

COS20030

**Malware Analysis**

*Lab 1*

**Abnormal behaviour, PCAP Analysis, Introduction to Ghidra**

# Purpose

This is the first lab in the Malware Analysis subject. Before we start learning how to analyse malware, we have to first learn what is the normal behaviour of the Windows OS and what is not.

In the first part of this lab, we will learn about normal characteristics and behaviour of some of the main processes of Windows Operating System. We will also learn about some examples of abnormal behaviour of these processes which could be signs of a malware infection on a machine.

In the second part of this lab, we will analyse network PCAP(Packet Capture) files to find malicious network activities.

We will end this lab by starting to use Ghidra, a free open-source disassembler tool. A disassembler is one of the most important tools in a malware analyst’s toolset. In later labs we will learn how Ghidra can be used in malware analysis.

# Outcome

* Learn to detect abnormal behaviour of OS essential processes
* Learn to use WireShark and NetMiner for network analysis
* Learn to use Ghidra to disassemble binary files

# Abnormal system behaviour

Malwares often try to pretend to be a part of the operating system in order not to raise any suspicion. To detect this sort of behaviour, it’s important to learn what is the normal behaviour of the system processes and what is the abnormal behaviour.

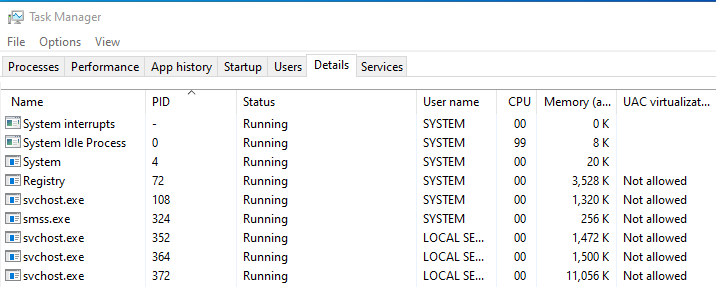
The following article summarises the normal and abnormal behaviour of some of the essential processes of the Windows operating system.

[https://0xcybery.github.io/blog/Core-Processes-In-Windows-System##smss.exe-(Session-Manager-Subsystem)](https://0xcybery.github.io/blog/Core-Processes-In-Windows-System)

Read the article and based on the knowledge you gained, try to find the abnormal behaviour shown in each screenshot below.

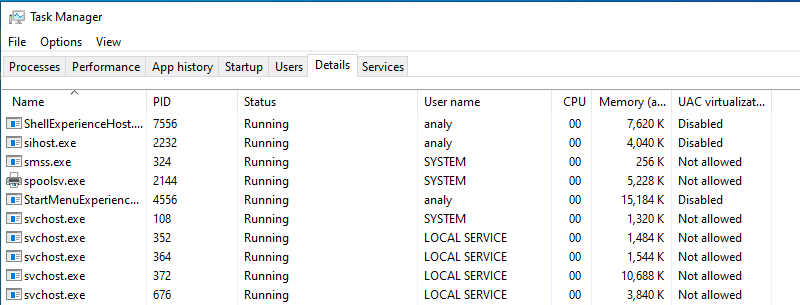
##### Exercise 1

|  |  |
| --- | --- |
| Question | Answer |
| List all the abnormalities seen in the screenshot | System pid not 4, multiple svchost.exe, one of them has high memory usage |



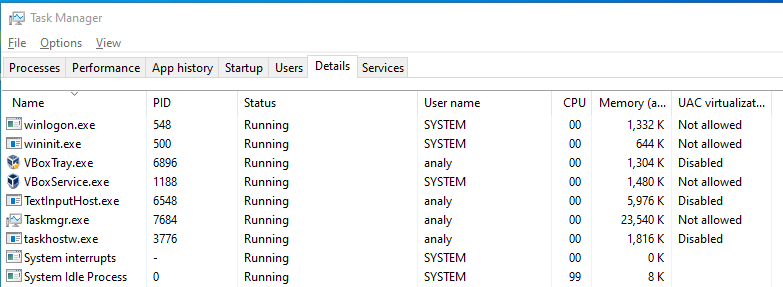
##### Exercise 2

|  |  |
| --- | --- |
| Question | Answer |
| List all the abnormalities seen in the screenshot | Multiple instances of smss.exe, unusual PIDs for system processes, high memory usage by StartMenuExperienceHost, UAC virtualization is disabled |



