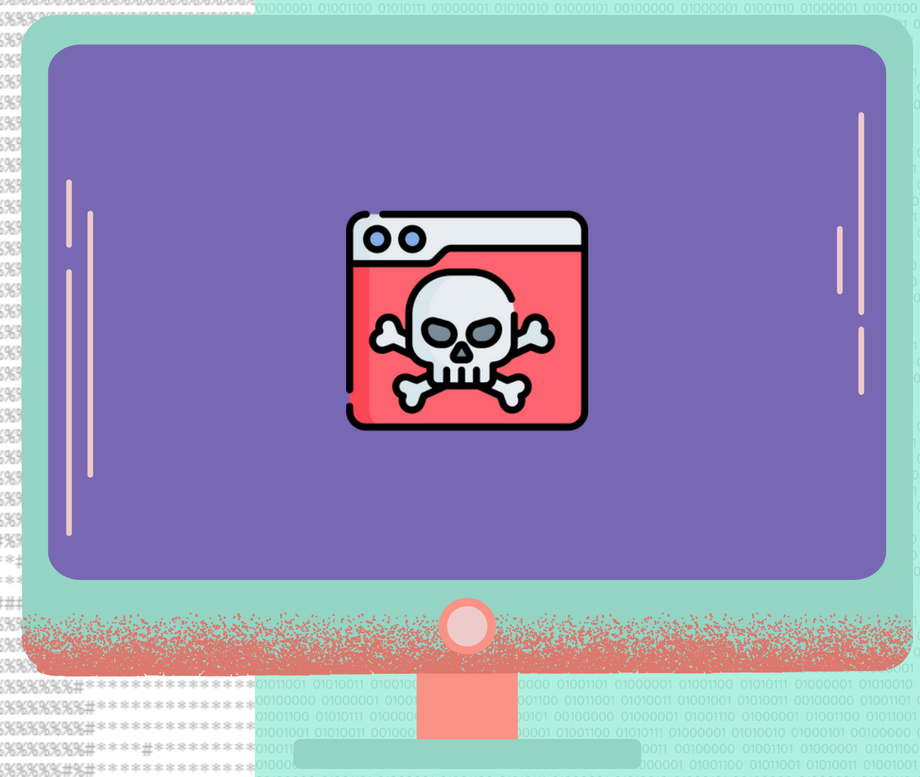
MALWARE ANALYSIS

By OS1RIS



$ WHOAMI

Shell

```
┌──(osiris㉿ALICE)-[~]
└─$ whoami

AFFAN NAJIY aka
```

An Incident Response Analyst with experience handling real-world cyber threats. Specialized in malware analysis and digital forensics, focusing on identifying IOCs, analyzing attack behavior, and dissecting malicious code.

Involved in detecting threats, assessing impact, and working with tools like EDR, XDR, and log sources. Continuously growing in memory analysis, reverse engineering, and threat hunting.

No i don't watch anime and i don't like playing CTF. i like playing single player game especially soulslike genre <3

But i love to learn everything, everywhere, anywhere. Feel free to call me nerd :)

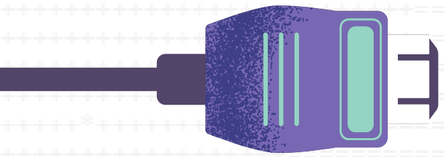
Background Study

1. MARA-Japan Industrial Institute (MJII) - Diploma in Electronic Engineering of Embedded System
2. Universiti Kuala Lumpur (UniKL) - Bachelor's Degree, Information Technology in Computer System Security

Disclaimer

This workshop involves real malware samples. Participants are advised to proceed at their own risk. The organizers will not be held responsible for any damage to your system, environment, or data. Any unethical conduct by participants during or after the workshop is solely their responsibility.

Bengkel ini melibatkan sampel malware sebenar. Peserta dinasihatkan untuk menjalankan aktiviti ini atas risiko sendiri. Penganjur tidak akan bertanggungjawab terhadap sebarang kerosakan pada sistem, persekitaran, atau data anda. Sebarang kelakuan tidak beretika oleh peserta semasa atau selepas bengkel ini adalah tanggungjawab peserta sepenuhnya.



Password: **MeinKampf**

TOPIC

Introduction

Static Analysis

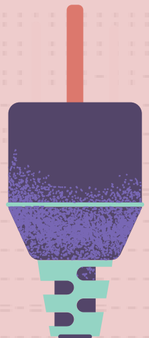
Dynamic Analysis

Bonus

Behavior Analysis

Real World Case Studied

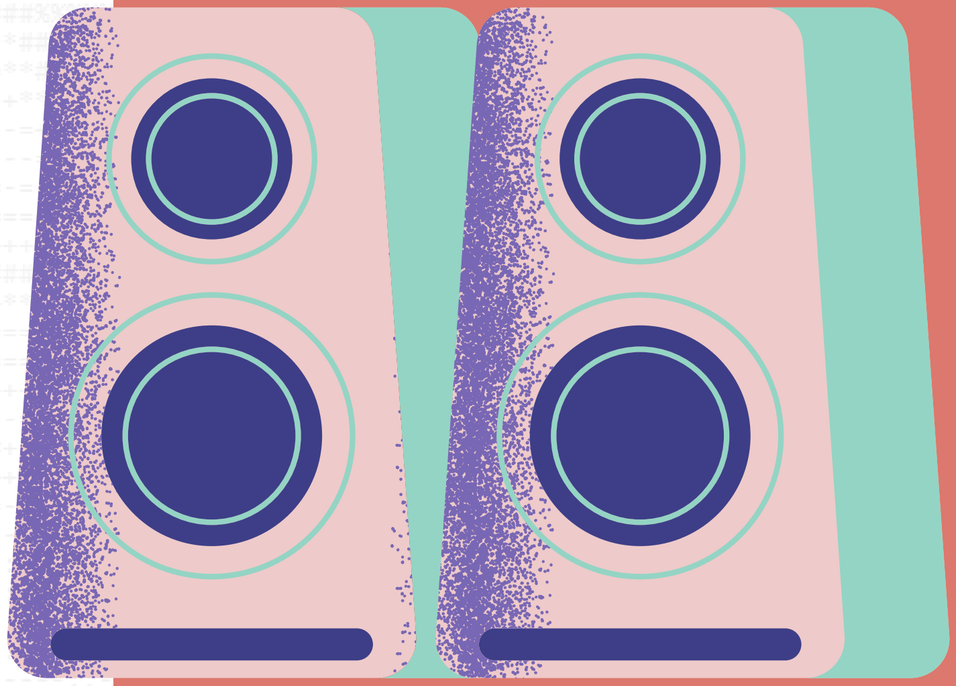
Not So
Threat Intel Integration



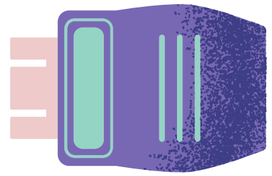
/'mælweə/

Malware

*Malware, short for malicious software, refers to any intrusive software developed by cybercriminals (often called hackers) to **steal data** and **damage** or **destroy** computers and computer systems.*



Types Of Malware

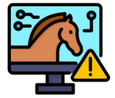


Virus

Attach to clean files and activate when opened. Spread through systems and damage or corrupt data.

