<https://blog.csdn.net/stpeace/article/details/47069215>

在linux中， 有些命令是大家通用的， 比如ls, rm, mv, cp等等， 这些我觉得没有必要再细说了。 而有些命令， 只有开发人员才会用到的， 这类命令， 作为程序员的我们， 是有必要了解的， 有的甚至需要熟练使用。

        有的人总说， 这些命令不重要， 用的时候去查就行了， 这么多么扯淡的说法啊。 具体用法细节是可以可查， 但至少得知道有ldd这个东西吧。连ldd都不知道， 怎么知道ldd是干啥的呢？

       在本文中， 我们来介绍一下ldd命令， 尽管它非常简单。  哦， 我突然想起来， 我有个朋友， 她的名字的是三个字， 首写字母刚好是l, d, d, 有点意思。  在linux中， ldd是list, dynamic, dependencies的缩写， 意思是， 列出动态库依赖关系。  当然， 你也可以用ldd --help或者man ldd来看其用法。 下面， 我们也来看看：

       test.h的内容为：

**[cpp]** [view plain](https://blog.csdn.net/stpeace/article/details/47069215) [copy](https://blog.csdn.net/stpeace/article/details/47069215)

1. **void** print();

       test.c的内容为：

**[cpp]** [view plain](https://blog.csdn.net/stpeace/article/details/47069215) [copy](https://blog.csdn.net/stpeace/article/details/47069215)

1. #include <stdio.h>
2. #include "test.h"
4. **void** print()
5. {
6. printf("rainy days\n");
7. }

       main.c的内容为：

**[cpp]** [view plain](https://blog.csdn.net/stpeace/article/details/47069215) [copy](https://blog.csdn.net/stpeace/article/details/47069215)

1. #include "test.h"
3. **int** main()
4. {
5. print();
6. **return** 0;
7. }

      进行一系列的编译， 并用ldd命令， 得到：

**[plain]** [view plain](https://blog.csdn.net/stpeace/article/details/47069215) [copy](https://blog.csdn.net/stpeace/article/details/47069215)

1. [taoge@localhost learn\_ldd]$ ls
2. main.c  test.c  test.h
3. [taoge@localhost learn\_ldd]$ gcc -c main.c test.c
4. [taoge@localhost learn\_ldd]$ gcc main.o test.o
5. [taoge@localhost learn\_ldd]$ ls
6. a.out  main.c  main.o  test.c  test.h  test.o
7. [taoge@localhost learn\_ldd]$ ./a.out
8. rainy days
9. [taoge@localhost learn\_ldd]$
10. [taoge@localhost learn\_ldd]$
11. [taoge@localhost learn\_ldd]$
12. [taoge@localhost learn\_ldd]$ ldd \*
13. a.out:
14. linux-gate.so.1 =>  (0x00ba1000)
15. libc.so.6 => /lib/libc.so.6 (0x0087e000)
16. /lib/ld-linux.so.2 (0x00858000)
17. main.c:
18. ldd: warning: you do not have execution permission for `./main.c'
19. not a dynamic executable
20. main.o:
21. ldd: warning: you do not have execution permission for `./main.o'
22. not a dynamic executable
23. test.c:
24. ldd: warning: you do not have execution permission for `./test.c'
25. not a dynamic executable
26. test.h:
27. ldd: warning: you do not have execution permission for `./test.h'
28. lddlibc4: cannot read header from `./test.h'
29. test.o:
30. ldd: warning: you do not have execution permission for `./test.o'
31. not a dynamic executable
32. [taoge@localhost learn\_ldd]$

      可以看到a.out依赖于libc.so.6这个库， 而这个库的路径为/lib/libc.so.6

      我们继续看使用静态链接库的情形：

**[plain]** [view plain](https://blog.csdn.net/stpeace/article/details/47069215) [copy](https://blog.csdn.net/stpeace/article/details/47069215)

1. [taoge@localhost learn\_ldd]$ ls
2. main.c  test.c  test.h
3. [taoge@localhost learn\_ldd]$ gcc -c test.c
4. [taoge@localhost learn\_ldd]$ ar rcs libtest.a test.o
5. [taoge@localhost learn\_ldd]$ gcc main.c -L. -ltest
6. [taoge@localhost learn\_ldd]$ ls
7. a.out  libtest.a  main.c  test.c  test.h  test.o
8. [taoge@localhost learn\_ldd]$ ./a.out
9. rainy days
10. [taoge@localhost learn\_ldd]$
11. [taoge@localhost learn\_ldd]$
12. [taoge@localhost learn\_ldd]$
13. [taoge@localhost learn\_ldd]$ ldd \*
14. a.out:
15. linux-gate.so.1 =>  (0x00e7c000)
16. libc.so.6 => /lib/libc.so.6 (0x0087e000)
17. /lib/ld-linux.so.2 (0x00858000)
18. libtest.a:
19. ldd: warning: you do not have execution permission for `./libtest.a'
20. not a dynamic executable
21. main.c:
22. ldd: warning: you do not have execution permission for `./main.c'
23. not a dynamic executable
24. test.c:
25. ldd: warning: you do not have execution permission for `./test.c'
26. not a dynamic executable
27. test.h:
28. ldd: warning: you do not have execution permission for `./test.h'
29. lddlibc4: cannot read header from `./test.h'
30. test.o:
31. ldd: warning: you do not have execution permission for `./test.o'
32. not a dynamic executable
33. [taoge@localhost learn\_ldd]$