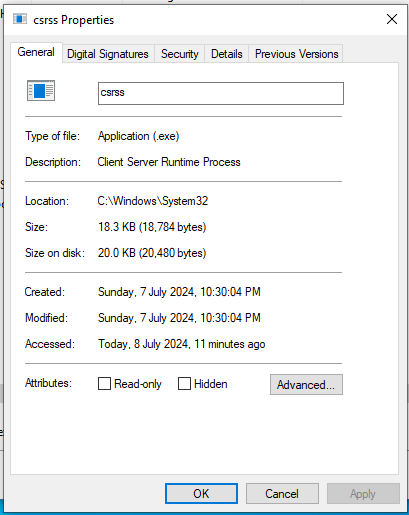
##### Exercise 3

|  |  |
| --- | --- |
| Question | Answer |
| List all the abnormalities seen in the screenshot | Multiple instances of virtual machine (only one should be running), unusual memory usage by Taskmgr.exe (23k), high cpu usage by System Idle Process, system interrupts shows 0k memory. |



##### Exercise 4

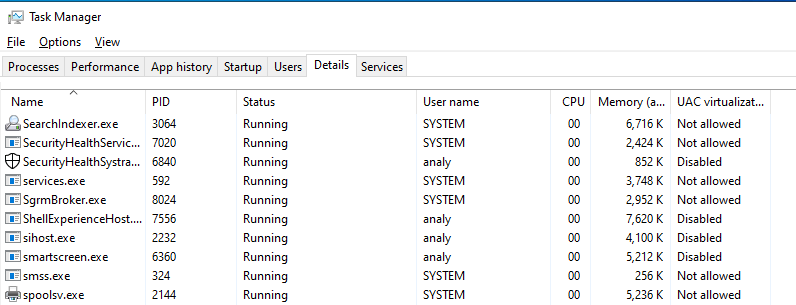
|  |  |
| --- | --- |
| Question | Answer |
| List all the abnormalities seen in the screenshot |  |



##### 

##### Exercise 5

|  |  |
| --- | --- |
| Question | Answer |
| List all the abnormalities seen in the screenshot |  |



# PCAP Analysis

##### Sample 1

In this exercise we want to analyse a packet capture (PCAP) file which contains the network traffic captured from a system infected with a malicious downloader program. The malicious program connects to a remote server, downloads another malware and executes this malware. The downloaded malware is a sample of the Lokibot malware. You can read about this malware here:

<https://www.cisa.gov/news-events/cybersecurity-advisories/aa20-266a>

In the PCAP file there is evidence of connection to the remote server and downloading the LokiBot executable file. We are also aware that LokiBot was also executed on the infected machine and attempted to communicate with its command and control server.

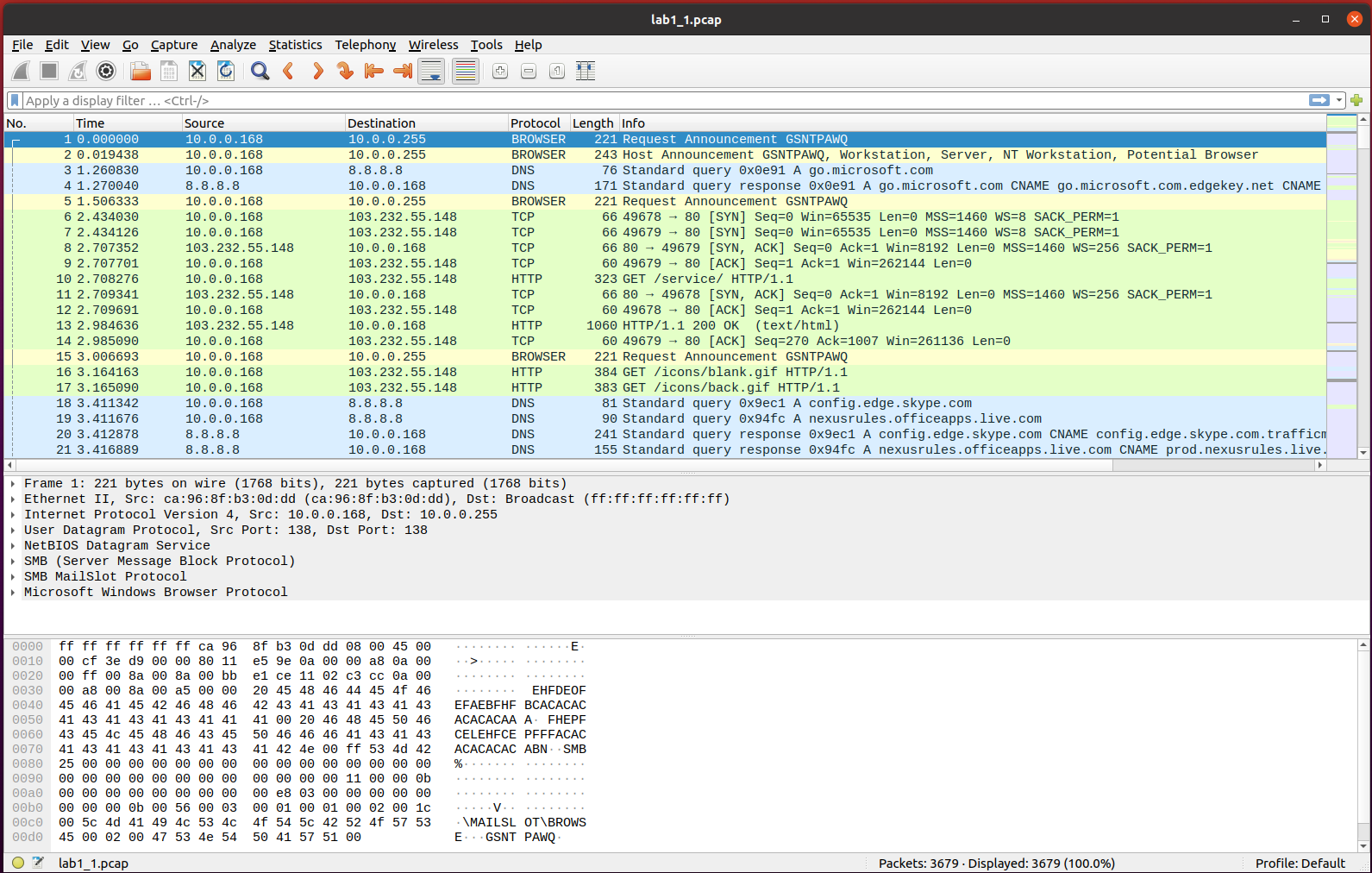
Our goal is to find the address of the download server, extract the downloaded LokiBot and find the address of the server that LokiBot communicates with.

