**SIT703 ADVANCED DIGITAL FORENSICS**

**2019 TRIMESTER 2**

**PRACTICAL SESSION 10: Malware analysis**

### Overview

In this class, we will work with malware and learn the basics of malware analysis or reverse engineering. Malware stands for “*malicious software*”, which is any software that causes a user’s PC to behave in an unintended way that compromises confidentiality, integrity and availability. 98% of cybersecurity breaches involve malware in at least 1 stage of the attack. Most often, it delivers the final payload that performs the malicious act. It can also be used to generate files and documents that are utilised by attackers, or design web pages that trick people into giving away their access credentials.

Malware analysis is a vital part of the incident response process. Knowing what the malware has performed in a victim’s PC or network helps inform your next steps: containment, eradication and recovery.

Malware analysis is divided into two categories: static and dynamic:

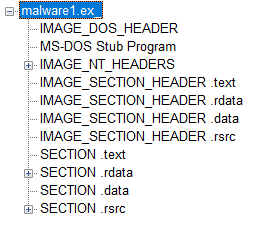
* In static analysis, the analyst reviews the code and extracts *indicators of compromise* “IoC” that help identify the intended behaviour of a malware sample. This steps also allows the extraction of signatures, utilised by traditional anti-malware software. It is important to note that in this type of analysis, the malware sample is not executed.
* In dynamic analysis, the malware sample is executed in a safe and isolated environment (called “*Sandbox*”) to determine how the malware behaves. Several tools are used to monitor several aspects and components of the operating system. This type of analysis usually yields a higher amount of information than static analysis.

Traditionally, malware is delivered to victims as executable files (.exe), however, several file types can be used to do so:

* Other types of Windows executable files such as .bin, .cmd, .com, msi, .bat, .scr, .dll, .bat.
* Linux executables: .elf.
* Java .jar files
* Mobile OS executables: .apk, .app.
* HTML and other webpage files.
* Documents: .doc, .docx, .xlsx, .pdf.
* Scripts: .vbs, .js, .py,

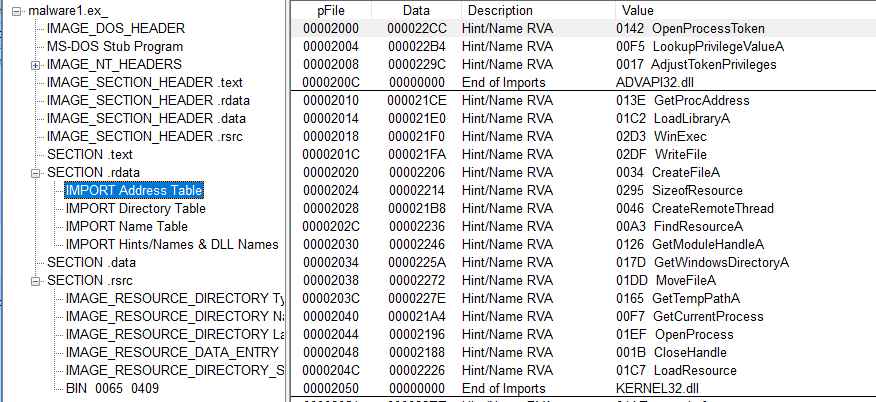
In this workshop, we will perform static analysis on exe and PDF files, which are the most popular and often utilised in widespread internet attack campaigns and phishing emails.

# Executable malware

A portable executable (PE) files consists of several sections. Of most significance to malware analysts are the .text, .rsrc and .data sections. The text section contains the code, which is the instructions for the processor to execute, usually in the form of machine code. The data and resources sections contain non-executable resources, such as strings, pictures and other types of information that help the executable behave in the intended way.

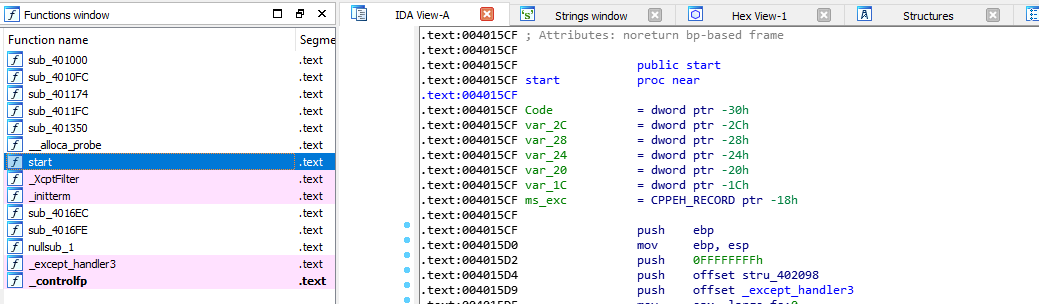
We can get a rough idea of the intended functionality of a malware sample by inspecting the various sections. Unzip this week’s resource folder. **DO NOT run any of these programs as they contain real malware and might severely harm your computer!**

