



Reverse Engineering (REV3)

## **UE 02 – Statische Analyse – Protokoll**

Jakob Mayr

WS 2023/2024

### **Einleitung**

...

# Aufgabe 1 - Statische Analyse Windows

## Erstellen der Dateien

In Aufgabe 1 ist eine Anwendung, welche "infected" auf `stdout` ausgibt in c zu schreiben und mit Visual Studio auf 4 verschiedene Varianten zu bauen.

Die vier Varianten mit deren Eigenschaften und Unterschieden (Quelle: ChatGPT):

### 1. Static Release (/MT):

- (a) Laufzeitbibliothek: Statisch
- (b) Debug-Informationen: Nein
- (c) Eigenschaften:
  - i. Die Laufzeitbibliothek wird in die ausführbare Datei eingebettet.
  - ii. Größere Dateigröße, da der Code der CRT (C Runtime Library) direkt in die Anwendung eingefügt wird.
  - iii. Keine Abhängigkeit von DLLs (Dynamically Linked Libraries) zur Laufzeit.
  - iv. Optimiert für Geschwindigkeit und nicht für das Debugging.

### 2. Static Debug (/MTd):

- (a) Laufzeitbibliothek: Statisch
- (b) Debug-Informationen: Ja
- (c) Eigenschaften:
  - i. Ähnlich wie /MT, aber mit zusätzlichen Debug-Informationen und weniger Optimierungen.
  - ii. Erleichtert das Debugging, da Variablen leichter überwacht werden können.
  - iii. Größerer Speicherbedarf und langsamere Ausführung im Vergleich zur Release-Version.

### 3. Dynamic Release (/MD):

- (a) Laufzeitbibliothek: Dynamisch
- (b) Debug-Informationen: Nein
- (c) Eigenschaften:
  - i. Verlinkt dynamisch mit den CRT-DLLs (Z.B. `msvcrXXX.dll`).
  - ii. Kleinere Dateigröße, da die Laufzeitbibliothek nicht eingebettet ist.
  - iii. Erfordert, dass die CRT-DLLs zur Laufzeit verfügbar sind.
  - iv. Optimiert für Geschwindigkeit.

### 4. Dynamic Debug (/MDd):

- (a) Laufzeitbibliothek: Dynamisch
- (b) Debug-Informationen: Ja
- (c) Eigenschaften:
  - i. Ähnlich wie /MD, aber mit zusätzlichen Debug-Informationen und weniger Optimierungen.
  - ii. Erleichtert das Debugging.
  - iii. Erfordert, dass die Debug-Version der CRT-DLLs zur Laufzeit verfügbar ist.

## Unterschiede:

1. **Statische vs. Dynamische Verlinkung:** /MT und /MTd binden die Laufzeitbibliothek statisch ein, wodurch die ausführbare Datei unabhängig von externen DLLs ist. /MD und /MDd verlinken dynamisch und erfordern, dass die entsprechenden DLLs zur Laufzeit vorhanden sind.
2. **Debug vs. Release:** Die Debug-Optionen (/MTd und /MDd) enthalten zusätzliche Debug-Informationen und sind nicht so stark optimiert wie die Release-Optionen (/MT und /MD), was das Debugging erleichtert, aber die Leistung beeinträchtigen kann.

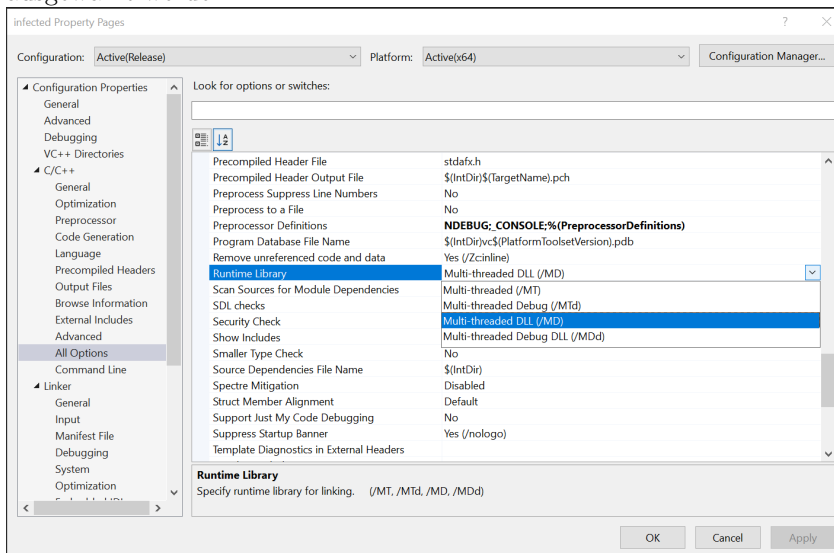
## Der Programm-Code für die gewünschte Executable:

```

1  #include <stdio.h>
2
3  int main() {
4      printf("infected");
5      return 0;
6  }
7

```

In den Einstellungen des Visual Studio Projekts kann die gewünschte Variante für die "Runtime Library" ausgewählt werden:



Die Files nach erstellen, hier ist direkt ersichtlich, dass beide statischen Varianten größer sind (Length), da sich der Code der Libraries im File befindet:

```

PS C:\Users\admin_sin\Desktop\rev3-s2210239021\releases> dir

Directory: C:\Users\admin_sin\Desktop\rev3-s2210239021\releases

Mode                LastWriteTime         Length Name
----                -
-a----          24.10.2023   16:15           10752 infected-md.exe
-a----          24.10.2023   16:16           14336 infected-MDd.exe
-a----          24.10.2023   16:10          139264 infected-mt.exe
-a----          24.10.2023   16:14          376832 infected-mtd.exe

PS C:\Users\admin_sin\Desktop\rev3-s2210239021\releases>

```

## Analyse der Dateien

Um die erzeugten Executables zu analysieren sollen zumindest die Tools **dumpbin** und **strings** verwendet werden.

### dumpbin

**dumpbin** ist ein Command-Line Tool, zur Verfügung gestellt von Microsoft Visual Studio um PE-Files (Portable Executable) zu analysieren. Es können beispielsweise die flags **/DEPENDENTS**, **/EXPORTS**, **/HEADERS** oder **/ALL** verwendet werden, um alle vom Tool lieferbaren Informationen zu erhalten.

Folgende Aufrufe zeigen die Ausgabe ohne Parameter:

```
PS C:\Users\admin_sin\Desktop\rev3-s2210239021\releases> dumpbin.exe .\infected-mt.exe
Microsoft (R) COFF/PE Dumper Version 14.36.32535.0
Copyright (C) Microsoft Corporation. All rights reserved.

Dump of file .\infected-mt.exe

File Type: EXECUTABLE IMAGE

Summary

   2000 .data
   2000 .pdata
   8000 .rdata
   1000 .reloc
   1000 .rsrc
  15000 .text
   1000 .RDATA

PS C:\Users\admin_sin\Desktop\rev3-s2210239021\releases>
```

(a) dumpbin mt-file

```
PS C:\Users\admin_sin\Desktop\rev3-s2210239021\releases> dumpbin.exe .\infected-mtd.exe
Microsoft (R) COFF/PE Dumper Version 14.36.32535.0
Copyright (C) Microsoft Corporation. All rights reserved.

Dump of file .\infected-mtd.exe

File Type: EXECUTABLE IMAGE

Summary

   3000 .data
   4000 .pdata
  15000 .rdata
   1000 .reloc
   1000 .rsrc
  43000 .text
   1000 .RDATA

PS C:\Users\admin_sin\Desktop\rev3-s2210239021\releases>
```

(b) dumpbin mtd-file

```
PS C:\Users\admin_sin\Desktop\rev3-s2210239021\releases> dumpbin.exe .\infected-md.exe
Microsoft (R) COFF/PE Dumper Version 14.36.32535.0
Copyright (C) Microsoft Corporation. All rights reserved.

Dump of file .\infected-md.exe

File Type: EXECUTABLE IMAGE

Summary

   1000 .data
   1000 .pdata
   1000 .rdata
   1000 .reloc
   1000 .rsrc
   1000 .text

PS C:\Users\admin_sin\Desktop\rev3-s2210239021\releases>
```

(c) dumpbin md-file

```
PS C:\Users\admin_sin\Desktop\rev3-s2210239021\releases> dumpbin.exe .\infected-MDd.exe
Microsoft (R) COFF/PE Dumper Version 14.36.32535.0
Copyright (C) Microsoft Corporation. All rights reserved.

Dump of file .\infected-MDd.exe

File Type: EXECUTABLE IMAGE

Summary

   1000 .data
   1000 .pdata
   1000 .rdata
   1000 .reloc
   1000 .rsrc
   2000 .text

PS C:\Users\admin_sin\Desktop\rev3-s2210239021\releases>
```

(d) dumpbin mdd-file

Figure 1: All infected.exe files analysed with dumpbin

Die Debug-Versionen (infected-MDd.exe und infected-mtd.exe) haben tendenziell größere .text Sektionen als ihre entsprechenden Release-Versionen, da sie zusätzliche Debug-Informationen enthalten.