###### WireShark

First tool to use is WireShark. Wireshark is a tool capable of capturing network traffic on a machine as well as analysing a PCAP file created previously on the same machine or on a different machine.

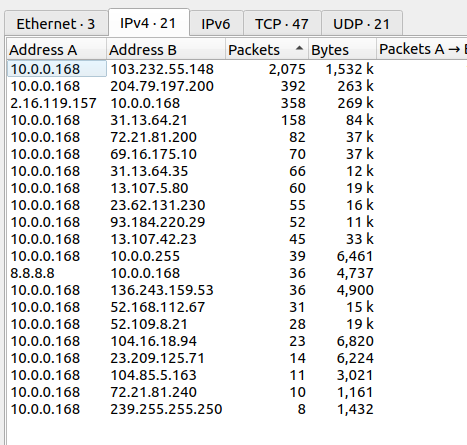
1. Extract lab1.zip with the password “infected”
2. Open Wireshark via its desktop shortcut
3. From File-> Open, navigate to the extracted lab1\_1.pcap file.

Once the PCAP file is loaded, all the sent and received packets are shown. You can use different features of this tool to find valuable information about the network traffic. You can also create different filters to see only specific packets.



Let’s first find the IP address with which the highest number of packets were communicated. This can be the download server's IP address, but we need to confirm this hypothesis.

1. Go to Statistics-> Conversations
2. Switch to IPv4 tab
3. Click on “Packets” column to sort the list by number of packets transmitted between each couple of IP addresses



From this list we can clearly see the IP address of the infected machine is 10.0.0.168 and we also found the IP address of the download server we were looking for.

Now, let’s look at the network traffic between the source IP and this IP address. To only see the communication between these two IP addresses we need to create a filter with the following instruction:

1. Right-clicking on the line which shows these two IP addresses and select

“Apply As Filter -> Selected -> A A black line with arrows

Description automatically generated B”

1. Close the statistics window and return to the main Wireshark window.
2. You can add more items to the filter created for you. In this case, let’s add “http” to the current filter, like below:



The packets are now filtered to only HTTP packets transmitted between the infected machine and the suspected IP address.

