large number of packets sent**, by some remote active OS fingerprinting tools, may cause a **denial of service** condition to the network, overloading switches/routers, or to overwhelm end devices
- Some active OS fingerprinting tools uses **crafted packets** with their fingerprinting process. Those packets **may cause a denial of service** to intermediate devices or to end devices (i.e. CPEs)
- Continues...

# Issues with Remote Active OS fingerprinting

## Generic Issues

## What do we fingerprint? – The Issue with Hardware-based Devices

- When fingerprinting **operating systems** we fingerprint the way an **operating system (the software)** reacts to different fingerprinting probes a tool uses
- With a **hardware based device** we fingerprint the way a **device's firmware** reacts to the different fingerprinting probes
- Hardware based devices of **the same manufacture** will usually **run the same**, or a **slightly different, firmware** (a.k.a software) **version**
- It will be either **one version for all**, or a **particular version for a particular functionality**

# What do we fingerprint? – The Issue with Hardware-based Devices

## ■ Example: Cisco IOS

- A Cisco 7200 router will be fingerprinted exactly the same as Cisco's Aironet 1100/1200 wireless access points
- It is not possible to distinguish between different hardware based products, and their functionality, manufactured by Cisco and using IOS, when using traditional active operating system fingerprinting methods
- It is possible to identify these devices as manufactured by Cisco and using IOS
- It is also possible to divide these devices into groups according to fingerprints differences with the IOS versions they are using, but not to discover their functionality

## What do we fingerprint? – The Issue with Hardware-based Devices

- Other examples: Foundry Networks IronWare operating system (Net/Fast/Big Iron family), Printers (i.e. HP Printers – it is not about their modules but rather it is their firmware), etc.
- If the designers of a fingerprinting tool failed to understand these issues, the results received, which are **based on a corrupted database**, will be **unreliable** (at best)

# The Way Probe Results Are Being Matched

- **A Strict Signature Matching based Tool**
  - Would search for a 100% match between the **received results** and the tool's **signature database**
  - If a 100% match is not found, than **no match is found** and the run fails
  - Extremely **sensitive to environmental affects** on the probed target, and on the network which the probed target resides on

# The Way Probe Results Are Being Matched

- **Fuzzy Logic**
  - **Xprobe2**
    - **First to implement a statistical analysis based mathematical algorithm** to provide with a best effort match between probe results, received from a targeted system, to a signature database
    - Uses one of the simplest forms of **Optical Character Recognition (OCR)**, by **utilizing a matrix based** fingerprints matching based on statistical calculation of scores for each test performed
  - Using a fuzzy logic approach, **provides better resistance against environmental affects** which might take their toll on a target system and on probe packets

# The Way Probe Results Are Being Matched

- Fuzzy Logic (continue)

- The **quality of the results** produced with an active operating system fingerprinting tool using a fuzzy logic approach would be with **higher quality**
- This is if the tool will not suffer from design flaws, and will **use a large base of fingerprinting tests**
- The fuzzy logic implementation with Xprobe2 **still misses** the ability to **assign different weights to different fingerprinting tests**
- This ability is required since **some fingerprinting tests** should have **bigger impact** over the overall fingerprinting results

## The Use of a Fixed Number of Fingerprinting Tests

- A fixed number of fingerprinting tests are used
- A fixed number of parameters are examined
- In theory:
  - Possible matches = the number of tests X number of parameters examines X parameter's permutations
- Although the overall number of possible matches is currently much higher than the number of the current available network elements, **certain test classes cannot deliver the expected results** and to provide with a clear distinction between different OSs

## The Use of a Fixed Number of Fingerprinting Tests

- A better tool for active OS fingerprinting would be required to **utilize fingerprinting tests**, which would **examine many parameter values** with a probe's reply
- These parameter values **would need to be different among many TCP/IP stack implementations**
- Therefore a number of those tests are needed in order to **achieve a broader distinction between different TCP/IP stack implementations**

## The Use of a Fixed Number of Fingerprinting Tests

- It suggests that the **usage** of more **parameter rich fingerprinting tests** with an active operating fingerprinting tool will provide **better overall results**
- An active operating system fingerprinting tool must, therefore, **reserve the ability** to be able to **support new fingerprinting methods as they are published**

## The inability to implement new fingerprinting tests due to DB population and control problems

- When a new fingerprinting test is implemented a signature DB of an active OS fingerprinting tool needs to be updated to reflect the addition of the new test
- An **uncontrolled signature DB cannot handle new fingerprinting tests**, since some of its signatures **cannot be rebuilt, expanded, or recreated** to reflect the addition of the new test
- This can create differences in the quality of the signatures

