PE Code Injection

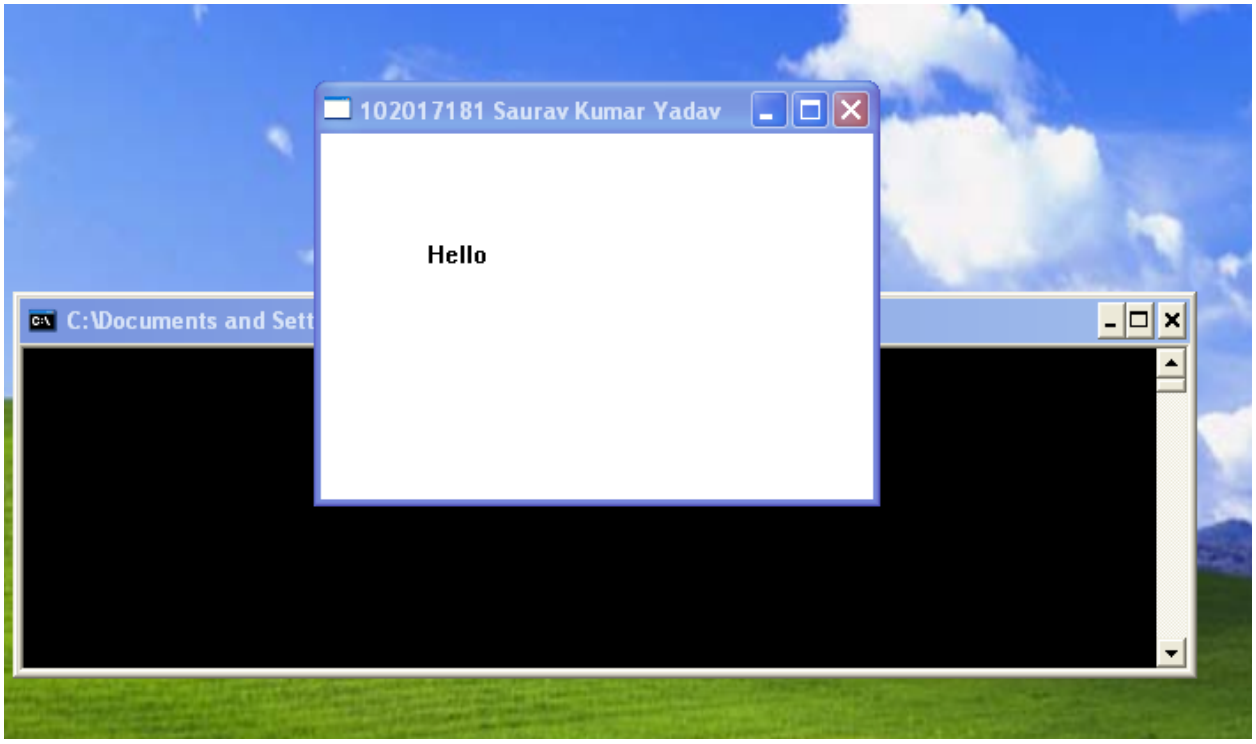
Implement your own Windows code to display simple Hello Text in the body of the Application. Now hijack control of this program and inject MessageBox code into it, Caption should show your Name_RollNumber and text in the message box should be "You have been Hacked". After the initial display of this MessageBox, retrieve back the original entry point. Create an answer document listing all the steps with appropriate screen shots.

A simple hello world windows program was written using codeblocks IDE in win xp.

```
#include <windows.h>

LRESULT CALLBACK WndProc(HWND, UINT, WPARAM, LPARAM);

static char gszClassName[]  = "102017181 Saurav Kumar Yadav";
static HINSTANCE ghInstance = NULL;

int WINAPI WinMain(HINSTANCE hInstance, HINSTANCE hPrevInstance, LPSTR lpCmdLine, int nCmdS
        WNDCLASSEX WndClass;
        HWND hwnd;
        MSG Msg;

        ghInstance = hInstance;

        WndClass.cbSize        = sizeof(WNDCLASSEX);
        WndClass.style         = NULL;
        WndClass.lpfnWndProc   = WndProc;
        WndClass.cbClsExtra    = 0;
        WndClass.cbWndExtra    = 0;
        WndClass.hInstance     = ghInstance;
        WndClass.hIcon         = LoadIcon(NULL, IDI_APPLICATION);
        WndClass.hCursor       = LoadCursor(NULL, IDC_ARROW);
        WndClass.hbrBackground = (HBRUSH)(COLOR_WINDOW+1);
        WndClass.lpszMenuName  = NULL;
        WndClass.lpszClassName = gszClassName;
        WndClass.hIconSm       = LoadIcon(NULL, IDI_APPLICATION);

        if(!RegisterClassEx(&WndClass)) {
                MessageBox(0, "Error Registering Window!", "Error!", MB_ICONSTOP | MB_OK);
                return 0;
        }

        hwnd = CreateWindowEx(
                WS_EX_STATICEDGE,
                gszClassName,
                "102017181 Saurav Kumar Yadav",
                WS_OVERLAPPEDWINDOW,
                CW_USEDEFAULT, CW_USEDEFAULT,
                320, 240,
                NULL, NULL,
                ghInstance,
                NULL);
```

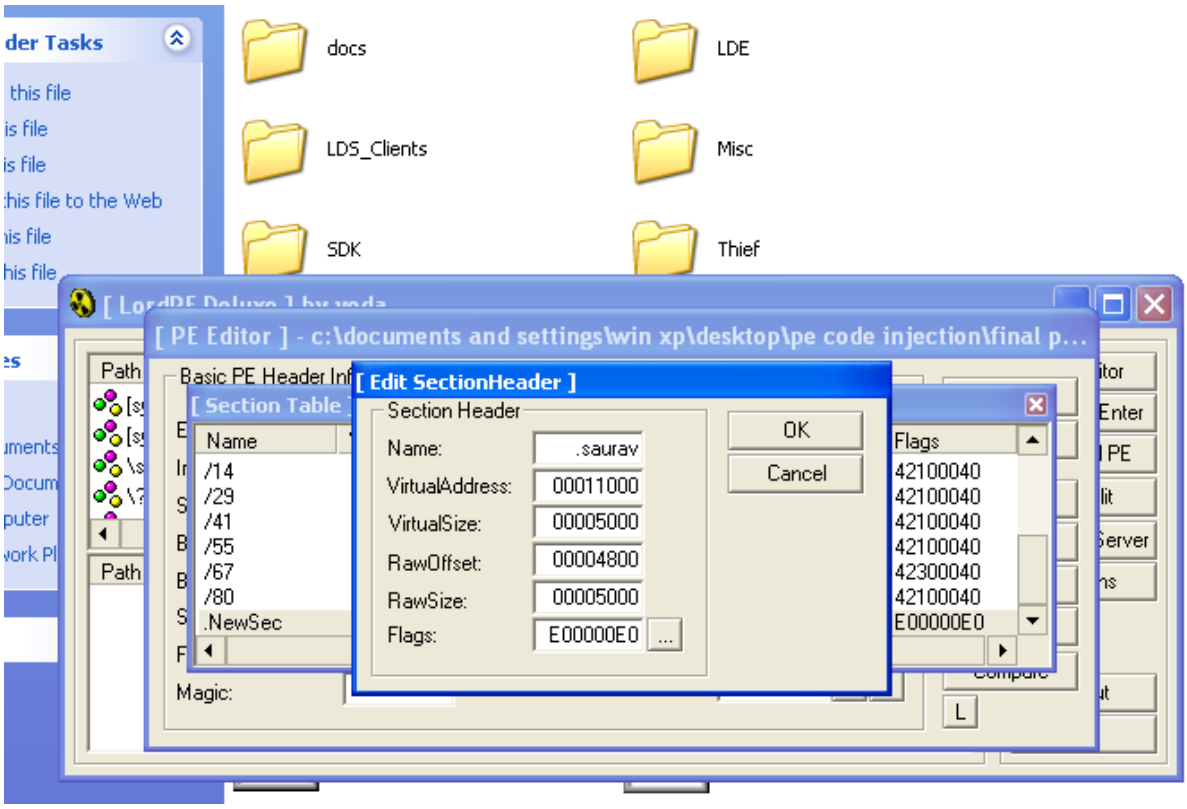
```
40                  NULL, NULL,
41                  ghInstance,
42                  NULL);
43
44          if(hwnd == NULL) {
45                  MessageBox(0, "Window Creation Failed!", "Error!", MB_ICONSTOP | MB_OK);
46                  return 0;
47          }
48
49          ShowWindow(hwnd, nCmdShow);
50          UpdateWindow(hwnd);
51
52          while(GetMessage(&Msg, NULL, 0, 0)) {
53                  TranslateMessage(&Msg);
54                  DispatchMessage(&Msg);
55          }
56          return Msg.wParam;
57  }
58
59  LRESULT CALLBACK WndProc(HWND hwnd, UINT Message, WPARAM wParam, LPARAM lParam) {
60          HDC hdc;
61          PAINTSTRUCT ps;
62          LPSTR szMessage = "Hello!";
63
64          switch(Message) {
65                  case WM_PAINT:
66                          hdc = BeginPaint(hwnd, &ps);
67                          TextOut(hdc, 70, 50, szMessage, strlen(szMessage));
68                          EndPaint(hwnd, &ps);
69                          break;
70                  case WM_CLOSE:
71                          DestroyWindow(hwnd);
72                          break;
73                  case WM_DESTROY:
74                          PostQuitMessage(0);
75                          break;
76                  default:
77                          return DefWindowProc(hwnd, Message, wParam, lParam);
78          }
79          return 0;
80  }
81
```