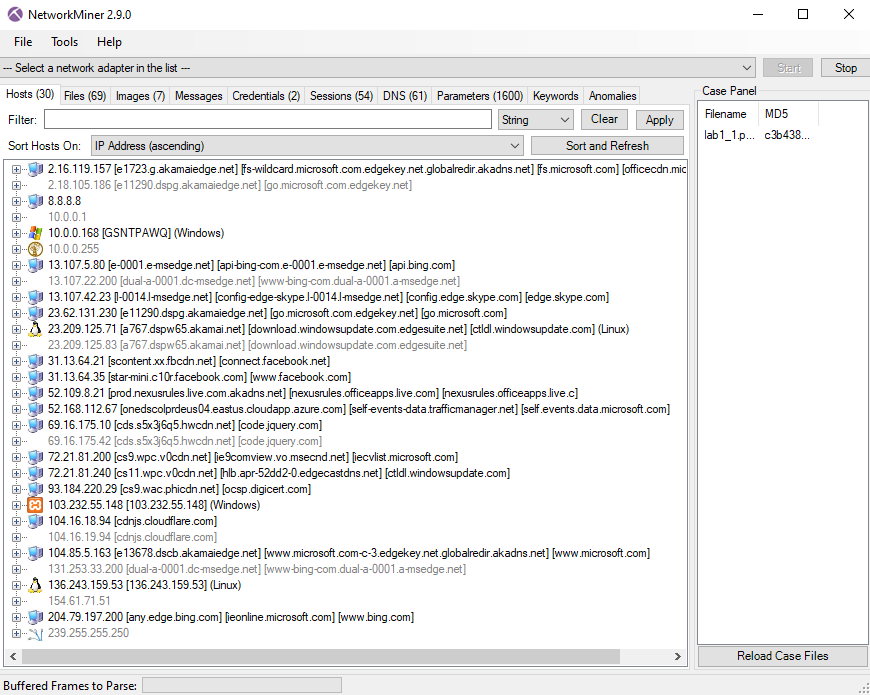
1. Now, just scroll down and look through the “Info” column to find a reference to a GET packet requesting a file with “.exe” extension. This is the LokiBot sample downloaded from the remote server.
2. Right click on the line showing the GET request and choose “Follow -> TCP Stream” to see the full TCP communication in this session.



In the TCP stream, we can see the executable file being downloaded from the server. “MZ” string at the beginning of the packet is the signature for the Windows Portable Executable(PE) file. We will learn more about PE files in later weeks.

###### NetworkMiner

Next step is to extract this file for further investigation (note: we are not going to analyse this file in this lab). In order to extract the downloaded file from a PCAP file, we use a different tool called NetworkMiner.

1. Open NetworkMiner via its desktop shortcut
2. From File -> Open, navigate to the lab1\_1.pcap file to see the extracted information like the screenshot below.

NetworkMiner is a very useful tool for network analysis and it can extract important details from PCAP files. It also automatically extracts all the files being sent or received in the network traffic.

In the first tab which is the “Hosts” tab, you can find various details about all the hosts found in the PCAP file listed under their IP addresses. Remember, we found the IP address where the LokiBot sample is downloaded from, using WireShark.

1. Find the IP address where the LokiBot sample is downloaded from in this list.
2. Click on the + next to the IP address to expand the details and fill in the table below.

|  |  |
| --- | --- |
| Question | Answer |
| IP Address | 10.0.0.168 |
| MAC address | CA968FB30DDD |
| The type of OS | Windows |
| Number of incoming sessions | 0 |

1. From the “Files” tab, find the line which corresponds to the LokiBot sample. You can do this, by looking for “exe” in the “Extenston” column.
2. Right-click on the line and choose “Open Folder” to find the extracted file. Make a copy of the file in the “Malware Samples/lab1/” folder and rename it to lab1\_3.exe\_. You will use this file to practise using Ghidra in the last part of this lab.

Go to Windows Defender

A screenshot of a computer

Description automatically generated

Click on Virus & threat protection

A screenshot of a computer

Description automatically generated

Click on Protection History

A screenshot of a computer

Description automatically generated

Go to the last “Threat blocked” 🡪 Click on that 🡪 Yes/No popped up 🡪 Choose Yes

A screenshot of a computer

Description automatically generated

From Actions: Choose Allow

Go to the second last “Threat blocked” 🡪 Yes/No popped up 🡪 Choose Yes

A screenshot of a computer

Description automatically generatedFrom Actions: Choose Restore 🡪 Now you should be able to see the file.

A screenshot of a computer

Description automatically generated

###### 

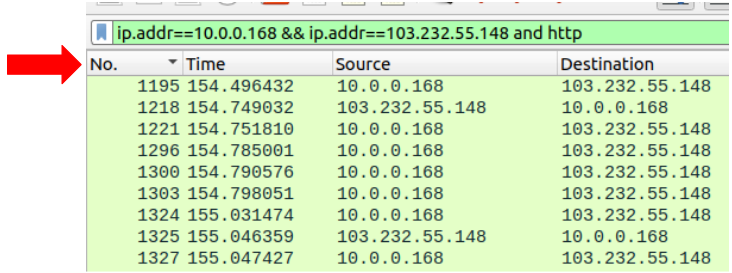
###### LokiBot communication with its command-and-control server

Our next goal is to find the address of the command and control server which the LokiBot sample(audiodg.exe file) communicates with, from the same lab1\_1.pcap file.

