

## On the Current State of Remote Active OS Fingerprinting Tools

**Ofir Arkin**  
CTO  
ofir.arkin@insightix.com  
Defcon 13



### 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
- Presents the new version of Xprobe2 (Xprobe2 v0.3)

## Ofir Arkin

- CTO and 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 (The Active OS Fingerprinting Tool)
  - VoIP Security
  - Information Warfare
- Member
  - VoIPSA (Board member, chair security research committee)



## Remote Active OS Fingerprinting An Introduction



## An Introduction

- Remote Active operating system fingerprinting is the process of actively determining a targeted network node's underlying operating system by probing the targeted system with several packets and examining the response(s), or lack thereof, received
- The traditional approach is to examine the TCP/IP stack behavior (IP, TCP, UDP, and ICMP protocols) of a targeted network element when probed with several legitimate and/or malformed packets

## An Introduction

- The received results would then be compared to a signature database in an effort to find an appropriate match
- Remote active OS fingerprinting is not limited to the IP and Transport layers only
- The application layer can be used as another venue for information gathering about the underlying operating system

## An Introduction

- The way of operation of an active OS fingerprinting tool varies from one remote active OS fingerprinting tool to another
  - The OS fingerprinting tests used (i.e. what does it check for)
  - The type of packets sent (i.e. RFC compliant, crafted)
  - The number of packets sent
  - Other variables

## An Introduction

- Identifying the underlying operating system of a network element, whether remote or local, is an important parameter for the success of many processes in the networking and security world
- Building a network inventory, getting the right context for network intrusion detection systems, and performing a vulnerability analysis are all good examples among many other for the use of active operating system fingerprinting

## Remote Active OS Fingerprinting Strengths

## Strengths

- 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 cover entire IP address ranges
- Can be used from a single point to scan multi-points
- Can be used from multi points to scan multi-points

# Remote Active OS Fingerprinting Weaknesses

## Weaknesses

- The weaknesses of remote active OS fingerprinting are divided into:
  - Scanning conditions and environmental effects
  - Operation of the OS fingerprinting process
  - Signature DB related issues
  - Tool related issues

## Weaknesses Scanning conditions and environmental effects

## Scanning Conditions & Environmental Effects

- 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 terrain and the limitations of the scan, switching scanning tactics, 'understand' results)

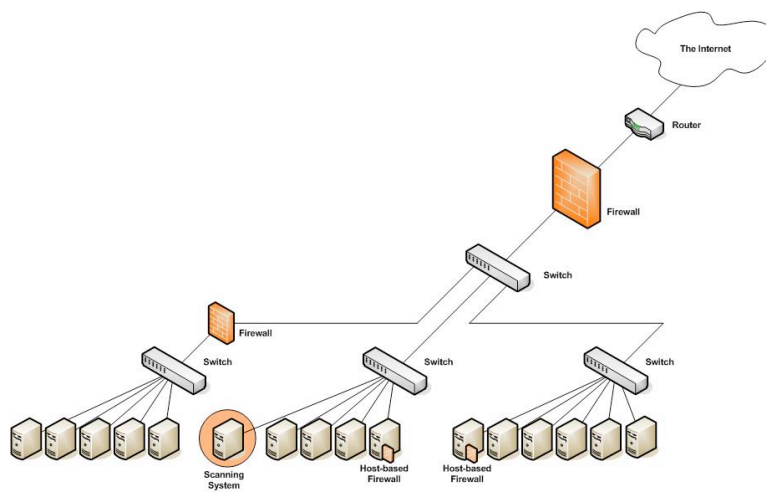
## Network Obstacles

- A remote active OS fingerprinting tool will be able to detect the underlying operating system of an element which will be operational ('up') on the network at the time of the scan. This is if the packets sent by the tool are able to reach the probed elements, and that the probed element's OS signature is included with the tool's signature DB
- Network obstacles such as Network firewalls, host-based firewalls, NAT enabled devices, load balancers and other, may block probe packets from reaching their target

