The Use of Xprobe2 in a Corporate Environment

Ofir Arkin
ofir@sys-security.com
Founder
Agenda

- Introduction
- Why using Xprobe2 in a Corporate environment?
- Parameters effecting the accuracy of active operating system fingerprinting
- Xprobe2 v0.2 RC1 and RC2 advanced functionality
- How to effectively use Xprobe2 in a Corporate environment?
- The future
- Questions
Material Available for Download
http://www.sys-security.com

- Xprobe2 0.2 RC 2 Source Code
- This presentation
Ofir Arkin

- CISO of an International Telephone Carrier
- Founder, The Sys-Security Group
- Computer Security Researcher
  - Etherleak: Ethernet frame padding information leakage (with Josh Anderson)
  - IP Telephony Security (Security risk factors with IP Telephony based networks, numerous advisories and white papers)
  - ICMP Usage In Scanning (Security related issues with the ICMP protocol)
  - Information Warfare (trace-back)
- Member, the Honeynet project
Xprobe/Xprobe2 Project

- Open Source Project
- Developers
  - Ofir Arkin
  - Fyodor Yarochkin
  - Meder Kydyraliev
- Xprobe2 is a remote active operating system fingerprinting tool
- Xprobe2 presents an alternative to other remote active operating system fingerprinting tools
- Voted one of the top 75 security tools (at the top 50)
Xprobe/Xprobe2 Project

- The project reflects our beliefs and ideas
- We hope it contributes to the security community at large
- Contributions are always welcomed
- We dedicate our spare time to work on the project
- Did we say Open Source?
Xprobe/Xprobe2 Project History

- Initial **alpha release** (Xprobe v0.0.1) at the **Blackhat briefings USA 2001**, June 2001
  
  - Relying on **ICMP-based active OS fingerprinting methods** found by Ofir Arkin (specified in the “ICMP Usage In Scanning” research paper)
  
  - **Static** decision tree
  
  - Was not signature-based
  
  - **It was only a mission statement** - Alpha – **limited in functionality**
Xprobe/Xprobe2 Project History

- Xprobe2 0.1 beta was released last year at Defcon X:
  - Based on a signature database
  - First open source fingerprinting tool to use fuzzy logic matching algorithm between probe results to a signature database (strict signature matching suffers from a number of accuracy issues)
  - Xprobe2 0.1 beta was using only ICMP-based fingerprinting tests
Xprobe/Xprobe2 Project History

- **Xprobe2 0.1 release** (April 2003)
  - Sends RFC compliant packets
  - A lot of **bug fixes**
  - Support for **IP ID = SENT** fingerprinting method
  - Major **signature DB** update
  - Documentation on **how to add your own signatures**
Xprobe/Xprobe2 Project History

- **Xprobe2 v0.2 RC1** released July 31\textsuperscript{st} 2003


- Many enhancements to the tool’s functionality were introduced with this version (will be explained later in the presentation)
Xprobe/Xprobe2 Project History

- Xprobe2 v0.2 RC2 released today
Why Using Xprobe2 in a Corporate Environment?
Why Xprobe2 in a Corporate Environment?
Why Xprobe2 in a Corporate Environment?

- An **intimate knowledge** of an organization’s infrastructure is required in order to effectively combat threats.
- It is extremely important to understand and to know your infrastructure (and to be up-to-date).
Why Xprobe2 in a Corporate Environment?

- The right **context** is a must for:
  - **Precise Vulnerability Assessment**
    - Must get the right operating system before “matching the vulnerabilities”
  - **Meaningful Intrusion detection Systems**
    - “Phantom Alerts” – when alerting on an IIS hack attempt where the target is an Apache Web Server on Linux
  - **Accurate Risk Management**
    - Without the correct inventory it is impossible to correctly asses the risk to an organization
Issues with Active OS fingerprinting
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 (or software) version.

- It will be either one version for all, or a particular version for a particular functionality.
The Issue with Hardware-based Devices

- **Example**: 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.
The Issue with Hardware-based Devices

- Another example is the Foundry Network’s Net/Fast/Big Iron family

- 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.
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 **higher**
  - 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 is used
- A fixed number of parameters are examined
- **In theory:** Possible matches = \( \text{tests} \times \text{parameters} \times \text{examines} \times \text{parameters permutations} \)
- 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 provide with a clear distinction between different network elements
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 the probe’s reply.

- These parameter values would need to be different among many network elements.

- Therefore, a number of this kind of tests is required to be used in order to achieve a broader distinction between different network elements.

- It suggests that the usage of more parameter rich fingerprinting tests with an active operating fingerprinting tool will provide better overall results.
The Use of a Certain Fingerprinting Niche

**nmap**

- Sending TCP [various]
- Sending UDP

**nmap**

- Examining TCP
  - P
  - Link

**nmap**

- Examining ICMP [one]
  - P
  - Link
The Use of a Certain Fingerprinting Niche

```
Xprobe2 v0.1

sending

sending

sending

UDP

ICMP [various]

Xprobe2 v0.1

examining

ICMP

P

Link
```
The Use of a Certain Fingerprinting Niche

- This fixation brings into light the inability of such tools to deal with situations were the fingerprinting tests they use do not yield an adequate result about a certain operating system or even a class of operating systems.
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
No Changes Are Made To the TCP/IP Stacks Of New Versions Of Operating Systems

[root@angelfire NG]# xprobe2 -v x.x.x.x

XProbe2 v.0.1 Copyright (c) 2002-2003 fygrave@tigerteam.net, ofir@sys-security.com, meder@areopag.net