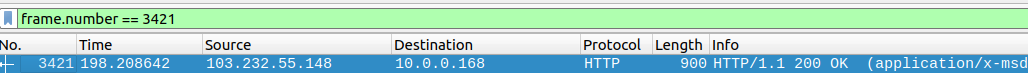
Because the communication of the LokiBot sample and its server could only happen after LokiBot is downloaded on the machine, we need to filter out all the network packets prior to the download of the LokiBot sample.

First step is to find the last packet of communication before the LokiBot sample is downloaded.

1. Keep the previous filter and click on the first column which is the “No.” (Frame Number) column to sort the packets based on their frame number.



1. Scroll all the way down to find the last packet.
2. Right-click on the frame number and create a filter by choosing “Apply As a Filter -> Selected”.



1. Edit the filter to show the frame numbers greater than the value and also add “http” to the filter.

A screen shot of a computer

Description automatically generated

If you have done everything correctly, you can see the command and control server’s IP address in the packets shown.

1. Right-click on the first HTTP packet and choose “Follow->TCP Stream” to see the whole communication in this session.
2. Fill in the table below to summarise your findings.

|  |  |
| --- | --- |
| Question | Answer |
| What is the Command and Control server’s IP address? | 136.243.159.53 |
| What does the “Info” column show for POST requests? This is the URI used for POST requests. | POST /~element/page.php?id=484 HTTP/1.0 |
| There is a specific string ending in “.ru” sent to the server. Find this string in the packet data. What is the full string? | ckav.ru |

##### Sample 2

Use the lab1\_2.pcap file for this exercise. In this exercise, we have a PCAP file that is captured from a machine infected with multi-stage malware.

The PCAP file starts with the packets generated by the first-stage malware which connects to a domain and downloads two files. The first file is downloaded as a base64-encoded stream and the second one is a password-protected zip file.

The second stage of the attacks is decoded from the base64-encoded stream. This stage then generates more network traffic. The second stage malware is very deceptive malware and is responsible for downloading the final payload of the attack. It uses a number of tricks to imitate the normal network traffic of the system. For example, it adds fake “host” and fake “referrer” fields to the HTTP packet headers to mimic normal network traffic. It also uses “.js” as the extension of the file it downloads from its server, even though the file is a sophisticated standalone shellcode and not a JavaScript file.

The goal of this exercise is to find information about the malicious domains used for downloading malware files.

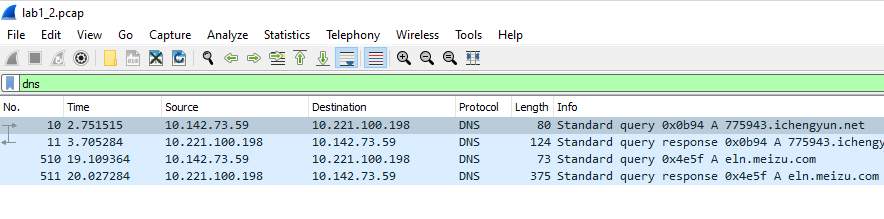
Go through the following steps and fill the table with relevant information.

1. Open WireShark via its desktop shortcut
2. Navigate to lab1\_2.pcap from “File -> Open” menu
3. Go to “Statistics -> Conversations” and switch to IPv4 tab
4. Find the top two IP addresses where the highest number of packets were sent to. These IP addresses are the destination of the network packets and are found in the “Address B” column.
5. Write down the IP addresses in the Table below.

|  |  |
| --- | --- |
| Question | Answer |
| What is the destination IP address that received the highest number of packets? | 210.16.189.8 |
| What is the destination IP address that received the second highest number of packets? | 122.225.34.198 |

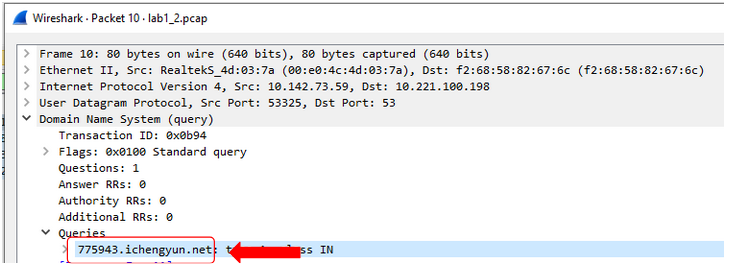
Next task is to find the domain names that were hosted on these two IP addresses. In order to do that, we need to find the DNS requests for the domains resulting in these two IP addresses.

1. Close the statistics window and go back to the main window of WireShark.
2. Create a filter on “dns” to see all the DNS packets.