15

insightix

## Network Obstacles

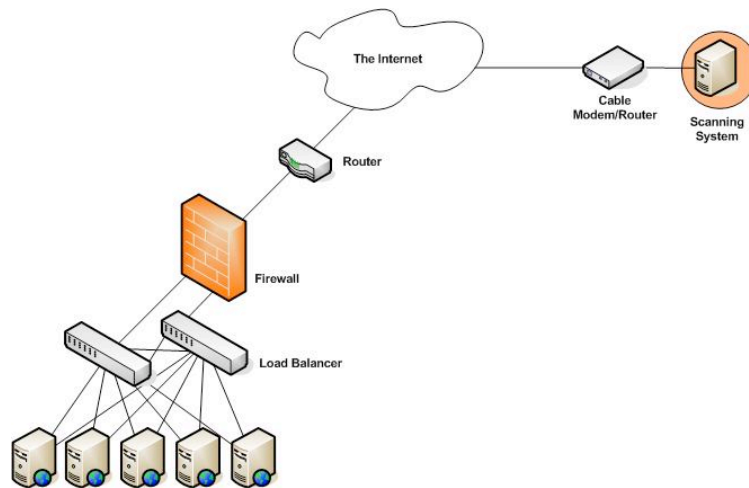


16

insightix



## Network Obstacles



17

insightix

## Network Obstacles

- 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

18

insightix

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

- A networking device along the path between the source system to its destination may alter one, or more, field values which are relevant to the OS fingerprinting process
- This would result with issues determining what is the targeted machines underlying operating system
- Example: Scrubbers

## The Use of Crafted/Malformed Packets

- If malformed packets are used with the OS fingerprinting process, a filtering device (and even an end point device) 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 may 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

## Weaknesses Signature DB Related Issues

## Signature DB related issues

- The signature DB is one of the most important parts of an active OS fingerprinting tool
- 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

## Producing Signatures

- Producing signatures is an important process
  - Signatures must be produced in a controlled environment
  - The test device must be controlled
  - The terrain must not harm the process
  - A lab is the most appropriate signature production environment
- Examples for problems
  - Lab environment vs. Internet
  - When there is no understanding of the process: 'Let's fingerprint this firewalled device'

25



## Strictly controlled vs. Loosely controlled signature DB

- Strictly controlled
  - Signatures are inserted to the DB only after verification (usually only by a tool maintainer)
  - Usually is a slower process
  - Limitation of the number of signatures / devices
  - Extremely accurate
- Loosely controlled
  - Signatures are submitted over the internet
  - Signature creation process is not controlled
  - Many signatures are produced in a wrongful manner
  - Creates an uncontrolled DB
  - Extremely inaccurate

26



## What do we fingerprinting?

- 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

## What do we fingerprinting?

- 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 fingerprinting?

- Example I: Cisco IOS
  - A Cisco 7200 router will be fingerprinted exactly the same as Cisco's Aironet 1100/1200 wireless access points
  - They run the same operating system - Cisco IOS
  - It is impossible to tell their use (i.e. router, wireless access point) according to traditional TCP/IP stack based active OS fingerprinting
- Example II:
  - Foundry Networks IronWare operating system (Net/Fast/Big Iron family)
- Example III:
  - Printers (i.e. HP Printers – it is not about their modules but rather it is their firmware), etc.

## What do we fingerprinting?

- Unfortunately with many active OS fingerprinting tools these issues were not taken into account
- These tools have a corrupted DB

## 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

## Weaknesses Operation of the OS fingerprinting tool



## The Way Probe Results Are Being Matched

- Scan results needs to be compared to the signature DB in order to find a match
- The comparison process can be done either using:
  - Strict signature matching
  - Statistical analysis approach

## The Way Probe Results Are Being Matched

- A Strict Signature Matching based Tool
  - Would search for a 100% match between the received results and a tool's signature database
  - If a 100% match is not found, than there are no results
  - 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

- Statistical based algorithms (the best match)
  - Using statistical based algorithms a tool is able to provide with better resistance against environmental affects which might take their toll on a target system and on probe packets
  - Some fingerprinting tests may have bigger impact over the overall accuracy of the test results compared with other tests used. Their failure may or may not harm with the ability to provide with granular results (i.e. not grouped)
  - Remark: Xprobe2 was the first open source tool to implement a statistical analysis based mathematical algorithm ('fuzzy logic') to provide with a best effort match between probe results to a signature database

## 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
- 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

## 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

## 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 Windows 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 OS service pack version is installed (if any at all)

## 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

## Weaknesses Summary

## Weaknesses - Summary

- 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

## Weaknesses - Summary

- 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

## Weaknesses - Summary

- The currently used TCP/IP-based OS fingerprinting test are not granular enough with their results (i.e. Microsoft Windows based OSs)

## Other approaches and their limitations



## Other approaches and their limitations

- 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

## Other approaches and their limitations

- The main problem of 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 quality of results when run against an opened TCP port with a single packet...