## No Changes Are Made To the TCP/IP Stacks Of New Versions Of Operating Systems

- The behavior of the TCP/IP stack of **newly released operating systems** hardly changes compared to an older version of the same operating system, or
- Changes made to a newly released operating system's TCP/IP stack **might affect a certain protocol behavior only**
- The result? **Inability** of some active operating system fingerprinting tools which rely on a certain fingerprinting niche **to distinguish between different versions of the same operating system** or even between a class of the same operating system family

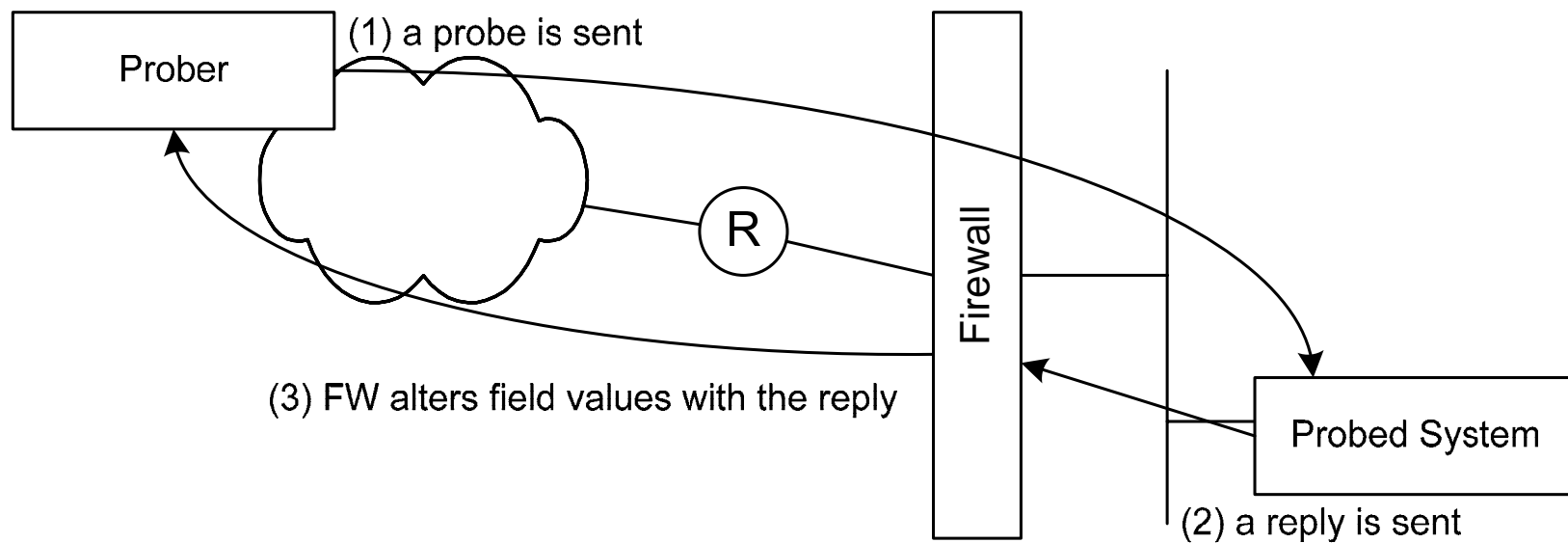
## The Inability to Determine the Exact OS Service Pack

- Traditional active operating system fingerprinting tools are usually **unable to identify the installation of software service packs on a targeted machine**
- For example, traditional active operating system fingerprinting tools will identify a targeted machine runs Microsoft Windows 2000, but **will not be able to determine which service pack version is installed (if any at all)**

## Some Fingerprinting Tests May Have Bigger Impact on the Overall Results

- Some fingerprinting tests have bigger **impact** over the **overall accuracy** of the test results compared with other tests used
- If these tests **fail**, for some reason, **the quality of the produced results will be significantly lowered**

# Different Networking Devices May Alter A Packet's Field Value



# A Firewallled Target Systems

- Probed systems **might be firewallled**
- If a remote active operating system fingerprinting tool relies on sending and/or receiving of particular packet types and those packets are **dropped** by a firewall protecting the target system(s) chances are that the **quality of the results** would be **degraded** to the point **false results** or **no results at all** will be produced

