Lab 6

The purpose of this lab was to introduce **static malware analysis**, which means analysing a file **without executing it**. This is how malware analysts **safely triage suspicious files** before deciding whether deeper reverse engineering is required.

Instead of analysing real malware (which is dangerous), we analysed a **benign Windows executable**:

- Process Monitor (Procmon.exe)

By doing this, I learned how security analysts:

- Extract **Indicators of Compromise (IOCs)**
- Inspect **Portable Executable (PE) files**
- Detect suspicious behaviour using **YARA rules**
- Build a **complete static investigation workflow**

Create function for computing of the hash and output

Lab 6

```python
import re

def extract_strings(path):
    with open(path, "rb") as f:
        data = f.read()
        pattern = rb"[ -~]{4,}"
        return re.findall(pattern, data)

strings = extract_strings(sample)
for s in strings[:20]:
    print(s.decode(errors="ignore"))
```

0.0s

Extraction of strings and output

```
!This program cannot be run in DOS mode.
V*0T
0RichU
.text
`.rdata
@.data
.rsrc
@.reloc
hpqQ
h`EN
h|nN
h\nN
hlnN
=UUU
h_rM
hDLN
h`GO
hDLN
h|GO
hDLN
```

```python
import pefile

pe = pefile.PE(sample)

print("Entry Point:", hex(pe.OPTIONAL_HEADER.AddressOfEntryPoint))
print("Image Base:", hex(pe.OPTIONAL_HEADER.ImageBase))

print("\nImported DLLs and functions:")
for entry in pe.DIRECTORY_ENTRY_IMPORT:
    print(" ", entry.dll.decode())
    for imp in entry.imports[:5]:
        print(" -", imp.name.decode() if imp.name else "None")
```

0.4s Python

Importation of DLLS and function

Lab 6

```
Entry Point: 0xa7f70
Image Base: 0x400000

Imported DLLs and functions:
  WS2_32.dll
 - getsockname
 - listen
 - recv
 - closesocket
 - socket
  VERSION.dll
 - GetFileVersionInfoW
 - VerQueryValueW
 - GetFileVersionInfoSizeW
  COMCTL32.dll
 - ImageList_ReplaceIcon
 - ImageList_SetBkColor
 - ImageList_AddMasked
 - ImageList_BeginDrag
 - ImageList_EndDrag
  FLTLIB.DLL
 - FilterSendMessage
 - FilterGetMessage
 - FilterReplyMessage
```

```
   COMCTL32.dll
  - ImageList_ReplaceIcon
  - ImageList_SetBkColor
  - ImageList_AddMasked
  - ImageList_BeginDrag
  - ImageList_EndDrag
   FLTLIB.DLL
  - FilterSendMessage
  - FilterGetMessage
  - FilterReplyMessage
  - FilterConnectCommunicationPort
 ...
  - DwmSetWindowAttribute
  - DwmDefWindowProc
   ntdll.dll
  - RtlGetVersion
```

Output is truncated. View as a scrollable element or open in a text editor. Adjust cell output settings...

Lab 6

Application of YARA Rule

Full code and output

Lab 6

```python
import hashlib, pefile, re, yara

# sample = "samples/procmon.exe"

def compute_hashes(path):
    algos = ["md5", "sha1", "sha256"]
    output = {}
    for a in algos:
        h = hashlib.new(a)
        with open(path, "rb") as f:
            h.update(f.read())
        output[a] = h.hexdigest()
    return output

def extract_strings(path):
    with open(path, "rb") as f:
        data = f.read()
        return re.findall(rb"[ -~]{4,}", data)

print("Hashes:", compute_hashes(sample))
print("\nStrings:")
print(extract_strings(sample)[:10])
```

```python
print("\nImports:")
pe = pefile.PE(sample)
for entry in pe.DIRECTORY_ENTRY_IMPORT:
    print(entry.dll.decode())

print("\nIOCs:")
decoded = open(sample, "rb").read().decode(errors="ignore")
print("URLs:", re.findall(r"https?://[^\s\"']+", decoded))
print("IPs:", re.findall(r"\b\d{1,3}(?:\.\d{1,3}){3}\b", decoded))

print("\nYARA:")
rule = yara.compile(source="""
rule Simple {
    strings: $s = "http"
    condition: $s
}
""")
print(rule.match(sample))
```

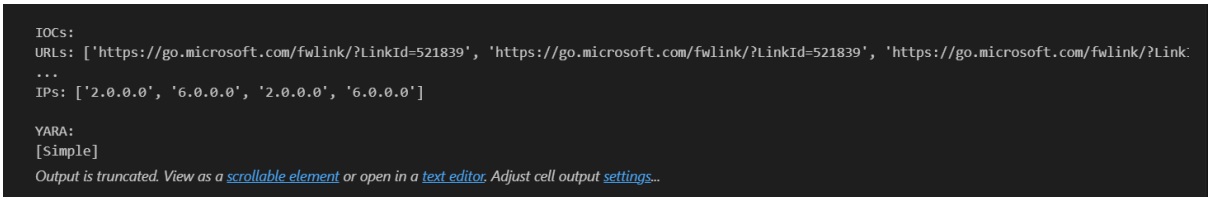
0.5s Python

```
Hashes: {'md5': 'c3e77b6959cc68baee9825c84dc41d9c', 'sha1': 'bc18a67ad4057dd36f896a4d411b8fc5b06e5b2f', 'sha256': '3b7ea4318c3c1508701102cf966f65(

Strings:
[b'!This program cannot be run in DOS mode.', b'V*0T', b'0RichU', b'.text', b'`.rdata', b'@.data', b'.rsrc', b'@.reloc', b'hpqQ', b'h`EN']

Imports:
WS2_32.dll
VERSION.dll
COMCTL32.dll
FLTLIB.DLL
KERNEL32.dll
USER32.dll
GDI32.dll
COMDLG32.dll
ADVAPI32.dll
SHELL32.dll
ole32.dll
OLEAUT32.dll
SHLWAPI.dll
UxTheme.dll
dwmapi.dll
ntdll.dll
```

Lab 6



DLLS, Urls and IPS