[+] Target is x.x.x.x
[+] Loading modules.
[+] Following modules are loaded:
   [ ] 1 ICMP echo (ping)
   [ ] 2 TTL distance
   [ ] 3 ICMP echo
   [ ] 4 ICMP Timestamp
   [ ] 5 ICMP Address
   [ ] 6 ICMP Info Request
   [ ] 7 ICMP port unreach

[+] 7 modules registered
[+] Initializing scan engine
[+] Running scan engine
[+] Host: x.x.x.x is up (Guess probability: 100%)
[+] Target: x.x.x.x is alive
[+] Primary guess:
[+] Host x.x.x.x Running OS: "Sun Solaris 5 (SunOS 2.5)" (Guess probability: 100%)
[+] Other guesses:
[+] Host x.x.x.x Running OS: "Sun Solaris 6 (SunOS 2.6)" (Guess probability: 100%)
[+] Host x.x.x.x Running OS: "Sun Solaris 7 (SunOS 2.7)" (Guess probability: 100%)
[+] Host x.x.x.x Running OS: "Sun Solaris 8 (SunOS 2.8)" (Guess probability: 100%)
[+] Host x.x.x.x Running OS: "Sun Solaris 9 (SunOS 2.9)" (Guess probability: 100%)
No Changes Are Made To the TCP/IP Stacks Of New Versions Of Operating Systems

[root@angelfire NG]# /usr/local/bin/nmap -sT -O x.x.x.x

Starting nmap 3.28 ( www.insecure.org/nmap/ ) at 2003-06-18 19:14 IDT
Interesting ports on x.x.x.x:
(The 1628 ports scanned but not shown below are in state: closed)

<table>
<thead>
<tr>
<th>Port</th>
<th>State</th>
<th>Service</th>
</tr>
</thead>
<tbody>
<tr>
<td>21/tcp</td>
<td>filtered</td>
<td>ftp</td>
</tr>
<tr>
<td>22/tcp</td>
<td>filtered</td>
<td>ssh</td>
</tr>
<tr>
<td>25/tcp</td>
<td>open</td>
<td>smtp</td>
</tr>
<tr>
<td>80/tcp</td>
<td>open</td>
<td>http</td>
</tr>
<tr>
<td>135/tcp</td>
<td>open</td>
<td>loc-srv</td>
</tr>
<tr>
<td>139/tcp</td>
<td>open</td>
<td>netbios-ssn</td>
</tr>
<tr>
<td>443/tcp</td>
<td>open</td>
<td>https</td>
</tr>
<tr>
<td>465/tcp</td>
<td>open</td>
<td>smtps</td>
</tr>
<tr>
<td>1029/tcp</td>
<td>open</td>
<td>ms-1sa</td>
</tr>
<tr>
<td>1433/tcp</td>
<td>open</td>
<td>ms-sql-s</td>
</tr>
<tr>
<td>2301/tcp</td>
<td>open</td>
<td>compaqdiag</td>
</tr>
<tr>
<td>5555/tcp</td>
<td>open</td>
<td>freeeciv</td>
</tr>
<tr>
<td>5800/tcp</td>
<td>open</td>
<td>vnc-http</td>
</tr>
<tr>
<td>5900/tcp</td>
<td>open</td>
<td>vnc</td>
</tr>
<tr>
<td>6000/tcp</td>
<td>filtered</td>
<td>X11</td>
</tr>
</tbody>
</table>

Remote operating system guess: Windows NT 3.51 SP5, NT4 or 95/98/98SE
Nmap run completed -- 1 IP address (1 host up) scanned in 3.334 seconds
The Inability to Determine the Exact Software 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 may have bigger impact over the overall accuracy of the test results compared to other fingerprinting tests used.

- If these tests fail, for some reason, the quality of the produced results will be lowered significantly, especially with tools using strict signature matching.

- The affect of a failure of a mark key test on the results a tool using a fuzzy logic approach produces will be less significant, although it might take its toll as well.
Some Fingerprinting Tests May Have Bigger Impact on the Overall Results

spanion:~ # xprobe2 -v x.x.x.x
XProbe2 v.0.1 Copyright (c) 2002-2003 fygrave@tigerteam.net, ofir@sys-security.com, meder@areopag.net
[+] Target is x.x.x.x
[+] Loading modules.
[+] Following modules are loaded:
  [x][1] ICMP echo (ping)
  [x][2] TTL distance
  [x][3] ICMP echo
  [x][4] ICMP Timestamp
  [x][5] ICMP Address
  [x][6] ICMP Info Request
  [x][7] ICMP port unreach
[+] 7 modules registered
[+] Initializing scan engine
[+] Running scan engine
[+] Host: x.x.x.x is up (Guess probability: 100%)
[+] Target: x.x.x.x is alive
[+] Primary guess:
[+] Host x.x.x.x Running OS: "Microsoft Windows XP Professional / XP Professional SP1" (Guess probability: 100%)
[+] Other guesses:
[+] Host x.x.x.x Running OS: "Microsoft Windows 2000/2000SP1/2000SP2/2000SP3" (Guess probability: 100%)
Some Fingerprinting Tests May Have Bigger Impact on the Overall Results

spanion:~ # xprobe2 -v -D 1 -D 2 -D 3 x.x.x.x
XProbe2 v.0.1 Copyright (c) 2002-2003 fygrave@tigerteam.net, ofir@sys-security.com, meder@areopag.net
[+] Target is x.x.x.x
[+] Loading modules.
[+] Following modules are loaded:
   [x][1] ICMP Timestamp
   [x][2] ICMP Address
   [x][3] ICMP Info Request
   [x][4] ICMP port unreach
[+] 4 modules registered
[+] Initializing scan engine
[+] Running scan engine
[+] All alive tests disabled
[+] Target: x.x.x.x is alive
[+] Primary guess:
[+] Host x.x.x.x Running OS: "Microsoft Windows XP Professional / XP Professional SP1" (Guess probability: 100%)
[+] Other guesses:
[+] Host x.x.x.x Running OS: "Microsoft Windows 2000/2000SP1/2000SP2/2000SP3" (Guess probability: 100%)
[+] Host x.x.x.x Running OS: "Microsoft Windows ME" (Guess probability: 100%)