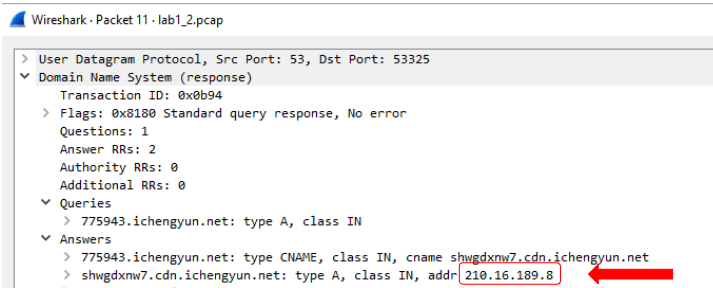


There are two sets of DNS query and DNS query response in the log. Each set belongs to one of the domains and the IP address it's hosted on.

1. Double-click on the first DNS query packet and from the top windows, scroll down to the last field which is “Queries”
2. Expand the details of this field and find the domain.



1. Close this window and double-click on the DNS query response packet which is the next packet in the packets shown in the main wireshark window.
2. scroll down to the last field which is “Answers” and expand the details to find the IP address.



1. Repeat the steps 8 to 11 for the second pair of DNS query and DNS query response packets.
2. Fill out the table below.

|  |  |
| --- | --- |
| Question | Answer |
| Domain from the first stage of the attack | 775934.ichengyun.net |
| The IP address hosted the first domain | 210.16.189.8 |
| Domain from the second stage of the attack | eln.meizu.com |
| The IP address hosted the second domain | 122.225.34.0 |

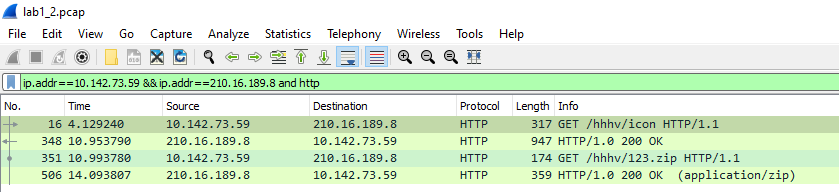
Next task is to find the URIs where the malicious files are downloaded from for each of the two IP addresses.

1. Open Statistics -> Conversations and switch to IPv4 tab.
2. Right-clicking on the line which shows the first malicious IP address and select

“Apply As Filter -> Selected -> A A black line with arrows

Description automatically generated B”

1. Close the statistics window and return to the main Wireshark window.
2. You can add more items to the filter created for you. In this case, let’s add “http” to the current filter, like below



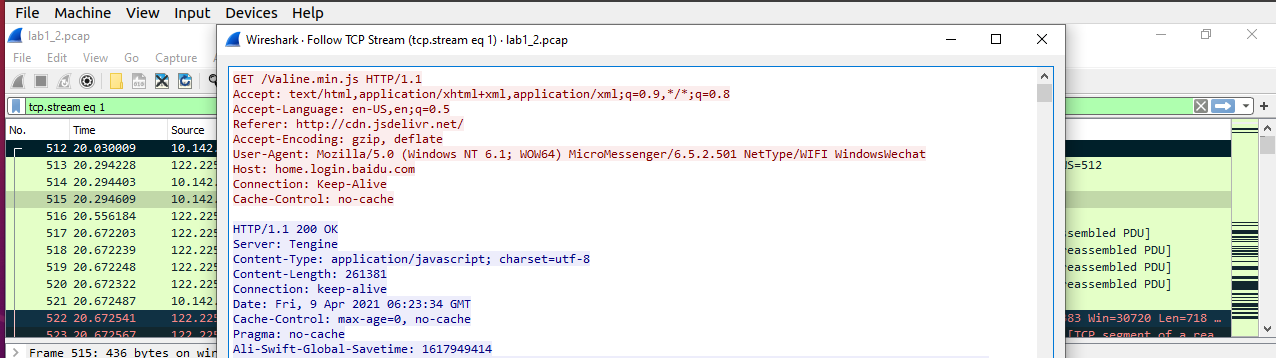
1. From the “Info” column, find the URIs for GET requests.
2. Repeat the steps 14 to 18 for the second malicious IP address and find the URL for downloading a file with “.js” extension.
3. Fill out the following table.

|  |  |
| --- | --- |
| Question | Answer |
| The URIs were the files downloaded from the first domain | GET /hhhv/icon HTTP/1.1 and GET /hhhv/123.zip HTTP/1.1 |
| The URIs were the files downloaded from the second domain | GET /Valine.min.js HTTP/1.1 |

Last task of this exercise is to find the fake “host” and fake “referrer” values the malware uses for its network packets to blend-in with normal network traffic of the machine.

To find “host” and “referrer” values, we need to look at the network packet content.

1. Right-click on the packet found in step 19 and choose “Follow -> TCP stream” to see the details of the packet data like the screenshot below.