The app displays following window

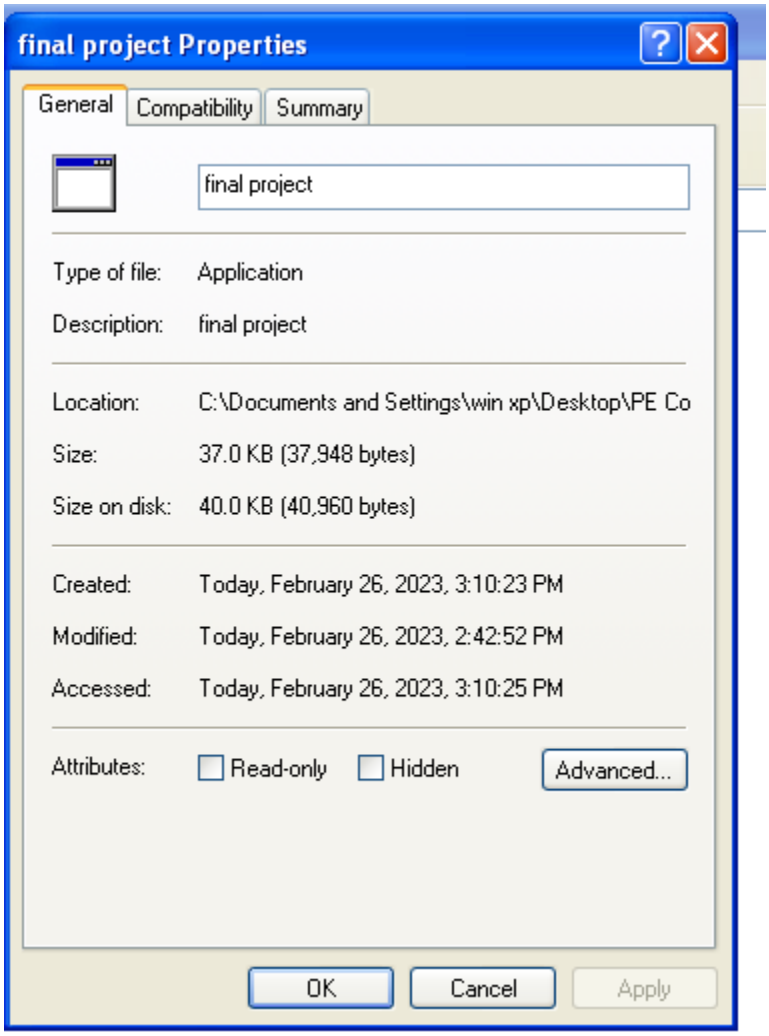


Now we implement PE code Injection .The steps are as follows:

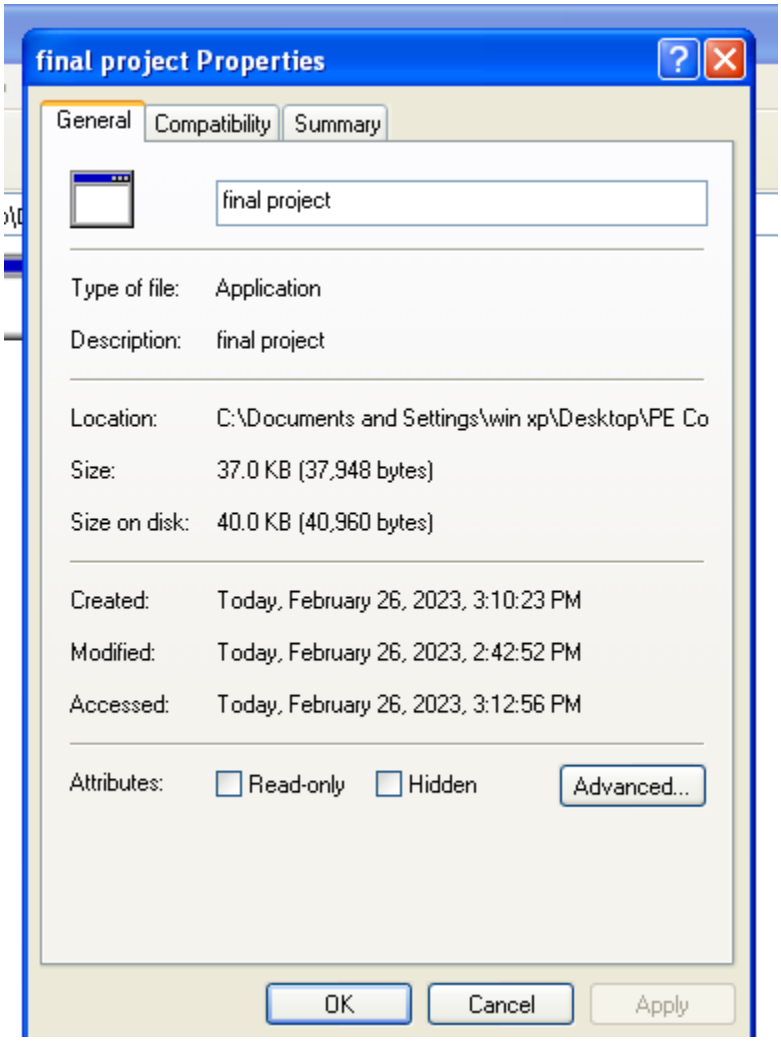
Using LordPE a new section of 5000 bytes is added to the above program.