Not using the ICMP echo fingerprinting module
Different Networking Devices May Alter A Packet’s Field Value

(1) a probe is sent

R

(3) FW alters field values with the reply

Prober

Firewall

Probed System

(2) a reply is sent
A Firewalled Target Systems
A Firewalled Target Systems

- Probed systems **might be firewalled**

- 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 may drop the packets, if the filtering 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 can be **altered** by a machine’s system administrator:
  - **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

- 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 active operating system fingerprinting tool is not only a factor of programming and topology.

- It is much affected from the way the signature database of the tool was and is built.

- 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.
The Quality of the Signature Database

- One can find false information quite easily in signature databases of some tools
- For example: nmap has a TCP “EOL” in the middle of a TCP Options list of some fingerprints (when it should not be there)
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 extremely 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

(1) a probe is blocked
(2) a different probe is used and is successful
(3) a reply is sent
Xprobe2 v0.1 RC1 Advanced Functionality
Xprobe2 v0.2 RC1 Advanced Functionality

- New Discovery Modules
- The ability to totally control the tools’ modules operation
- The use of best of breed TCP/IP stack fingerprinting techniques (more fingerprinting modules)
- A port scanner
- A mechanism for automatic calculation of the receive timeout(s) of modules
- A new signature DB built from scratch
New Discovery Modules

- Designed to perform host detection, firewall detection, and to provide information for the automatic receive timeout mechanism

- Two new discovery modules are introduced, the “TCP ping” and “UDP ping” discovery modules

- They are not executed by default

- One must specify, using the “-p” command line option, an open or a closed TCP port for the TCP ping discovery module to be executed, and a closed UDP port for the UDP ping discovery module to be executed
New Discovery Modules

- The aim of the discovery modules is to **elicit a response from a targeted host**, either a **SYN|ACK** or a **RST** as a response for the **TCP ping discovery module** and an **ICMP port unreachable** as a response for the **UDP ping discovery module**.

- The round trip time calculated for a successful run of a discovery module is used with the automatic receives timeout mechanism.
The Ability to Totally Control the Modules Operation with Xprobe2 – Module Execution

- With the “-D” command line option one can specify which Xprobe2 modules *not to use*

- With the “-M” command line option one can specify which Xprobe2 modules to *use*
The Ability to Totally Control the Modules Operation with Xprobe2 – Module Execution Order
The Ability to Totally Control the Modules Operation with Xprobe2 – Module Execution, Example “- D”

[ root@fremont src]# ./xprobe2 -v -c ../etc/xprobe2.conf -D infogather:ttl_calc -D 9 -p
TCP:139:open x.x.x.x
Xprobe2 v.0.2 Copyright (c) 2002-2003 fygrave@tigerteam.net, ofir@sys-security.com, meder@areopag.net
[+] Target is x.x.x.x
[+] Loading modules.
[+] Following modules are loaded:
[ x] [ 1] ping:icmp_ping  -  ICMP echo discovery module
[ x] [ 2] ping:tcp_ping  -  TCP-based ping discovery module
[ x] [ 3] ping:udp_ping  -  UDP-based ping discovery module
[ x] [ 4] infogather:portscan  -  TCP and UDP PortScanner
[ x] [ 5] fingerprint:icmp_echo  -  ICMP Echo request fingerprinting module
[ x] [ 6] fingerprint:icmp_tstamp  -  ICMP Timestamp request fingerprinting module
[ x] [ 7] fingerprint:icmp_amask  -  ICMP Address mask request fingerprinting module
[ x] [ 8] fingerprint:icmp_port_unreach  -  ICMP port unreachable fingerprinting module
[ x] [ 9] fingerprint:tcp_hshake  -  TCP Handshake fingerprinting module
[+] 9 modules registered
[+] Initializing scan engine
[+] Running scan engine
[-] ping:udp_ping module: no closed/open UDP ports known on x.x.x.x. Module test failed
[+] Host: x.x.x.x is up (Guess probability: 66%)
[+] Target: x.x.x.x is alive. Round-Trip Time: 0.09818 sec
[+] Selected safe Round-Trip Time value is: 0.19636 sec
[+] Primary guess:
[+] Host x.x.x.x Running OS: "Microsoft Windows 2000 Server Service Pack 3" (Guess probability: 100%)
The Ability to Totally Control the Modules Operation with Xprobe2 – Module Execution, Example "- M"