Worms

Self-replicate across networks without user action, causing widespread disruption



Trojans

Disguised as legit software. Create backdoors for attackers to steal data or take control.

Spyware

Silently collects user data, keystrokes, and activity

Adware

Displays unwanted ads, tracks user behavior, and slows down systems.

Botnet

Networks of infected devices (bots) under a hacker's control via a C2 server. Used to launch DDoS attacks, mine crypto, or send spam.

Rootkit

Hide deep in the OS to maintain stealthy control. Modify system processes to avoid detection and allow persistent access.

RAT

Allow remote control over a system, used to steal data or deploy more malware.

Droppers

Install additional malware onto the system secretly.

Ransomware

Encrypts files and demands ransom for decryption. Causes severe financial and data loss.

Fileless

Operates in memory without leaving files on disk. Uses tools like PowerShell or WMI ,Macro, dynamic link injection and many

Infostealer

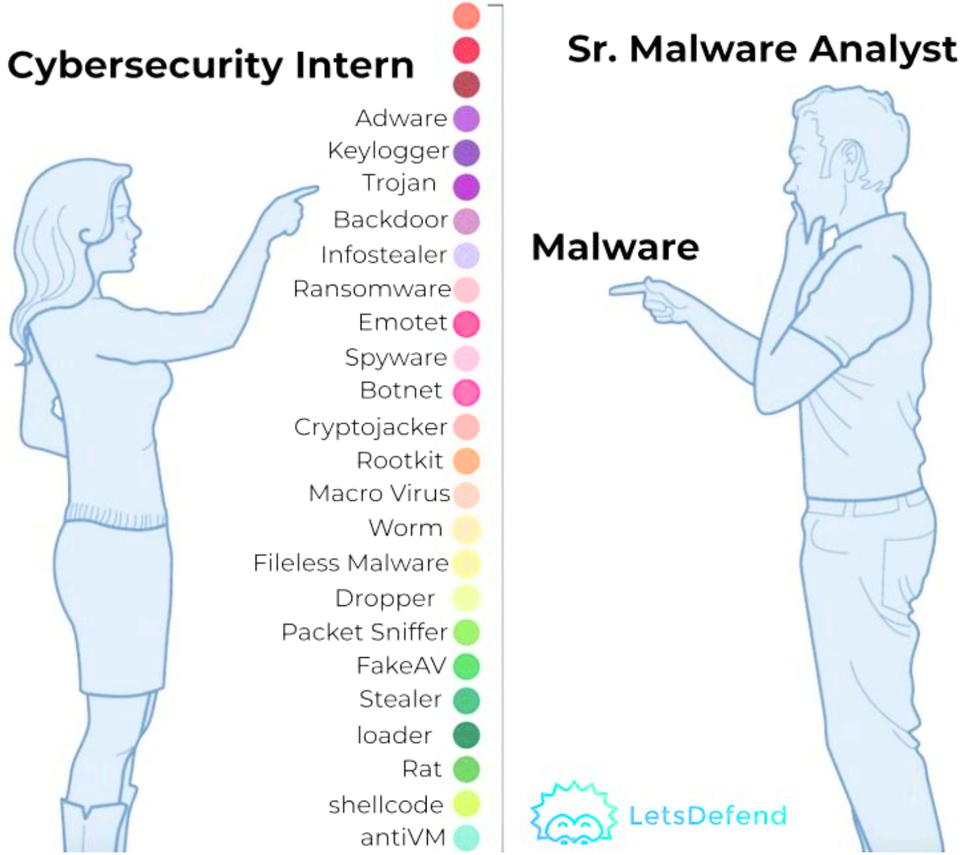
Extract sensitive info like passwords, banking data, or personal files.

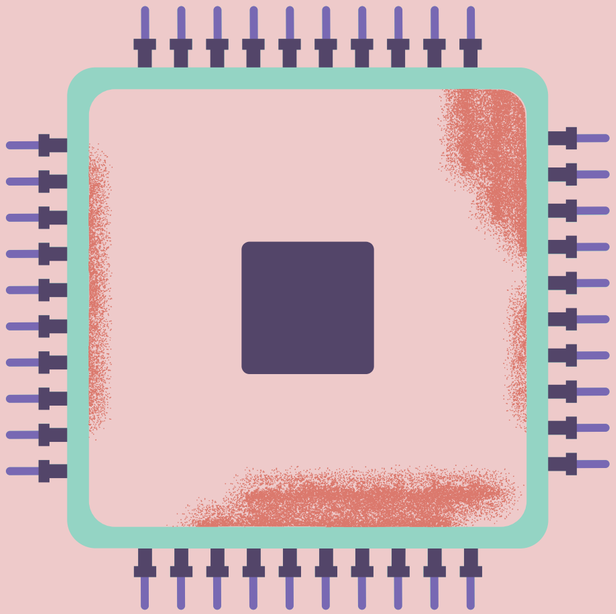
REALITY LIFE

Cybersecurity Terminology

Then	Now
Virus	malware
Keylogger	malware
Worm	malware
Trojan	malware
Rootkit	malware
Backdoor	malware
Adware	malware

LetsDefend





Got Infected?

What to do?

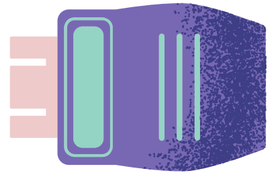
01 **DISCONNECT NETWORK** | Unplug from the internet and local network to stop the malware from spreading or communicating with attackers

02 **SCAN AND REMOVE** | Eliminate the malicious file or use any trusted antivirus or malware removal tools (e.g. Malwarebytes, Windows Defender)

03 **RESTORE BACKUP** | Restore clean backups if files were damaged or encrypted. Keep monitoring for unusual activity to ensure full recovery

04 **REFORMAT** | Reset PC Wipes the system clean when malware can't be fully removed

05 **BUY NEW MACHINE** | If you rich go buy a new laptop or pc.