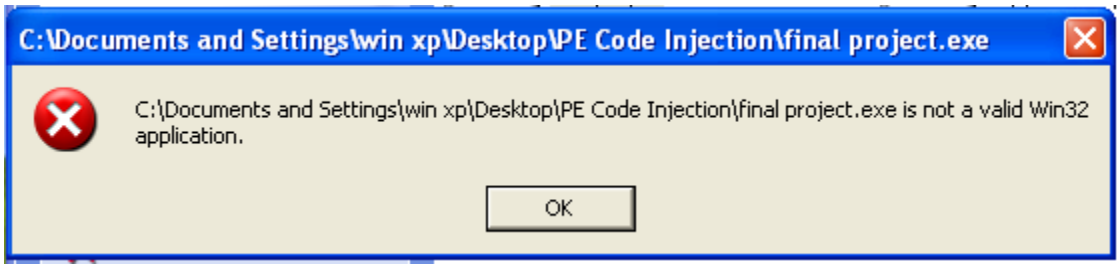
Size of app before adding a new section:



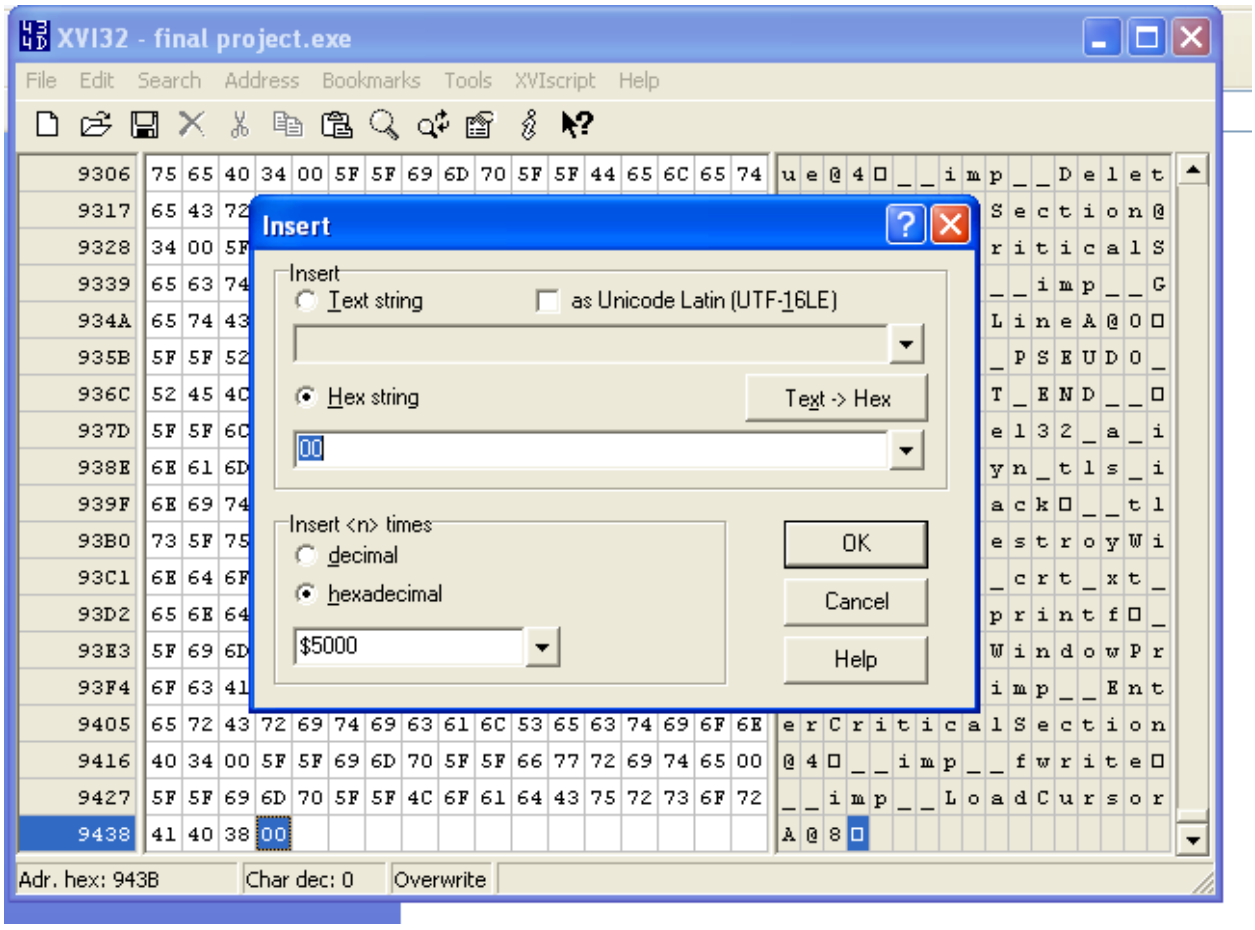
Size of the application after adding 5000 bytes:



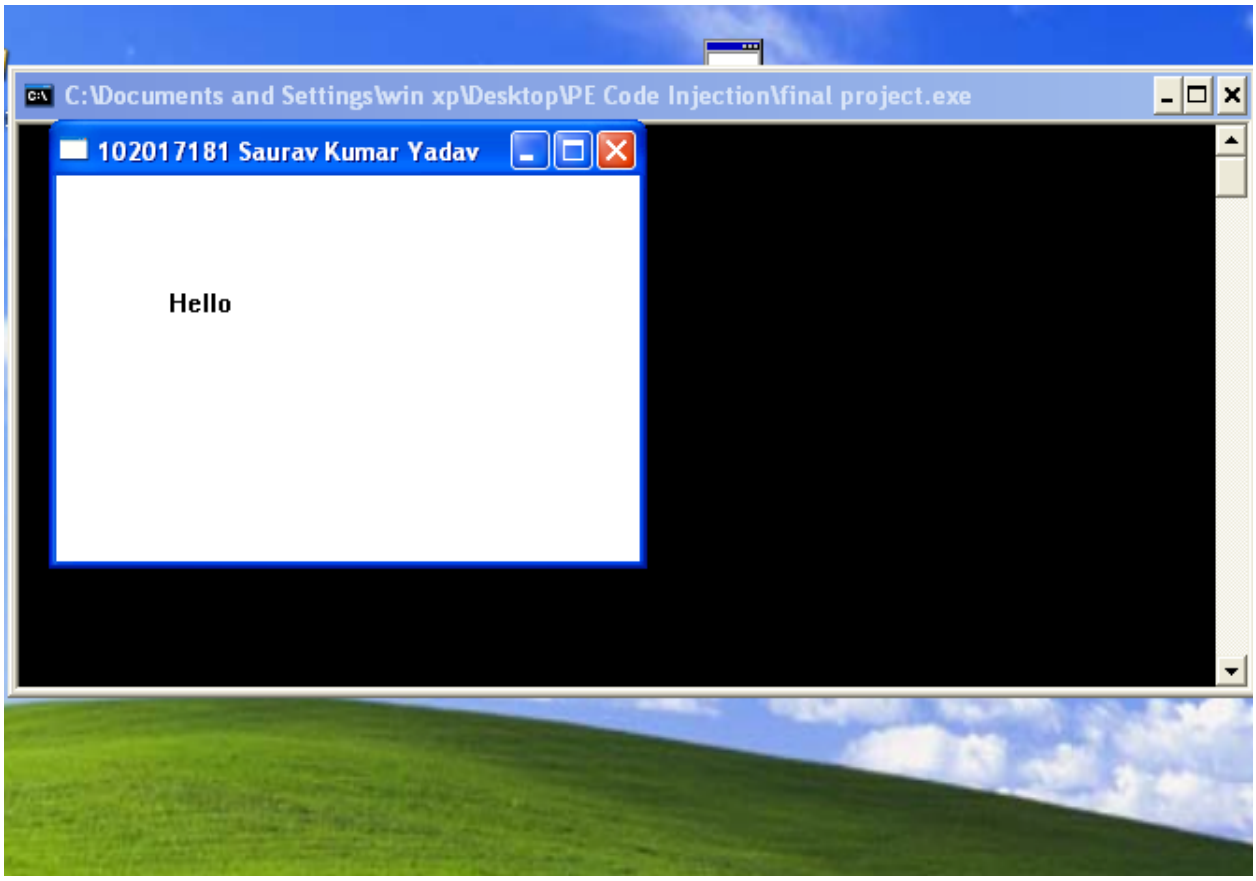
Since the size of application doesnot match the size of the header,Windows displays the following error message while trying to open the program.



XVI32 is used to add 5000 bytes of zeros to the program so that the total size of header match the total size of the program.

