Malware Analysis and Creation of a Detection Application

BSc(Hons) Applied Computer Science

Introduction

This project focuses on malware analysis to evaluate whether programs exhibit dangerous features, leading into the development of a Windows desktop application to detect such malware programs.

Methodology

The methodology consists of defining the project scope and objectives, collecting a malware sample, performing static and dynamic analysis, identifying malware features, developing OOAD models and implementing the detection application using an iterative waterfall approach

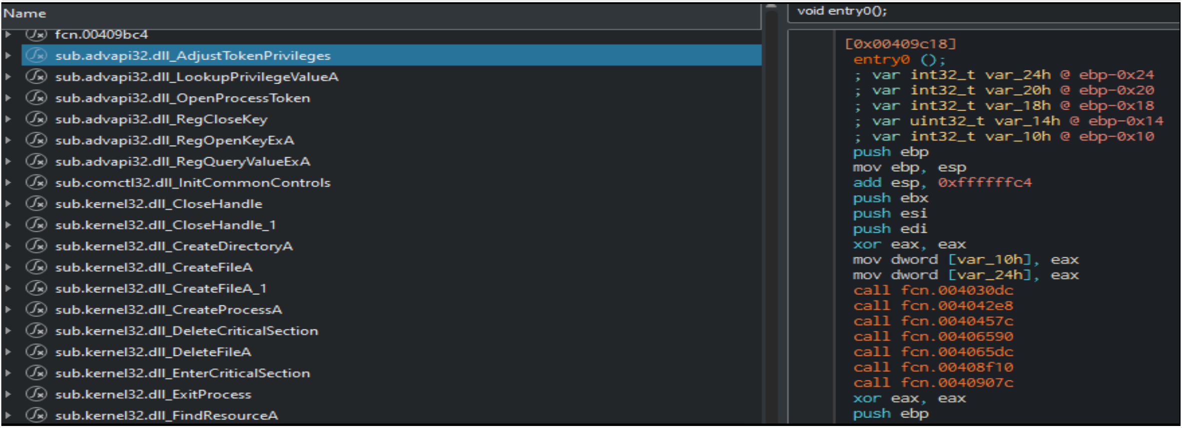
Projected Outcomes

1.Successful collection and analysis of a malware sample.

2. Identification of common and unique features of the malware.

3. Creation of an easy-to-use desktop application for detecting malware on Windows machines.

Figure 1 – Static Analysis

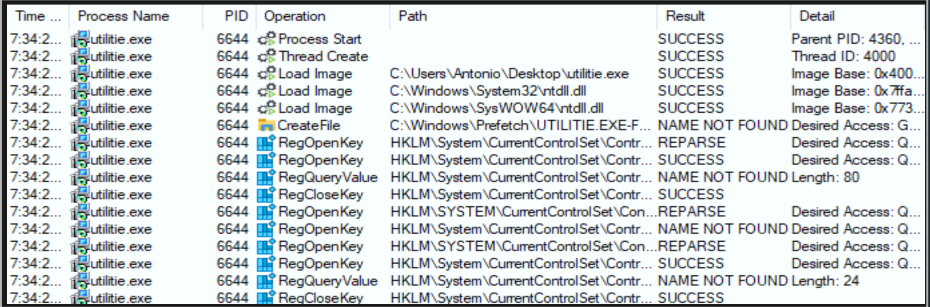


```
Name
fcn.00409bc4
sub.advapi32.dll_AdjustTokenPrivileges
sub.advapi32.dll_LookupPrivilegeValueA
sub.advapi32.dll_OpenProcessToken
sub.advapi32.dll_RegCloseKey
sub.advapi32.dll_RegOpenKeyExA
sub.advapi32.dll_RegQueryValueExA
sub.comctl32.dll_InitCommonControls
sub.kernel32.dll_CloseHandle
sub.kernel32.dll_CloseHandle_1
sub.kernel32.dll_CreateDirectoryA
sub.kernel32.dll_CreateFileA
sub.kernel32.dll_CreateFileA_1
sub.kernel32.dll_CreateProcessA
sub.kernel32.dll_DeleteCriticalSection
sub.kernel32.dll_DeleteFileA
sub.kernel32.dll_EnterCriticalSection
sub.kernel32.dll_ExitProcess
sub.kernel32.dll_FindResourceA
```

```
void entry0();

[0x00409c18]
 entry0 ();
 ; var int32_t var_24h @ ebp-0x24
 ; var int32_t var_20h @ ebp-0x20
 ; var int32_t var_18h @ ebp-0x18
 ; var uint32_t var_14h @ ebp-0x14
 ; var int32_t var_10h @ ebp-0x10
 push ebp
 mov ebp, esp
 add esp, 0xffffffc4
 push ebx
 push esi
 push edi
 xor eax, eax
 mov dword [var_10h], eax
 mov dword [var_24h], eax
 call fcn.004030dc
 call fcn.004042e8
 call fcn.0040457c
 call fcn.00406590
 call fcn.004065dc
 call fcn.00408f10
 call fcn.0040907c
 xor eax, eax
 push ebp
```