Die statisch verlinkten Versionen (infected-mt.exe und infected-mtd.exe) haben größere .text und .rdata Sektionen im Vergleich zu den dynamisch verlinkten Versionen, da die C Runtime Library direkt in die ausführbare Datei eingebettet ist.

Die dynamisch verlinkten Versionen (infected-md.exe und infected-MDd.exe) sind im Allgemeinen kleiner, weil sie zur Laufzeit externe DLLs verwenden.

## strings

Der Befehl strings wird verwendet, um alle druckbaren Zeichenfolgen in einer Binärdatei zu extrahieren und anzuzeigen. Eine Zeichenfolge in diesem Kontext ist eine Sequenz von druckbaren Zeichen, die typischerweise durch Null-Bytes (\0) oder durch nicht-druckbare Zeichen getrennt ist. Dieses Tool ist nützlich, um Text, Pfade, URLs, Versionen und andere Informationen aus ausführbaren Dateien, Bibliotheken oder beliebigen Binärdateien zu extrahieren.

Folgende zwei screenshots zeigen sowohl den Anfang als auch das Ende der 4 verschiedenen Files:

```

1 output-strings-md.txt
2 [This program cannot be run in DOS mode.]
3 C:\Users\ladmin_sin\Desktop\rev3-s2210239021\infected
4 .text$000
5 .rdata$000
6 .rdata$zzzz
7 .C_specific_handler
8 __current_exception
9 __current_exception_context
10 VC_RUNTIME140.dll
11 __acrt_iob_func
12 __stdio_common_vfprintf
13 _seh_filter_exe
14 __set_app_type
15 __setusermatherr
16 __configure_narrow_argv
17 _initialize_narrow_environment
18 _get_initial_narrow_environment
19 _initter_e
20 _set_fnmode
21 __p_argc
22 __p_argv
23 _register_thread_local_exe_atexit_callback
24 __configthreadlocale
25 __set_new_mode
26 __p_commode
27 _initialize_onexit_table
28 _register_onexit_function
29 _crt_atexit
30 ucrtbased.dll
31 api-ms-win-crt-stdio-l1-1-0.dll
32 api-ms-win-crt-runtime-l1-1-0.dll
33 api-ms-win-crt-math-l1-1-0.dll
34 api-ms-win-crt-locale-l1-1-0.dll
35 api-ms-win-crt-heap-l1-1-0.dll
36 RtlCaptureContext
37 RtlLookupFunctionEntry
38 RTVirtualUnwind
39 UnhandledExceptionFilter
40 SetUnhandledExceptionFilter
41 GetCurrentProcess
42 TerminateProcess
43 IsProcessorFeaturePresent
44 QueryPerformanceCounter
45 GetCurrentProcessId
46 GetCurrentThreadId
47 IsSystemTimeActive
48 InitializeListHead
49 IsDebuggerPresent
50 GetModuleHandleW
51 KERNEL32.dll
52 <xml version='1.0' encoding='UTF-8' standalone='yes'
53 <assembly xmlns='urn:schemas-microsoft-com:asm.v1' m
54 <trustInfo xmlns='urn:schemas-microsoft-com:asm.v3
55 <security
56 <requestedPrivileges
57 <requestedExecutionLevel level='asInvoker' u
58 </security
59 </trustInfo
60 </assembly>
61 main 297

1 output-strings-mdt.txt
2 [This program cannot be run in DOS mode.]
3 C:\Users\ladmin_sin\Desktop\rev3-s2210239021\infected
4 .text$000
5 .rdata$000
6 .rdata$zzzz
7 .C_specific_handler
8 __current_exception
9 __current_exception_context
10 VC_RUNTIME140.dll
11 __acrt_iob_func
12 __stdio_common_vfprintf
13 _seh_filter_exe
14 __set_app_type
15 __setusermatherr
16 __configure_narrow_argv
17 _initialize_narrow_environment
18 _get_initial_narrow_environment
19 _initter_e
20 _set_fnmode
21 __p_argc
22 __p_argv
23 _register_thread_local_exe_atexit_callback
24 __configthreadlocale
25 __set_new_mode
26 __p_commode
27 _initialize_onexit_table
28 _register_onexit_function
29 _crt_atexit
30 ucrtbased.dll
31 api-ms-win-crt-stdio-l1-1-0.dll
32 api-ms-win-crt-runtime-l1-1-0.dll
33 api-ms-win-crt-math-l1-1-0.dll
34 api-ms-win-crt-locale-l1-1-0.dll
35 api-ms-win-crt-heap-l1-1-0.dll
36 RtlCaptureContext
37 RtlLookupFunctionEntry
38 RTVirtualUnwind
39 UnhandledExceptionFilter
40 SetUnhandledExceptionFilter
41 GetCurrentProcess
42 TerminateProcess
43 IsProcessorFeaturePresent
44 QueryPerformanceCounter
45 GetCurrentProcessId
46 GetCurrentThreadId
47 IsSystemTimeActive
48 InitializeListHead
49 IsDebuggerPresent
50 GetModuleHandleW
51 KERNEL32.dll
52 <xml version='1.0' encoding='UTF-8' standalone='yes'
53 <assembly xmlns='urn:schemas-microsoft-com:asm.v1' m
54 <trustInfo xmlns='urn:schemas-microsoft-com:asm.v3
55 <security
56 <requestedPrivileges
57 <requestedExecutionLevel level='asInvoker' u
58 </security
59 </trustInfo
60 </assembly>
61 main 297

1 output-strings-mt.txt
2 [This program cannot be run in DOS mode.]
3 C:\Users\ladmin_sin\Desktop\rev3-s2210239021\infected
4 .text$000
5 .rdata$000
6 .rdata$zzzz
7 .C_specific_handler
8 __current_exception
9 __current_exception_context
10 VC_RUNTIME140.dll
11 __acrt_iob_func
12 __stdio_common_vfprintf
13 _seh_filter_exe
14 __set_app_type
15 __setusermatherr
16 __configure_narrow_argv
17 _initialize_narrow_environment
18 _get_initial_narrow_environment
19 _initter_e
20 _set_fnmode
21 __p_argc
22 __p_argv
23 _register_thread_local_exe_atexit_callback
24 __configthreadlocale
25 __set_new_mode
26 __p_commode
27 _initialize_onexit_table
28 _register_onexit_function
29 _crt_atexit
30 ucrtbased.dll
31 api-ms-win-crt-stdio-l1-1-0.dll
32 api-ms-win-crt-runtime-l1-1-0.dll
33 api-ms-win-crt-math-l1-1-0.dll
34 api-ms-win-crt-locale-l1-1-0.dll
35 api-ms-win-crt-heap-l1-1-0.dll
36 RtlCaptureContext
37 RtlLookupFunctionEntry
38 RTVirtualUnwind
39 UnhandledExceptionFilter
40 SetUnhandledExceptionFilter
41 GetCurrentProcess
42 TerminateProcess
43 IsProcessorFeaturePresent
44 QueryPerformanceCounter
45 GetCurrentProcessId
46 GetCurrentThreadId
47 IsSystemTimeActive
48 InitializeListHead
49 IsDebuggerPresent
50 GetModuleHandleW
51 KERNEL32.dll
52 <xml version='1.0' encoding='UTF-8' standalone='yes'
53 <assembly xmlns='urn:schemas-microsoft-com:asm.v1' m
54 <trustInfo xmlns='urn:schemas-microsoft-com:asm.v3
55 <security
56 <requestedPrivileges
57 <requestedExecutionLevel level='asInvoker' u
58 </security
59 </trustInfo
60 </assembly>
61 main 297

1 output-strings-mtd.txt
2 [This program cannot be run in DOS mode.]
3 C:\Users\ladmin_sin\Desktop\rev3-s2210239021\infected
4 .text$000
5 .rdata$000
6 .rdata$zzzz
7 .C_specific_handler
8 __current_exception
9 __current_exception_context
10 VC_RUNTIME140.dll
11 __acrt_iob_func
12 __stdio_common_vfprintf
13 _seh_filter_exe
14 __set_app_type
15 __setusermatherr
16 __configure_narrow_argv
17 _initialize_narrow_environment
18 _get_initial_narrow_environment
19 _initter_e
20 _set_fnmode
21 __p_argc
22 __p_argv
23 _register_thread_local_exe_atexit_callback
24 __configthreadlocale
25 __set_new_mode
26 __p_commode
27 _initialize_onexit_table
28 _register_onexit_function
29 _crt_atexit
30 ucrtbased.dll
31 api-ms-win-crt-stdio-l1-1-0.dll
32 api-ms-win-crt-runtime-l1-1-0.dll
33 api-ms-win-crt-math-l1-1-0.dll
34 api-ms-win-crt-locale-l1-1-0.dll
35 api-ms-win-crt-heap-l1-1-0.dll
36 RtlCaptureContext
37 RtlLookupFunctionEntry
38 RTVirtualUnwind
39 UnhandledExceptionFilter
40 SetUnhandledExceptionFilter
41 GetCurrentProcess
42 TerminateProcess
43 IsProcessorFeaturePresent
44 QueryPerformanceCounter
45 GetCurrentProcessId
46 GetCurrentThreadId
47 IsSystemTimeActive
48 InitializeListHead
49 IsDebuggerPresent
50 GetModuleHandleW
51 KERNEL32.dll
52 <xml version='1.0' encoding='UTF-8' standalone='yes'
53 <assembly xmlns='urn:schemas-microsoft-com:asm.v1' m
54 <trustInfo xmlns='urn:schemas-microsoft-com:asm.v3
55 <security
56 <requestedPrivileges
57 <requestedExecutionLevel level='asInvoker' u
58 </security
59 </trustInfo
60 </assembly>
61 main 297