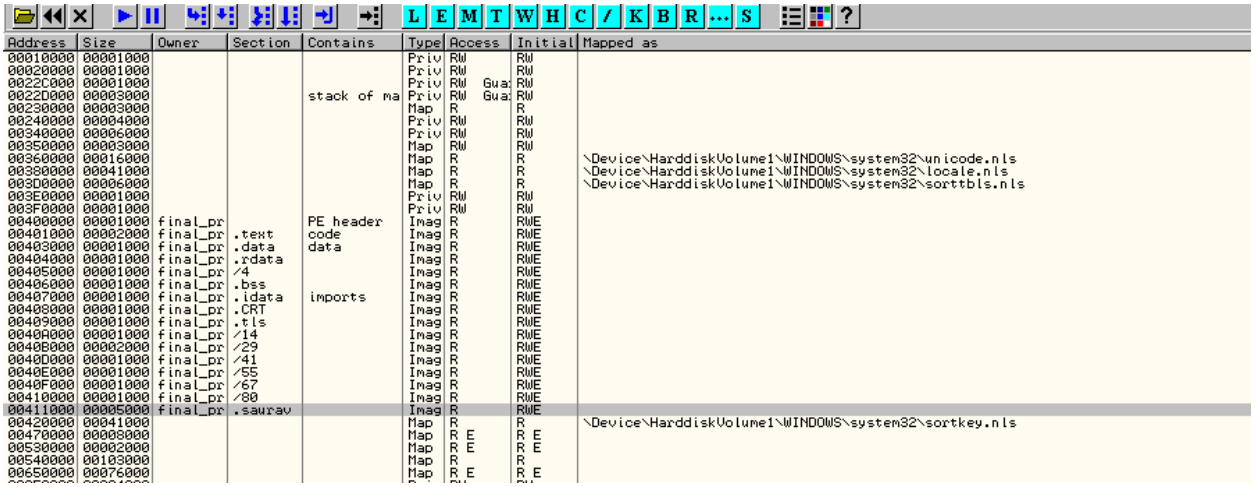


The program is then saved. Now the program works as before.



Now using ollydbg PE Code injection is done.

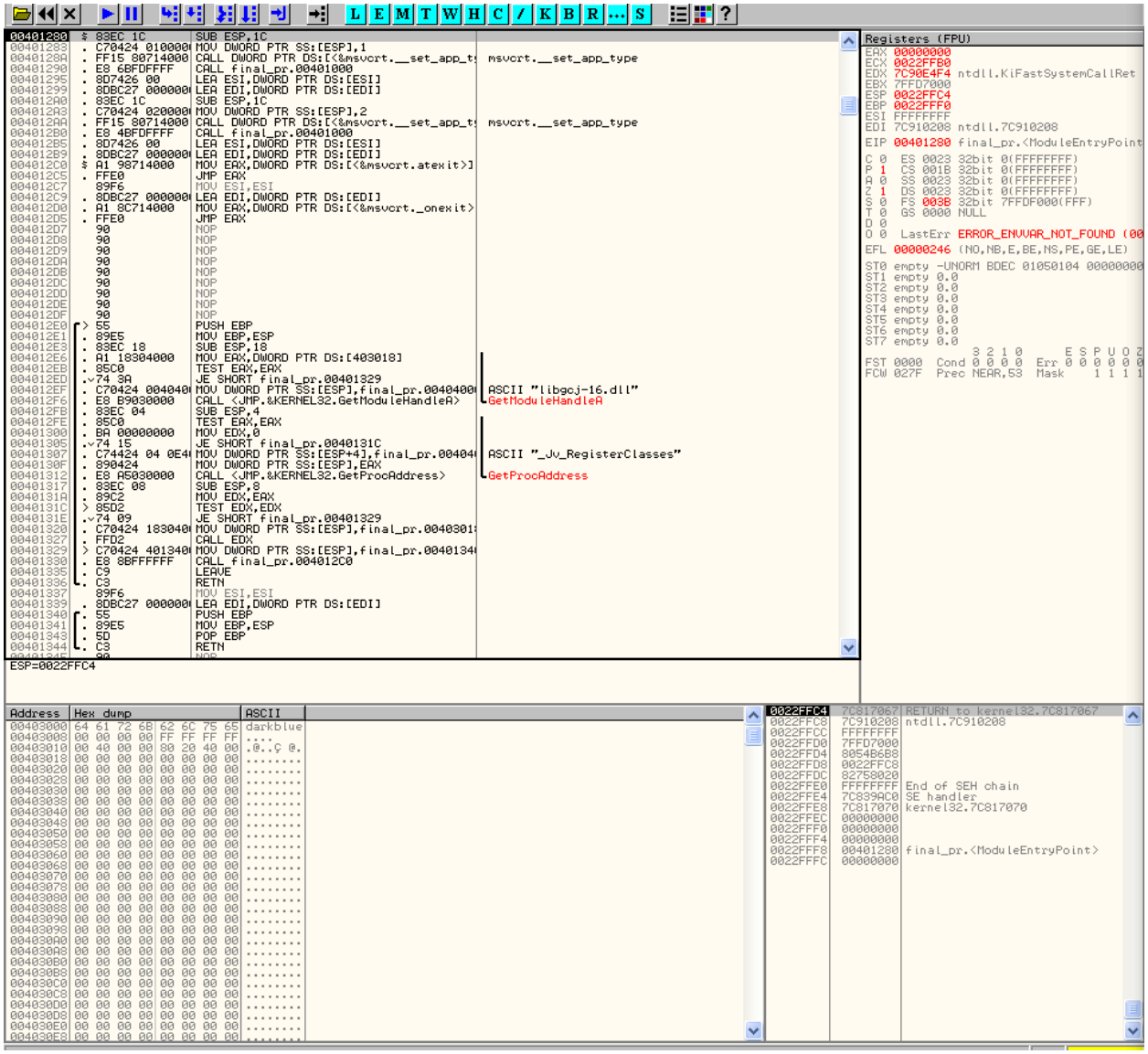
From the memory map, The address of the header which we added is found.



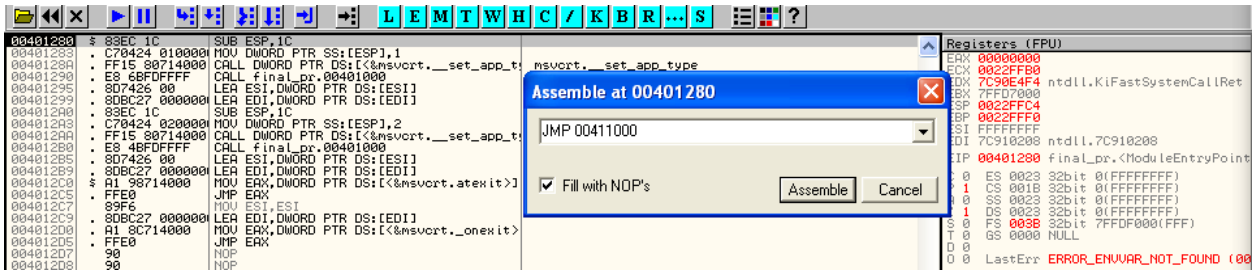
Address	Size	Owner	Section	Contains	Type	Access	Initial	Mapped as
00010000	00001000				Priv	RW	RW	
00020000	00001000				Priv	RW	RW	
0022C000	00001000				Priv	RW Gua	RW	
0022D000	00003000			stack of ma	Priv	RW Gua	RW	
00230000	00003000				Map	R	R	
00240000	00004000				Priv	RW	RW	
00340000	00006000				Priv	RW	RW	
00350000	00003000				Map	RW	RW	
00360000	00016000				Map	R	R	\Device\HarddiskVolume1\WINDOWS\system32\unicode.nls
00380000	00041000				Map	R	R	\Device\HarddiskVolume1\WINDOWS\system32\locale.nls
003D0000	00006000				Map	R	R	\Device\HarddiskVolume1\WINDOWS\system32\sorttbls.nls
003E0000	00001000				Priv	RW	RW	
003F0000	00001000				Priv	RW	RW	
00400000	00001000	final_pr		PE header	Imag	R	RWE	
00401000	00002000	final_pr	.text	code	Imag	R	RWE	
00403000	00001000	final_pr	.data	data	Imag	R	RWE	
00404000	00001000	final_pr	.rdata		Imag	R	RWE	
00405000	00001000	final_pr	/4		Imag	R	RWE	
00406000	00001000	final_pr	.bss		Imag	R	RWE	
00407000	00001000	final_pr	.idata	imports	Imag	R	RWE	
00408000	00001000	final_pr	.CRT		Imag	R	RWE	
00409000	00001000	final_pr	.tls		Imag	R	RWE	
0040A000	00001000	final_pr	/14		Imag	R	RWE	
0040B000	00002000	final_pr	/29		Imag	R	RWE	
0040D000	00001000	final_pr	/41		Imag	R	RWE	
0040E000	00001000	final_pr	/55		Imag	R	RWE	
0040F000	00001000	final_pr	/67		Imag	R	RWE	
00410000	00001000	final_pr	/80		Imag	R	RWE	
00411000	00005000	final_pr	.saurav		Imag	R	RWE	
00420000	00041000				Map	R	R	\Device\HarddiskVolume1\WINDOWS\system32\sortkey.nls
00470000	00008000				Map	R E	R E	
00530000	00002000				Map	R E	R E	
00540000	00103000				Map	R	R	
00650000	00076000				Map	R E	R E	

The address of new header is 0X00411000.This address is where the MessageBoxA() code will be injected to display message box.

