

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ
ВЫСШЕГО ОБРАЗОВАНИЯ
«БЕЛГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНОЛОГИЧЕСКИЙ
УНИВЕРСИТЕТ им. В. Г. ШУХОВА»
(БГТУ им. В.Г. Шухова)



ИНСТИТУТ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ И УПРАВЛЯЮЩИХ СИСТЕМ

РГЗ

по дисциплине: Архитектура вычислительных систем
тема: «Создание программы, выводящей собственный стек вызова»

Выполнил: ст. группы ПВ-223
Пахомов Владислав Андреевич

Проверил: Осипов Олег Васильевич

Белгород 2024 г.

Оглавление

- 1 Формулировка задачи**
- 2 Алгоритм работы программы, используемые библиотеки.**
- 3 Исходная программа.**
- 4 Результаты выполнения программы.**
- 5 Заключение**

1 Формулировка задачи

Разработать программное обеспечение, выводящее собственный стек вызова при помощи Assembler.

2 Алгоритм работы программы, используемые библиотеки.

Будем разрабатывать программное обеспечение для операционной системы Windows. Для анализа данных процесса будем использовать библиотеку dbghelp.h, входящую в состав Win32. SymInitialize инициализирует обработчик символов для процесса, после чего получим адрес функции, выполняющий трейсбек стека из kernel32.dll под названием RtlCaptureStackBackTrace, и после чего вызываем её, получая стектрейс - массив адресов инструкций.

После этого будем анализировать массив адресов инструкций. Можно получить информацию о модуле при помощи GetModuleHandleExA. SymGetSymFromAddr позволяет получить информацию об имени функции, SymGetLineFromAddr позволяет получить номер строки в исходном тексте программы.

3 Исходная программа.

hw.asm

```
.686
.model flat, stdcall
option casemap: none

include windows.inc
include user32.inc
include kernel32.inc
include msvcrt.inc
include dbghelp.inc

includelib msvcrt.lib
includelib user32.lib
includelib kernel32.lib
; 32-битную версию dbghelp.lib можно взять из x64dbg/pluginsdk
includelib dbghelp.lib

.data
kernel32dllname db "kernel32.dll", 0
RtlCaptureStackBackTracename db "RtlCaptureStackBackTrace", 0
mainprintlevel db "----- Level %d, address 0x%p -----", 13, 10, 0
printtraceheadingprint db "Defining function name and module at address 0x%p", 13, 10, 0
newline db 13, 10, 0
callers db 1024 dup(?)
printtracemodule db "Module begin address: 0x%p", 13, 10
db "Module name: %s", 13, 10, 0
```

```

symgetsymfromaddr64failedmsg db "SymGetLineFromAddr failed. Exit code: %d", 13, 10, 0
symgetsymfromaddr64subprogram db "Subprogram name: %s, Instruction offset from function beginning: %u", 13, 10, 0
symgetlinefromaddrloc db "Subprogram '%s' is located at '%s'", 13, 10
db "Address: 0x%p", 13, 10
db "Instruction offset from beginning of line: %d", 13, 10
db "Line: %d", 13, 10, 0

```

.code

```

print_trace proc instructionAddress: DWORD
    invoke crt_printf, offset printtraceheadingprint, instructionAddress

```

```

    sub esp, 4

```

```

    mov eax, GET_MODULE_HANDLE_EX_FLAG_FROM_ADDRESS
    xor eax, GET_MODULE_HANDLE_EX_FLAG_UNCHANGED_REFCOUNT

```

```

    invoke GetModuleHandleExA, eax, instructionAddress, esp

```

```

    cmp eax, 0

```

```

    je print_trace_modulehandlenotexists

```

```

        ; char module_name[1024];

```

```

        sub esp, 261

```

```

        ; flags

```

```

        mov eax, GET_MODULE_HANDLE_EX_FLAG_FROM_ADDRESS
        xor eax, GET_MODULE_HANDLE_EX_FLAG_UNCHANGED_REFCOUNT

```

```

        mov edx, esp
        add edx, 261
        mov edx, [edx]

```

```

        mov ecx, esp

```

```

        ; GetModuleFileNameA(hModule, module_name, MAX_PATH);

```

```

        invoke GetModuleFileNameA, edx, ecx, 260

```

```

        mov ecx, esp
        add ecx, 261
        mov ecx, [ecx]

```

```

        invoke crt_printf, offset printtracemodule, ecx, esp

```

```

        add esp, 261

```

```

        ; DWORD disp = 0;

```

```

        sub esp, 4

```

```

        mov dword ptr [esp], 0

```

```

        ;struct

```

```

        ;{

```

```

        ;     IMAGEHLP_SYMBOL64 symbolInfo = { };

```

```

        ;     char name_buffer[1024];