```

```

1 output-strings-md.txt
2 [This program cannot be run in DOS mode.]
3 C:\Users\ladmin_sin\Desktop\rev3-s2210239021\infected
4 .text$000
5 .rdata$000
6 .rdata$zzzz
7 .C_specific_handler
8 __current_exception
9 __current_exception_context
10 VC_RUNTIME140.dll
11 __acrt_iob_func
12 __stdio_common_vfprintf
13 _seh_filter_exe
14 __set_app_type
15 __setusermatherr
16 __configure_narrow_argv
17 _initialize_narrow_environment
18 _get_initial_narrow_environment
19 _initter_e
20 _set_fnmode
21 __p_argc
22 __p_argv
23 _register_thread_local_exe_atexit_callback
24 __configthreadlocale
25 __set_new_mode
26 __p_commode
27 _initialize_onexit_table
28 _register_onexit_function
29 _crt_atexit
30 ucrtbased.dll
31 api-ms-win-crt-stdio-l1-1-0.dll
32 api-ms-win-crt-runtime-l1-1-0.dll
33 api-ms-win-crt-math-l1-1-0.dll
34 api-ms-win-crt-locale-l1-1-0.dll
35 api-ms-win-crt-heap-l1-1-0.dll
36 RtlCaptureContext
37 RtlLookupFunctionEntry
38 RTVirtualUnwind
39 UnhandledExceptionFilter
40 SetUnhandledExceptionFilter
41 GetCurrentProcess
42 TerminateProcess
43 IsProcessorFeaturePresent
44 QueryPerformanceCounter
45 GetCurrentProcessId
46 GetCurrentThreadId
47 IsSystemTimeActive
48 InitializeListHead
49 IsDebuggerPresent
50 GetModuleHandleW
51 KERNEL32.dll
52 <xml version='1.0' encoding='UTF-8' standalone='yes'
53 <assembly xmlns='urn:schemas-microsoft-com:asm.v1' m
54 <trustInfo xmlns='urn:schemas-microsoft-com:asm.v3
55 <security
56 <requestedPrivileges
57 <requestedExecutionLevel level='asInvoker' u
58 </security
59 </trustInfo
60 </assembly>
61 main 297

1 output-strings-mdt.txt
2 [This program cannot be run in DOS mode.]
3 C:\Users\ladmin_sin\Desktop\rev3-s2210239021\infected
4 .text$000
5 .rdata$000
6 .rdata$zzzz
7 .C_specific_handler
8 __current_exception
9 __current_exception_context
10 VC_RUNTIME140.dll
11 __acrt_iob_func
12 __stdio_common_vfprintf
13 _seh_filter_exe
14 __set_app_type
15 __setusermatherr
16 __configure_narrow_argv
17 _initialize_narrow_environment
18 _get_initial_narrow_environment
19 _initter_e
20 _set_fnmode
21 __p_argc
22 __p_argv
23 _register_thread_local_exe_atexit_callback
24 __configthreadlocale
25 __set_new_mode
26 __p_commode
27 _initialize_onexit_table
28 _register_onexit_function
29 _crt_atexit
30 ucrtbased.dll
31 api-ms-win-crt-stdio-l1-1-0.dll
32 api-ms-win-crt-runtime-l1-1-0.dll
33 api-ms-win-crt-math-l1-1-0.dll
34 api-ms-win-crt-locale-l1-1-0.dll
35 api-ms-win-crt-heap-l1-1-0.dll
36 RtlCaptureContext
37 RtlLookupFunctionEntry
38 RTVirtualUnwind
39 UnhandledExceptionFilter
40 SetUnhandledExceptionFilter
41 GetCurrentProcess
42 TerminateProcess
43 IsProcessorFeaturePresent
44 QueryPerformanceCounter
45 GetCurrentProcessId
46 GetCurrentThreadId
47 IsSystemTimeActive
48 InitializeListHead
49 IsDebuggerPresent
50 GetModuleHandleW
51 KERNEL32.dll
52 <xml version='1.0' encoding='UTF-8' standalone='yes'
53 <assembly xmlns='urn:schemas-microsoft-com:asm.v1' m
54 <trustInfo xmlns='urn:schemas-microsoft-com:asm.v3
55 <security
56 <requestedPrivileges
57 <requestedExecutionLevel level='asInvoker' u
58 </security
59 </trustInfo
60 </assembly>
61 main 297

1 output-strings-mt.txt
2 [This program cannot be run in DOS mode.]
3 C:\Users\ladmin_sin\Desktop\rev3-s2210239021\infected
4 .text$000
5 .rdata$000
6 .rdata$zzzz
7 .C_specific_handler
8 __current_exception
9 __current_exception_context
10 VC_RUNTIME140.dll
11 __acrt_iob_func
12 __stdio_common_vfprintf
13 _seh_filter_exe
14 __set_app_type
15 __setusermatherr
16 __configure_narrow_argv
17 _initialize_narrow_environment
18 _get_initial_narrow_environment
19 _initter_e
20 _set_fnmode
21 __p_argc
22 __p_argv
23 _register_thread_local_exe_atexit_callback
24 __configthreadlocale
25 __set_new_mode
26 __p_commode
27 _initialize_onexit_table
28 _register_onexit_function
29 _crt_atexit
30 ucrtbased.dll
31 api-ms-win-crt-stdio-l1-1-0.dll
32 api-ms-win-crt-runtime-l1-1-0.dll
33 api-ms-win-crt-math-l1-1-0.dll
34 api-ms-win-crt-locale-l1-1-0.dll
35 api-ms-win-crt-heap-l1-1-0.dll
36 RtlCaptureContext
37 RtlLookupFunctionEntry
38 RTVirtualUnwind
39 UnhandledExceptionFilter
40 SetUnhandledExceptionFilter
41 GetCurrentProcess
42 TerminateProcess
43 IsProcessorFeaturePresent
44 QueryPerformanceCounter
45 GetCurrentProcessId
46 GetCurrentThreadId
47 IsSystemTimeActive
48 InitializeListHead
49 IsDebuggerPresent
50 GetModuleHandleW
51 KERNEL32.dll
52 <xml version='1.0' encoding='UTF-8' standalone='yes'
53 <assembly xmlns='urn:schemas-microsoft-com:asm.v1' m
54 <trustInfo xmlns='urn:schemas-microsoft-com:asm.v3
55 <security
56 <requestedPrivileges
57 <requestedExecutionLevel level='asInvoker' u
58 </security
59 </trustInfo
60 </assembly>
61 main 297

1 output-strings-mtd.txt
2 [This program cannot be run in DOS mode.]
3 C:\Users\ladmin_sin\Desktop\rev3-s2210239021\infected
4 .text$000
5 .rdata$000
6 .rdata$zzzz
7 .C_specific_handler
8 __current_exception
9 __current_exception_context
10 VC_RUNTIME140.dll
11 __acrt_iob_func
12 __stdio_common_vfprintf
13 _seh_filter_exe
14 __set_app_type
15 __setusermatherr
16 __configure_narrow_argv
17 _initialize_narrow_environment
18 _get_initial_narrow_environment
19 _initter_e
20 _set_fnmode
21 __p_argc
22 __p_argv
23 _register_thread_local_exe_atexit_callback
24 __configthreadlocale
25 __set_new_mode
26 __p_commode
27 _initialize_onexit_table
28 _register_onexit_function
29 _crt_atexit
30 ucrtbased.dll
31 api-ms-win-crt-stdio-l1-1-0.dll
32 api-ms-win-crt-runtime-l1-1-0.dll
33 api-ms-win-crt-math-l1-1-0.dll
34 api-ms-win-crt-locale-l1-1-0.dll
35 api-ms-win-crt-heap-l1-1-0.dll
36 RtlCaptureContext
37 RtlLookupFunctionEntry
38 RTVirtualUnwind
39 UnhandledExceptionFilter
40 SetUnhandledExceptionFilter
41 GetCurrentProcess
42 TerminateProcess
43 IsProcessorFeaturePresent
44 QueryPerformanceCounter
45 GetCurrentProcessId
46 GetCurrentThreadId
47 IsSystemTimeActive
48 InitializeListHead
49 IsDebuggerPresent
50 GetModuleHandleW
51 KERNEL32.dll
52 <xml version='1.0' encoding='UTF-8' standalone='yes'
53 <assembly xmlns='urn:schemas-microsoft-com:asm.v1' m
54 <trustInfo xmlns='urn:schemas-microsoft-com:asm.v3
55 <security
56 <requestedPrivileges
57 <requestedExecutionLevel level='asInvoker' u
58 </security
59 </trustInfo
60 </assembly>
61 main 297