Figure 2 – Dynamic Analysis



Time ...	Process Name	PID	Operation	Path	Result	Detail
7:34:2...	utilitie.exe	6644	Process Start		SUCCESS	Parent PID: 4360, ...
7:34:2...	utilitie.exe	6644	Thread Create		SUCCESS	Thread ID: 4000
7:34:2...	utilitie.exe	6644	Load Image	C:\Users\Antonio\Desktop\utilitie.exe	SUCCESS	Image Base: 0x400...
7:34:2...	utilitie.exe	6644	Load Image	C:\Windows\System32\ntdll.dll	SUCCESS	Image Base: 0x7ffa...
7:34:2...	utilitie.exe	6644	Load Image	C:\Windows\SysWOW64\ntdll.dll	SUCCESS	Image Base: 0x773...
7:34:2...	utilitie.exe	6644	CreateFile	C:\Windows\Prefetch\UTILITIE.EXE-F...	NAME NOT FOUND	Desired Access: G...
7:34:2...	utilitie.exe	6644	RegOpenKey	HKLM\System\CurrentControlSet\Contr...	REPARSE	Desired Access: Q...
7:34:2...	utilitie.exe	6644	RegOpenKey	HKLM\System\CurrentControlSet\Contr...	SUCCESS	Desired Access: Q...
7:34:2...	utilitie.exe	6644	RegQueryValue	HKLM\System\CurrentControlSet\Contr...	NAME NOT FOUND	Length: 80
7:34:2...	utilitie.exe	6644	RegCloseKey	HKLM\System\CurrentControlSet\Contr...	SUCCESS	
7:34:2...	utilitie.exe	6644	RegOpenKey	HKLM\SYSTEM\CurrentControlSet\Con...	REPARSE	Desired Access: Q...
7:34:2...	utilitie.exe	6644	RegOpenKey	HKLM\System\CurrentControlSet\Contr...	NAME NOT FOUND	Desired Access: Q...
7:34:2...	utilitie.exe	6644	RegOpenKey	HKLM\SYSTEM\CurrentControlSet\Con...	REPARSE	Desired Access: Q...
7:34:2...	utilitie.exe	6644	RegOpenKey	HKLM\System\CurrentControlSet\Contr...	SUCCESS	Desired Access: Q...
7:34:2...	utilitie.exe	6644	RegQueryValue	HKLM\System\CurrentControlSet\Contr...	NAME NOT FOUND	Length: 24
7:34:2...	utilitie.exe	6644	RegCloseKey	HKLM\System\CurrentControlSet\Contr...	SUCCESS	

Figure 3 – App Search for Malware

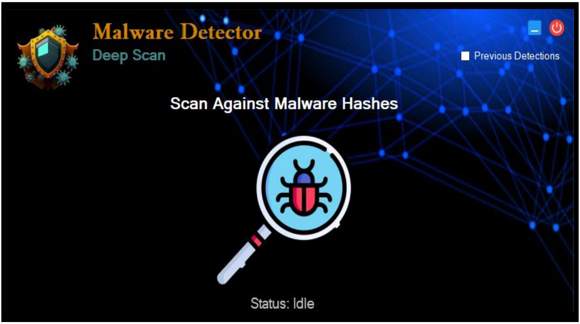
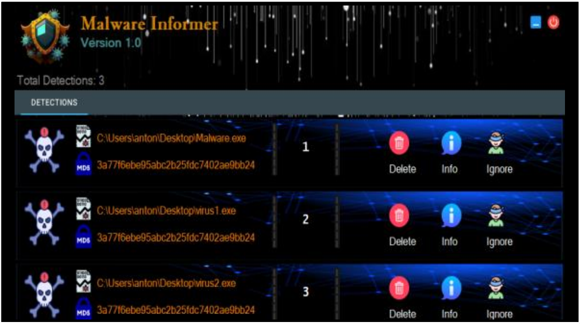


Figure 4 – App Malware Detected



Total Detections: 3

C:\Users\anton\Desktop\Malware.exe
3a77f6ebe95abc2b25fdc7402ae9bb24
1

C:\Users\anton\Desktop\virus1.exe
3a77f6ebe95abc2b25fdc7402ae9bb24
2

C:\Users\anton\Desktop\virus2.exe
3a77f6ebe95abc2b25fdc7402ae9bb24
3

Preliminary Results

1. A single malware sample (Utilitie.exe) was collected and analysed.
2. Static analysis shown in figure 1 revealed obfuscation techniques and functionalities (d0x, 2021) of the identified trojan.
3. Dynamic analysis displayed in figure 2 showed that the sample had behaviour patterns consistent with findings in static analysis.
4. OOAD models were developed to guide the implementation of the detection application.
5. The detection application seen in figures 3 and 4 was implemented, and preliminary testing showed promising results in terms of detecting malware based on their signatures.

References

d0x, M., 2021. MalAPI.io. [online] Malapi.io. Available at: <https://malapi.io/> [Accessed 11 March 2023].