The Program has the following instructions in main module:

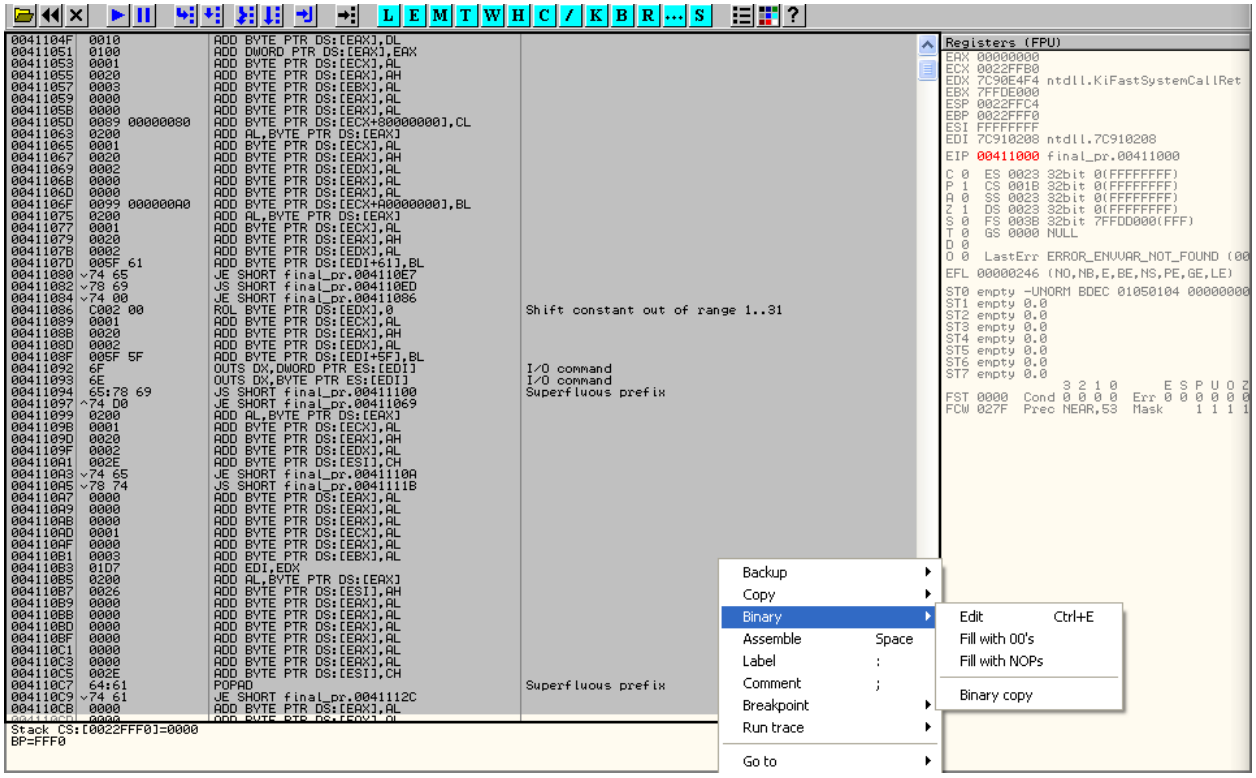


From the main thread, the first few instructions are copied for later reference.At the entry point of the program the instruction is changed form “SUB ESP,1C” to “JUMP 00411000” due to which there is change in the flow of the program, and the program jump to 0X00411000 and instructions from address starts getting executed.



The changes are then copied to the executables and saved to the program.

From the memory address 0X00411000 some portion is selected and filled with 00 using Ollydbg. This is where we will be injecting new code.



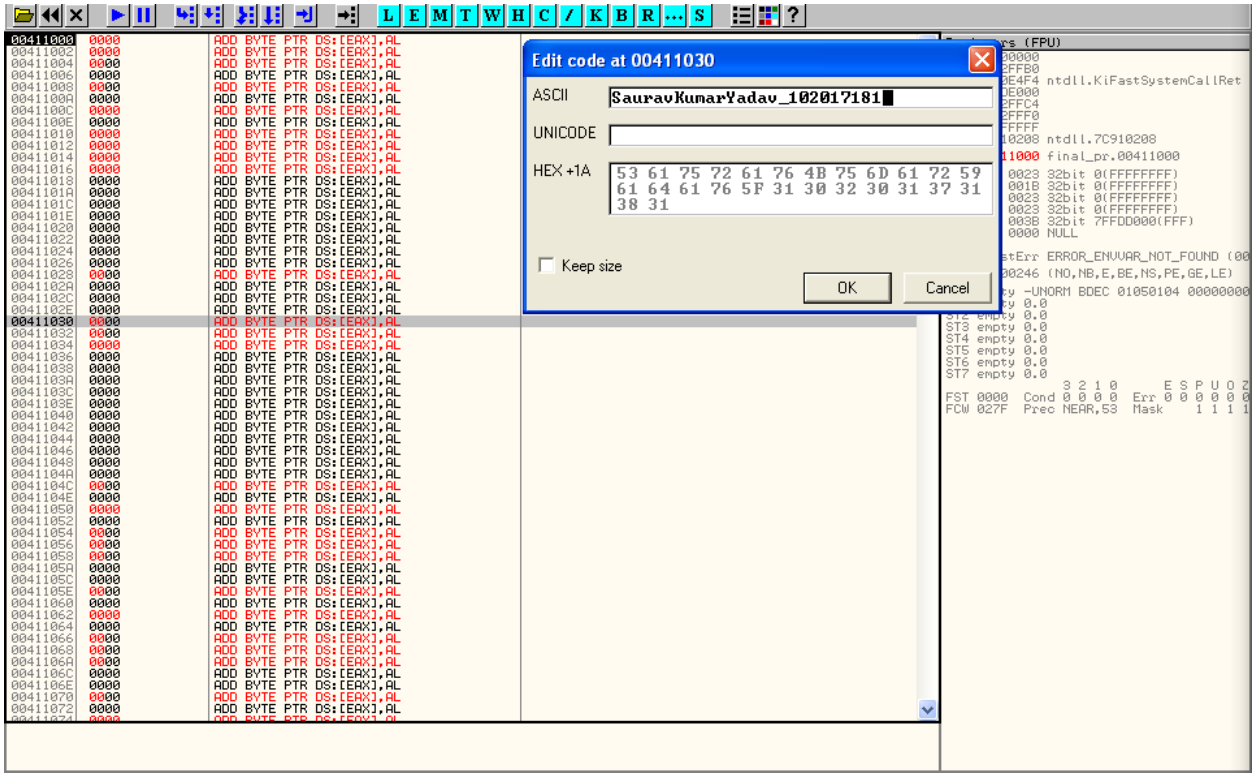
We will be injecting MessageBoxA() into the app. The syntax of message box is as follow

int MessageBoxA(
[in, optional] HWND hWnd,
[in, optional] LPCSTR lpText,
[in, optional] LPCSTR lpCaption,
[in] UINT uType
);