```

## Fragen

### 1. Welche Imports werden verwendet?

Durch den Aufruf von `dumpbin /IMPORTS <PE-filename>` können die Imports ermittelt werden:

```
PS C:\Users\Quickemu\repos\REV3\UE02\rev3-s2210239021\windows\releases> dumpbin /IMPORTS .\infected-mt.exe
Microsoft (R) COFF/PE Dumper Version 14.37.32825.0
Copyright (C) Microsoft Corporation. All rights reserved.

Dump of file .\infected-mt.exe

File Type: EXECUTABLE IMAGE

Section contains the following imports:

  KERNEL32.dll
    140016000 Import Address Table
    14001FE98 Import Name Table
    0 time date stamp
    0 Index of first forwarder reference

    4F5 RtlCaptureContext
    4FD RtlLookupFunctionEntry
    504 RtlVirtualUnwind
    5E6 UnhandledExceptionFilter
    5A4 SetUnhandledExceptionFilter
    232 GetCurrentProcess
    5C4 TerminateProcess
    3A8 IsProcessorFeaturePresent
    470 QueryPerformanceCounter
    233 GetCurrentProcessId
    237 GetCurrentThreadId
    30A GetSystemTimeAsFileTime
    38A InitializeSlistHead
```

- (a) infected-mt.exe
  - i. KERNEL32.dll
- (b) infected-mtd.exe
  - i. KERNEL32.dll
- (c) infected-md.exe
  - i. VCRUNTIME140.dll
  - ii. api-ms-win-crt-stdio-l1-1-0.dll
  - iii. api-ms-win-crt-runtime-l1-1-0.dll
  - iv. api-ms-win-crt-math-l1-1-0.dll
  - v. api-ms-win-crt-locale-l1-1-0.dll
  - vi. api-ms-win-crt-heap-l1-1-0.dll
  - vii. KERNEL32.dll
- (d) infected-MDd.exe
  - i. VCRUNTIME140D.dll
  - ii. ucrtbased.dll
  - iii. KERNEL32.dll

## 2. Welche Sektionen werden verwendet?

Durch den Aufruf von `dumpbin <PE-filename>` können die Sektionen ermittelt werden:

```
PS C:\Users\admin_sin\Desktop\rev3-s2210239021\releases> dumpbin.exe .\infected-md.exe
Microsoft (R) COFF/PE Dumper Version 14.36.32535.0
Copyright (C) Microsoft Corporation. All rights reserved.

Dump of file .\infected-md.exe

File Type: EXECUTABLE IMAGE

Summary

           1000 .data
           1000 .pdata
           1000 .rdata
           1000 .reloc
           1000 .rsrc
           1000 .text
PS C:\Users\admin_sin\Desktop\rev3-s2210239021\releases> |
```

(a) infected-mt.exe

- i. .text
- ii. .rdata
- iii. .data
- iv. .pdata
- v. \_RDATA
- vi. .rsrc
- vii. .reloc

(b) infected-mtd.exe

- i. .text
- ii. .rdata
- iii. .data
- iv. .pdata
- v. \_RDATA
- vi. .rsrc
- vii. .reloc

(c) infected-md.exe

- i. .text
- ii. .rdata
- iii. .data
- iv. .pdata
- v. .rsrc
- vi. .reloc

(d) infected-MDd.exe

- i. .text
- ii. .rdata
- iii. .data
- iv. .pdata
- v. .rsrc
- vi. .reloc

3. **Was kannst du über die\*den Author\*in sagen?** Ein direkter Author ist nicht erkennbar. Allerdings könnte man auf "admin.sin" schließen, da das File für die "Program Database" (infected.pdb) im Pfad den User nennt:

```

1      Debug Directories
2
3      Time Type          Size          RVA      Pointer
4      -----
5      6537D1B0 cv          66 00003464      2264      Format: RSDS, {6F72B84A-D687
-4743-A104-E88EF40E7E96}, 4, C:\Users\admin.sin\Desktop\rev3-s2210239021\
infected\x64\Release\infected.pdb
6      6537D1B0 feat        14 000034CC      22CC      Counts: Pre-VC++ 11.00=0, C/C
++=30, /GS=30, /sdl=1, guardN=29
7      6537D1B0 coffgrp      284 000034E0      22E0      4C544347 (LTCG)
8      6537D1B0 iltcg         0 00000000         0
9
10

```

Dies ist in allen Files gleich herauszulesen.

4. **Was kannst du über die Umgebung, in der das Sample erzeugt wurde, sagen?**  
 Auskünfte über die Umgebung können durch die Compiler-Version, eingebettete Ressourcen oder Abhängigkeiten und der Weiteren gegeben werden.

#### HEADERS

Durch den Aufruf von `dumpbin /HEADERS <PE-filename>` können mehrere Informationen über die Umgebung herausgefunden werden:

```

PS C:\Users\Quickemu\repos\REV3\UE02\rev3-s2210239021\windows\releases> dumpbin /headers .\infected-md.exe
Microsoft (R) COFF/PE Dumper Version 14.37.32825.0
Copyright (C) Microsoft Corporation. All rights reserved.

Dump of file .\infected-md.exe

PE signature found

File Type: EXECUTABLE IMAGE

FILE HEADER VALUES
      8664 machine (x64)
           6 number of sections
      6537D186 time date stamp Tue Oct 24 16:15:34 2023
           0 file pointer to symbol table
           0 number of symbols
           F0 size of optional header
          22 characteristics
              Executable
              Application can handle large (>2GB) addresses

```

(Der Screenshot zeigt nur den Beginn des outputs...)

