CS165-lab3

setup environment

- Copy the program file 'try_me' to your own Linux PC or server
- Command to make it as executable file:
  - ```
    gli076@zerg:~/project3$ chmod 777 try_me
    ```
- Then run it with input

  ```
  gli076@zerg:~/project3$ ./try_me abc
  file name: /home/admin/uid_74146_crack
  You have input: abc
  ```
- Or input with hex code
  - The character of string "abc" is

    ```
    gli076@zerg:~/project3$ ./try_me $(printf "\x61\x62\x63")
    file name: /home/admin/uid_74146_crack
    You have input: abc
    ```
    ```

overflow

- strcpy()

```
void test(char* input)
{
    char test[17] = "abc";
    strcpy(test, input);

    printf("You have input: %s\n", test);
}
```

gdb

- Open in gdb
```
gli076@zerg:~/project3$ gdb ./try_me
```
- set breakpoint
  - by line number

```
(gdb) b 12
Breakpoint 1 at 0x8048e4a: file test.c, line 12.
```

- run in gdb
  - input hex number

```
(gdb) r $(printf "\x61\x62\x63")
Starting program: /home/gli076/project3/try_me $(printf "\x61\x62\x63")
file name: /home/admin/uid_74146_crack
You have input: abc
```

- Print memory content
  - For example, print content of stack, you can see the input which is copied in test[17] string

```
(gdb) x/20x $esp
0xffffd0c0:     0x00000000      0x080ea00c      0xffffd118      0x6104f220
0xffffd0d0:     0x00006362      0x00000000      0x00000000      0x00000000
0xffffd0e0:     0x080da304      0x080eaf84      0xffffd118      0x08048f68
0xffffd0f0:     0xffffd33d      0x080ebf40      0x000121a2      0x08048f27
0xffffd100:     0x00000002      0xffffd1b4      0xffffd1c0      0x000121a2
```

```
 8   void test(char* input)
 9   {
10       char test[17] = "abc";
11       strcpy(test, input);
```

gdb

- see the asm code
-

```
(gdb) disassemble
Dump of assembler code for function test:
   0x08048e24 <+0>:     push   %ebp
   0x08048e25 <+1>:     mov    %esp,%ebp
   0x08048e27 <+3>:     sub    $0x28,%esp
   0x08048e2a <+6>:     movl   $0x636261,-0x19(%ebp)
   0x08048e31 <+13>:    movl   $0x0,-0x15(%ebp)
   0x08048e38 <+20>:    movl   $0x0,-0x11(%ebp)
   0x08048e3f <+27>:    movl   $0x0,-0xd(%ebp)
   0x08048e46 <+34>:    movb   $0x0,-0x9(%ebp)
=> 0x08048e4a <+38>:    sub    $0x8,%esp
   0x08048e4d <+41>:    pushl  0x8(%ebp)
   0x08048e50 <+44>:    lea    -0x19(%ebp),%eax
   0x08048e53 <+47>:    push   %eax
   0x08048e54 <+48>:    call   0x80481e0
   0x08048e59 <+53>:    add    $0x10,%esp
   0x08048e5c <+56>:    sub    $0x8,%esp
   0x08048e5f <+59>:    lea    -0x19(%ebp),%eax
   0x08048e62 <+62>:    push   %eax
   0x08048e63 <+63>:    push   $0x80be2c8
   0x08048e68 <+68>:    call   0x804f200 <printf>
   0x08048e6d <+73>:    add    $0x10,%esp
   0x08048e70 <+76>:    leave
   0x08048e71 <+77>:    ret
End of assembler dump.
```

gdb

- Other popular commands
  - si: execute one more asm instruction
  - step: execute until reach a new source code line (C code)

CS165-lab2

Reverse Engineering

Use ‘cmd’ to open the authenticate_yourself.exe

```
D:\学习\UCR\CS165\lab2>authenticate_yourself.exe
Please enter your username and password to be authenticated:
Username: 123
Password: 123
Incorrect username. You are not allowed to enter the system.
```