[root@fremont src]# ./xprobe2 -v -c ../etc/xprobe2.conf -M ping:icmp_ping -M fingerprint:icmp_echo -M fingerprint:icmp_port_unreach -p TCP:23:open x.x.x.x
Xprobe2 v.0.2 Copyright (c) 2002-2003 fygrave@tigerteam.net, ofir@sys-security.com, meder@areopag.net
[+] Target is x.x.x.x
[+] Loading modules.
[+] Following modules are loaded:
[x] [1] ping:icmp_ping - ICMP echo discovery module
[x] [2] fingerprint:icmp_echo - ICMP Echo request fingerprinting module
[x] [3] fingerprint:icmp_port_unreach - ICMP port unreachable fingerprinting module
[+] 3 modules registered
[+] Initializing scan engine
[+] Running scan engine
[+] Target: x.x.x.x is up (Guess probability: 100%)
[+] Target: x.x.x.x is alive. Round-Trip Time: 0.05988 sec
[+] Selected safe Round-Trip Time value is: 0.11975 sec
[+] Primary guess:
[+] Host x.x.x.x Running OS: "Cisco IOS 11.2" (Guess probability: 100%)
[+] Other guesses:
[+] Host x.x.x.x Running OS: "Cisco IOS 11.1" (Guess probability: 100%)
[+] Host x.x.x.x Running OS: "NetBSD 1.5.3" (Guess probability: 93%)
[+] Host x.x.x.x Running OS: "NetBSD 1.5.2" (Guess probability: 93%)
[+] Host x.x.x.x Running OS: "NetBSD 1.5.1" (Guess probability: 93%)
[+] Host x.x.x.x Running OS: "NetBSD 1.5" (Guess probability: 93%)
[+] Host x.x.x.x Running OS: "OpenBSD 2.5" (Guess probability: 93%)
[+] Host x.x.x.x Running OS: "NetBSD 1.4.3" (Guess probability: 93%)
[+] Host x.x.x.x Running OS: "NetBSD 1.4.2" (Guess probability: 93%)
[+] Host x.x.x.x Running OS: "Cisco IOS 11.3" (Guess probability: 93%)
[+] Cleaning up scan engine
[+] Modules deinitialized
[+] Execution completed.
The Ability to Totally Control the Modules Operation with Xprobe2

- Except for the “TCP ping”, “UDP ping”, and the port scanner module, all Xprobe2’s modules will be executed by default

- Combined with Xprobe2’s other command line options (such as “-s” and “-t”), the complete control over the tool’s operations and usage is given to the end user

- This complete control over Xprobe2’s way of operation allows one to execute different modules according to the topology it is facing

- The “-L” command line option can be used to list available modules
The Ability to Totally Control Modules and Features of Xprobe2

```
[root@fremont src]# ./xprobe2 -L
Xprobe2 v.0.2 Copyright (c) 2002-2003 fygrave@tigerteam.net, ofir@sys-security.com,
meder@areopag.net
Following modules are available (by keyword)
ping:icmp_ping  ping:tcp_ping  ping:udp_ping  infogather:ttl_calc
infogather:portscan  fingerprint:icmp_echo  fingerprint:icmp_tstamp
fingerprint:icmp_amask
fingerprint:icmp_info  fingerprint:icmp_port_unreach
fingerprint:tcp_hshake
usage: ./xprobe2 [ options] target
Options:
    -v    Be verbose
    -r    Show route to target(traceroute)
-p <proto:portnum:state>  Specify portnumber, protocol and state.
    Example: tcp:23:open, UDP:53:CLOSED
    -c <configfile>  Specify config file to use.
    -h    Print this help.
    -o <fname>  Use logfile to log everything.
-t <time_sec>  Set initial receive timeout or roundtrip time.
-s <send_delay>  Set packetsend delay (milliseconds).
-d <debuglvl>  Specify debugging level.
-D <modnum>  Disable module number <modnum>.
-M <modnum>  Enable module number <modnum>.
-L    Display modules.
-m <numofmatches>  Specify number of matches to print.
-P    Enable portscanning module
-T <portspec>  Specify TCP port(s) to scan.
    Example: -T21-23,53,110
-U <portspec>  Specify UDP port(s) to scan.
    -f    force fixed round-trip time (-t opt).
```
The Use of Best of Breed TCP/IP Stack Fingerprinting Techniques

- Searched for a TCP-based fingerprinting module with maximum impact over the overall fingerprinting results

- A test which will use as much parameters as possible and provide with a real added value

- We have decided on adding a TCP module based on the TCP 3-way handshake
The Use of Best of Breed TCP/IP Stack Fingerprinting Techniques

- The **parameters** with the SYN request sent **resembles** the parameters used with a Linux `telnet` request

- Unlike other tools, which use a similar module, Xprobe2 **examines** parameters found in the IP and TCP layers

`sys-security group`
The Use of Best of Breed TCP/IP Stack Fingerprinting Techniques

Implemented with RC2
The Use of Best of Breed TCP/IP Stack Fingerprinting Techniques – Example (without)

[root@fremont src]# ./xprobe2 -v -c ../etc/xprobe2.conf -p TCP:25:open -D 11 x.x.x.x
Xprobe2 v.0.2 Copyright (c) 2002-2003 fygrave@tigerteam.net, ofir@sys-security.com, meder@areopag.net
[+] Target is x.x.x.x
[+] Loading modules.
[+] Following modules are loaded:
[x] [1] ping:icmp_ping - ICMP echo discovery module
[x] [2] ping:tcp_ping - TCP-based ping discovery module
[x] [3] ping:udp_ping - UDP-based ping discovery module
[x] [4] infogather:ttl_calc - TCP and UDP based TTL distance calculation
[x] [5] infogather:portscan - TCP and UDP PortScanner
[x] [6] fingerprint:icmp_echo - ICMP Echo request fingerprinting module
[x] [7] fingerprint:icmp_tstamp - ICMP Timestamp request fingerprinting module
[x] [8] fingerprint:icmp_anmask - ICMP Address mask request fingerprinting module
[x] [9] fingerprint:icmp_info - ICMP Information request fingerprinting module
[x] [10] fingerprint:icmp_port_unreachable - ICMP port unreachable fingerprinting module
[+] 10 modules registered
[+] Initializing scan engine
[+] Running scan engine
[-] ping:udp_ping module: no closed/open UDP ports known on x.x.x.x. Module test failed
[+] Host: x.x.x.x is up (Guess probability: 75%)
[+] Target: x.x.x.x is alive. Round-Trip Time: 0.00156 sec
[+] Selected safe Round-Trip Time value is: 0.00312 sec
[+] Primary guess:
[+] Host x.x.x.x Running OS: "Sun Solaris 6 (SunOS 2.6)" (Guess probability: 100%)
[+] Other guesses:
[+] Host x.x.x.x Running OS: "Sun Solaris 7 (SunOS 2.7)" (Guess probability: 100%)
[+] Host x.x.x.x Running OS: "Sun Solaris 8 (SunOS 2.8)" (Guess probability: 100%)
[+] Host x.x.x.x Running OS: "Sun Solaris 9 (SunOS 2.9)" (Guess probability: 100%)
[+] Host x.x.x.x Running OS: "HP UX 11.0" (Guess probability: 95%)
The Use of Best of Breed TCP/IP Stack Fingerprinting Techniques – Example (with)