Folgende Informationen können über die Files gefunden werden:

- (a) infected-mt.exe
- i. **Architektur:** 8664 machine (x64)
  - ii. **Linker-Version:** 14.36 (typisch für Visual Studio-Version)
  - iii. **Zeitstempel:** Tue Oct 24 16:10:20 2023
  - iv. **Subsystem:** Windows CUI (Konsolenanwendung)
  - v. **DLL-Charakteristika:**  
 Kompatibel mit "High Entropy Virtual Addresses", "Dynamic Base", "NX (No eXecute)", "Terminal Server Aware".  
 Diese Eigenschaften sind Sicherheitsmerkmale, die oft in modernen Anwendungen verwendet werden.



- vi. Speicherreservierung: Die Größen für Stack- und Heap-Reservierung und -Commit sind angegeben, was Hinweise auf den für die Ausführung der Anwendung benötigten Speicher gibt.
- vii. **Verzeichnisse und Sektionen:** Verschiedene Verzeichnisse wie das Importverzeichnis, das Resourceverzeichnis, das Exceptionverzeichnis usw. sind aufgelistet. Diese geben Informationen über die Struktur der ausführbaren Datei und welche externen Funktionen oder Ressourcen sie verwendet.  
Beispielsweise nur für mt-Version angeführt:

```

1          10 number of directories
2          0 [          0] RVA [size] of Export Directory
3          28DC [          A0] RVA [size] of Import Directory
4          5000 [          1E0] RVA [size] of Resource Directory
5          4000 [          174] RVA [size] of Exception Directory
6          0 [          0] RVA [size] of Certificates Directory
7          6000 [          30] RVA [size] of Base Relocation Directory
8          23A0 [          70] RVA [size] of Debug Directory
9          0 [          0] RVA [size] of Architecture Directory
10         0 [          0] RVA [size] of Global Pointer Directory
11         0 [          0] RVA [size] of Thread Storage Directory
12        2260 [          140] RVA [size] of Load Configuration Directory
13         0 [          0] RVA [size] of Bound Import Directory
14        2000 [          1A0] RVA [size] of Import Address Table Directory
15         0 [          0] RVA [size] of Delay Import Directory
16         0 [          0] RVA [size] of COM Descriptor Directory
17         0 [          0] RVA [size] of Reserved Directory
18

```

- viii. **Betriebssystem- und Subsystem-Version:** Das Betriebssystem und das Subsystem sind auf Version 6.00 eingestellt. Dies könnte auf eine Kompatibilität mit bestimmten Windows-Versionen hinweisen (z.B. Windows Vista, 7, 8, 10), die alle NT 6.x-Versionen sind.
  - ix. **Art der Ausführbaren:** Die Ausführbare ist als "Executable" markiert, was darauf hindeutet, dass es sich um eine Standard-Ausführbare (und nicht um eine DLL) handelt.
- (b) infected-mtd.exe
- i. **Architektur:** 8664 machine (x64)
  - ii. **Linker-Version:** 14.36 (typisch für Visual Studio-Version)
  - iii. **Zeitstempel:** Tue Oct 24 16:14:17 2023
  - iv. **Subsystem:** Windows CUI (Konsolenanwendung)
  - v. **DLL-Charakteristika:**  
Kompatibel mit "High Entropy Virtual Addresses", "Dynamic Base", "NX (No eXecute)", "Terminal Server Aware".  
Diese Eigenschaften sind Sicherheitsmerkmale, die oft in modernen Anwendungen verwendet werden.
  - vi. Speicherreservierung: Die Größen für Stack- und Heap-Reservierung und -Commit sind angegeben, was Hinweise auf den für die Ausführung der Anwendung benötigten Speicher gibt.
  - vii. **Verzeichnisse und Sektionen:** Verschiedene Verzeichnisse wie das Importverzeichnis, das Resourceverzeichnis, das Exceptionverzeichnis usw. sind aufgelistet. Diese geben Informationen über die Struktur der ausführbaren Datei und welche externen Funktionen oder Ressourcen sie verwendet.
  - viii. **Betriebssystem- und Subsystem-Version:** Das Betriebssystem und das Subsystem sind auf Version 6.00 eingestellt. Dies könnte auf eine Kompatibilität mit bestimmten Windows-Versionen hinweisen (z.B. Windows Vista, 7, 8, 10), die alle NT 6.x-Versionen sind.
  - ix. **Art der Ausführbaren:** Die Ausführbare ist als "Executable" markiert, was darauf hindeutet, dass es sich um eine Standard-Ausführbare (und nicht um eine DLL) handelt.

- (c) infected-md.exe
- i. **Architektur:** 8664 machine (x64)
  - ii. **Linker-Version:** 14.36 (typisch für Visual Studio-Version)
  - iii. **Zeitstempel:** Tue Oct 24 16:15:34 2023
  - iv. **Subsystem:** Windows CUI (Konsolenanwendung)
  - v. **DLL-Charakteristika:**  
Kompatibel mit "High Entropy Virtual Addresses", "Dynamic Base", "NX (No eXecute)", "Terminal Server Aware".  
Diese Eigenschaften sind Sicherheitsmerkmale, die oft in modernen Anwendungen verwendet werden.
  - vi. **Speicherreservierung:** Die Größen für Stack- und Heap-Reservierung und -Commit sind angegeben, was Hinweise auf den für die Ausführung der Anwendung benötigten Speicher gibt.
  - vii. **Verzeichnisse und Sektionen:** Verschiedene Verzeichnisse wie das Importverzeichnis, das Resourceverzeichnis, das Exceptionverzeichnis usw. sind aufgelistet. Diese geben Informationen über die Struktur der ausführbaren Datei und welche externen Funktionen oder Ressourcen sie verwendet.
  - viii. **Betriebssystem- und Subsystem-Version:** Das Betriebssystem und das Subsystem sind auf Version 6.00 eingestellt. Dies könnte auf eine Kompatibilität mit bestimmten Windows-Versionen hinweisen (z.B. Windows Vista, 7, 8, 10), die alle NT 6.x-Versionen sind.
  - ix. **Art der Ausführbaren:** Die Ausführbare ist als "Executable" markiert, was darauf hindeutet, dass es sich um eine Standard-Ausführbare (und nicht um eine DLL) handelt.
- (d) infected-MDd.exe
- i. **Architektur:** 8664 machine (x64)
  - ii. **Linker-Version:** 14.36 (typisch für Visual Studio-Version)
  - iii. **Zeitstempel:** Tue Oct 24 16:16:16 2023
  - iv. **Subsystem:** Windows CUI (Konsolenanwendung)
  - v. **DLL-Charakteristika:**  
Kompatibel mit "High Entropy Virtual Addresses", "Dynamic Base", "NX (No eXecute)", "Terminal Server Aware".  
Diese Eigenschaften sind Sicherheitsmerkmale, die oft in modernen Anwendungen verwendet werden.
  - vi. **Speicherreservierung:** Die Größen für Stack- und Heap-Reservierung und -Commit sind angegeben, was Hinweise auf den für die Ausführung der Anwendung benötigten Speicher gibt.
  - vii. **Verzeichnisse und Sektionen:** Verschiedene Verzeichnisse wie das Importverzeichnis, das Resourceverzeichnis, das Exceptionverzeichnis usw. sind aufgelistet. Diese geben Informationen über die Struktur der ausführbaren Datei und welche externen Funktionen oder Ressourcen sie verwendet.
  - viii. **Betriebssystem- und Subsystem-Version:** Das Betriebssystem und das Subsystem sind auf Version 6.00 eingestellt. Dies könnte auf eine Kompatibilität mit bestimmten Windows-Versionen hinweisen (z.B. Windows Vista, 7, 8, 10), die alle NT 6.x-Versionen sind.
  - ix. **Art der Ausführbaren:** Die Ausführbare ist als "Executable" markiert, was darauf hindeutet, dass es sich um eine Standard-Ausführbare (und nicht um eine DLL) handelt.