Safely Handle the Malware

1

Use an Isolated Environment

Always analyze in a VM or sandbox with no access to your main system or network.

2

Disabled Internet Access (if not needed)

Prevent malware from calling out to C2 servers or downloading more payloads.

3

Take Snapshots

Create VM snapshots before analysis so you can revert quickly after testing.

4

Defang file type

Renaming the extension to ".exe.something" to prevent accidental execution.

5

Label and Hash Sample

Keep track of file hashes (MD5/SHA256) to identify and manage samples.

6

Don't Use Personal Accounts

Never log-in to personal emails or systems from your malware lab.

7

Practice Safe Extraction

Store malware in password-protected ZIPs when sharing or archiving.

8

Avoid Execute

Avoid double clicking or run the file unless you know what you handling or doing.

9

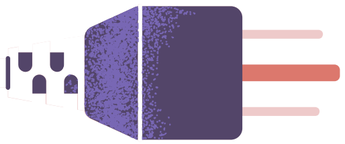
Log Everything

Track all behavior. Use tools like Procmon, Wireshark etc.

10

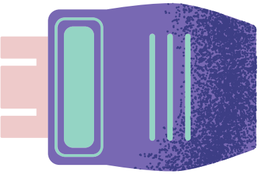
Be Smart

Don't Skill Issue yourself. One wrong click, and you'll crying regretting your own decision.



Learning Malware Analysis

Does It Matters?



Understand the Threat

01 Malware analysis allows analysts to deeply understand **how a threat behaves, what files it touches, what processes it load, and how it moves through systems**. This knowledge is a boost for identifying the attacker's goals, techniques, and potential damage.

Support Incident Response

02 During or after an incident, malware analysis provides clear insights into the **infection chain, the system changes made, and any backdoors or persistence mechanisms**. This speeds up containment, eradication, and recovery steps, helping teams respond more effectively.

Bypass Anti-Analysis Tricks

03 Analyzing malware teaches how it **detects and evades virtual machines, sandboxes, or debuggers**. By understanding these anti-analysis methods, analysts can adjust their tools and environment to bypass such checks and continue their investigation effectively.

Supports Research

04 Malware analysis helps researchers **discover new techniques used by attackers**. This knowledge contributes to developing better security tools, improving threat intelligence, and staying ahead in the cyber arms race.

In Normal working environment

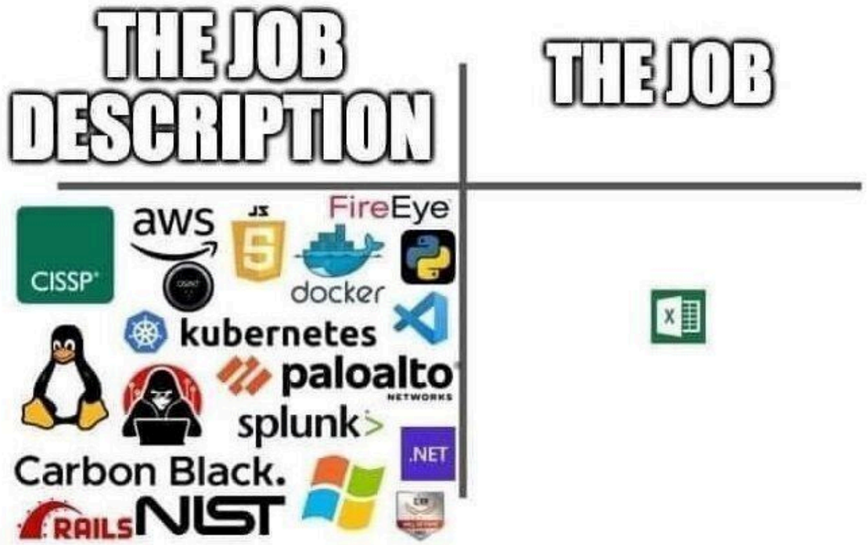
Does It Matters?

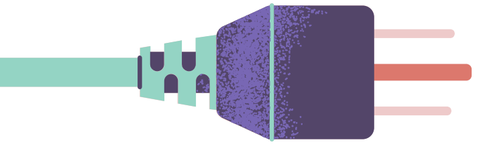
90% OF THE TIME

05

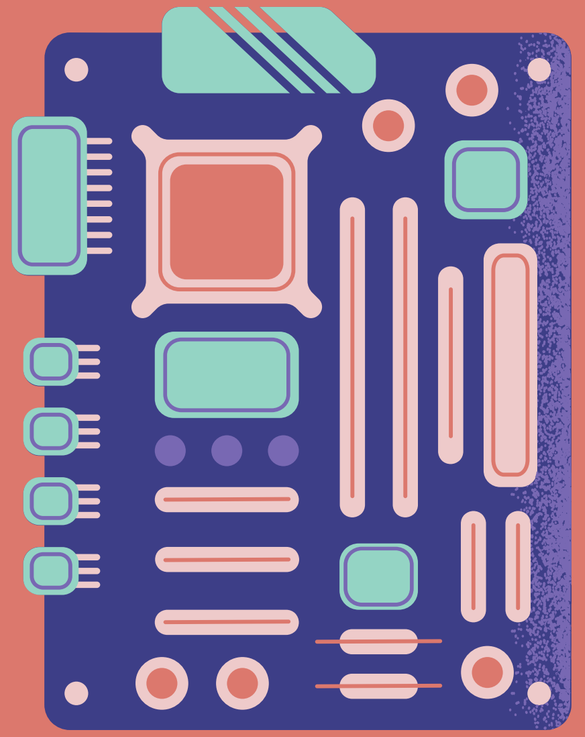
NO!

Yes if you want to **support your findings** to SOC teams.