[ root@fremont src]# ./xprobe2 -v -c ../etc/xprobe2.conf -p TCP:25:open x.x.x.x
Xprobe2 v.0.2 Copyright (c) 2002-2003 fygrave@tigerteam.net, ofir@sys-security.com, meder@areopag.net
[+] Target is x.x.x.x
[+] Loading modules.
[+] Following modules are loaded:
[ x] [ 1] ping:icmp_ping - ICMP echo discovery module
[ x] [ 2] ping:tcp_ping - TCP-based ping discovery module
[ x] [ 3] ping:udp_ping - UDP-based ping discovery module
[ x] [ 4] infogather:ttl_calc - TCP and UDP based TTL distance calculation
[ x] [ 5] infogather:portscan - TCP and UDP PortScanner
[ x] [ 6] fingerprint:icmp_echo - ICMP Echo request fingerprinting module
[ x] [ 7] fingerprint:icmp_timestamp - ICMP Timestamp request fingerprinting module
[ x] [ 8] fingerprint:icmp_addressmask - ICMP Address mask request fingerprinting module
[ x] [ 9] fingerprint:icmp_info - ICMP Information request fingerprinting module
[ x] [10] fingerprint:icmp_port_unreachable - ICMP port unreachable fingerprinting module
[ x] [11] fingerprint:tcp_handshake - TCP Handshake fingerprinting module
[+] 11 modules registered
[+] Initializing scan engine
[+] Running scan engine
[-] ping:udp_ping module: no closed/open UDP ports known on x.x.x.x. Module test failed
[+] Host: x.x.x.x is up (Guess probability: 75%)
[+] Target: x.x.x.x is alive. Round-Trip Time: 0.09717 sec
[+] Selected safe Round-Trip Time value is: 0.19434 sec
[+] Primary guess:
[+] Host x.x.x.x Running OS: "Sun Solaris 8 (SunOS 2.8)" (Guess probability: 100%)
[+] Other guesses:
[+] Host x.x.x.x Running OS: "Sun Solaris 7 (SunOS 2.7)" (Guess probability: 93%)
[+] Host x.x.x.x Running OS: "Sun Solaris 9 (SunOS 2.9)" (Guess probability: 93%)
[+] Host x.x.x.x Running OS: "Sun Solaris 6 (SunOS 2.6)" (Guess probability: 90%)
...
The Use of Best of Breed TCP/IP Stack Fingerprinting Techniques

- Combined with Xprobe2’s other fingerprinting modules, the TCP handshake module greatly enhance Xprobe2’s abilities, overall accuracy, and the ability to provide results when executed against different topologies.
Port Scanner

- The success of executing some of Xprobe2’s fingerprinting modules depends on successfully probing an open TCP port and a closed UDP port.

- Therefore we have implemented a port scanner module as an independent module to Xprobe2 0.2 RC1.

- By default Xprobe2 does not tie the port scanner module with its fingerprinting modules and therefore it maintains the minimal usage of packets to discover a targeted system’s underlying operating system.
Port Scanner, Usage

[root@fremont src]# ./xprobe2 -v -c ../etc/xprobe2.conf -s 0.1 -P -T 20-40,80
x.x.x.x
Xprobe2 v.0.2 Copyright (c) 2002-2003 fygrave@tigerteam.net,
ofir@sys-security.com, meder@areopag.net
[+] Target is x.x.x.x
[+] Loading modules.
[+] Following modules are loaded:
[x] [1] ping:icmp_ping  -  ICMP echo discovery module
[x] [2] ping:tcp_ping  -  TCP-based ping discovery module
[x] [3] ping:udp_ping  -  UDP-based ping discovery module
[x] [4] infogather:ttl_calc  -  TCP and UDP based TTL distance calculation
[x] [5] infogather:portscan  -  TCP and UDP PortScanner
[x] [6] fingerprint:icmp_echo  -  ICMP Echo request fingerprinting module
[x] [7] fingerprint:icmp_tstamp  -  ICMP Timestamp request fingerprinting module
[x] [8] fingerprint:icmp_amask  -  ICMP Address mask request fingerprinting module
[x] [9] fingerprint:icmp_info  -  ICMP Information request fingerprinting module
[x] [10] fingerprint:icmp_port_unreach  -  ICMP port unreachable fingerprinting module
[x] [11] fingerprint:tcp_hshake  -  TCP Handshake fingerprinting module
[+] 11 modules registered
Initializing scan engine
Running scan engine
ping:tcp_ping module: no closed/open TCP ports known on x.x.x.x. Module test failed
ping:udp_ping module: no closed/open UDP ports known on x.x.x.x. Module test failed
No distance calculation. x.x.x.x appears to be dead or no ports known
Host: x.x.x.x is up (Guess probability: 25%)
Target: x.x.x.x is alive. Round-Trip Time: 0.00149 sec
Selected safe Round-Trip Time value is: 0.00298 sec

