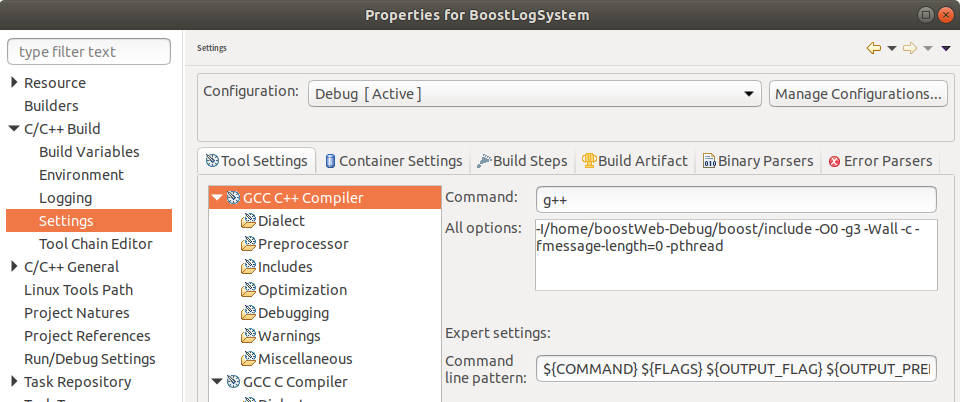
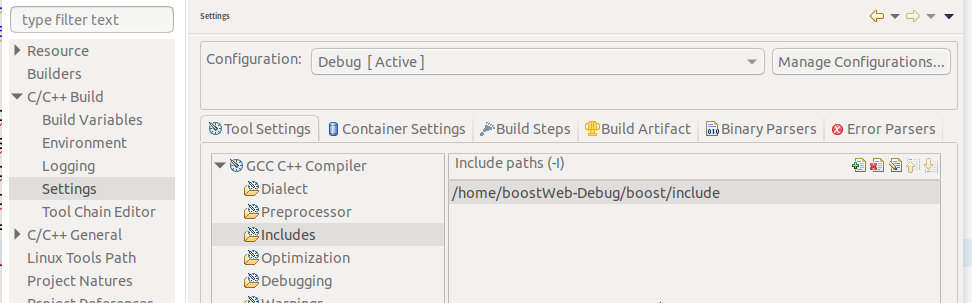
Ubuntu与Fedaro装载库差异

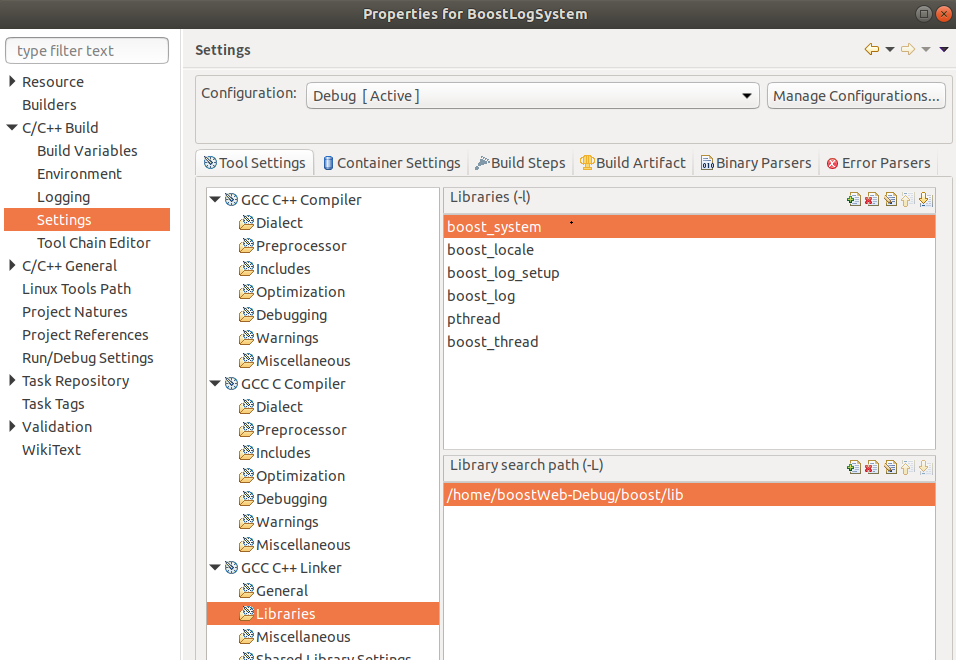
问题：基于boost库的日志系统，在fedora28下正常运行，在Ubuntu下，无法运行，出现以下错误：