Common Tools



Windows Environment ·····→

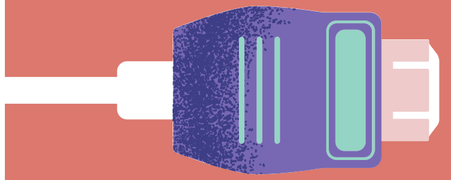
- **FLARE-VM**

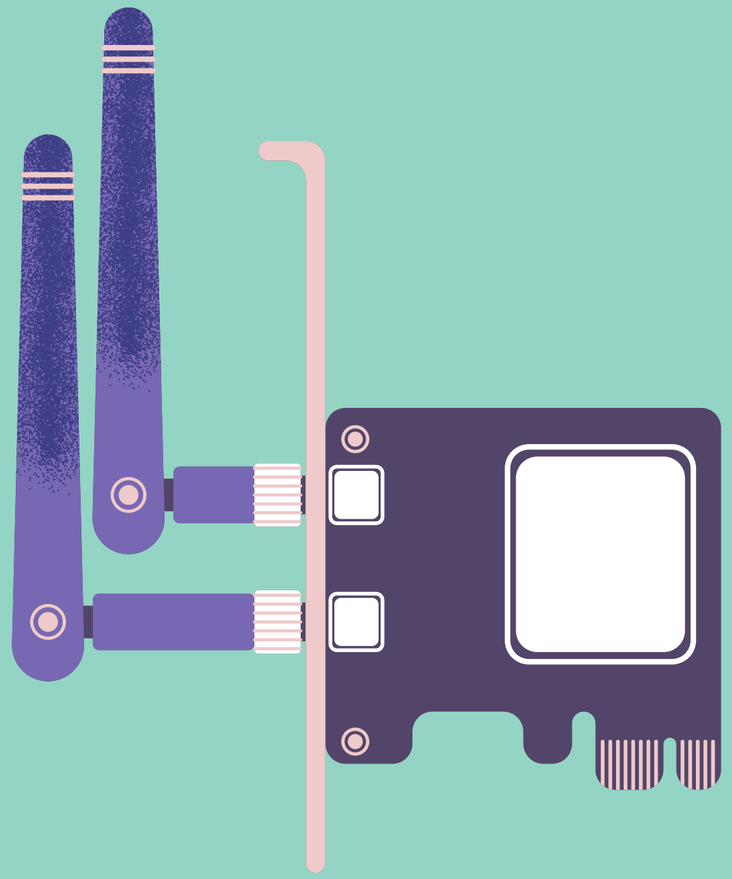
- PEStudio
- Detect It Easy (DIE)
- BinText
- Resource Hacker
- dnSpy
- Process Monitor (Procmon)
- Process Explorer
- Regshot
- Fakenet-NG
- Wireshark
- x64dbg
- IDA Free
- Ghidra
- Cutter
- HashMyFiles
- HxD

Linux Environment ·····→

- **REMnux**

- Ghidra
- Radare2
- Cutter
- strace
- ltrace
- GDB
- objdump
- strings
- file
- hexdump
- tcpdump
- Wireshark
- QEMU
- Volatility
- yara
- binwalk
- upx





/ˈstætɪk əˈnæləsɪs/

STATIC ANALYSIS

Static analysis methodically **examines the contents of files** and programs from the inside out for signs of potentially malicious intent, looking specifically for known malware signatures. Static malware analysis **does not require program execution**, which helps in identifying vulnerabilities like code injection points.

What To Look For 🎯

01 File type and architecture

02 Hash File

03 Strings and Hardcoded

04 Imports and Functions

05 Application Programming Interfaces (APIs)

DEMO: SAMPLE01

Learning Outcome:

1. Understand how to gather malware indicators without execution
2. Identify suspicious patterns, strings, and functions in binaries
3. Recognize file structure, imports, and possible obfuscation techniques

```
┌──(osiris㉿ALICE)-[~/sample]
└─$ tree
.
├── Sample01
│   ├── dev.exe.zip
│   ├── your_app.exe.zip
│   └── Hash.txt
```

TOOLS:
Strings
File
PEStudio
Detect It Easy (DIE)

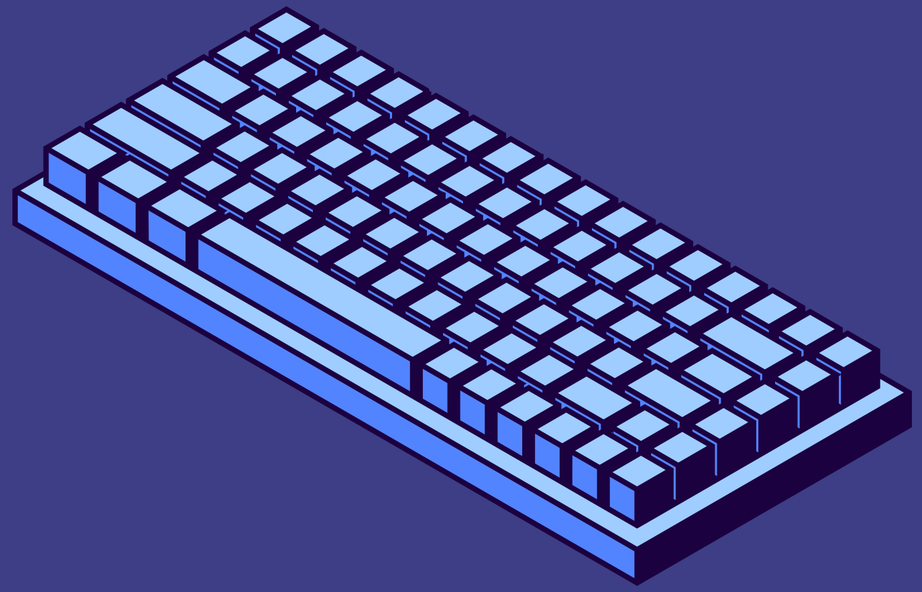
Filename:
your_app.exe.bin
SHA256:
0d38f8bf831f1dbbe9a058930127171f24c3df8dae81e6aa66c430a63cbe0509

1. What is the **MD5** of this file?
2. What is the **Architecture Mode**?
3. What is the **Entropy** of this file?
4. What is the **Compiler** of this malware?
5. On PESTUDIO how many **API imports** was flag?
6. While String, what three of the **commandline** that can be seen?

TOOLS:
Strings
File
PEStudio
Detect It Easy (DIE)

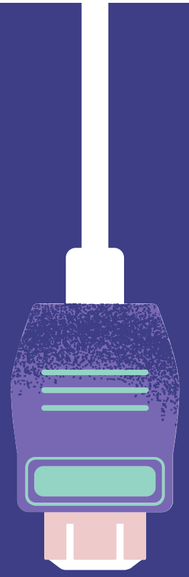
Filename:
dev.exe
SHA256:
00acf5d0db7ef50140dae7a3482d9db80704ec98670bd1607e76c99382a4888c

1. What is the **SHA1** of this file?
2. What is the **Architecture Mode**?
3. What is the **Debug name** of this file?
4. What is the **Compiler** of this malware?
5. On PESTUDIO, What is the **Name or Malware Category** of this malware?
6. On PESTUDIO, **Entrypoint in Hex** of the program?



/daɪˈnæmɪk əˈnæləsɪs/

DYNAMIC ANALYSIS



Dynamic analysis, on the other hand, involves **executing or “detonating”** a suspicious program within a virtual sandbox environment and closely monitoring its **behavior, interactions with the system, and response** to various inputs.