Portscan results for x.x.x.x:
Stats:
TCP: 4 - open, 18 - closed, 0 - filtered
UDP: 0 - open, 0 - closed, 0 - filtered
Portscan took 2.50 seconds.
Details:
Proto Port Num. State Serv. Name
TCP 21 open ftp
TCP 22 open ssh
telnet
TCP 37 open time
Other ports are in closed state.
Primary guess:
Host x.x.x.x Running OS: "HP UX 11.0" (Guess probability: 100%)
Other guesses:
Host x.x.x.x Running OS: "HP UX 11.0i" (Guess probability: 96%)
Host x.x.x.x Running OS: "Sun Solaris 8 (SunOS 2.8)" (Guess probability: 90%)
Host x.x.x.x Running OS: "Sun Solaris 9 (SunOS 2.9)" (Guess probability: 90%)
Host x.x.x.x Running OS: "Sun Solaris 6 (SunOS 2.6)" (Guess probability: 87%)
Host x.x.x.x Running OS: "OpenBSD 2.5" (Guess probability: 78%)
Host x.x.x.x Running OS: "OpenBSD 2.9" (Guess probability: 78%)
Host x.x.x.x Running OS: "NetBSD 1.4" (Guess probability: 78%)
Host x.x.x.x Running OS: "NetBSD 1.4.1" (Guess probability: 78%)
Cleaning up scan engine
Modules deinitialized
Execution completed.
Port Scanner, Usage

- When the port scanner module is used, knowledge about opened TCP ports, and closed UDP ports will be used as parameters for other modules.

- For example, the port used for the TCP handshake module will be one that was already discovered as opened by the port scanner.

- Currently with Xprobe2 v0.2RC1 the modules which receive input from the port scanner module are the ICMP port unreachable module and the TCP handshake module.
Port Scanner, -P and -p

Start

Is -P defined?

Yes

Are the ports scanned open/closed?

No

Success with other dependent modules

Yes

Is -p defined?

No

Use defaults

Are the ports used open/closed?

Yes

Success with other dependent modules

No

Failure
Port Scanner, Controlling the Sending Stream

- A command line option, “-s”, was added to control the sending stream of packets.
- The command line controls the time interval between each SYN packet sent and/or UDP datagram sent.
- The value given is represented in milliseconds.
- If the “-s” command line option is not used, Xprobe2 RC1 will use a default time interval of 10 milliseconds between SYN packets sent and/or UDP datagram sent (i.e. 100 packets per second).
Port Scanner, Controlling the Sending Stream

- Controlling the stream of packets the port scanner will generate is an important feature, allowing one to **adjust the paste of the scan**, not allowing **denial of service conditions to be introduced** (against networking gear the packets go through, or even against the targeted machine), and **adjusting the port scanner’s paste to accommodate network and host related issues** (the network is congested, old networking gear, etc.)
Port Scanner, Controlling the Sending Stream

- In some situations one must use the "-s" option to specify a longer delay between packets sent, since a target operating system (i.e. FreeBSD, Cisco routers) might have an automatic feature to rate-limit the number of replies it sends per a certain amount of time.
Port Scanner, Controlling the Sending Stream

spanion:~/tmp/xprobe2-0.2rc1/src # ./xprobe2 -v -c ../etc/xprobe2.conf -P -U 516-520 -T 20-30 192.168.0.150
Xprobe2 v.0.2rc1 Copyright (c) 2002-2003 fygrave@tigerteam.net, ofir@sys-security.com, meder@areopag.net
[+] Target is 192.168.0.150
[+] Loading modules.
[+] Following modules are loaded:
[ x] [ 1] ping:icmp_ping - ICMP echo discovery module
[ x] [ 2] ping:tcp_ping - TCP-based ping discovery module
[ x] [ 3] ping:udp_ping - UDP-based ping discovery module
[ x] [ 4] infogather:ttl_calc - TCP and UDP based TTL distance calculation
[ x] [ 5] infogather:portscan - TCP and UDP PortScanner
[ x] [ 6] fingerprint:icmp_echo - ICMP Echo request fingerprinting module
[ x] [ 7] fingerprint:icmp_tstamp - ICMP Timestamp request fingerprinting module
[ x] [ 8] fingerprint:icmp_amask - ICMP Address mask request fingerprinting module
[ x] [ 9] fingerprint:icmp_info - ICMP Information request fingerprinting module
[ x] [10] fingerprint:icmp_port_unreach - ICMP port unreachable fingerprinting module
[ x] [11] fingerprint:tcp_hshake - TCP Handshake fingerprinting module
[ +] 11 modules registered
[ +] Initializing scan engine
[ +] Running scan engine
[-] ping:tcp_ping module: no closed/open TCP ports known on 192.168.0.150. Module test failed
[-] ping:udp_ping module: no closed/open UDP ports known on 192.168.0.150. Module test failed
[ +] No distance calculation. 192.168.0.150 appears to be dead or no ports known
[ +] Host: 192.168.0.150 is up (Guess probability: 25%)
[ +] Target: 192.168.0.150 is alive. Round-Trip Time: 0.00064 sec
[ +] Selected safe Round-Trip Time value is: 0.00128 sec
Port Scanner, Controlling the Sending Stream