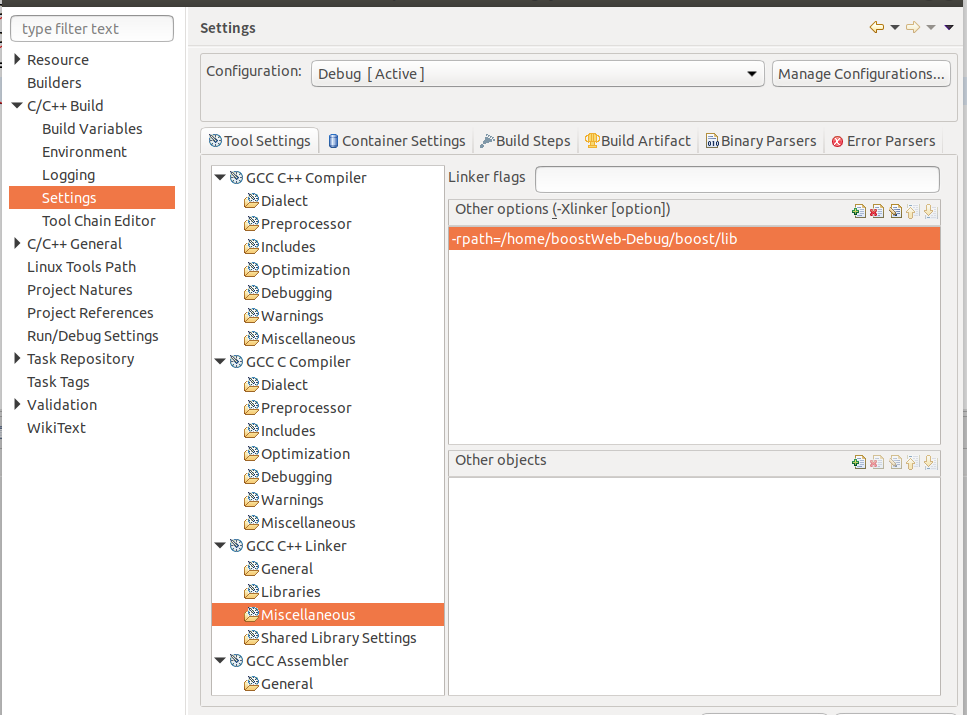
/home/wangh/workspace/BoostLogSystem/Debug/BoostLogSystem: error while loading shared libraries: libboost\_atomic.so.1.67.0: cannot open shared object file: No such file or directory

配置详细见下图：









问题原因：

是ubuntu内核ld的问题，ld主要用于程序编译最后库的链接

1.GCC编译的程序都是elf格式的，elf格式的程序对库的寻找路径有两个不同的概念一个是rpath，一个是runpath

2.rpath的寻找优先级是高于所有的，如果elf文件中写入了rpath，程序执行时候首先从这个指定的路径去寻找库，这个路径优先级高于所有环境变量，并且全局生效

3.runpath的寻找优先级是低于系统环境变量的，而且只能寻找本程序直接依赖的库，多重依赖库情况下，往下递归寻找的时候这个路径是不会生效的，所以多重依赖库会找不到（只能通过加入环境变量来弥补这个问题）

rpath与runpath区别：[https://www.bbsmax.com/A/A7zg2ROkd4/](https://www.bbsmax.com/A/A7zg2ROkd4/" \t "_blank)

**基于以上三点特性，我们使用同样的链接参数-Wl,-rpath=dir(或者-XLink -rpath=dir)**

Fedora下ld会默认生成rpath，所以我们寻找库路径正常

ubuntu下ld会默认生成runpath，所以我们底层依赖的库找不到路径

ubuntu下ld会这么做的原因是因为rpath这个东西正因为优先级太高，所以在安全方面留下了不安全的后门，如果有人恶意通过修改elf文件的rpath，把rpath改成他自定义的目录，通过一些技术手段可以入侵到没有权限入侵的系统中去。所以ubuntu内核就自以为是的出于安全考虑直接默认禁用了rpath，即便是指定了rpath生成的也是runpath

D\_RPATH后门技术：[https://www.cnblogs.com/gm-201705/p/9864099.html](https://www.cnblogs.com/gm-201705/p/9864099.html" \t "_blank)

