Дубовский Алексей Владимирович 151004

Lab3

**1  
Runtime**

#define a 10

#define b 3

typedef void(\_\_cdecl\* PROC\_MATH)(int , int);

typedef void(\_\_cdecl\* PROC\_RES)();

int main() {

HINSTANCE hLib = LoadLibrary(TEXT("MathDll.dll"));

PROC\_MATH ProcMath1 = (PROC\_MATH)GetProcAddress(hLib, "Addition");

if (ProcMath1 != NULL) {

ProcMath1(a, b);

}

PROC\_MATH ProcMath2 = (PROC\_MATH)GetProcAddress(hLib, "Subtraction");

if (ProcMath2 != NULL) {

ProcMath2(a, b);

}

PROC\_RES ProcRes = (PROC\_RES)GetProcAddress(hLib, "PrintResult");

if (ProcRes != NULL) {

ProcRes();

}

FreeLibrary(hLib);

int c = 0;

scanf\_s("%d", c);

return 0;

}

MathDll

int addition = 0;

\_\_declspec(dllexport) void Addition(int a, int b) {

addition = a + b;

}

int subtraction = 0;

\_\_declspec(dllexport) void Subtraction(int a, int b) {

subtraction = a - b;

}

\_\_declspec(dllexport) void PrintResult() {

if (0 != addition) {

printf("addition = %d \n", addition);

}

if (0 != subtraction) {

printf("subtraction = %d \n", subtraction);

}

}

**2**

**Loadtime**

#define a 10

#define b 3

void Addition(int , int );

void Subtraction(int, int);

void PrintResult();

int main() {

Addition(a, b);

Subtraction(a, b);

PrintResult();

int c = 0;

scanf\_s("%d", c);

return 0;

}

MathLib

int addition = 0;

\_\_declspec(dllexport) void Addition(int a, int b) {

addition = a + b;

}

int subtraction = 0;

\_\_declspec(dllexport) void Subtraction(int a, int b) {

subtraction = a - b;

}

\_\_declspec(dllexport) void PrintResult() {

if (0 != addition) {

printf("addition = %d \n", addition);

}

if (0 != subtraction) {

printf("subtraction = %d \n", subtraction);

}

}

**3**

**ChangeString**

#define SIZE\_BUFF 1'000'000

char buff[SIZE\_BUFF];

\_\_declspec(dllexport) void ReplaceString(const char\* oldString, const char\* newString) {

HANDLE hProc = GetCurrentProcess();

int oldStringLen = strlen(oldString);

SYSTEM\_INFO siSysInfo;

GetSystemInfo(&siSysInfo);

PSIZE\_T minAddr = oldString - SIZE\_BUFF\*1 - 90;

PSIZE\_T maxAddr = siSysInfo.lpMaximumApplicationAddress;

char\* p = minAddr;

memset(buff, 0, sizeof(SIZE\_BUFF));

int j = 0;

while(p< maxAddr) {

ReadProcessMemory(hProc, p, &buff, SIZE\_BUFF, NULL);

for (int i = 0; i < SIZE\_BUFF; i++) {

if (buff[i] == oldString[j]) {

j++;

if (j == oldStringLen) {

WriteProcessMemory(hProc, p + i - j + 1, newString, oldStringLen, NULL);

CloseHandle(hProc);

return;

}

}

else {

j = 0;

}

}

p += SIZE\_BUFF;

}

CloseHandle(hProc);

}

Starter

typedef void(\_\_cdecl\* PROC1)(char\*,char\*);

int main() {

char str[] = "Old text";

printf(str);

printf("\n");

HINSTANCE hLib = LoadLibrary(TEXT("ChangeString.dll"));

PROC1 Proc = (PROC1)GetProcAddress(hLib, "ReplaceString");

if (Proc != NULL) {

Proc(str, "New text");

}

FreeLibrary(hLib);

printf(str);

scanf\_s(str);

return 0;

}

**4**

**IngectDll**

BOOL APIENTRY DllMain( HMODULE hModule,

DWORD ul\_reason\_for\_call,

LPVOID lpReserved)

{

FILE\* file;

switch (ul\_reason\_for\_call)

{

case DLL\_PROCESS\_ATTACH:

fopen\_s(&file,"D:\\example.txt", "w");

DWORD pid = GetCurrentProcessId();

HANDLE h = OpenProcess(PROCESS\_ALL\_ACCESS, 0, pid);

fprintf(file, "Process id = %d, Process Handle = %d", pid,h);

fclose(file);

case DLL\_THREAD\_ATTACH:

case DLL\_THREAD\_DETACH:

case DLL\_PROCESS\_DETACH:

break;

}

return TRUE;

}

Ingect

int main() {

const char\* dllPath = "D:\\InjectDll.dll";

DWORD pid = 20092;

HANDLE hProcess = OpenProcess(PROCESS\_ALL\_ACCESS, FALSE, pid);

if (hProcess)

{

LPVOID remoteString = VirtualAllocEx(hProcess, 0, MAX\_PATH, MEM\_COMMIT, PAGE\_READWRITE);

if (remoteString)

{

WriteProcessMemory(hProcess, remoteString, dllPath, strlen(dllPath) + 1, 0);

HANDLE hThread = CreateRemoteThread(hProcess, 0, 0, (LPTHREAD\_START\_ROUTINE)LoadLibraryA, remoteString, 0, 0);

if (hThread)

{

WaitForSingleObject(hThread, INFINITE);

CloseHandle(hThread);

}

VirtualFreeEx(hProcess, remoteString, 0, MEM\_RELEASE);

}

CloseHandle(hProcess);

}

return 0;

}