[+] Portscan results for 192.168.0.150:
[+] Stats:
[+] TCP: 3 - open, 7 - closed, 1 - filtered
[+] UDP: 2 - open, 3 - closed, 0 - filtered
[+] Portscan took 0.34 seconds.
[+] Details:
[+] Proto Port Num. State Serv. Name
[+] TCP 21 open ftp
[+] TCP 23 open telnet
[+] TCP 25 open smtp
[+] TCP 30 filtered N/A
[+] UDP 517 open talk
[+] UDP 518 open ntalk
[+] Other ports are in closed state.
[+] Primary guess:
[+] Host 192.168.0.150 Running OS: "Linux Kernel 2.2.22" (Guess probability: 100%)
[+] Other guesses:
[+] Host 192.168.0.150 Running OS: "Linux Kernel 2.2.23" (Guess probability: 100%)
[+] Host 192.168.0.150 Running OS: "Linux Kernel 2.2.24" (Guess probability: 100%)
[+] Host 192.168.0.150 Running OS: "Linux Kernel 2.2.25" (Guess probability: 100%)
[+] Host 192.168.0.150 Running OS: "Linux Kernel 2.2.26" (Guess probability: 100%)
[+] Host 192.168.0.150 Running OS: "Linux Kernel 2.2.27" (Guess probability: 100%)
[+] Host 192.168.0.150 Running OS: "Linux Kernel 2.2.28" (Guess probability: 100%)
[+] Host 192.168.0.150 Running OS: "Linux Kernel 2.2.29" (Guess probability: 100%)
[+] Host 192.168.0.150 Running OS: "Linux Kernel 2.2.30" (Guess probability: 100%)
[+] Cleaning up scan engine
[+] Modules deinitialized
[+] Execution completed.
Port Scanner, Controlling the Sending Stream

spanion:~/tmp/xprobe2-0.2rc1/src # ./xprobe2 -v -c ../etc/xprobe2.conf -s 0.04 -P -U 516-520 -T 20-30 192.168.0.150
Xprobe2 v.0.2rc1 Copyright (c) 2002-2003 fygrave@tigerteam.net, ofir@sys-security.com, meder@areopag.net
[+] Target is 192.168.0.150
[+] Loading modules.
[+] Following modules are loaded:
  [x] [ 1] ping:icmp_ping - ICMP echo discovery module
  [x] [ 2] ping:tcp_ping - TCP-based ping discovery module
  [x] [ 3] ping:udp_ping - UDP-based ping discovery module
  [x] [ 4] infogather:ttl_calc - TCP and UDP based TTL distance calculation
  [x] [ 5] infogather:portscan - TCP and UDP PortScanner
  [x] [ 6] fingerprint:icmp_echo - ICMP Echo request fingerprinting module
  [x] [ 7] fingerprint:icmp_tstamp - ICMP Timestamp request fingerprinting module
  [x] [ 8] fingerprint:icmp_amask - ICMP Address mask request fingerprinting module
  [x] [ 9] fingerprint:icmp_info - ICMP Information request fingerprinting module
  [x] [10] fingerprint:icmp_port_unreach - ICMP port unreachable fingerprinting module
  [x] [11] fingerprint:tcp_handshake - TCP Handshake fingerprinting module
[+] 11 modules registered
[+] Initializing scan engine
[+] Running scan engine
[-] ping:tcp_ping module: no closed/open TCP ports known on 192.168.0.150. Module test failed
[-] ping:udp_ping module: no closed/open UDP ports known on 192.168.0.150. Module test failed
[+] No distance calculation. 192.168.0.150 appears to be dead or no ports known
[+] Host: 192.168.0.150 is up (Guess probability: 25%)
[+] Target: 192.168.0.150 is alive. Round-Trip Time: 0.00080 sec
[+] Selected safe Round-Trip Time value is: 0.00159 sec
Port Scanner, Controlling the Sending Stream

[+] Portscan results for 192.168.0.150:
[+] Stats:
[+] TCP: 3 - open, 8 - closed, 0 - filtered
[+] UDP: 2 - open, 3 - closed, 0 - filtered
[+] Portscan took 0.81 seconds.
[+] Details:
[+] Proto Port Num. State Serv. Name
[+] TCP 21 open ftp
[+] TCP 23 open telnet
[+] TCP 25 open smtp
[+] UDP 517 open talk
[+] UDP 518 open ntalk
[+] Other ports are in closed state.
[+] Primary guess:
[+] Host 192.168.0.150 Running OS: "Linux Kernel 2.2.22" (Guess probability: 100%)
[+] Other guesses:
[+] Host 192.168.0.150 Running OS: "Linux Kernel 2.2.23" (Guess probability: 100%)
[+] Host 192.168.0.150 Running OS: "Linux Kernel 2.2.24" (Guess probability: 100%)
[+] Host 192.168.0.150 Running OS: "Linux Kernel 2.2.25" (Guess probability: 100%)
[+] Host 192.168.0.150 Running OS: "Linux Kernel 2.2.28" (Guess probability: 100%)
[+] Host 192.168.0.150 Running OS: "Linux Kernel 2.2.27" (Guess probability: 100%)
[+] Host 192.168.0.150 Running OS: "Linux Kernel 2.2.26" (Guess probability: 100%)
[+] Host 192.168.0.150 Running OS: "Linux Kernel 2.2.25" (Guess probability: 100%)
[+] Host 192.168.0.150 Running OS: "Linux Kernel 2.2.24" (Guess probability: 100%)
[+] Host 192.168.0.150 Running OS: "Linux Kernel 2.2.23" (Guess probability: 100%)
[+] Cleaning up scan engine
[+] Modules deinitialized
[+] Execution completed.
A Mechanism for Automatic Calculation of the Receive Timeout

- An automatic receive timeout calculation mechanism was implemented with Xprobe2 v0.2 RC1

- The mechanism allows Xprobe2 to be **time efficient taking into account the terrain it works in and against**

- Xprobe2 is fast: 0.5 – 2.0 seconds on a Local LAN for OS fingerprinting 1 IP address

- Xprobe2 uses three receive timeouts:
  - For the **discovery modules** (ICMP ping, TCP ping, UDP ping)
  - For the **port scanner**
  - For the **fingerprinting modules**
A Mechanism for Automatic Calculation of the Receive Timeout

- Xprobe2 uses the three different discovery modules in order to calculate the receiving timeout for its fingerprinting modules.