## The Needed Solution

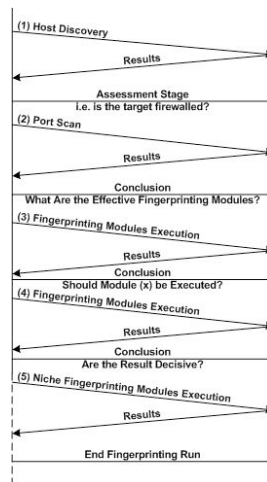
## The Needed Solution

- 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

## The Needed Solution

- 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

## The Needed Solution

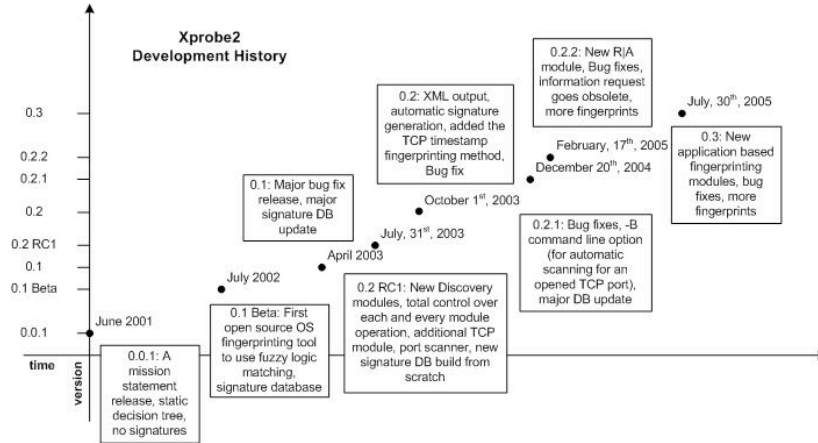


## Xprobe2

## The Xprobe2 Project

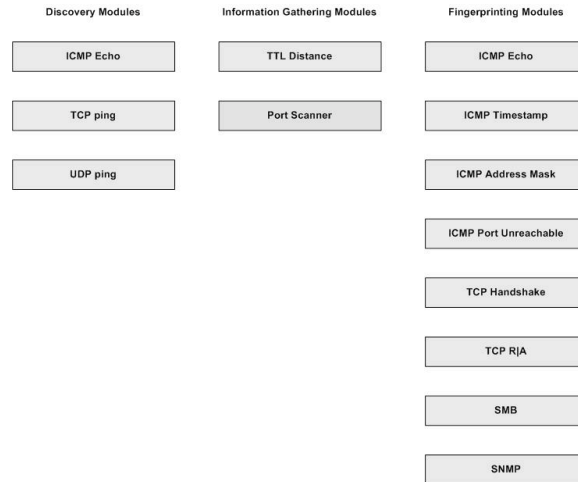
- An open source remote active OS fingerprinting tool, which presents an alternative to other remote active OS fingerprinting tools
- Developers
  - Fyodor Yarochkin
  - Ofir Arkin
  - Meder Kydyraliev
- The project represents our take, beliefs and ideas, and we hope it contributes to the community at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

# Xprobe2 - Project History



# Xprobe2 The OS Fingerprinting Modules

## The OS Fingerprinting Modules



59

## 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

60

## Introducing Xprobe2 v0.3

## Featured added to Xprobe2 v0.3

- Application-based OS fingerprinting modules
  - SMB
  - SNMP
- New signatures
  - Mac OS X 10.2.x, 10.3.x, 10.4.x
  - Linux Kernels 2.4.29, 2.4.30, 2.6.11, 2.6.12
  - FreeBSD 4.11, 5.4
  - OpenBSD 3.7
- Bug fixes
- Available for download from:  
<http://www.sys-security.com>

# Xprobe2 v0.3 Demo

# Xprobe2 v0.3 Demo



## Remote Active OS Fingerprinting Future Directions

## Future Directions

- Automating the active OS fingerprinting scan to understand the terrain
- Switch scanning tactics according to the terrain
- Evaluate the quality of the results received
- Determine if the results received are good enough to be presented
- Present results

## Questions?



## Resources

- 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.

## Resources

- 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.

## Thanks!