```
D:\学习\UCR\CS165\lab2>authenticate_yourself.exe
Please enter your username and password to be authenticated:
Username: admin
Password:
Here’s your flag:
```

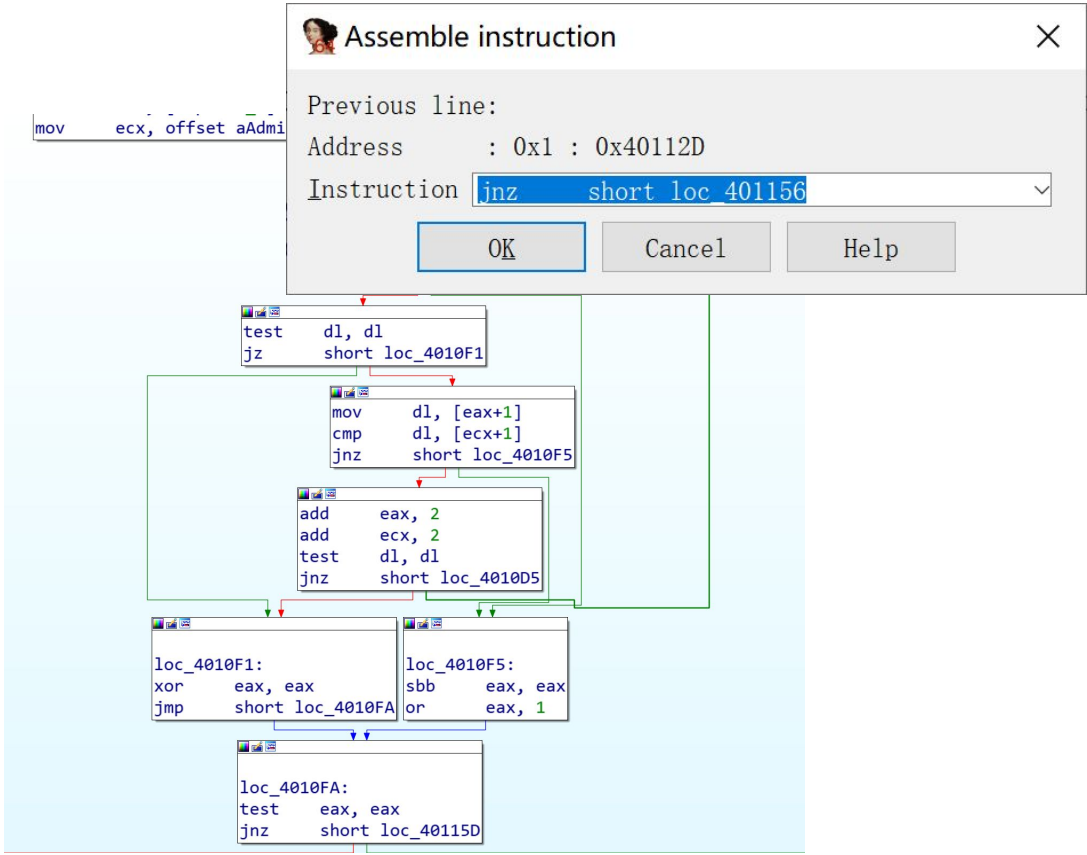
IDA pro Edit->patch program->assemble

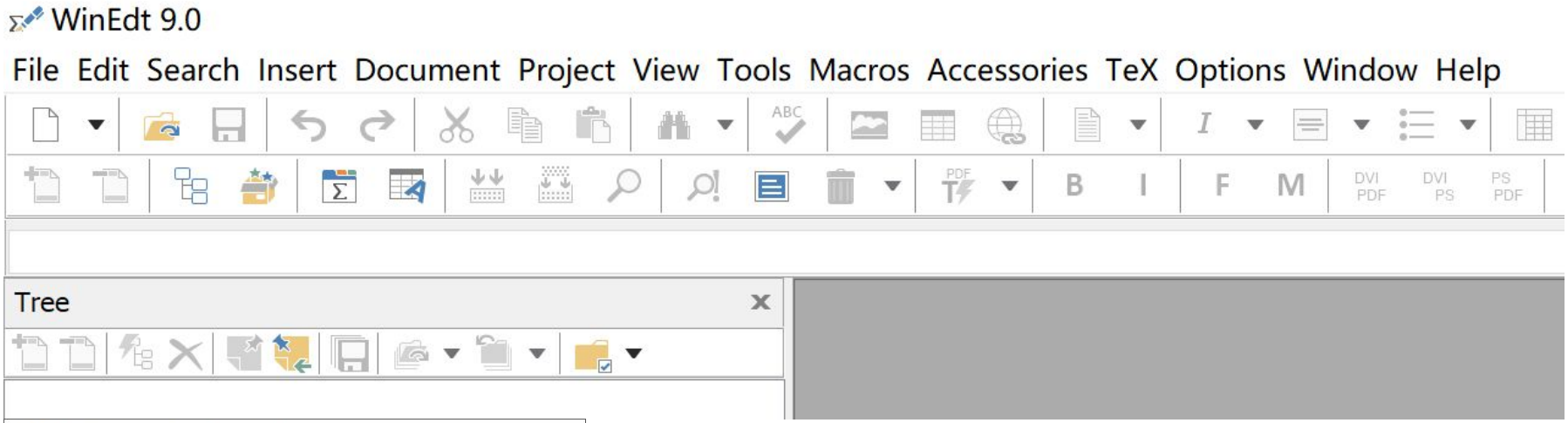
```
; Attributes: bp-based frame