- The timeout used is the longest round-trip time of a discovery modules used (ICMP echo, TCP ping, UDP ping) times two (RTT*2) measured in milliseconds.

- The TCP ping and UDP ping discovery modules will not be executed by default.

- In order to allow a proper receive timeout for the ICMP echo discovery module itself, one can use the “-t” command line option and specify the receiving timeout in milliseconds.
A Mechanism for Automatic Calculation of the Receive Timeout – The Port Scanner Module
Maintaining a Quality Signature Database

- Xprobe2’s signature database is tightly controlled.
- New signatures will be added to the database if, and only if, we can verify them against a test system we control or have legitimate access to.
- We see the signature database issue as a mandatory issue for the success of the tool.
- It is very easy to corrupt a signature database where it would lead to false and inaccurate results.
Maintaining a Quality Signature Database

- Xprobe2’s signature database was re-built from scratch currently containing over 160 signatures
  - All of the Microsoft based Operating Systems starting with Microsoft Windows 95. Uniquely identifying:
    - Microsoft Windows 2003 Standard Edition
    - Microsoft Windows 2003 Enterprise Edition
    - Microsoft Windows 2000 Server SP4
    - Microsoft Windows 2000 Server SP3
    - The entire Linux Kernel branches of 2.4.x, 2.2.x
Maintaining a Quality Signature Database

- FreeBSD 2.2.7, 2.2.8, 3.1, 3.2, 3.3, 3.4, 3.5.1, 4.0, 4.1, 4.2, 4.3, 4.4, 4.5, 4.6, 4.6.2, 4.7, 4.8, 5.0, 5.1
- OpenBSD 2.4, 2.5, 2.6, 2.7, 2.8, 2.9, 3.0, 3.1, 3.2, 3.3
- NetBSD 1.6.1, 1.6, 1.5.3, 1.5.2, 1.5.1, 1.5, 1.4.3, 1.4.2, 1.4.1, 1.4, 1.3.3, 1.3.2, 1.3.1, 1.3
- Cisco IOS 12.2, 12.0, 11.3, 11.2, 11.1
- And many more...
Added Functionality with Xprobe2 v0.2 RC2
Added Functionality with Xprobe2 v0.2 RC2

- Support for TCP Timestamp fingerprinting tests
  - First discussed at Blackhat USA 2003

- XML Output

- Automatic signature generation
  - Using the – F command line option

- Some cosmetic changes:
  - – L lists numbers for available modules
How to effectively use Xprobe2 in a Corporate environment
How to effectively use Xprobe2 in a Corporate environment
Xprobe2 v0.2 RC2 Demo
The Future of Active Operating System Fingerprinting
The Future of Active Operating System Fingerprinting

- The *terrain* we scan from and against is a major factor in producing accurate results.

- One cannot compare the *scanning conditions* of a local network versus a well fortified Internet web site.

- With the latter the number of TCP/IP based stack fingerprinting tests that can be successfully used is limited, usually to the opened service(s) on the Internet connected system (i.e. TCP port 80).
The Future of Active Operating System Fingerprinting

- We have already noted that an active operating system fingerprinting tool usually must use several different operating system fingerprinting tests in order to provide with accurate results.

- With only a limited success of its fingerprinting tests, the quality of the results produced by an active operating system fingerprinting tool will be significantly degraded.

- A way to compensate for operating system fingerprinting tests which we would not be able to use in situations that will prove them useless must be found.
The Future of Active Operating System Fingerprinting

- Another issue, which needs to be resolved, is the inability to differentiate between different operating systems of the same manufacture.

- A part of the remedy is to use application layer based fingerprinting tests tailored towards the services found opened on the targeted system(s) and/or a service commonly found with the operating system family in question.

- The criteria for adopting one such test, is that it should be hard or impossible to trick the test, and that it will produce accurate results when executed against well fortified Internet connected systems.
The Future of Active Operating System Fingerprinting

- Traditional TCP/IP stack based operating system fingerprinting tests must be tailored for maximum impact over the overall fingerprinting results when used in situation in which the targeted system has a limited exposure.

- The TCP/IP stack based operating system fingerprinting tests to be used must not be easily defeated by commonly found defense systems (i.e. a SYN | FIN scan will be useless when a site is being defended by a stateful inspection-based firewall).

- The other part of the remedy is much harder to be implemented:
Host Discovery

Results

Assessment Stage

i.e. is the target firewalled?

Port Scan

Results

Conclusion

What Are the Effective Fingerprinting Modules?

Fingerprinting Modules Execution

Results

Conclusion

Should Module (x) be Executed?

Fingerprinting Modules Execution

Results

Conclusion

Are the Result Decisive?

Niche Fingerprinting Modules Execution

Results

End Fingerprinting Run
Credits

- Sebastian Krahmer – libusi++
- Scut – Fuzzy logic comments and suggestions
- I. Levin – Fuzzy implementation comments
- Elie a.k.a. Lupin Bursztein – Patches, bug fixes
- Those behind the usage of hashing for port scanning (done in 1998 by an Israeli Hacker’s group)
Further Reading

- Arkin Ofir, “ICMP Usage in Scanning” research project
  
  http://www.sys-security.com

  

- Arkin Ofir & Fyodor Yarochkin, “X – Remote ICMP based OS fingerprinting Techniques”, August 2001 (This paper describes the first generation of Xprobe).
  
Further Reading

  
  [http://www.sys-security.com/archive/phrack/p57-0x07](http://www.sys-security.com/archive/phrack/p57-0x07)


Questions?