What To Look For 🎯

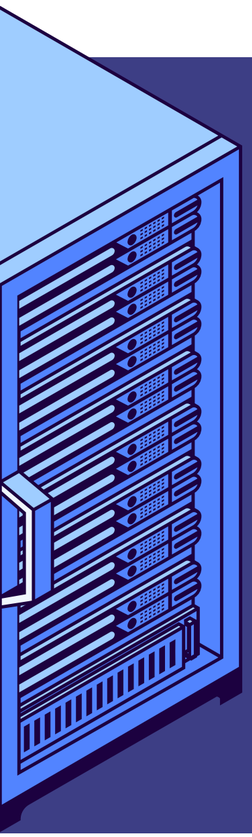
01 Process behavior

02 File system changes

03 Network activity

04 Registry modifications

05 API calls



DEMO: SAMPLE02

Learning Outcome:

1. Understand how malware behaves during execution.
2. Identify key system, file, and network indicators generated by the malware.
3. Learn how to trace malware actions such as persistence, registry changes, and process injection.

```
┌──(osiris㉿ALICE)-[~/sample]
└─$ tree
.
├── Sample02
│   ├── aaaa_protected.exe.zip
│   ├── emigma.zip
│   └── Hash.txt
```

TOOLS:
Fakenet
Procmon
Process Hacker 2

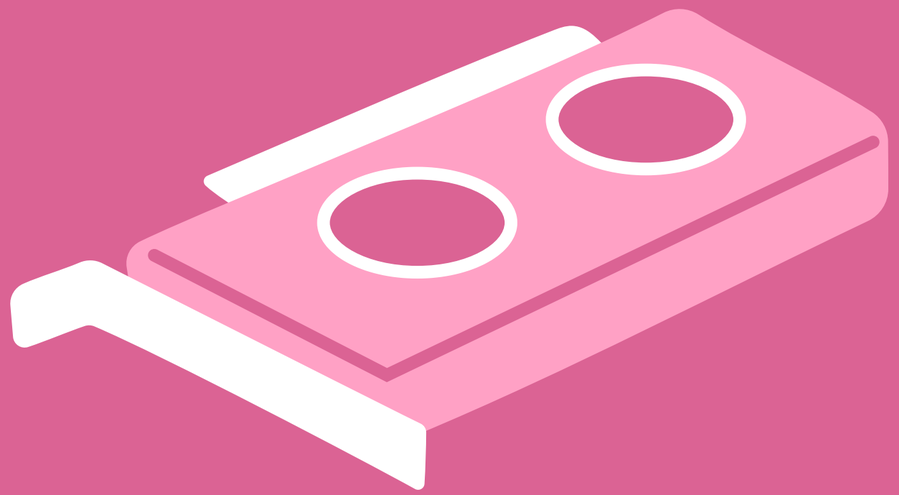
Filename:
emigma.exe
SHA256:
245bb11f09aebee6994a1e7cd201a9e3dd9cd5f166558fb21180af6509d89ba4

1. What is the **Architecture Mode**?
2. How many **Program it drop**? Name it
3. Do it have any **Connection attempt**?
4. What is the named of the **Domain** it try to connect?
5. What is the **Command line** that can be found?
6. Named the path of the **Registry** created
7. What is this **Malware type** can be called?

TOOLS:
Fakenet
Procmon
Process Hacker 2

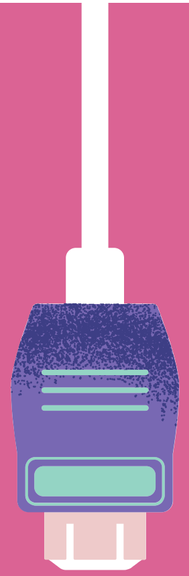
Filename:
aaaa_protected.exe
SHA256:
85706bed6a79755fc38025c615b1bb3535c26e85b4b627f0ed81964890b20ca8

1. What is the **Architecture Mode**?
2. Name of the **Program that is drop**?
3. How many **Path Location of the drop file**?
4. What is the named of the **Domain** it try to connect?
5. What is the **Command line** that can be found?
6. Using Process Hacker 2, When you **dump the drop** file. What can you see?



/bɪˈheɪvjər əˈnæləsɪs/

BEHAVIOR ANALYSIS



Is an continuation of Dynamic Analysis. Behavior analysis focuses on observing what malware does when it runs, such as **modifying files, creating processes, or sending data to external servers**.

What To Look For 🎯

01 File system changes

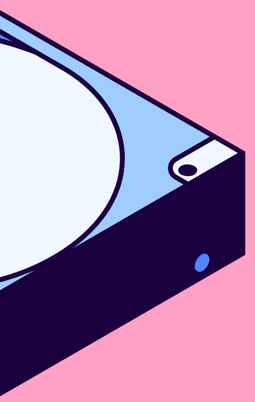
02 Registry modifications

03 Network activity

04 Process creation

05 Persistence methods

06 Command and Control (C2) communication



DEMO: SAMPLE02

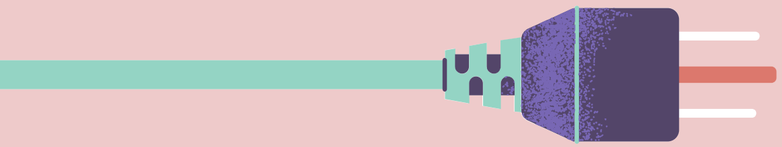
Learning Outcome:

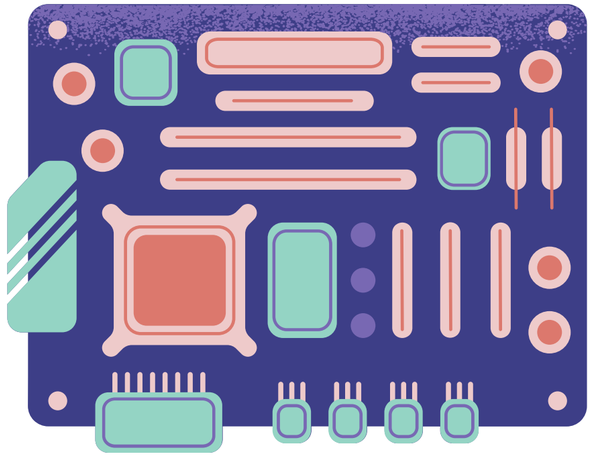
1. Understand how malware interacts with the operating system, files, network, and registry during execution.
2. Identify persistent techniques used by malware to maintain access or reinfect a system.
3. Recognize patterns or anomalies that indicate malicious behavior.