       这次用静态库， 结果还是差不多， 就没什么好说的了。

       我们继续看使用动态链接库时的情形：

**[plain]** [view plain](https://blog.csdn.net/stpeace/article/details/47069215) [copy](https://blog.csdn.net/stpeace/article/details/47069215)

1. [taoge@localhost learn\_ldd]$ ls
2. main.c  test.c  test.h
3. [taoge@localhost learn\_ldd]$ gcc -c test.c
4. [taoge@localhost learn\_ldd]$ gcc -shared -fPIC -o libtest.so test.o
5. [taoge@localhost learn\_ldd]$ gcc main.c -L. -ltest
6. [taoge@localhost learn\_ldd]$ ls
7. a.out  libtest.so  main.c  test.c  test.h  test.o
8. [taoge@localhost learn\_ldd]$ ./a.out
9. ./a.out: error while loading shared libraries: libtest.so: cannot open shared object file: No such file or directory
10. [taoge@localhost learn\_ldd]$
11. [taoge@localhost learn\_ldd]$
12. [taoge@localhost learn\_ldd]$
13. [taoge@localhost learn\_ldd]$ ldd \*
14. a.out:
15. linux-gate.so.1 =>  (0x00f3d000)
16. libtest.so => not found
17. libc.so.6 => /lib/libc.so.6 (0x0087e000)
18. /lib/ld-linux.so.2 (0x00858000)
19. libtest.so:
20. linux-gate.so.1 =>  (0x0031d000)
21. libc.so.6 => /lib/libc.so.6 (0x00110000)
22. /lib/ld-linux.so.2 (0x00858000)
23. main.c:
24. ldd: warning: you do not have execution permission for `./main.c'
25. not a dynamic executable
26. test.c:
27. ldd: warning: you do not have execution permission for `./test.c'
28. not a dynamic executable
29. test.h:
30. ldd: warning: you do not have execution permission for `./test.h'
31. lddlibc4: cannot read header from `./test.h'
32. test.o:
33. ldd: warning: you do not have execution permission for `./test.o'
34. not a dynamic executable
35. [taoge@localhost learn\_ldd]$
36. [taoge@localhost learn\_ldd]$ su root
37. Password:
38. [root@localhost learn\_ldd]# cp libtest.so /usr/lib/
39. [root@localhost learn\_ldd]# ./a.out
40. rainy days
41. [root@localhost learn\_ldd]# exit
42. exit
43. [taoge@localhost learn\_ldd]$ ./a.out
44. rainy days
45. [taoge@localhost learn\_ldd]$
46. [taoge@localhost learn\_ldd]$
47. [taoge@localhost learn\_ldd]$
48. [taoge@localhost learn\_ldd]$ ldd a.out
49. linux-gate.so.1 =>  (0x00510000)
50. libtest.so => /usr/libtest.so (0x00fe3000)
51. libc.so.6 => /lib/libc.so.6 (0x0087e000)
52. /lib/ld-linux.so.2 (0x00858000)
53. [taoge@localhost learn\_ldd]$

        首先， 我们可以看到， a.out依赖于libtest.so这个库， 但是， 结果是not found， 找不到。 为什么呢？ 因为在/usr/lib下面没有libtest.so， 后来， 我把libtest.so拷贝过去（需要root权限）， 就OK了。 另外， 我们也应该看到， libtest.so的依赖库也是可以通过ldd命令找到的。

         当然， 如果不想自己写程序， 但想试一下ldd命令， 那也可以， 直接如下：

**[plain]** [view plain](https://blog.csdn.net/stpeace/article/details/47069215) [copy](https://blog.csdn.net/stpeace/article/details/47069215)

1. [taoge@localhost learn\_ldd]$ ldd /bin/ls
2. linux-gate.so.1 =>  (0x0052b000)
3. libselinux.so.1 => /lib/libselinux.so.1 (0x00b52000)
4. librt.so.1 => /lib/librt.so.1 (0x00a5c000)
5. libcap.so.2 => /lib/libcap.so.2 (0x0489c000)
6. libacl.so.1 => /lib/libacl.so.1 (0x048c9000)
7. libc.so.6 => /lib/libc.so.6 (0x0087e000)
8. libdl.so.2 => /lib/libdl.so.2 (0x00a0c000)
9. /lib/ld-linux.so.2 (0x00858000)
10. libpthread.so.0 => /lib/libpthread.so.0 (0x00a13000)
11. libattr.so.1 => /lib/libattr.so.1 (0x04d99000)
12. [taoge@localhost learn\_ldd]$ ldd /bin/mv
13. linux-gate.so.1 =>  (0x00944000)
14. libselinux.so.1 => /lib/libselinux.so.1 (0x00b52000)
15. librt.so.1 => /lib/librt.so.1 (0x00a5c000)
16. libacl.so.1 => /lib/libacl.so.1 (0x048c9000)
17. libattr.so.1 => /lib/libattr.so.1 (0x04d99000)
18. libc.so.6 => /lib/libc.so.6 (0x00110000)
19. libdl.so.2 => /lib/libdl.so.2 (0x00a0c000)
20. /lib/ld-linux.so.2 (0x00858000)
21. libpthread.so.0 => /lib/libpthread.so.0 (0x00a13000)
22. [taoge@localhost learn\_ldd]$

        在实际linux开发与调试中， 要经常查看动态库依赖关系， ldd用得还是比较多的， 特别是出现故障的时候。OK, ldd命令就简单介绍到这里了， 虽然简单， 但很实用， 故不可不知。