1. From the top of the window, you can find the “Host” and “Referrer” values. Because we already know the malicious domains used in this malicious network communication, we can confirm that these “Host” and “Referrer” values are fake.
2. Fill out these values in the table below.

|  |  |
| --- | --- |
| Question | Answer |
| Fake “Host” value | home.login.baidu.com |
| Fake “Referrer” value | http://cdn.jsdelivr.net/ |

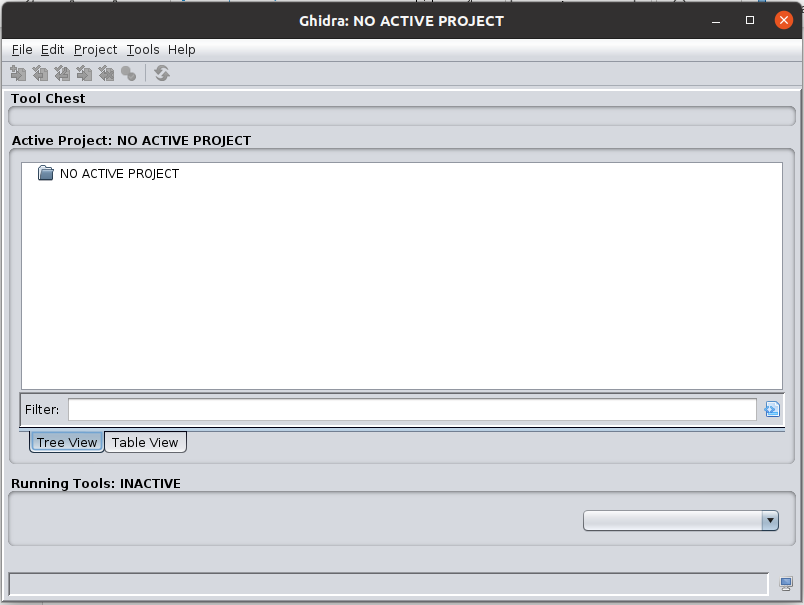
# Use Ghidra for static analysis

Ghidra is a free and open-source reverse engineering tool developed by the National Security Agency of the United States. Although the tool is relatively new compared to other tools with similar capabilities, it has been widely used by security experts since its release in 2019.

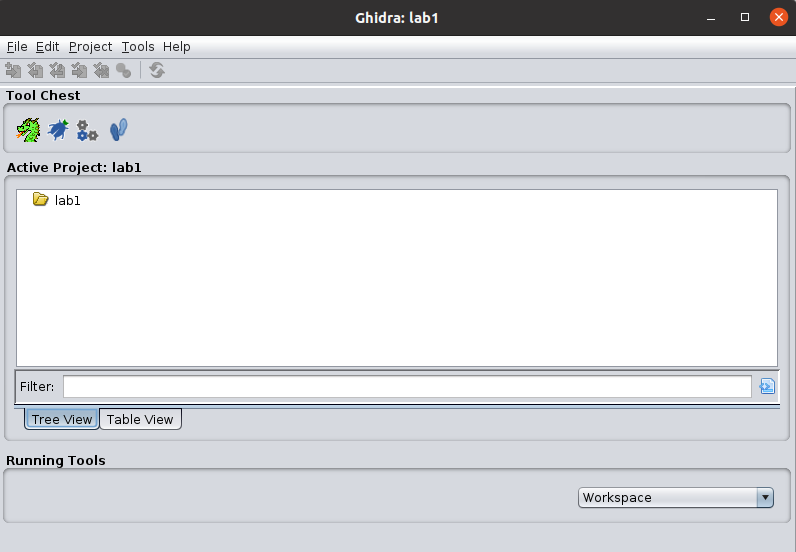
In this part of the lab, we are going to use Ghidra to have a quick look at the disassembled and decompiled code of a compiled program without getting into too much detail.

Execute Ghidra via its desktop shortcut. Everything in Ghidra happens in a project. Therefore, we need to create a new project for our analysis and then import the sample into this project.

The first time you open Ghidra, you will see a dialogue that shows there is no active project.

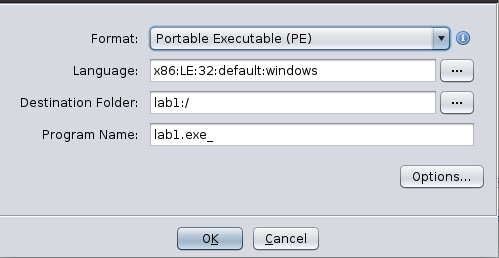


1. Go to File-> New Project and choose Non-shared Project.
2. Then you need to choose a name for your project. Write lab1 in the Project Name field. Now, you will see the project is created.