Unterschiede in den Versionen finden sich in der Anzahl der Sektionen, dem Zeitstempel, der Größe des Codes, der Relativ Virtuellen Adresse (RVA) sowie der Größe verschiedener Verzeichnisse (Import Directory, Resource Directory, Exception Directory, Base Relocation Directory, Debug Directory, Load Configuration Directory, Import Address Table Directory) wieder. Ebenfalls unterscheidet sich die Anzahl der Verzeichnisse.

## DEPENDENCIES

Durch den Aufruf von `dumpbin /DEPENDENCIES <PE-filename>` können mehrere Informationen über die Umgebung herausgefunden werden:

```
PS C:\Users\Quickemu\repos\REV3\UE02\rev3-s2210239021\windows\releases> dumpbin /dependents .\infected-mt.exe
Microsoft (R) COFF/PE Dumper Version 14.37.32825.0
Copyright (C) Microsoft Corporation. All rights reserved.

Dump of file .\infected-mt.exe
File Type: EXECUTABLE IMAGE

Image has the following dependencies:
    KERNEL32.dll

Summary
    2000 .data
    2000 .pdata
    5000 .rdata
    1000 .reloc
    1000 .rsrc
    1000 text
    1000 .idata
PS C:\Users\Quickemu\repos\REV3\UE02\rev3-s2210239021\windows\releases>
```

(a) dependencies mt-file

```
PS C:\Users\Quickemu\repos\REV3\UE02\rev3-s2210239021\windows\releases> dumpbin /dependents .\infected-mtd.exe
Microsoft (R) COFF/PE Dumper Version 14.37.32825.0
Copyright (C) Microsoft Corporation. All rights reserved.

Dump of file .\infected-mtd.exe
File Type: EXECUTABLE IMAGE

Image has the following dependencies:
    KERNEL32.dll

Summary
    3000 .data
    4000 .pdata
    15000 .rdata
    1000 .reloc
    1000 .rsrc
    43000 text
    1000 .idata
PS C:\Users\Quickemu\repos\REV3\UE02\rev3-s2210239021\windows\releases>
```

(b) dependencies mtd-file

```
PS C:\Users\Quickemu\repos\REV3\UE02\rev3-s2210239021\windows\releases> dumpbin /dependents .\infected-md.exe
Microsoft (R) COFF/PE Dumper Version 14.37.32825.0
Copyright (C) Microsoft Corporation. All rights reserved.

Dump of file .\infected-md.exe
File Type: EXECUTABLE IMAGE

Image has the following dependencies:
    VCRUNTIME140.dll
    api-ms-win-crt-stdio-l1-1-0.dll
    api-ms-win-crt-runtime-l1-1-0.dll
    api-ms-win-crt-math-l1-1-0.dll
    api-ms-win-crt-locale-l1-1-0.dll
    api-ms-win-crt-heap-l1-1-0.dll
    KERNEL32.dll

Summary
    1000 .data
    1000 .pdata
    1000 .rdata
    1000 .reloc
    1000 .rsrc
    1000 text
PS C:\Users\Quickemu\repos\REV3\UE02\rev3-s2210239021\windows\releases>
```

(c) dependencies md-file

```
PS C:\Users\Quickemu\repos\REV3\UE02\rev3-s2210239021\windows\releases> dumpbin /dependents .\infected-mdd.exe
Microsoft (R) COFF/PE Dumper Version 14.37.32825.0
Copyright (C) Microsoft Corporation. All rights reserved.

Dump of file .\infected-mdd.exe
File Type: EXECUTABLE IMAGE

Image has the following dependencies:
    VCRUNTIME140.dll
    ucrtbased.dll
    KERNEL32.dll

Summary
    1000 .data
    1000 .pdata
    1000 .rdata
    1000 .reloc
    1000 .rsrc
    2000 text
PS C:\Users\Quickemu\repos\REV3\UE02\rev3-s2210239021\windows\releases>
```

(d) dependencies mdd-file

Figure 2: All dependencies

Die Unterschiedlichen Abhängigkeiten geben ebenfalls Auskunft über die Umgebung.

## Weitere Informationen zur Umgebung

Es können natürlich noch weitere Informationen über die Umgebung gefunden werden, hilfreich können auch die Parameter `/DEBUG`, `/IMPORTS` oder `/RESSOURCES` sein, es kommt jedoch immer darauf an, wonach gesucht wird.

## 2. Aufgabe - Statische Analyse Linux

### create file

Gleicher code wie zuvor:

```

1  #include <stdio.h>
2
3  int main() {
4      printf("infected");
5      return = 0;
6  }
7

```

Kompilieren und auflisten der Executables:

```

mendacium fedora ./REV3 > UE02 > rev3-s2210239021 > linux
→ gcc -std=c99 -Wall -pedantic infected.c -o infected.out time:1ms
mendacium fedora ./REV3 > UE02 > rev3-s2210239021 > linux
→ gcc -std=c99 -Wall -pedantic infected.c -ggdb -o infected-ggdb.out
mendacium fedora ./REV3 > UE02 > rev3-s2210239021 > linux
→ gcc -std=c99 -Wall -pedantic infected.c -static -o infected-static.out
mendacium fedora ./REV3 > UE02 > rev3-s2210239021 > linux
→ gcc -std=c99 -Wall -pedantic infected.c -static -ggdb -o infected-static-ggdb.out
mendacium fedora ./REV3 > UE02 > rev3-s2210239021 > linux
→ time:80ms

```

```

mendacium fedora ./REV3 > UE02 > rev3-s2210239021 > linux
→ la time:1ms
total 1.6M
-rw-r--r-- 1 mendacium mendacium 69 Oct 24 16:54 infected.c
-rwxr-xr-x 1 mendacium mendacium 18K Oct 24 16:58 infected-ggdb.out*
-rwxr-xr-x 1 mendacium mendacium 17K Oct 24 16:55 infected.out*
-rwxr-xr-x 1 mendacium mendacium 777K Oct 24 17:01 infected-static-ggdb.out*
-rwxr-xr-x 1 mendacium mendacium 776K Oct 24 17:00 infected-static.out*
mendacium fedora ./REV3 > UE02 > rev3-s2210239021 > linux
→ time:1ms

```

### Fragen

#### 1. Welche Imports werden verwendet?

Um importierte Bibliotheken und Funktionen zu überprüfen kann `readelf -d <elf-filename>` oder `objdump -T <elf-filename>` verwendet werden.

Für die Beantwortung der Frage für die Imports kann `objdump` verwendet werden:

```

mendacium@mendacium ~$ ~/fh/REV3/UE02/rev3-s2210239021/linux $ main objdump -T infected.out
infected.out: file format elf64-x86-64

DYNAMIC SYMBOL TABLE:
0000000000000000 DF *UND* 0000000000000000 (GLIBC_2.34) __libc_start_main
0000000000000000 DF *UND* 0000000000000000 (GLIBC_2.2.5) printf
0000000000000000 w D *UND* 0000000000000000 Base __gmon_start__

```

#### (a) infected.out:

```

1  infected.out: file format elf64-x86-64
2
3  DYNAMIC SYMBOL TABLE:
4  0000000000000000 DF *UND* 0000000000000000 (GLIBC_2.34)
   __libc_start_main
5  0000000000000000 DF *UND* 0000000000000000 (GLIBC_2.2.5) printf
6  0000000000000000 w D *UND* 0000000000000000 Base
   __gmon_start__
7

```

#### i. \_\_libc\_start\_main

Dieser Eintrag zeigt, dass die Funktion `__libc_start_main` aus der Standard-C-Bibliothek (GLIBC, Version 2.34) importiert wird. Diese Funktion ist der übliche Einstiegspunkt

für C-Programme und wird intern vom System aufgerufen, um die Ausführung des Programms zu starten.

ii. **printf**

Dieser Eintrag zeigt, dass die Funktion `printf` ebenfalls aus der Standard-C-Bibliothek (GLIBC, Version 2.2.5 oder höher) importiert wird. Diese Funktion wird im Code verwendet, um den String "infected" auszugeben.

iii. **gmon\_start**

Dieser Eintrag ist ein schwacher undefinierter (weak undefined) Symbol, der mit der Profilerstellung (GNU gmon) zusammenhängt. Es ist nicht unbedingt notwendig für die Ausführung des Programms und wird nur aufgerufen, wenn es definiert ist.