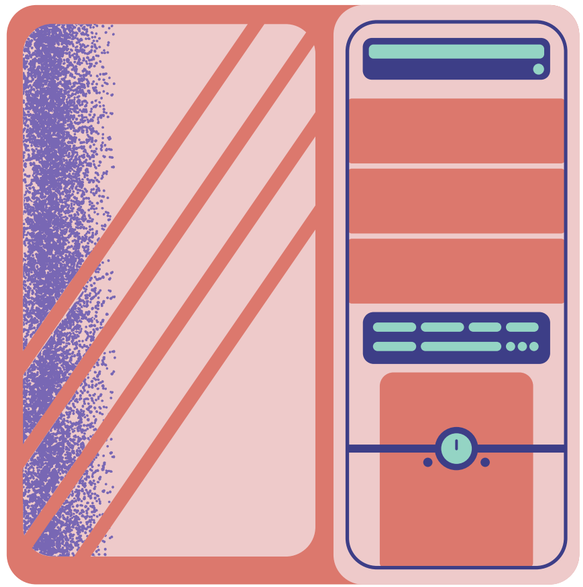
/rɪəl wɜːld keɪs ˈstʌdiːd/

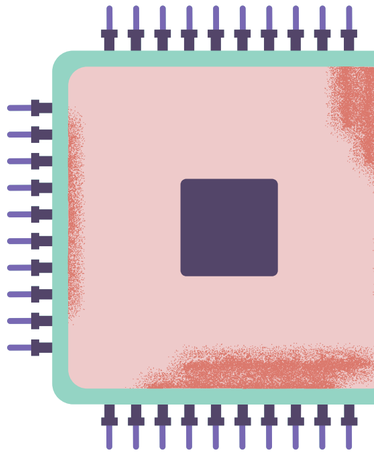
REAL WORLD CASE STUDIED

This section presents a real-world malware attack to show how analysis techniques are applied in practice. The case study helps relate theoretical knowledge to actual incidents, making the learning more practical and relevant. Refer '***Case Study***' Material









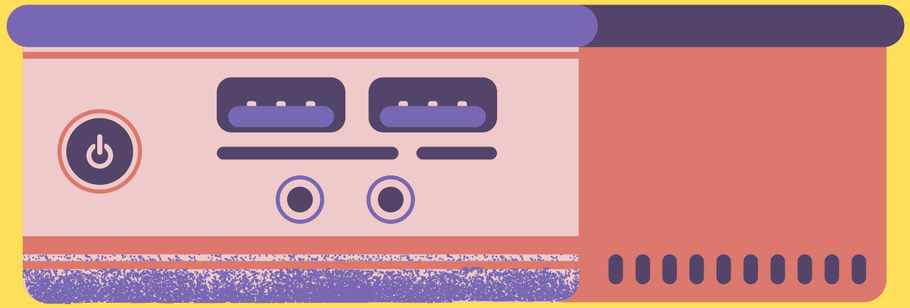
TOOLS:
None But Google

CASE STUDY

Learning Outcome:

1. Understand real-world malware behavior and its impact through hands-on analysis.

```
┌──(osiris㉿ALICE)-[~/sample]
└─$ tree
.
├── Case Study
│   └── Oyster Malware_Poisoning SEO(July 2025)
│       ├── Case-Studied.pcap
│       ├── Hash.txt
│       ├── KeePass-2.58-Setup.exe.zip
│       ├── WinSCP.zip
│       └── zqin.dll.zip
```



/θrɛt ˈɪntɛl ˌɪntɪˈgreɪʃᵊn/

THREAT INTEL INTEGRATION

Threat Intelligence is **information collected and analyzed about existing** or emerging cyber threats that helps organizations detect, understand, and respond to attacks more effectively.

What To Look For 🎯

01 **Tactical**: Indicator Of Compromised

02 **Operational**: TTPs, malware families

03 **Strategic**: High-level trends, motivations

Integration of Threat Intelligence

01 **Fast Threat Detection**: Speeds up identification of malware by matching indicators with known threat data.

02 **Behavioral Confirmation**: Helps verify observed actions during analysis by comparing with documented threat behaviors.

03 **Threat Correlation**: Connects the sample to known threat actors, campaigns, or malware families.

04 **Analysis Enrichment**: Support our static and dynamic findings using threat intel feeds or platforms.

DEMO: SAMPLE03

Learning Outcome:

1. Understand how malware interacts with the operating system, files, network, and registry during execution.
2. Identify persistent techniques used by malware to maintain access or reinfect a system.
3. Recognize patterns or anomalies that indicate malicious behavior.

```
┌──(osiris㉿ALICE)-[~/sample]
└─$ tree
.
├── Sample03
│   ├── ransom.zip
│   ├── malware.zip
│   └── Hash.txt
```

TOOLS:
VirusTotal [https://www.virustotal.com/]
anyRun [https://app.any.run/]
Triage [https://tria.ge/s]

Source: **ransom.zip**
SHA256:
0d38f8bf831f1dbbe9a058930127171f24c3df8dae81e6aa66c430a63cbe0509

1. What is the **name** of this ransomware?
2. What is the **original file name** found?
3. What is the **version** and **variant** name?
4. What is the **creation time** of this ransomware?
5. Which **operating system** does it target?
6. What is **another name** for this ransomware?
7. Why does **Any.Run** flag the file as “**No Threats Detected**”?

TOOLS:
VirusTotal [https://www.virustotal.com/]
anyRun [https://app.any.run/]
Triage [https://tria.ge/s]

Source: **malware.zip**
SHA256:
3c92bfc71004340ebc00146ced294bc94f49f6a5e212016ac05e7d10fcb3312c

1. What is the **name** of this malware?
2. What **type of malware category** is this?
3. What does this malware **remove** using a command?
4. What **file extension** is created by this malware?
5. What file does this **malware drop**, and what is its name?
6. What is the **code** to log into the chat?

BONUS

CHALLENGE

Test Your Skill

```
┌──(osirisⓀALICE)-[~/sample]
└─$ tree
.
├── Challenge
│   ├── 555df9a101322d17e040c0e39b528c08bbc3c8b040bdf491aea0efc5d4d3d661
│   ├── a3b797088da9ddb98ef5a04ff29d54e289cd7824a47ebf1f7b09ea4eb2c840bb
│   ├── a7fd97177186aff9f442beb9da6b1ab3aff47e611b94609404e755dd2f97dce8
│   ├── af259cf23af26d89b615a56795a53249721f3be718be6621ab941476108ab555
│   ├── d105536c7b43d74fb3c32f8179261835f52007d8fb83eb8bdf1972756156a1c9
│   └── pyinstxtractor.py
```

BONUS

WHAT NEXT?