# The Use of Malformed Packets

- If malformed packets are used, a filtering device (**and even end point devices**) **may drop** the packets, if the device analyzes packets for non-legitimate content
- Therefore **the quality** of the results produced by utilizing a fingerprinting tests relying on malformed packets will be **degraded** and in some cases even **fail**
- Malformed packets may have another affect, they **might cause some TCP/IP stacks to crash**

# A TCP/IP Stack's Behavior Might Be Altered

- Some **characteristics** of a TCP/IP stack's behavior might be **altered**:
  - **Tunable parameters** of the TCP/IP stack might be changed e.g. the `sysctl` command on the various \*BSDs, the `ndd` command on Sun Solaris, etc.
  - Numerous **patches exist** for some open source operating system's kernels that **alter the way** the particular operating system's TCP/IP stack responses to certain packets

# A TCP/IP Stack's Behavior Might Be Altered

- If a remote active operating system fingerprinting tool is using some of the TCP/IP based parameters that can be altered as part of its fingerprinting test, the quality of the results would be **effected** and **questionable** when these parameter values will be altered

# The Quality of the Signature Database

- The quality of the results produced by an OS fingerprinting tool is directly affected by **the way the signature database of a tool was built and is maintained**
- If **signatures submitted** to the database were and are **obtained in a wrongfully manner** than the signature database should be regarded as **corrupt**
- The results produced by the tool **will not be accurate**, **this even if the tool would use the most advanced fingerprinting tests**

## The Inability to Identify the Underlying Architecture Platform

- Usually, active operating system fingerprinting tools will identify the operating system of a network node, but **not its underlying platform**
- The knowledge about the underlying platform is **important** for tools performing vulnerability assessment, network inventory, etc., which rely on the results of the active operating system fingerprinting tool (i.e. nessus)

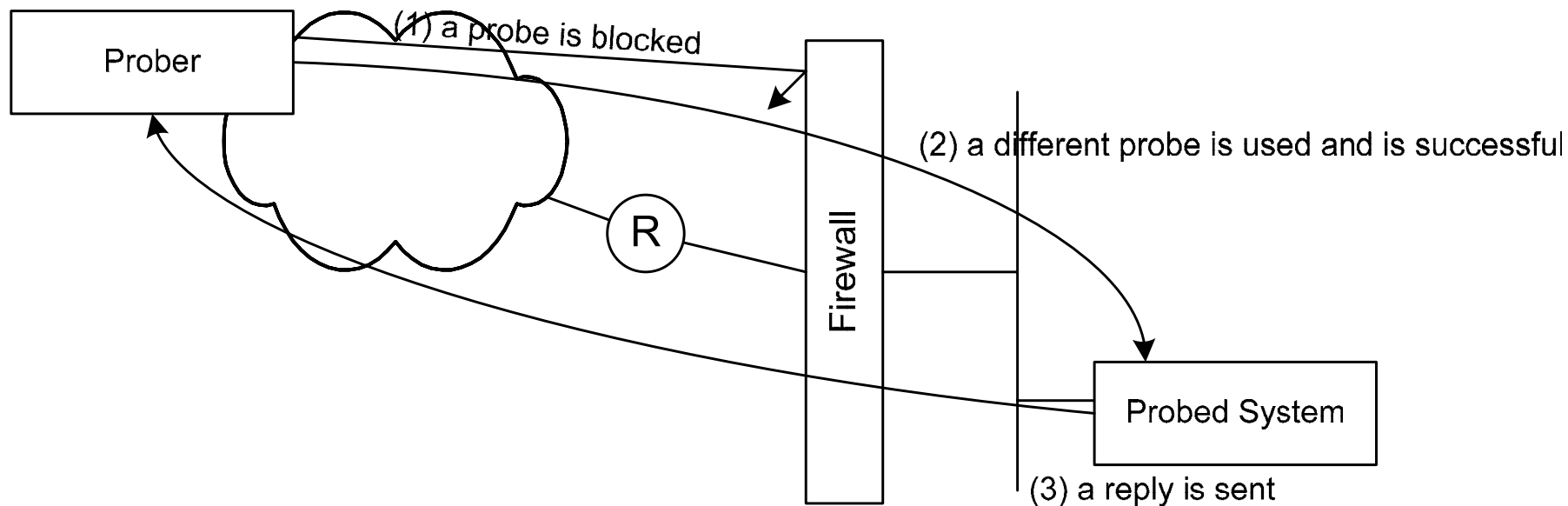
# The Inability to Scale

- An active operating system fingerprinting tool should have the **ability to scan large networks**
- Must **not use many packets** to do so
- For any router and switch there is an **upper limit** to the number of **packets per second it can process**
- Beyond that limit, some packets will be **dropped**, but more important, **the router/switch might suffer from a denial of service condition**
- Therefore it is very important to **balance the scan rate** with the network and network elements abilities