Launch PEView and navigate to **malware1.ex\_** inside the “workshop” folder. Spend a few minutes inspecting the various sections of this malware sample. Click on “SECTION .rdata” and choose the IMPORT address table. What is the functionality of the import address table?



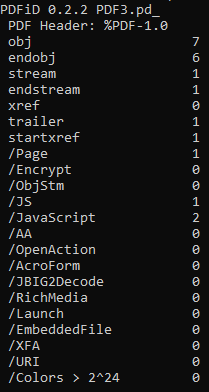
Can you predict what this malware can perform? Discuss your findings with a colleague or your tutor. After you finish this step, close PEView.

To analyse the text section, we need to reverse engineer the code, the process to achieve that is called *disassembling*. This is a deterministic process that converts machine code (hex) into assembly language instructions. Several tools can be used to perform this step, such as IDA, which is considered an industry standard.



Launch *IDAfree70*, navigate to malware1.ex\_. Spend some time reviewing the various parts of the screen. Try and locate the strings view. What do you think this malware performs?

# Maldoc: PDF

PDF files are binary files that are capable of executing code. Malicious content is usually delivered through the execution of JavaScript. PDF files have a hierarchical structure and consist of objects to build the document.

Locate and launch “PDFID” from the PDF folder. When a command prompt starts, navigate the workshop folder and type the following command:

**pdfid.py PDF3.pd\_**

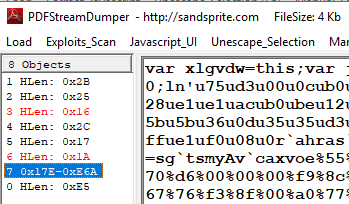
You should see the following screen:

This screenshot suggests that this PDF file contains seven objects, including one stream and 2 JS (JavaScript) object. This is a red flag, and we should investigate further.

Launch PDFStreamDumper and navigate to PDF3.pd\_. When the file loads, spend some time going through the content of each object. When you are done, locate object seven, then click on “Javascript\_UI” button.

Now click on “Format\_Javascript”. This option will break long lines and start a new line after every “;”, which helps make the code more readable.

Spend some time analysing the JavaScript code in this window. The code looks obfuscated, and there seems to be some hidden code in the content of the variable “joamhua2”. Malicious PDF files usually have anti-analysis traps that prevent us from performing automated analysis tasks. Let’s try clicking the “Run” button. The code will be executed in, and the result will be displayed in the bottom pane.

Copy the result and paste it on top of the current JavaScript code. There is a highly suspicious variable called “payload” which seems to contain an encoded shellcode! You need to figure out what type of encoding is used. The shellcode contains a hint.

Once you manage to decode the shellcode, you can use your knowledge from week 3 to analyse the shellcode.

Try and find another way to analyse the shellcode In PDFStreamdumper.

# Forensic task

1. In the exercise folder, analyse the file “malware2.dl\_”. What does the malware sample do?
2. In the exercise folder, analyse the PDF file “PDF1.pd\_”. What malicious action does it perform?
3. In the exercise folder, analyse the PDF file “PDF2.pd\_”. What malicious action does it perform?

# Further Reading about malware

* Whitepaper: How Malware Analysis Benefits Incident Response:

<https://informationsecurity.report/Resources/Whitepapers/51e831f9-aeef-41a4-b2e9-5162a2ac5f65_How%20Malware%20Analysis.pdf>

* Static malware analysis: <https://resources.infosecinstitute.com/malware-analysis-basics-static-analysis/>
* PDF malware analysis: <https://digital-forensics.sans.org/blog/2009/12/14/pdf-malware-analysis/>
* PDF structure: <https://blog.didierstevens.com/2008/04/09/quickpost-about-the-physical-and-logical-structure-of-pdf-files/>