objdump命令是Linux下的反汇编目标文件或者可执行文件的命令，它还有其他作用，下面以ELF格式可执行文件test为例详细介绍：

objdump -f test

显示test的文件头信息

objdump -d test

反汇编test中的需要执行指令的那些section

objdump -D test

与-d类似，但反汇编test中的所有section

objdump -h test

显示test的Section Header信息

objdump -x test

显示test的全部Header信息

objdump -s test

除了显示test的全部Header信息，还显示他们对应的十六进制文件代码

举例：

将C源代码和反汇编出来的指令对照：

1.

编译成目标文件（要加-g选项）

gcc -g -o test.c

2.

输出C源代码和反汇编出来的指令对照的格式

objdump -S test.o

如下：

```
 4
 5 test.o:     file format elf32-i386
 6
 7
 8 Disassembly of section .text:
 9
10 00000000 <main>:
11 #include <stdio.h>
12
13 void main()
14 {
15    0:   55                      push   %ebp
16    1:   89 e5                   mov    %esp,%ebp
17    3:   83 e4 f0                and    $0xfffffff0,%esp
18    6:   83 ec 10                sub    $0x10,%esp
19   printf("Hello World!\n");
20    9:   c7 04 24 00 00 00 00    movl   $0x0,(%esp)
21   10:   e8 fc ff ff ff          call   11 <main+0x11>
22 }
23   15:   c9                      leave
24   16:   c3                      ret
```

如何对任意一个二进制文件进行反汇编?

我们可以这样做：

objdump -D -b binary -m i386 a.bin

-D表示对全部文件进行反汇编，-b表示二进制，-m表示指令集架构，a.bin就是我们要反汇编的二进制文件

objdump -m可以查看更多支持的指令集架构，如i386:x86-64，i8086等

另外上面的所有objdump命令的参数同样适用于arm-linux-objdump。

同时我们也可以指定big-endian或little-endian（-EB或-EL），我们可以指定从某一个位置开始反汇编等。所以objdump命令是非常强大的!