## Inability to Control the Fingerprinting Modules to Be Executed

- When scanning different machines on different topologies **some tests would be proved useless**
- Controlling which tests to use would result with better accuracy and less chance of being detected
- One needs to **control the fingerprinting tests** a certain tool has to offer **according to her/his needs**
- Furthermore, we would like an active OS fingerprinting tool to be able to detect certain scanning conditions and to react, by **switching scanning tactics**

# Inability to Control the Fingerprinting Modules to Be Used



# Issues with Remote Active OS fingerprinting

## The OS Fingerprinting Methods

# The OS Fingerprinting Methods

- The OS fingerprinting methods a certain remote active OS fingerprinting tool uses **requires that the scanning conditions would meet several conditions** in order to **produce with a successful identification** of the underlying operating system of a remote machine
- **Some of those conditions cannot be met under several scanning terrains**
- One good example would be a web server behind a well fortified firewall

# The OS Fingerprinting Methods

- Since **some of the OS fingerprinting tests** a remote active OS fingerprinting tool would use **may fail, the accuracy** of the tool **will be degraded when optimal scanning conditions would not be met**
- If the OS fingerprinting tests which would fail, would be those with the bigger impact on the accuracy of the tool's result, the result the tool would produce would be poor at best

# The OS Fingerprinting Methods

## Other Approaches – TCP Stack Implementation

- Some researchers suggested to use a certain OS fingerprinting niche to fingerprint the underlying operating systems of remote machines in light of Internet conditions
- The suggested tests would use an opened TCP port, and only would examine the **TCP stack** implementation of the remote machine
- Some of those tests **requires specific data to be exchanged between the scanning system to its target element**, and **a great number of packets to be exchanged**

# The OS Fingerprinting Methods

## Other Approaches – TCP Stack Implementation

- The main issue with this approach is that this approach is ok to use when you wish to identify families of operating systems and not an exact operating system version
- Another issue with this approach is that some other tests, which are currently available with open source remote active OS fingerprinting tools, produces the same results when run against an opened TCP port...