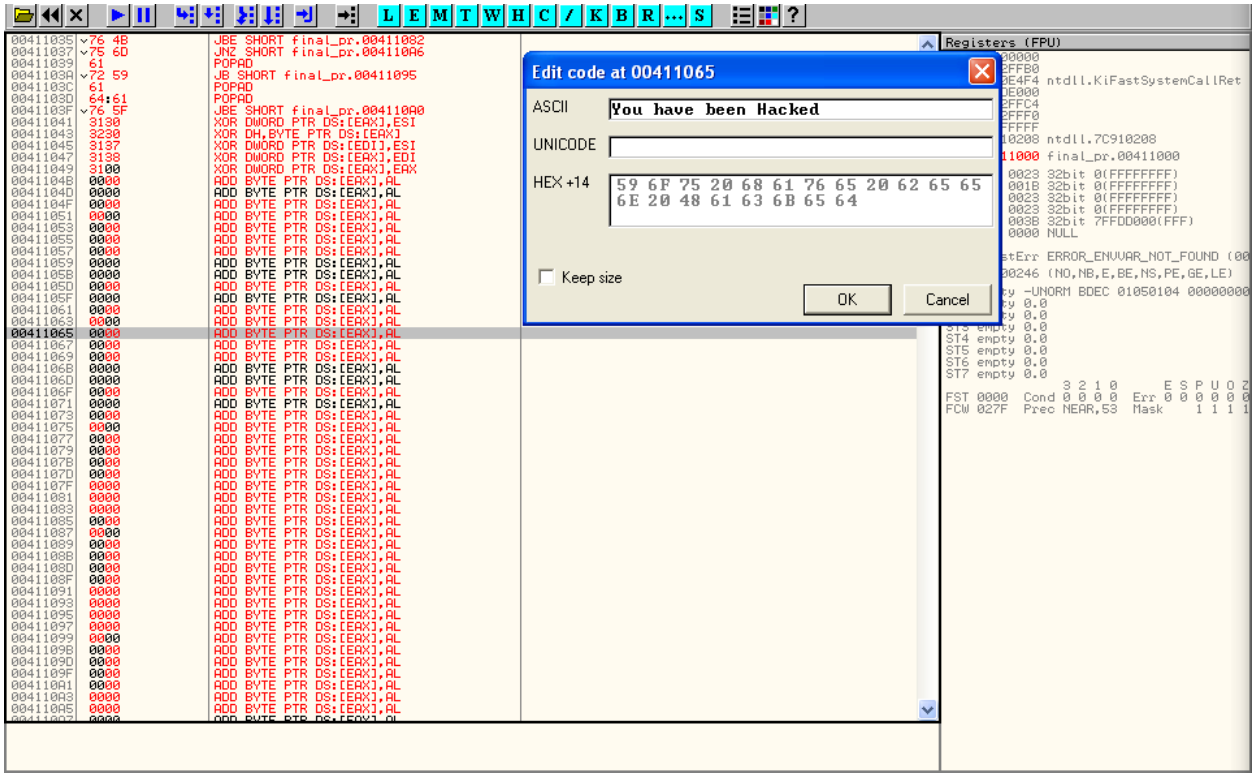
Where in our program hWnd will be ‘0’ signifying no owner for handle to owner window value, lpText will be “You have been Hacked”, lpCation will be “SauravKumarYadav_102017181”, and uType value will be ‘0’ for one push button OK.

Two strings are added into the memory , First string is SauravKumarYadav_102017181. Which will be pointed by lpCaption and second string is “You have been Hacked” which will be pointed by lpText.

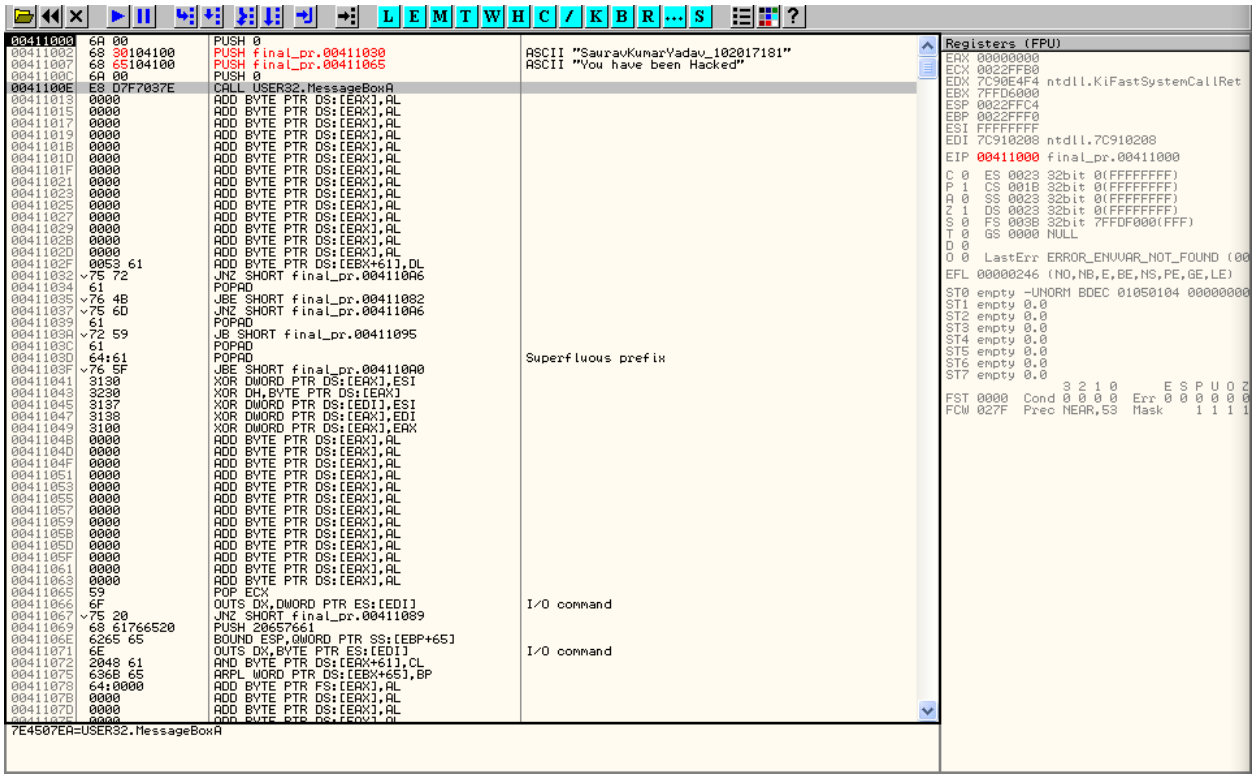
At 0x00411030 ,“SauravKumarYadav_102017181” is added.



