# Evidence Artifacts and Detailed Analysis

We started by booting up a Kali Distribution in Forensics Mode.

## Exercise 1.1

At the instance **2011-10-11 at 12:52:54** the client requests a malicious HTML page from <http://10.20.0.111:8080/banking.htm>. And this way the attack begun.

This site has infected with a GIF image that allow the attacker to exploit the client using the [CVE-2010-0249](https://nvd.nist.gov/vuln/detail/CVE-2010-0249).

|  |
| --- |
| Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1; SV1) |

Evidence 1: Client's browser user agent

The client (victim) was using the Internet Explorer version 6 and using Windows XP SP2 (Windows NT 5.1).

This exploit allows the attacker to execute malicious code on the client, therefore it is very likely that somewhere in the future the attacker would have sent some executable. At the instance **2011-10-11 at 12:53:13,** 4709 the client shuts down every TCP connection and stablishes one to **10.20.0.111**. This connection reveals an interesting and expected thing: the server is sending a malicious file to the client.

But how do we were able to know that is an executable? Since MS-DOS times every Windows compiled application carries a message that is shown when we run that windows application on a MS-DOS machine. That message is: “**This program cannot be run in DOS mode**”. Every application will have this message and so it is very likely that this may be a malicious application because this message is present (TCP Stream 104 🡪 4726 Pac. No.)

We were able to recover the following files from PCAP:

|  |  |
| --- | --- |
| File Name | MD5 |
| rawExtractChanged | ca0977e1038f25894d2ab1a837e361da9e687695 | |
| PE32 executable (DLL) (GUI) Intel 80386, for MS Windows | |

Evidence 2: File requested by the infected client

We perform an analysis on this file on a clean sandboxed Windows machine in order to retrieve more information about this mysterious file.

This file is a known malware identified by most antivirus products as **Trojan: Win32/Swrort.A[[1]](#footnote-2)**, whose goal is “… give a malicious hacker access to your PC to download other malware”, and it is obtained by “visiting a hacked or compromised webpage”. Which corresponds to this case.

To protect accidental execution of the malware, it is encrypted on a ZIP file with the password: “malware”.

## Exercise 1.2

The first strange thing that we discovered was the tremendous ping requests from the same machine (**10.20.0.110**) to several different machines whose IPs belong to the network **192.168.10.0/24**. This is an obvious Ping Scan.

The team suspects that this scan has the purpose of finding online hosts on the local network. These scans started at frame 8 (**2011-11-11 13:46:34**).

We observed at **13:46:42** **(**Frame **334)**, that **10.20.0.110** tries to find machines with HTTPS server running (port 443). 5 seconds later (**13:46:47**) **(**Frame **694)**, the machine tries to send an ACK message to port **80** of every machine in the network.

The team suspects that this approach changes (search specific for HTTPS and then HTTP) comes from SSH messages. In fact, these messages are received at the following time:

* 13:46:19 (Frame 1);
* 13:46:24 (Frame 5);
* 13:46:35 (Frame 30);
* **13:46:39 (Frame 219)**;
* 13:46:50 (Frame 863);
* 13:46:54 (Frame 1104);
* 13:47:09 (Frame 1959)
* ...

At this point the team wants to know what kind of tool / malware performs these actions. First, we tried to gather information that could leak what tool was used:

* TCP window size is always 1024 bytes.

With a [search on the Internet](https://bencane.com/2013/02/25/10-nmap-commands-every-sysadmin-should-know/) we found a tool, which is used by most network administrators to perform a variety of network operations such as finding which ports are open in the machines of the network. This tool is called **nmap**. The **TCP SYN** packets have a constant window size of 1024 which coincides with the behavior.

At **13:50:55 (**Frame **2119)**, the machine **10.20.0.110** tries to scan the following ports on the victim **192.168.10.11**:

* MySQL server (3306);
* HTTP server (80);
* SSH server (22);
* Telnet (23);
* HTTPS (443);
* IMAP (143);
* And many other (less known) protocols.

At minute 11 (from the start of the capture) **13:57:20** (Frame **17545**), another SSH message arrives and we suspect that UDP flag (-sU) was removed from nmap, because there are no more UDP messages (until **14:15:35** Frame **42440**). This forces nmap to check for opened port using only TCP and SYN and ACK messages.

At the final scan (last SSH message arrives at **14:17:10**, Frame **47210**) and we can see that the scan tries to obtain header information from FTP protocols and from HTTP: (For more detailed information check TCP Stream 23359)

|  |
| --- |
| GET / HTTP/1.0 |

Evidence 3: nmap request

|  |
| --- |
| HTTP/1.1 200 OK  Content-Type: text/html  Last-Modified: Fri, 30 Sep 2011 23:32:29 GMT  Accept-Ranges: bytes  ETag: "d8877838c97fcc1:0"  Server: **Microsoft-IIS/7.5**  X-Powered-By: ASP.NET  Date: Fri, 11 Nov 2011 14:17:19 GMT  Connection: keep-alive  Content-Length: 3069  … |

Evidence 4: server's reply

## MD5 Checksum of recovered files

|  |  |
| --- | --- |
| File name | MD5 |
| banking.gif | df3e567d6f16d040326c7a0ea29a4f41 |
| banking.htm | a576776febc36992086fe9a7d2662836 |
| c.gif | 32023bb33cfb2a1990a4ef2d85b6ac16 |
| malware.zip | 4ec19a35af2e9411f6d0e63a12d46a29 |
| | rawExtract.exe | da1447c27196f9d8bcbfff0dcd14780a |
| | rawExtractChanged.exe | b2ea0abb7f05298e8cea86c37d58dc18 |

Artifact 1: List of recovered files

1. More information about this trojan at [Microsoft Malware Encyclopedia](https://www.microsoft.com/en-us/wdsi/threats/malware-encyclopedia-description?name=Trojan%3aWin32%2fSwrort.A&threatid=2147630763) [↑](#footnote-ref-2)