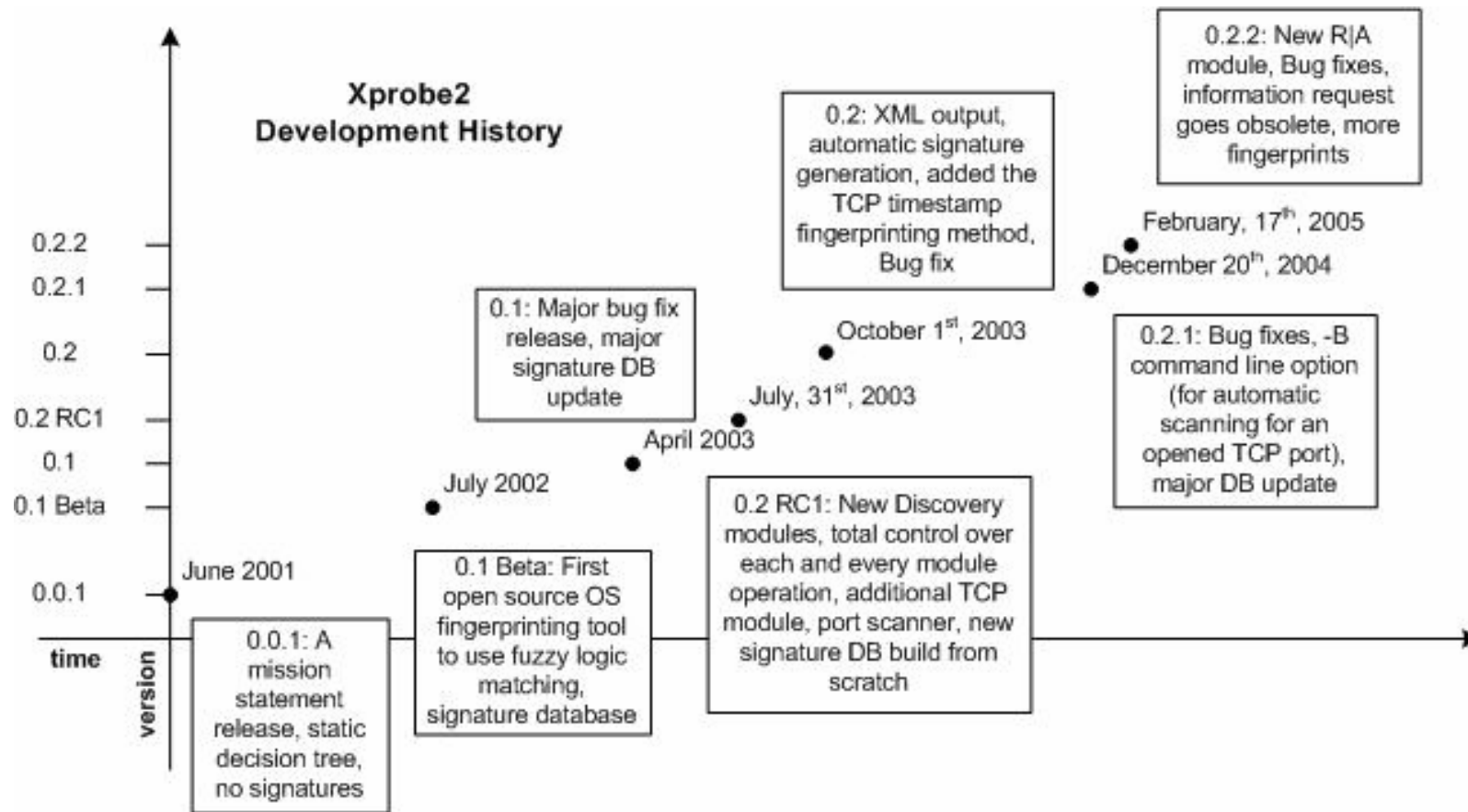
# Xprobe2



# Xprobe2 Project

- **An open source remote active OS fingerprinting tool, which presents an alternative to other remote active OS fingerprinting tools**
  
- **Developers**
  - Fyodor Yarochkin
  - Ofir Arkin
  - Meder Kydyraliev
  
- **The project represents our take, beliefs and ideas, and we hope it contributes to the community at large**
  
- **Voted one of the top 75 security tools (at the top 50)**

# Xprobe2 – Project History



# Xprobe2

## The OS Fingerprinting Modules

# Xprobe2 – The OS Fingerprinting Modules

## Discovery Modules

ICMP Echo

TCP ping

UDP ping

## Information Gathering Modules

TTL Distance

Port Scanner

## Fingerprinting Modules

ICMP Echo

ICMP Timestamp

ICMP Address Mask

ICMP Port Unreachable

TCP Handshake

TCP R|A

# Xprobe2 – The OS Fingerprinting Modules

- **What is usually needed?**

- Opened TCP port
- Closed TCP port
- Closed UDP port
- ICMP echo reply
- ICMP timestamp reply
- Address Mask reply

- **Impact**

# Remote Active OS Fingerprinting

## Accuracy

## Accuracy – Examined results in a sterilized environment

Tool	Target OS	Guessed OS	No. of Packets
nmap	Windows 2003 Enterprise Server	Microsoft Windows 2003 Server or XP SP2	3385
Xprobe2 0.2.2	Windows 2003 Enterprise Server	Windows 2003 Server	32
nmap	Windows XP SP2	Microsoft Windows Server 2003 or XP SP2	3402
Xprobe2 0.2.2	Windows XP SP2	Microsoft Windows Server 2003 or XP SP2	30
nmap	Windows 2000 SP4	Microsoft Windows Millennium Edition (Me), Windows 2000 Pro or Advanced Server, or Windows XP	3275
Xprobe2 0.2.2	Windows 2000 SP4	Windows 2000 SP4	18
nmap	Linux Kernel 2.6.8	2.4.18 - 2.6.7	3374
Xprobe2 0.2.2	Linux Kernel 2.6.8	Linux Kernel 2.6.8	20
nmap	Linux Kernel 2.6.0	Linux 2.4.18 - 2.6.7	3374
Xprobe2 0.2.2	Linux Kernel 2.6.0	Linux Kernel 2.6.0 – 2.6.7	18
nmap	Linux Kernel 2.4.18	Linux 2.4.0 - 2.5.20	3373
Xprobe2 0.2.2	Linux Kernel 2.4.18	Linux Kernel 2.4.5 – 2.4.18	28
nmap	Linux Kernel 2.4.4	Linux 2.4.0 - 2.5.20	3375
Xprobe2 0.2.2	Linux Kernel 2.4.4	Linux Kernel 2.4.4	26