Malware analysis doesn't stop at surface level.
The real skill lies in **Reverse Engineering**. Where true understanding begins. here few things you can look up to:

- Use **Assembly** to break down how malware behaves
  - **Static Analysis**: Understand logic via Decompiler views (e.g, *IDA*, *Ghidra*)
  - **Dynamic Analysis**: Watch runtime behavior in *x64dbg* / *x32dbg*

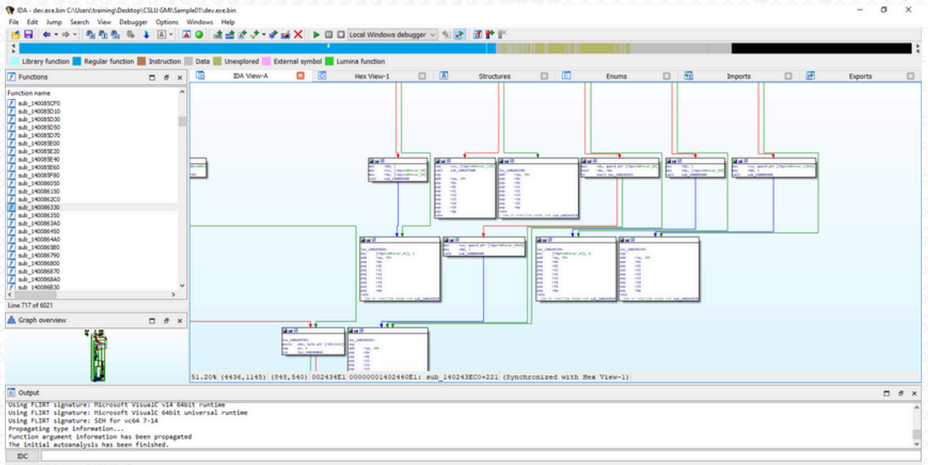
- Learn how malware hides, injects, evades

- Other things that will help boost your understanding, **learn basic programming** (Highly recommend C Programming)
  - Most malware is written in low-level languages like C/C++
  - Helps you read and understand assembly code easier
  - Understand how memory, stack, and function calls work

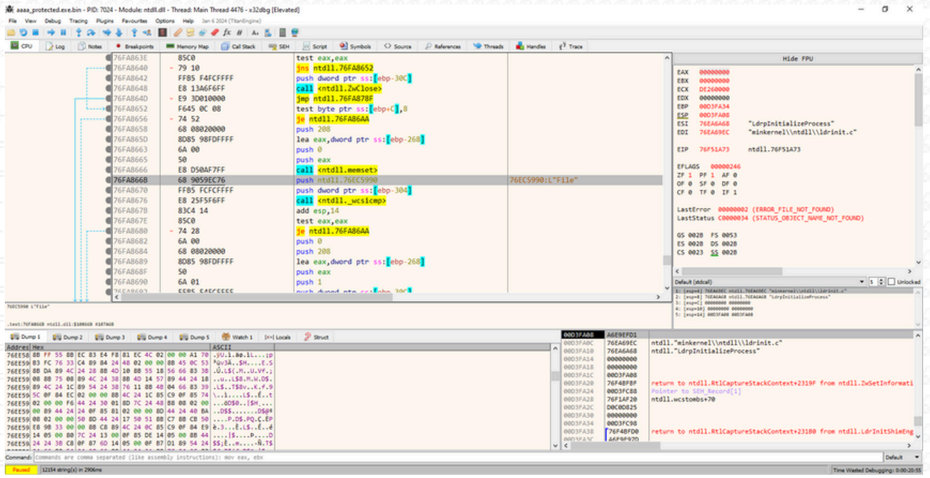
- Many indicators can't be seen from high-level code alone

- Easier to write precise **YARA rules** by matching opcode patterns

IDA:



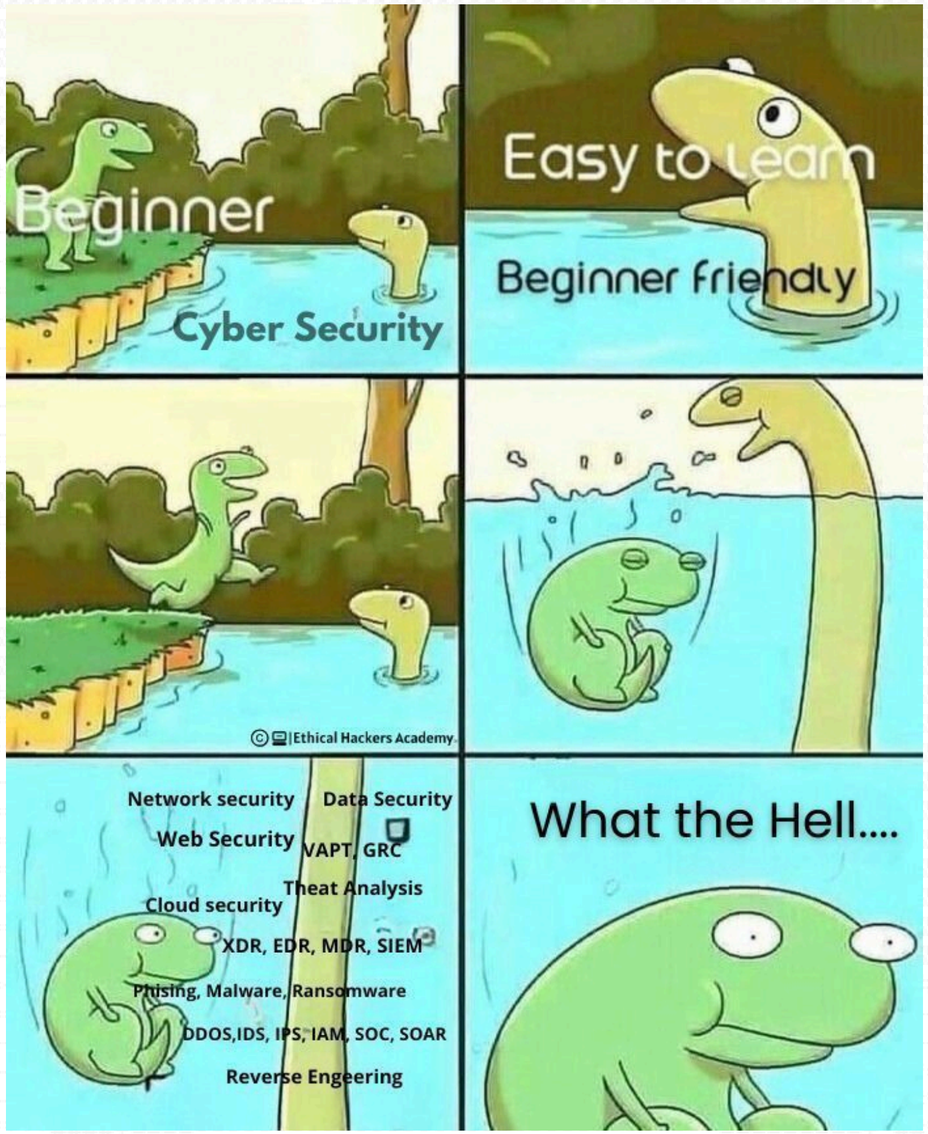
xdbg:



BONUS

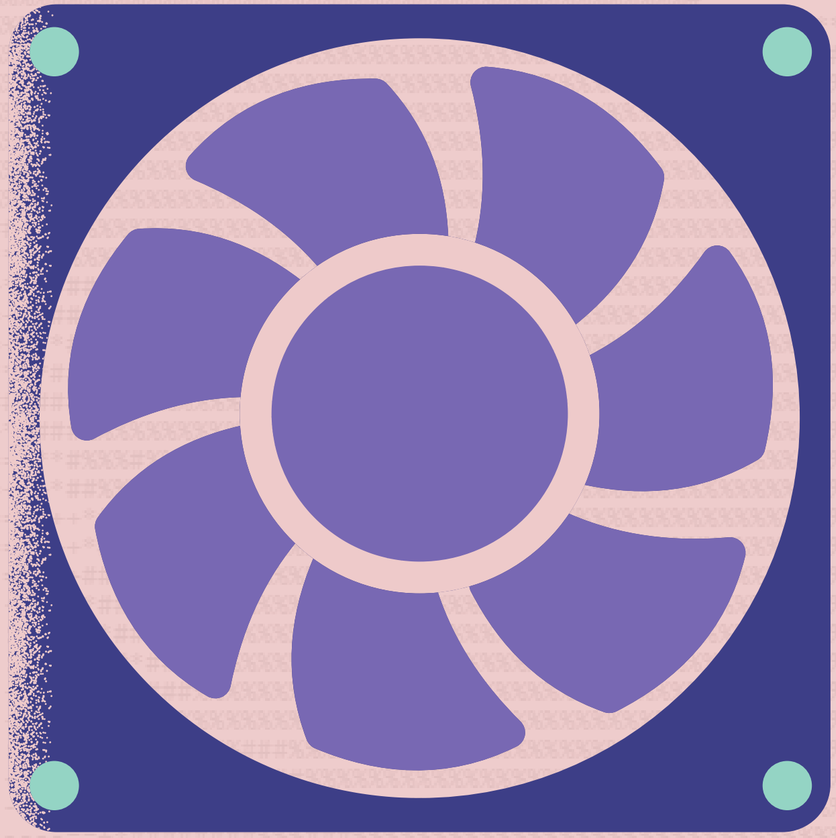
“Do the Reverse Engineering” they said

"ITS FUN” they said



The Future Direction

The future of malware analysis focuses on smarter tools, faster detection, and deeper insights. As threats become more complex, analysts will rely on automation, collaborative threat intelligence, and cross-platform visibility to respond quickly and accurately.



WRAP UP

01 This workshop only touched the surface of how malicious software operates, giving you a basic exposure to malware behavior and how to use tools. **You are not a malware expert after this**, there are still many techniques, tools, and analysis methods we didn't cover.

02 I hoped that you've **gained some knowledge** on how to handle malware safely, even at a beginner level.

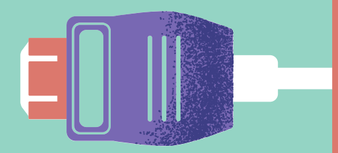
03 **Think before you click**, always stay alert, and be conscious of your actions when dealing with unknown files or links.

Terima Kasih
非常感谢

மிகவும் நன்றி

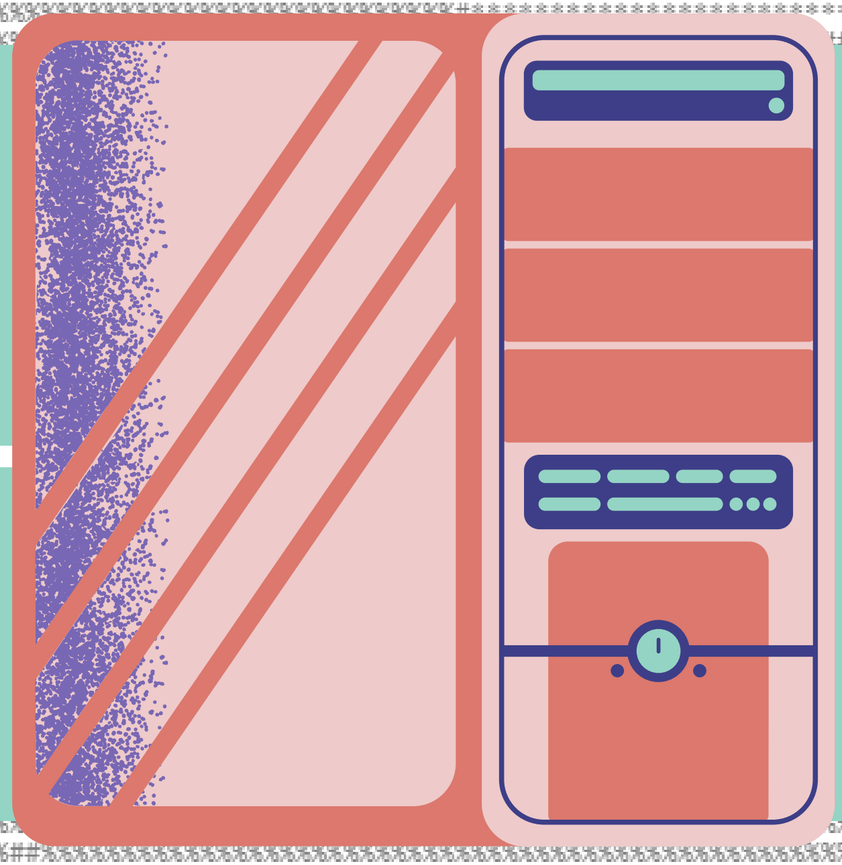
정말 감사합니다

Thank You



どうもありがとうございます

Vielen Dank



FIN