```

```

; } SYMBOL_DATA;
sub esp, 1048

; SYMBOL_DATA.symbolInfo.SizeOfStruct = sizeof(IMAGEHLP_SYMBOL64)
mov dword ptr [esp], 24
; SYMBOL_DATA.symbolInfo.MaxNameLength = 1024
mov dword ptr [esp + 16], 1024

; eax = CurrentProcess()
invoke GetCurrentProcess
; ecx = &disp
mov ecx, esp
add ecx, 1048

invoke SymGetSymFromAddr, eax, instructionAddress, ecx, esp

cmp eax, 0
je print_trace_symgetsymfromaddr64failed
    ; ecx = symbolInfo.name
    mov ecx, esp
    add ecx, 20

    ; edx = disp
    mov edx, esp
    add edx, 1048
    mov edx, [edx]

    invoke crt_printf, offset symgetsymfromaddr64subprogram, ecx, edx

    sub esp, 20
    mov dword ptr [esp], 20
    sub esp, 4

    ; ecx - ссылка на displacement
    mov ecx, esp

    ; edx - ссылка на line
    mov edx, esp
    add edx, 4

    invoke GetCurrentProcess
    invoke SymGetLineFromAddr, eax, instructionAddress, ecx, edx

    cmp eax, 0
    je print_trace_symgetlinefromaddr64failed
        ; eax = адрес SYMBOL_DATA.symbolInfo.Name
        mov eax, esp
        add eax, 24
        add eax, 20

        ; ecx = ссылка на line.FileName
        mov ecx, esp
        add ecx, 4
        add ecx, 12

```

```

        ; edx = SYMBOL_DATA.symbolInfo.Address
        mov edx, esp
        add edx, 24
        add edx, 20
        add edx, 4

        ; ebx = displacement
        mov ebx, esp
        mov ebx, [ebx]

        ; esi = line.LineNumber
        mov esi, esp
        add esi, 4
        add esi, 8
        mov esi, [esi]

        invoke crt_printf, eax, ecx, edx, ebx, esi
        jmp print_trace_symgetlinefromaddr64failedend
print_trace_symgetlinefromaddr64failed:
        invoke GetLastError
        invoke crt_printf, offset symgetsymfromaddr64failedmsg, eax
print_trace_symgetlinefromaddr64failedend:

        add esp, 24

        jmp print_trace_symgetsymfromaddr64failedend

print_trace_symgetsymfromaddr64failed:
        invoke GetLastError
        invoke crt_printf, offset symgetsymfromaddr64failedmsg, eax
print_trace_symgetsymfromaddr64failedend:

        add esp, 1052

print_trace_modulehandlenotexists:
        add esp, 4
        ret
print_trace endp

start:
        ; SymInitialize(GetCurrentProcess(), NULL, TRUE);
        invoke GetCurrentProcess
        invoke SymInitialize, eax, NULL, TRUE

        ; CaptureStackBackTraceType pCaptureStackBackTraceType =
        ↪ (CaptureStackBackTraceType)(GetProcAddress(LoadLibraryA("kernel32.dll"), "RtlCaptureStackBackTrace"));
        invoke LoadLibraryA, offset kernel32dllname
        invoke GetProcAddress, eax, offset RtlCaptureStackBackTracename

        cmp eax, 0
        jne pCaptureStackBackTraceTypeNotNull

```

```
;
push 1
call ExitProcess
```

pCaptureStackTraceTypeNotNull:

```
; pCaptureStackTraceType(0, 1024, callers, NULL);
push NULL
push offset callers
push 1024
push 0
call eax

; Сохраняем в стек count
push eax
; i = 0
mov ecx, 0
```

callersloop:

```
; i >= count?
cmp ecx, [esp]
; Выход из цикла
jge endcallersloop

mov ebp, dword ptr callers[ecx * 4]

; printf("----- Level %d, address 0x%p ----- \n", i + 1, callers[i]);
push ecx
invoke crt_printf, offset mainprintlevel, ecx, ebp
pop ecx

mov ebp, dword ptr callers[ecx * 4]
push ecx
invoke print_trace, ebp
pop ecx

push ecx
invoke crt_printf, offset newline
pop ecx

inc ecx
jmp callersloop
```

endcallersloop:

```
call crt__getch ; Задержка ввода, getch()
; Вызов функции ExitProcess(0)
push 0 ; Поместить аргумент функции в стек
call ExitProcess ; Выход из программы
```

end start

4 Результаты выполнения программы.

```
C:\Users\vladi\workspace\Assembler\computing_systems_architecture\hw>pause
Для продолжения нажмите любую клавишу . . .

C:\Users\vladi\workspace\Assembler\computing_systems_architecture\hw>hw.exe
----- Level 0, address 0x004011B3 -----
Defining function name and module at address 0x004011B3
Module begin address: 0x00400000
Module name: C:\Users\vladi\workspace\Assembler\computing_systems_architecture\hw\hw.exe
SymGetLineFromAddr failed. Exit code: 487

----- Level 1, address 0x77B9CEBB -----
Defining function name and module at address 0x77B9CEBB
Module begin address: 0x77B30000
Module name: C:\WINDOWS\SYSTEM32\ntdll.dll
Subprogram name: RtlInitializeExceptionChain, Instruction offset from function beginning: 107
SymGetLineFromAddr failed. Exit code: 487

----- Level 2, address 0x77B9CE41 -----
Defining function name and module at address 0x77B9CE41
Module begin address: 0x77B30000
Module name: C:\WINDOWS\SYSTEM32\ntdll.dll
Subprogram name: RtlGetAppContainerNamedObjectPath, Instruction offset from function beginning: 561
SymGetLineFromAddr failed. Exit code: 487
```

5 Заключение

Разработали на Assembler при помощи библиотек Windows программу, позволяющую отследить стектрейс вызова текущей программы.