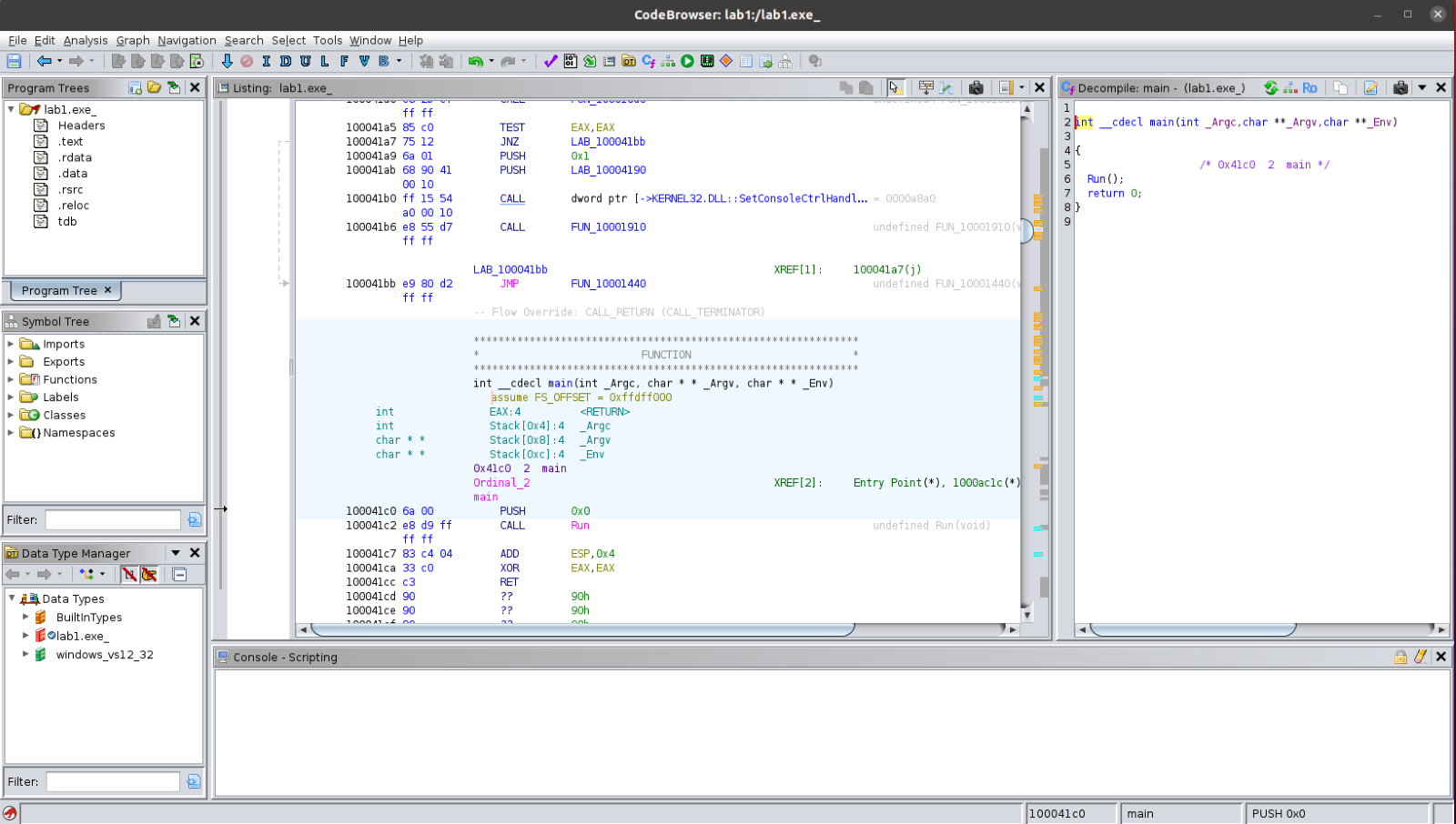
1. From File -> Import file, import the lab1\_3.exe\_ file to this project.

In the next window, you can see that Ghidra automatically recognises the file format and the platform the file is compiled for.



1. Click “OK” on the next two steps and allow the file to be imported.
2. Finally, double-click on the imported file to start the analysis.
3. When asked if you want to analyse the file, click “Yes” and then “Analyze”.

Once the initial analysis by Ghidra is finished and the file is ready for our investigation, you will see the following layout.



In Ghidra you can see the disassembly and the decompiled code in the two main windows next to each other. As the two windows are synchronised, you can navigate through the code in each of the windows. To move to a new function, you can simply double-click on the name of the function in the code and to go back to the previous function, you can use Alt + left.

To better see the functions of the executable file, go to Window -> Functions.

Familiarise yourself with the layout of the tool. We will use this tool for analysis in the coming weeks.

*End of Lab*