(b) **infected-ggdb.out:**

```

1      infected-ggdb.out:      file format elf64-x86-64
2
3      DYNAMIC SYMBOL TABLE:
4      0000000000000000      DF *UND*  0000000000000000  (GLIBC_2.34)
      __libc_start_main
5      0000000000000000      DF *UND*  0000000000000000  (GLIBC_2.2.5) printf
6      0000000000000000      w  D  *UND*  0000000000000000  Base
      __gmon_start__
7
```

Die Ergebnisse in `infected-ggdb.out` sind gleich zu erklären wie in `infected.out`.

(c) **infected-static.out:**

```

1      infected-static.out:      file format elf64-x86-64
2
3      objdump: infected-static.out: not a dynamic object
4      DYNAMIC SYMBOL TABLE:
5      no symbols
6
```

Da alle benötigten Bibliotheken und Abhängigkeiten direkt in die ausführbare Datei eingebettet werden, sind auch keine dynamischen Objekte/Symbole zu finden.

(d) **infected-static-ggdb.out:**

```

1      infected-static-ggdb.out:      file format elf64-x86-64
2
3      objdump: infected-static-ggdb.out: not a dynamic object
4      DYNAMIC SYMBOL TABLE:
5      no symbols
6
```

Die Ergebnisse in `infected-static-ggdb.out` sind gleich zu erklären wie in `infected-static.out`.

2. Welche Sektionen sind vorhanden? Mit dem Befehl `readelf -S <elf-filename>` können die vorhandenen Sektionen ermittelt werden:

```

x mendacium@mendacium ~/FH/REV3/UE02/rev3-s2210239021/linux % main readelf -S infected.out
There are 32 section headers, starting at offset 0x3930:

Section Headers:
[Nr] Name           Type             Address          Offset
Size             EntSize          Flags Link Info Align
[ 0]                NULL             0000000000000000 0 0 0
0000000000000000 0000000000000000
[ 1] .interp          PROGBITS         000000000400318 00000318
000000000000001c 0000000000000000 A 0 0 1
[ 2] .note.gnu.pr[...] NOTE             000000000400338 00000338
0000000000000040 0000000000000000 A 0 0 8
[ 3] .note.gnu.bu[...] NOTE             000000000400378 00000378
0000000000000024 0000000000000000 A 0 0 4
[ 4] .note.ABI-tag    NOTE             00000000040039c 0000039c
0000000000000020 0000000000000000 A 0 0 4
[ 5] .gnu.hash        GNU_HASH         0000000004003c0 000003c0
000000000000001c 0000000000000000 A 6 0 8
[ 6] .dynsym          DYNSYM           0000000004003e0 000003e0
0000000000000060 0000000000000018 A 7 1 8
[ 7] .dynstr          STRTAB           000000000400440 00000440
000000000000004a 0000000000000000 A 0 0 1
[ 8] .gnu.version     VERSYM           00000000040048a 0000048a
0000000000000008 0000000000000002 A 6 0 2
[ 9] .gnu.version_r   VERNEED         000000000400498 00000498
0000000000000030 0000000000000000 A 7 1 8
[10] .rela.dyn        RELA             0000000004004c8 000004c8
0000000000000030 0000000000000018 A 6 0 8

```

(Der Screenshot zeigt nur den Beginn des Outputs.)

Folglich werden nur die Sektionsnamen beginnend mit einem Punkt durch den Befehl `readelf -S <elf-filename> | awk 'print $2' | grep '^\..'` angezeigt.

(a) infected.out:

```

1      .rela.dyn
2      .rela.plt
3      .init
4      .plt
5      .text
6      .fini
7      .rodata
8      .eh_frame_hdr
9      .eh_frame
10     .init_array
11     .fini_array
12     .dynamic
13     .got
14     .got.plt
15     .data
16     .bss
17     .comment
18     .annobin.notes
19     .gnu.build.a[...]
20     .symtab
21     .strtab
22     .shstrtab
23

```

(b) infected-ggdb.out:

```

1      .rela.dyn
2      .rela.plt
3      .init
4      .plt
5      .text
6      .fini
7      .rodata
8      .eh_frame_hdr
9      .eh_frame
10     .init_array
11     .fini_array

```

```
12      .dynamic
13      .got
14      .got.plt
15      .data
16      .bss
17      .comment
18      .annobin.notes
19      .gnu.build.a[...]
20      .debug_aranges
21      .debug_info
22      .debug_abbrev
23      .debug_line
24      .debug_str
25      .debug_line_str
26      .symtab
27      .strtab
28      .shstrtab
29
```

(c) infected-static.out:

```
1      .rodata
2      .stapsdt.base
3      .eh_frame
4      .gcc_except_table
5      .tdata
6      .tbss
7      .init_array
8      .fini_array
9      .data.rel.ro
10     .got
11     .got.plt
12     .data
13     .bss
14     .comment
15     .annobin.notes
16     .note.stapsdt
17     .gnu.build.a[...]
18     .symtab
19     .strtab
20     .shstrtab
21
```

(d) infected-static-ggdb.out:

```
1      .rodata
2      .stapsdt.base
3      .eh_frame
4      .gcc_except_table
5      .tdata
6      .tbss
7      .init_array
8      .fini_array
9      .data.rel.ro
10     .got
11     .got.plt
12     .data
13     .bss
14     .comment
15     .annobin.notes
16     .note.stapsdt
17     .gnu.build.a[...]
18     .debug_aranges
19     .debug_info
20     .debug_abbrev
21     .debug_line
22     .debug_str
```

```

23         .debug_line_str
24         .symtab
25         .strtab
26         .shstrtab
27

```

### 3. Was kannst du über die\*den Autor\*in sagen?

Es ist oft schwierig, spezifische Informationen über den Autor aus einer ausführbaren Datei zu extrahieren. Der kompilierte Code, enthält keine solchen Informationen. In der Regel können solche Informationen (falls vorhanden) in den Metadaten oder in den Kommentaren im Code gefunden werden, aber in diesem Fall gibt es keine solchen Informationen.

Der Aufruf mit `string <elf-filename>` zeigt ebenfalls keine Zeichenketten welche auf den Author schließen lassen würden.

### 4. Was kannst du über die Umgebung, in der das Sample erzeugt wurde, sagen?

Durch die Aufrufe von `file <elf-filename>` und `readelf -h <elf-filename>` können grundlegende Informationen über die Architektur und das Betriebssystem für welche die Datei kompiliert wurden herausgefunden werden:

```

mendactum@mendactum ~/fh/REV3/UE02/rev3-s2210233921/linux % cat file infected.out
infected.out: ELF 64-bit LSB executable, x86-64, version 1 (SYSV), dynamically linked, interpreter /lib64/ld-linux-
x86-64.so.2, BuildID[sha1]=30c06d9c747c6e7a097d5dca07c4354e1c6402b1, for GNU/Linux 3.2.0, not stripped
mendactum@mendactum ~/fh/REV3/UE02/rev3-s2210233921/linux % main readelf -h infected.out
ELF Header:
  Magic:   7f 45 4c 46 02 01 01 00 00 00 00 00 00 00 00 00
  Class:             ELF64
  Data:              2's complement, little endian
  Version:           1 (current)
  OS/ABI:            UNIX - System V
  ABI Version:       0
  Type:              EXEC (Executable file)
  Machine:           Advanced Micro Devices X86-64
  Version:           0x1
  Entry point address: 0x401040
  Start of program headers: 64 (bytes into file)
  Start of section headers: 14640 (bytes into file)
  Flags:             0x0
  Size of this header: 64 (bytes)
  Size of program headers: 56 (bytes)
  Number of program headers: 13
  Size of section headers: 64 (bytes)
  Number of section headers: 32
  Section header string table index: 31
mendactum@mendactum ~/fh/REV3/UE02/rev3-s2210233921/linux % main

```

#### (a) infected.out:

Grundlegende Informationen:

```

1         infected.out: ELF 64-bit LSB executable, x86-64, version 1 (SYSV),
        dynamically linked, interpreter /lib64/ld-linux-x86-64.so.2, BuildID[sha1
        ]=30c06d9c747c6e7a097d5dca07c4354e1c6402b1, for GNU/Linux 3.2.0, not
        stripped
2