; int __cdecl main(int argc, const char **argv, const char **envp)
_main proc near

var_70= byte ptr -70h
var_C= byte ptr -0Ch
var_4= dword ptr -4
argc= dword ptr  8
argv= dword ptr  0Ch
envp= dword ptr  10h

push    ebp
mov     ebp, esp
sub     esp, 70h
mov     eax, ___security_cookie
xor     eax, ebp
mov     [ebp+var_4], eax
push    offset aPleaseEnterYou ; "Please enter your username and password"...
call    sub_401020
push    offset aUsername ; "Username: "
call    sub_401020
lea     eax, [ebp+var_C]
push    eax
push    offset aS       ; "%s"
call    sub_401050
push    offset aPassword ; "Password: "
call    sub_401020
lea     eax, [ebp+var_70]
push    eax
push    offset a100s    ; "%100s"
call    sub_401050
add     esp, 1Ch
lea     eax, [ebp+var_C]
mov     ecx, offset aAdmin ; "admin"
```




WinEdt 9.0
File Edit Search Insert Document Project View Tools Macros Accessories TeX Options Window Help
Tree
WinEdt Registration Reminder
WinEdt's trial period of 31 days has elapsed.
You can continue to use this unregistered copy of the program. However, as the time goes on, WinEdt will be issuing
(annoying) registration reminders with increasing frequency ...
If you want to use WinEdt please visit
http://www.winedt.com
or contact the author by email at
support@winedt.com
to arrange the registration details.

CS165-lab1

Password crack

- https://www.vidarholen.net/contents/blog/?p=32
- Design a program, input password, output crypt result
- Bruteforce
  - Try password from 'a' to 'aaaaaa' to 'zzzzzz'
- ASCII
  - https://en.wikipedia.org/wiki/ASCII
  - 'a' : 97 (hex: 61)
  - 'z' : 122 (hex: 7A)
- MD5 hash
  - A function
    - Input: string of any length
      - 'az': 61 7A (16 bit: 0110 0001 0111 1010)
    - Output: 128 bit number
      - 3b 00 b4 72 a2 a9 58 80 8f 55 38 e6 4c 39 d9 0a (';¡´r¢©X□□U8æL9Ù' + ...)
      - ';'
    - What if there is 00 ?
      - Pay attention to your programming language

Compute crypt result - 1

- team0:1hfT7jp2q$wPwz7GC6xLt9eQZ9eJkaq.:16653:0:99999:7:::
- Give a password
  - Input
    - password: 'zhgnnd'
    - salt: 'hfT7jp2q'
    - magic: '1'
  - output
    - 'wPwz7GC6xLt9eQZ9eJkaq.'

Compute crypt result - 2

- Input
  - password: 'zhgnnd'
  - salt: 'hfT7jp2q'
  - magic: '1'
- output
  - 'wPwz7GC6xLt9eQZ9eJkaq.'

- Alternate sum:
  - alternate_sum = md5('zhgnndhfT7jp2qzhgnnd') = 'aabbccddeeffgghh'
  - 'aabbccddeeffgghh' (61 61 62 62 63 63 64 64 65 65 66 66 67 67 68 68)
- Len(password)
  - 6 (1 1 0)
  - 3 (1 1)
- Intermediate$_0$
  - concatenation = 'zhgnnd1hfT7jp2q' + 'aabbcc' + 'z' (hex: 7A) + 0 (NULL byte, not '00' string) + 0
    - if password is 'zhg', the len(password) = 3 (binary: 1 1)
      - then, concatenation = 'zhg1hfT7jp2q' + 'aab' + 0 + 0
  - intermediate$_0$ = md5(concatenation) = 'hhggffeeddccbbaa' (68 68 67 67 66 66 65 65 64 64 63 63 62 62 61 61)
- Intermediate$_1$ (i = 0)
  - concatenation = 'hhggffeeddccbbaa' + 'zhgnnd' = 'hhggffeeddccbbaazhgnnd'
  - intermediate1 = md5(concatenation) = 128bit number

Compute crypt result - 2

- Input
  - password: 'zhgnnd'
  - salt: 'hfT7jp2q'
  - magic: '1'
- output
  - 'wPwz7GC6xLt9eQZ9eJkaq.'

- Intermediate$_{1000}$:
  - b'aassddffgghhjjkk' (61 61 73 73 64 64 66 66 67 67 68 68 6a 6a 6b 6b)
- Replacement:
  - b'hdhdsgksgkafjafj' (68 64 68 64 73 67 6b 73 67 6b 61 66 6a 61 66 6a)
  - b'\x68\x64\x68\x64'
  - 011010000110010001101000011001000111001101100111011010110111001101100111011010110110000101100110011010100110000101100110011010
- Crypt base 64:
  - ./0123456789ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz
  - group0: 101010 (42) --- 'e'
  - gourp1:011001(25) ---'N'
  - ...
  - group21: 01(1) ---'/'
  - 'eN...../' (22 digits)
  - //128 = 6 * 21 + 2