

openEuler默认使能LTO探索与实践

Compiler SIG Contributor 王淳洋

目录

- LTO 介绍与趋势

- 基本原理

- 社区洞察

- LTO 优化收益

- 性能提升

- 体积缩减

- LTO 在 openEuler 上的应用

- 使能方案

- 应用适配

- 当前缺陷

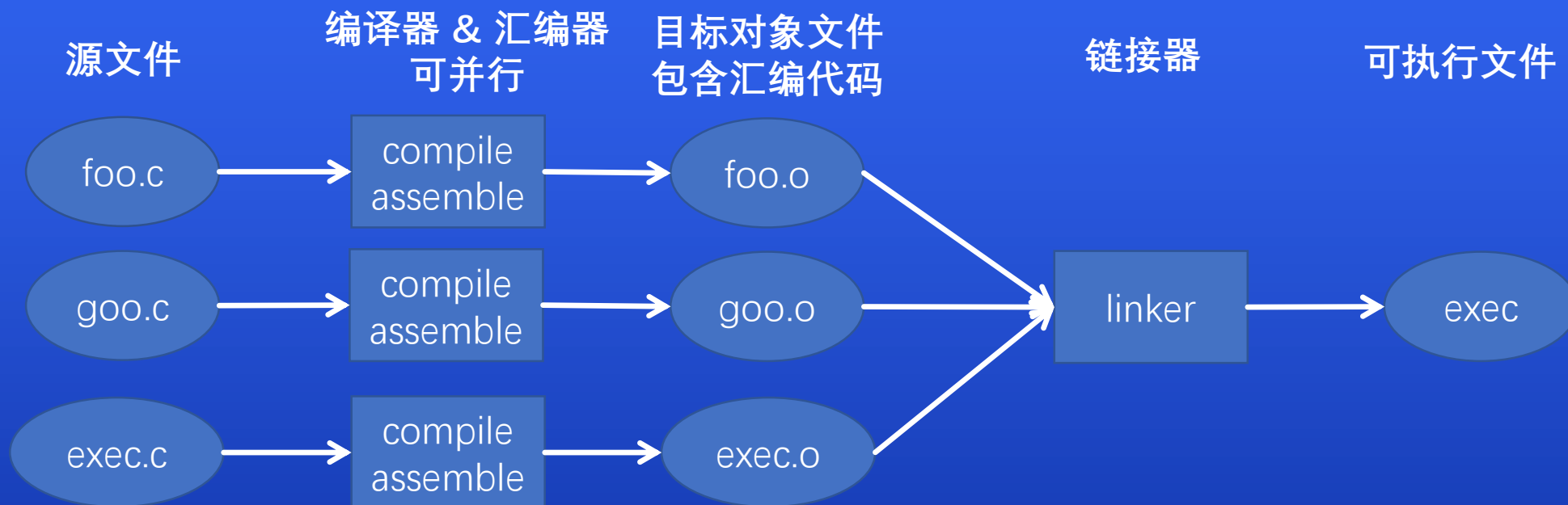