```

ELF-Header:

```

1         ELF Header:
2         Magic:   7f 45 4c 46 02 01 01 00 00 00 00 00 00 00 00 00
3         Class:             ELF64
4         Data:              2's complement, little endian
5         Version:           1 (current)
6         OS/ABI:            UNIX - System V
7         ABI Version:       0
8         Type:              EXEC (Executable file)
9         Machine:           Advanced Micro Devices X86-64
10        Version:           0x1
11        Entry point address: 0x401040
12        Start of program headers: 64 (bytes into file)
13        Start of section headers: 14640 (bytes into file)
14        Flags:             0x0
15        Size of this header: 64 (bytes)
16        Size of program headers: 56 (bytes)

```



```
17      Number of program headers:      13
18      Size of section headers:        64 (bytes)
19      Number of section headers:      32
20      Section header string table index: 31
21
```

(b) **infected-ggdb.out:**

Grundlegende Informationen:

```
1      infected-ggdb.out: ELF 64-bit LSB executable, x86-64, version 1 (SYSV),
      dynamically linked, interpreter /lib64/ld-linux-x86-64.so.2, BuildID[sha1
      ]=e6d61ad5610dcddb5ee77ac374dc9ec665b38c8, for GNU/Linux 3.2.0, with
      debug_info, not stripped
2
```

ELF-Header:

```
1      ELF Header:
2      Magic:    7f 45 4c 46 02 01 01 00 00 00 00 00 00 00 00 00
3      Class:                                ELF64
4      Data:                                2's complement, little endian
5      Version:                              1 (current)
6      OS/ABI:                                UNIX - System V
7      ABI Version:                          0
8      Type:                                  EXEC (Executable file)
9      Machine:                               Advanced Micro Devices X86-64
10     Version:                               0x1
11     Entry point address:                   0x401040
12     Start of program headers:              64 (bytes into file)
13     Start of section headers:             15408 (bytes into file)
14     Flags:                                 0x0
15     Size of this header:                   64 (bytes)
16     Size of program headers:               56 (bytes)
17     Number of program headers:             13
18     Size of section headers:               64 (bytes)
19     Number of section headers:             38
20     Section header string table index: 37
21
```

(c) **infected-static.out:** Grundlegende Informationen:

```
1      infected-static.out: ELF 64-bit LSB executable, x86-64, version 1 (GNU/
      Linux), statically linked, BuildID[sha1]=
      f1c38229763e1666d63a59346fc0ed51ccc67cc2, for GNU/Linux 3.2.0, not stripped
      , too many notes (256)
2
```

ELF-Header:

```
1      ELF Header:
2      Magic:    7f 45 4c 46 02 01 01 03 00 00 00 00 00 00 00 00
3      Class:                                ELF64
4      Data:                                2's complement, little endian
5      Version:                              1 (current)
6      OS/ABI:                                UNIX - GNU
7      ABI Version:                          0
8      Type:                                  EXEC (Executable file)
9      Machine:                               Advanced Micro Devices X86-64
10     Version:                               0x1
11     Entry point address:                   0x401670
12     Start of program headers:              64 (bytes into file)
13     Start of section headers:             791888 (bytes into file)
14     Flags:                                 0x0
15     Size of this header:                   64 (bytes)
16     Size of program headers:               56 (bytes)
17     Number of program headers:             10
18     Size of section headers:               64 (bytes)
19     Number of section headers:             34
20     Section header string table index: 33
21
```

(d) **infected-static-ggdb.out**: Grundlegende Informationen:

```
1      infected-static-ggdb.out: ELF 64-bit LSB executable, x86-64, version 1
      (GNU/Linux), statically linked, BuildID[sha1]=7
      c4c65df8e34fb67889fc06f4cc6fdbfb9b668c8, for GNU/Linux 3.2.0, with
      debug_info, not stripped, too many notes (256)
```

2

## ELF-Header:

```
1      ELF Header:
2      Magic:    7f 45 4c 46 02 01 01 03 00 00 00 00 00 00 00 00
3      Class:                                ELF64
4      Data:                                2's complement, little endian
5      Version:                               1 (current)
6      OS/ABI:                                UNIX - GNU
7      ABI Version:                           0
8      Type:                                  EXEC (Executable file)
9      Machine:                               Advanced Micro Devices X86-64
10     Version:                               0x1
11     Entry point address:                   0x401670
12     Start of program headers:              64 (bytes into file)
13     Start of section headers:              792656 (bytes into file)
14     Flags:                                 0x0
15     Size of this header:                   64 (bytes)
16     Size of program headers:               56 (bytes)
17     Number of program headers:              10
18     Size of section headers:               64 (bytes)
19     Number of section headers:              40
20     Section header string table index:     39
21
```

## Analyse Datei "a"

Um einen groben Eindruck der Datei zu bekommen kann zuallererst der **strings**-Befehl verwendet werden.

Der erste Erkennbare string **"!This program cannot be run in DOS mode."** ist typisch für PE-Files (Windows):

```
mendacium@mendacium ~/fh/REV3/UE02/rev3-s2210239021/a % main strings a | head -n 1
!This program cannot be run in DOS mode.
```

Aufgrund dessen wird weiters versucht mit **dumpbin** (für Windows-Executables das passende Tool) und **strings** die Fragen zu beantworten.

## Fragen

### 1. Projektname:

Lässt man sich wieder die Headers mit **dumpbin /HEADERS a** anzeigen, so kann in den "Debug Directories" wieder die Pfad-Angabe für das ".pdb"-File (programm database) herausgelesen werden:

```
Debug Directories
-----
Time Type      Size      RVA Pointer
-----
617A416A cv      66 00016130 5A10 Format: RID5, (CC35618F-81D2-498C-BAE0-9A48F87E226), 4, C:\Users\P22225\Documents\Visual Studio 2013\Projects\Joshua\Debug\Joshua.pdb
617A416A fixup  14 00016078 5A70 Counts: PrevVC= 11,00=0, CCI=27, /GS=27, /dli=1, guard=0
```

Der Pfadname deutet darauf hin, dass das Projekt **"Joshua"** hieß.

### 2. Autor\*in:

Ebenfalls aus dem Output des vorherigen Aufrufes kann auf den/die Autor/in geschlossen werden (User-Verzeichnis):

**"P22225"**

### 3. Zeitstempel:

Der Zeitstempel steht in den "HEADER VALUES" und kann ebenfalls mit dem Befehl **dumpbin /HEADERS a** angezeigt werden:

```
PS C:\Users\jakim\repos\REV3\UE02\rev3-s2210239021\a> dumpbin /HEADERS a
Microsoft (R) COFF/PE Dumper Version 14.37.32824.0
Copyright (C) Microsoft Corporation. All rights reserved.

Dump of file a

PE signature found

File Type: EXECUTABLE IMAGE

FILE HEADER VALUES
      14C machine (x86)
        7 number of sections
    617A416A time date stamp Thu Oct 28 08:21:30 2021
        0 file pointer to symbol table
        0 number of symbols
        E0 size of optional header
    102 characteristics
      Executable
    32 bit word machine
```

**"Thu Oct 28 08:21:30 2021"**

#### 4. relevante Strings:

Welche strings von Relevanz sind liegt natürlich im Auge des Betrachters...

Ganz voran gehen allerdings die zwei links:

YT-Video "WarGames (1/11) Movie CLIP - Asexual Reproduction (1983) HD": <https://www.youtube.com/watch?v=LwDbgE54QYE&list=PLZbXA4lyCtqpG0S2KC1mAAKaGwbup-DQt>

SIB Studiengangs-Homepage:

<https://www.fh-ooe.at/campus-hagenberg/studiengaenge/bachelor/sichere-informationssysteme/>

```
61 Debug
62 Starting
63 https://www.youtube.com/watch?v=LwDbgE54QYE&list=PLZbXA4lyCtqpG0S2KC1mAAKaGwbup-DQt
64 open
65 sib.html
66 https://www.fh-ooe.at/campus-hagenberg/studiengaenge/bachelor/sichere-informationssysteme/
67 Stack around the variable '
68 ' was corrupted
```