## Accuracy – Examined results in a sterilized environment

Tool	Target OS	Guessed OS	No. of Packets
nmap	FreeBSD 5.3	FreeBSD 5.2-CURRENT (Jan 2004) on x86	3388
Xprobe2 0.2.2	FreeBSD 5.3	FreeBSD 5.3	22
nmap	FreeBSD 5.2	FreeBSD 5.2-CURRENT (Jan 2004) on x86	
Xprobe2 0.2.2	FreeBSD 5.2	FreeBSD 5.2 – 5.2.1	
nmap	FreeBSD 4.10	4.6.2-RELEASE - 4.8-RELEASE	3552
Xprobe2 0.2.2	FreeBSD 4.10	FreeBSD 4.7 – 4.10	26
nmap	OpenBSD 3.6	OpenBSD 3.4 – 3.6	
Xprobe2 0.2.2	OpenBSD 3.6	OpenBSD 3.5 – 3.6	
nmap	Sun Solaris 7	Sun Solaris 2.6 - 8 (SPARC)	3388
Xprobe2 0.2.2	Sun Solaris 7	Sun Solaris 7	20

# Remote Active OS Fingerprinting

## Future

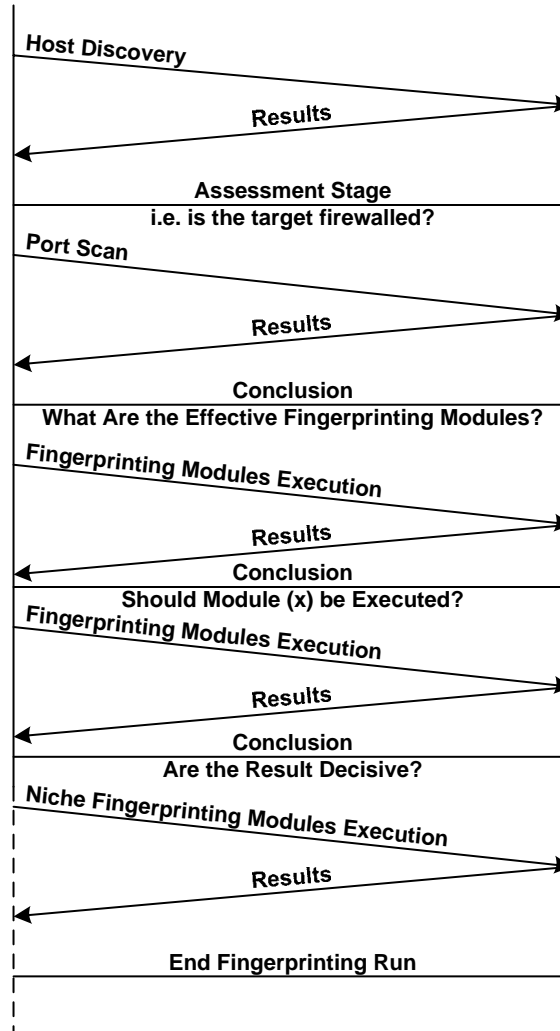
# Future

- **Intelligence in scanning must be introduced**
- **Understanding of the terrain a tool operates in is crucial**
- **An active OS fingerprinting tool must understand the quality of the results received**
- **More tests needs to be evaluated in order to find more OS fingerprinting tests which will have significance in the OS fingerprinting process**

# Future

- An integration between Stack-based OS fingerprinting tests and **application layer based fingerprinting tests tailored towards the services found opened on a targeted system(s) and/or a service commonly found with the operating system family in question, must be created**

# Future



# Questions?

# Reference

- **Ofir Arkin's Web Site:**  
**<http://www.sys-security.com>**
- **Arkin Ofir, "ICMP Usage in Scanning" version 3.0, June 2001**
- **Arkin Ofir & Fyodor Yarochkin, "X – Remote ICMP based OS fingerprinting Techniques", August 2001.**
- **Arkin Ofir & Fyodor Yarochkin, "ICMP based remote OS TCP/IP stack fingerprinting techniques", Phrack Magazine, Volume 11, Issue 57, File 7 of 12, Published August 11, 2001.**

# Reference

- Arkin Ofir & Fyodor Yarochkin, “Xprobe2 - A ‘Fuzzy’ Approach to Remote Active Operating System Fingerprinting”, August 2002.
- Arkin Ofir, Fyodor Yarochkin, Meder Kydyraliev, “The Present & Future of Xprobe2 – Next Generation Active Operating System Fingerprinting ”, July 2003.