The best reason for dynamic linking is ease of upgrades. Every application with a built-in static library needs to be recompiled and linked if you need to upgrade the library. With dynamic linking, one update works across all the binaries.

Dynamic linking is used in almost all cases because it saves disk space and eases upgrades

You can use which to know the absolute url of a command

For example:

which ln

=> /usr/bin/ln

which sln

=> /usr/sbin/sln

**ldd /usr/sbin/sln**

/sbin/sln:

not a dynamic executable

ldd is statically linked

ldd /usr/bin/ln

/bin/ln:

linux-vdso.so.1 (0x00007ffedd31e000)

libc.so.6 => /lib64/libc.so.6 (0x00007f2d3bd5d000)

/lib64/ld-linux-x86-64.so.2 (0x00007f2d3c11d000)

ln is dynamically linked

**linux-vdso.so.1**

The library that you see as linux-vdso.so.1 is a virtual library, or Virtual Dynamic Shared Object, that is located only in each program's address space. Some systems call this linux-gate.so.1. This virtual library provides the necessary logic to allow user programs to access system functions through the fastest means available on the particular processor, either interrupt, or with most newer processors, fast system call.

**libc.so.6**

has a pointer to /lib64/libc.so.6 or /lib/i386-linux-gnu/libc.so.6.

**/lib64/ld-linux-x86-64.so.2**

is the absolute path to another library.