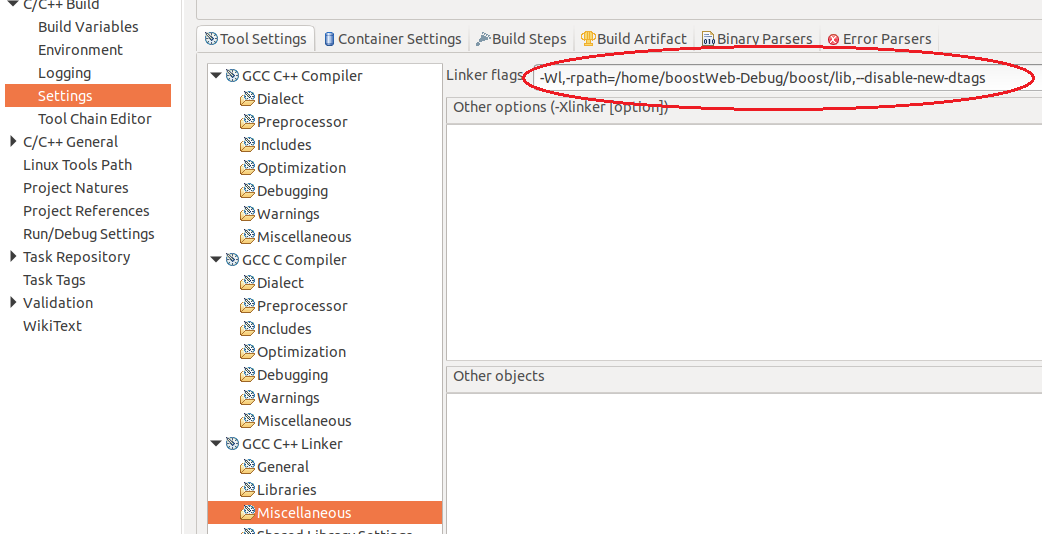
解决方法：

程序链接阶段取消ld的默认禁用，加入参数--disable-new-dtags(禁止添加新的ld标签，runpath就不会生成,可以正常生成rpath)

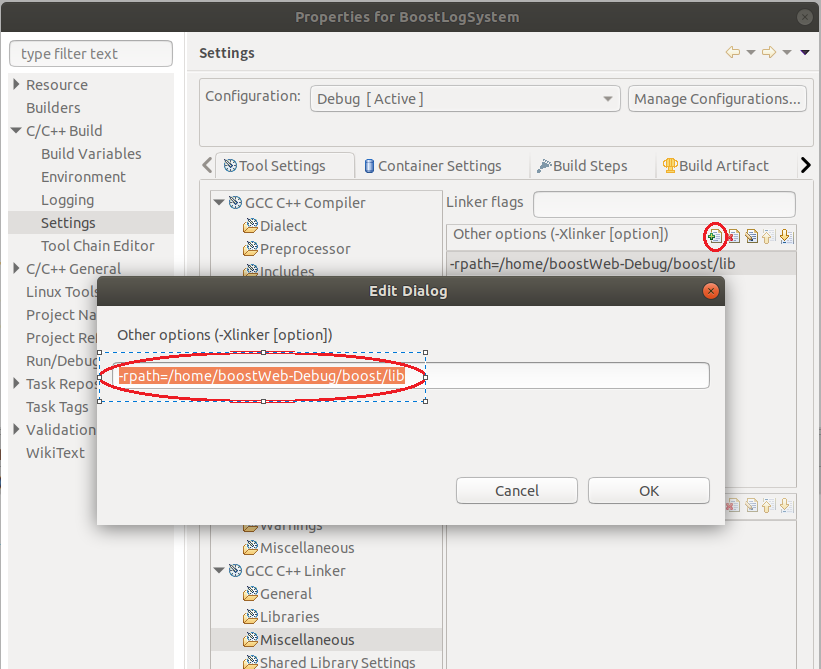
方法一：设置Linker flags，操作如下

Project—》右键——》propertiers——》c/c++ build——》Setting——》GCC C++ Linker——》Miscellaneous 如下图，在Linker flags中输入以下文字，红色圈所示。

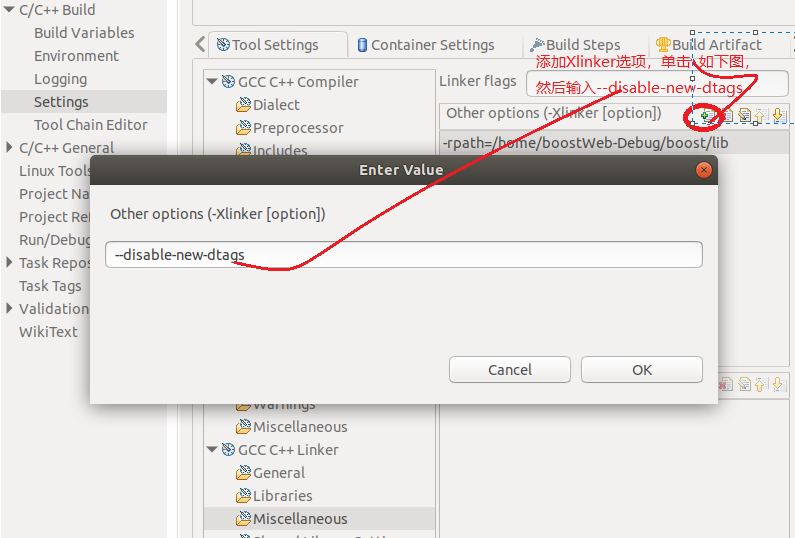
-Wl,-rpath=/home/boost/lib,--disable-new-dtags

 方法二：修改linker选项

1. 设置rpath：Project—》右键——》propertiers——》c/c++ build——》Setting——》GCC C++ Linker——》Miscellaneous



2、禁止添加runpath



最后设置如下图