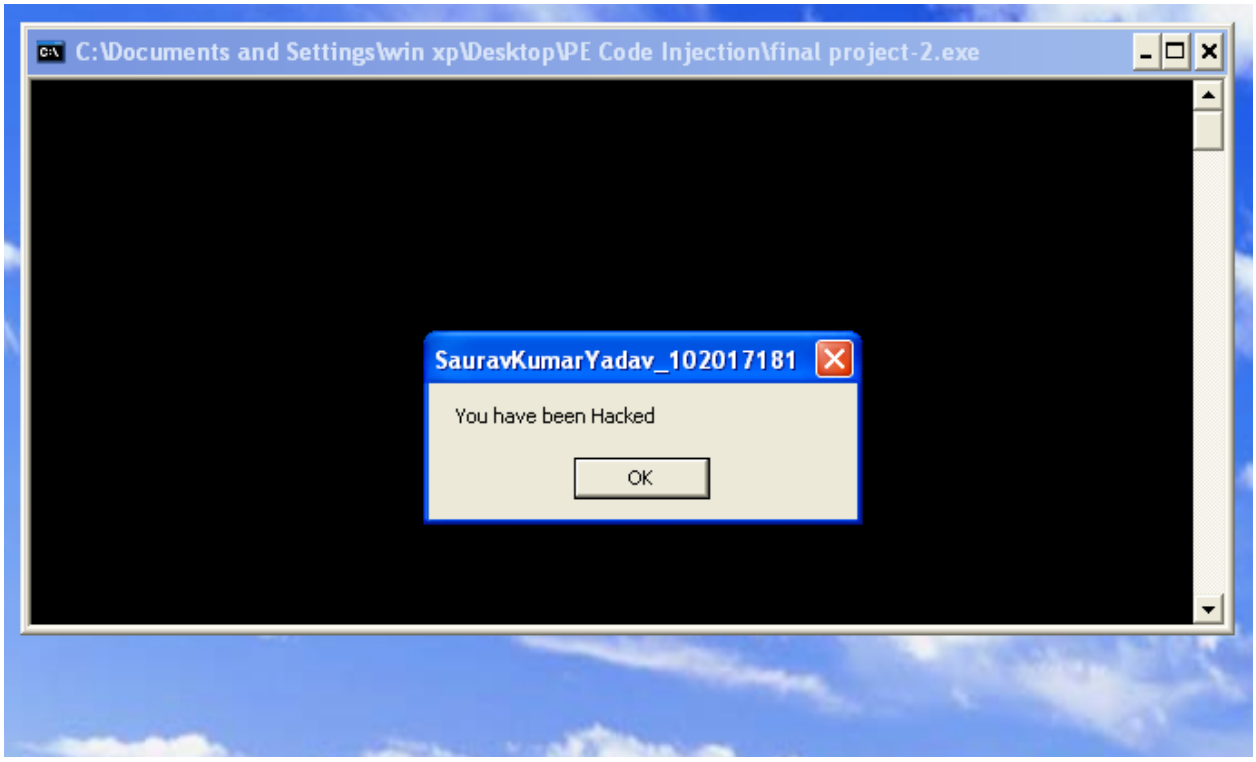
At 0x00411065 ,“You have been Hacked” is added.



At the 00411000 the parameters for MessageBoxA() are added and a call is made to MessageBoxA.

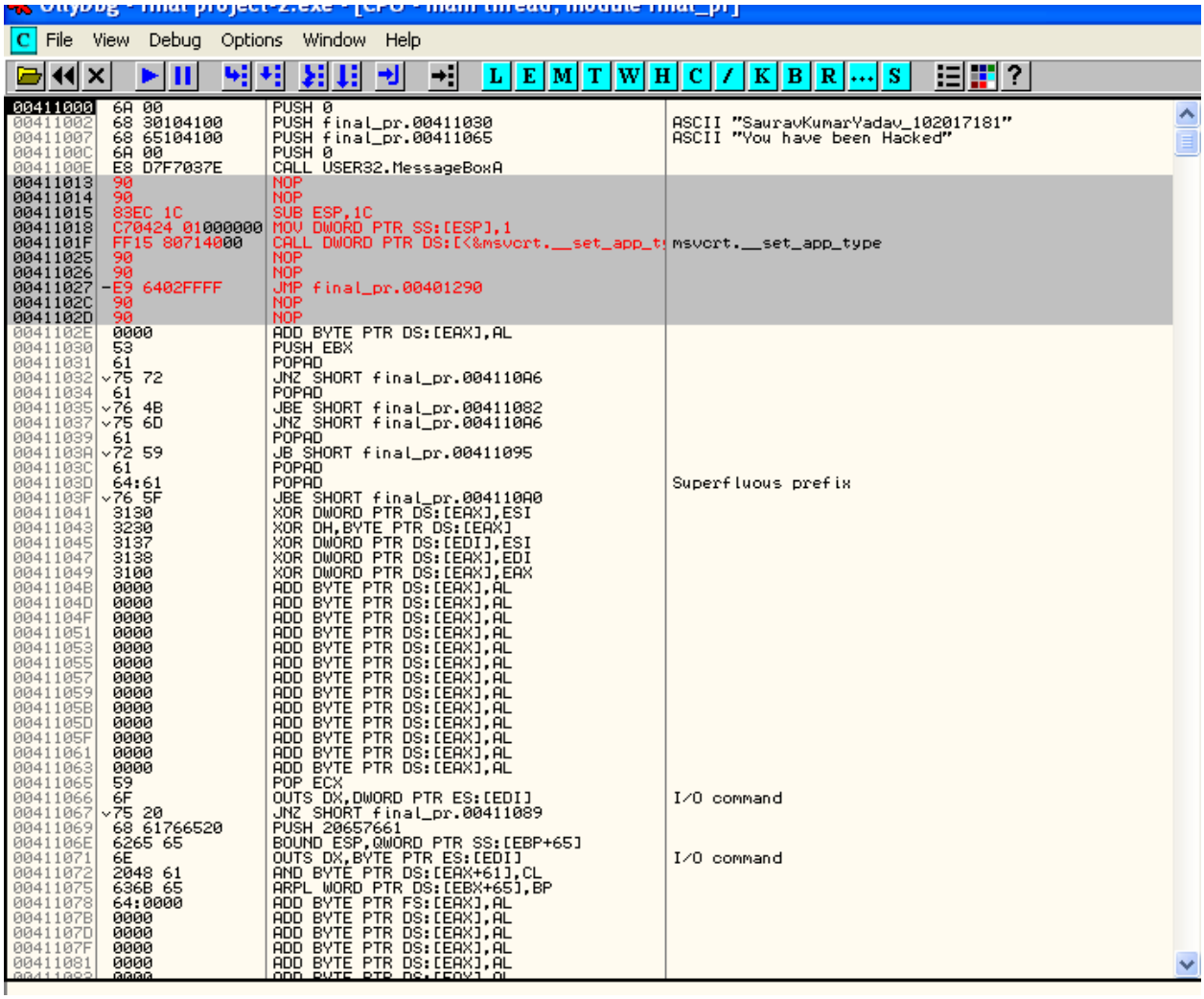


The changes are then copied to executable and then saved to the file.
On running the newly created app after injecting the code. The following window is displayed.

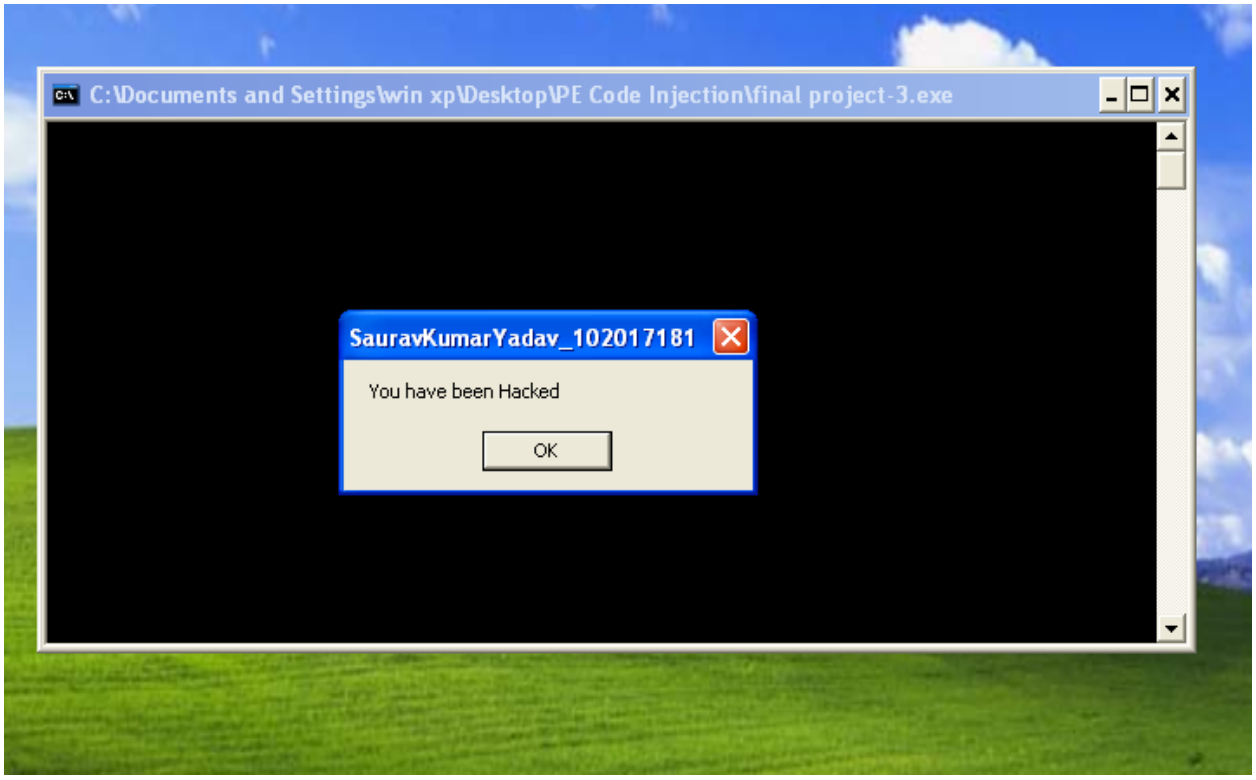


The insertion of MessageBox is successful .Since we have not returned the program to its normal flow yet. It breaks down when ok is pressed. So now the program is returned to it's old flow using ollygdbg.

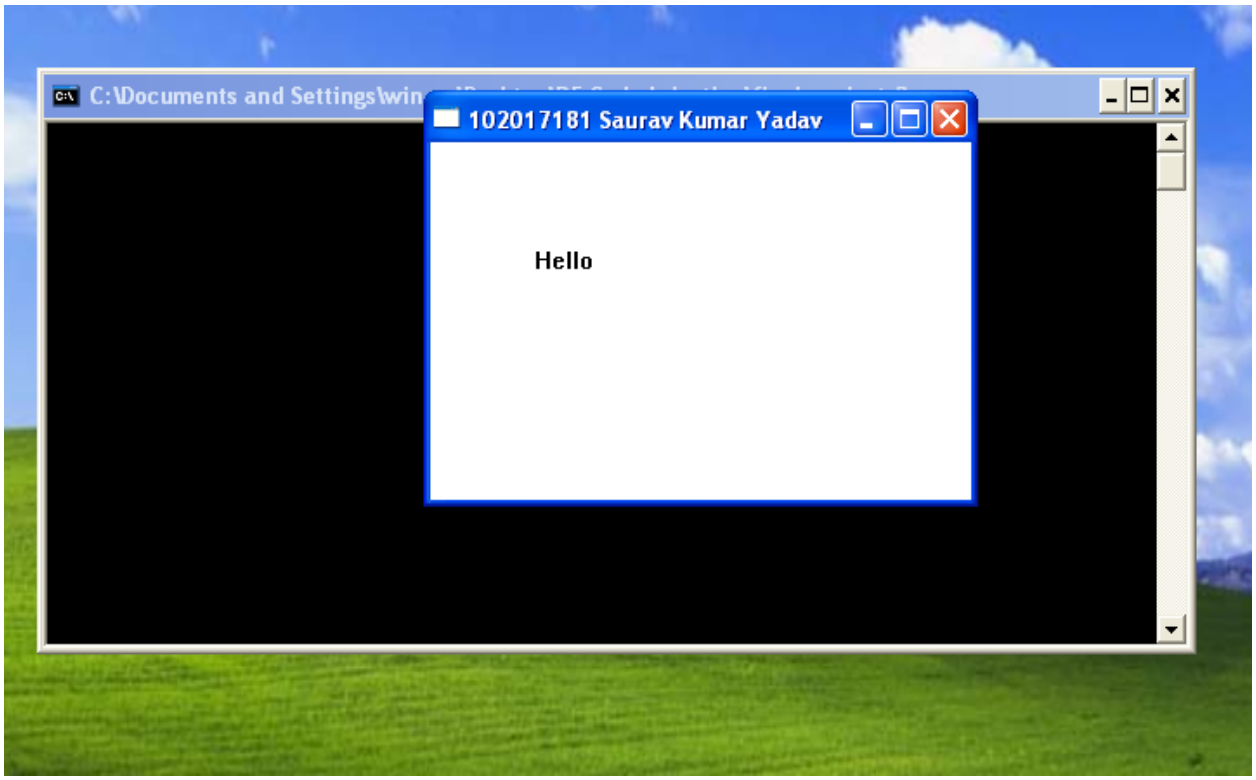
After the MessageBoxA() is called, we add the instructions that we had overwritten with “JMP 00411000”. Then we make the jump to the origin entrypoint using “JMP 00401290” which is the address from where original instructions follow. The selection is then copied to the executable and saved.



The new program behaves as follow:



After ok is pressed following appear



Which is the original program.

Hence a prgram was created using Win32 to display “Hello”. This program was hijacked and MessageBox was sucessfully injected in the program to display “Saurav Kumar Yadav_102017181” as caption and “You have been Hacked” as text . Then the original entry point of the program was retrieved and the flow of program was returned to original.

Code used to create the app:

```
#include <windows.h>

LRESULT CALLBACK WndProc(HWND, UINT, WPARAM, LPARAM);

static char gszClassName[]  = "102017181 Saurav Kumar Yadav";
static HINSTANCE ghInstance = NULL;

int WINAPI WinMain(HINSTANCE hInstance, HINSTANCE hPrevInstance, LPSTR lpCmdLine,
int nCmdShow) {
	WNDCLASSEX WndClass;
	HWND hwnd;
	MSG Msg;

	ghInstance = hInstance;

	WndClass.cbSize        = sizeof(WNDCLASSEX);
	WndClass.style         = NULL;
	WndClass.lpfnWndProc   = WndProc;
	WndClass.cbClsExtra    = 0;
	WndClass.cbWndExtra    = 0;
	WndClass.hInstance     = ghInstance;
	WndClass.hIcon         = LoadIcon(NULL, IDI_APPLICATION);
	WndClass.hCursor       = LoadCursor(NULL, IDC_ARROW);
	WndClass.hbrBackground = (HBRUSH)(COLOR_WINDOW+1);
	WndClass.lpszMenuName  = NULL;
	WndClass.lpszClassName = gszClassName;
	WndClass.hIconSm       = LoadIcon(NULL, IDI_APPLICATION);

	if(!RegisterClassEx(&WndClass)) {
		MessageBox(0, "Error Registering Window!", "Error!", MB_ICONSTOP | MB_OK);
		return 0;
	}

	hwnd = CreateWindowEx(
		WS_EX_STATICEDGE,
		gszClassName,
		"102017181 Saurav Kumar Yadav",
		WS_OVERLAPPEDWINDOW,
		CW_USEDEFAULT, CW_USEDEFAULT,
		320, 240,
		NULL, NULL,
		ghInstance,
		NULL);
```

```
	if(hwnd == NULL) {
		MessageBox(0, "Window Creation Failed!", "Error!", MB_ICONSTOP | MB_OK);
		return 0;
	}

	ShowWindow(hwnd, nCmdShow);
	UpdateWindow(hwnd);

	while(GetMessage(&Msg, NULL, 0, 0)) {
		TranslateMessage(&Msg);
		DispatchMessage(&Msg);
	}
	return Msg.wParam;
}

LRESULT CALLBACK WndProc(HWND hwnd, UINT Message, WPARAM wParam, LPARAM
lParam) {
	HDC hdc;
	PAINTSTRUCT ps;SECURE CODING-UCS638
	LPSTR szMessage = "Hello!";

	switch(Message) {
		case WM_PAINT:
			hdc = BeginPaint(hwnd, &ps);
			TextOut(hdc, 70, 50, szMessage, strlen(szMessage));
			EndPaint(hwnd, &ps);
			break;
		case WM_CLOSE:
			DestroyWindow(hwnd);
			break;
		case WM_DESTROY:
			PostQuitMessage(0);
			break;
		default:
			return DefWindowProc(hwnd, Message, wParam, lParam);
	}
	return 0;
}
```