- 未来规划

LTO 介绍与趋势

基本原理

常规编译流程

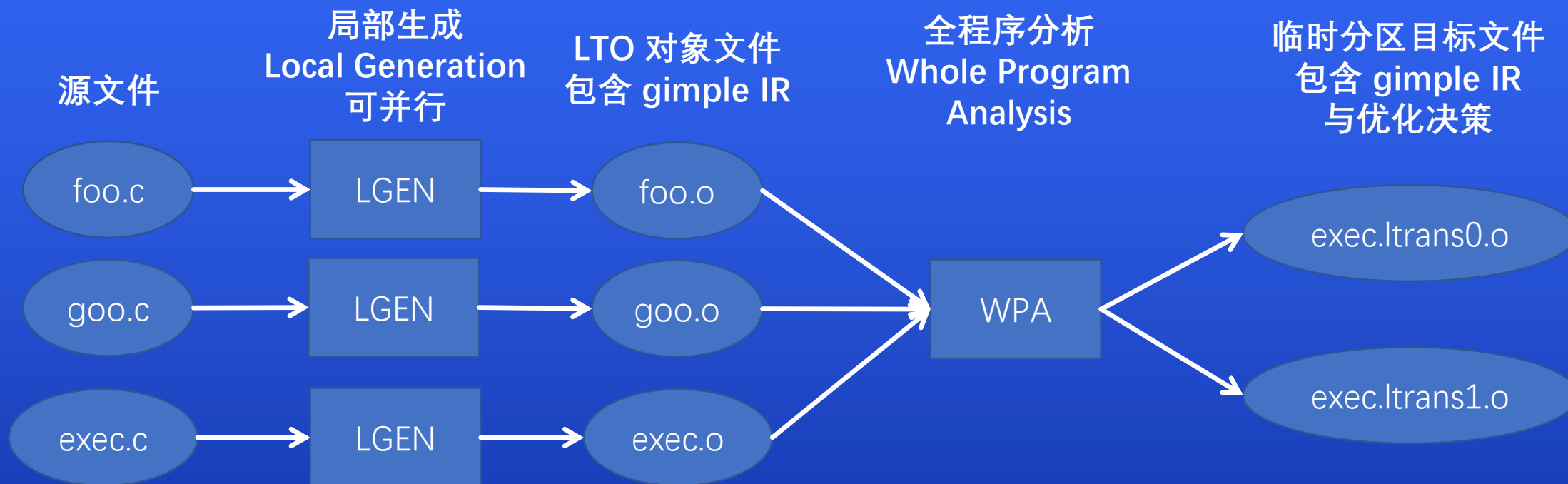
非 LTO:



基本原理

LTO 编译流程

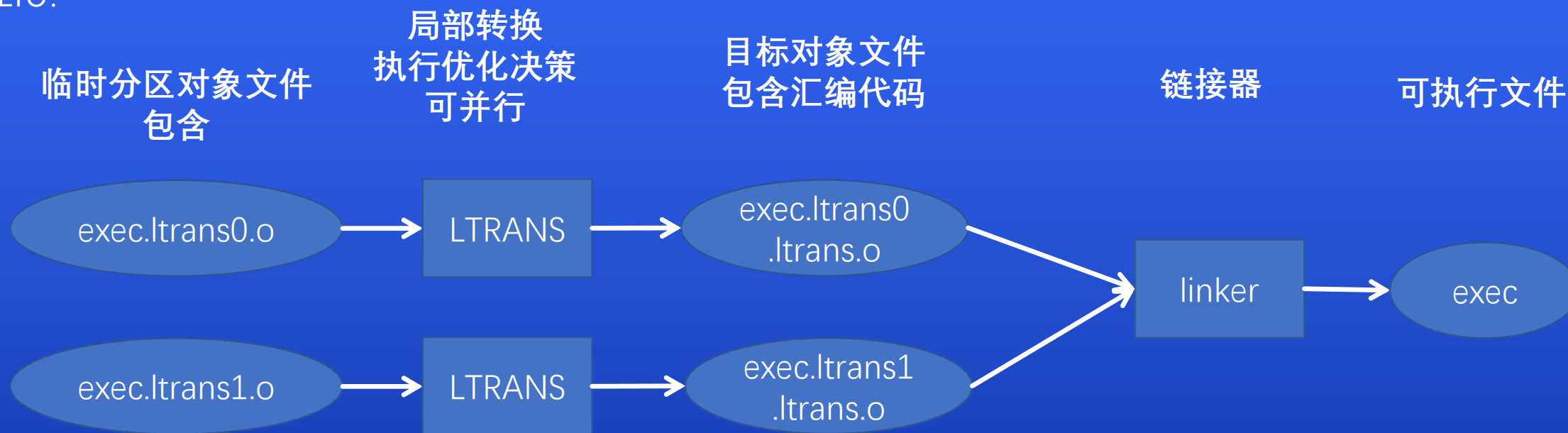
LTO:



基本原理

LTO 编译流程

LTO:

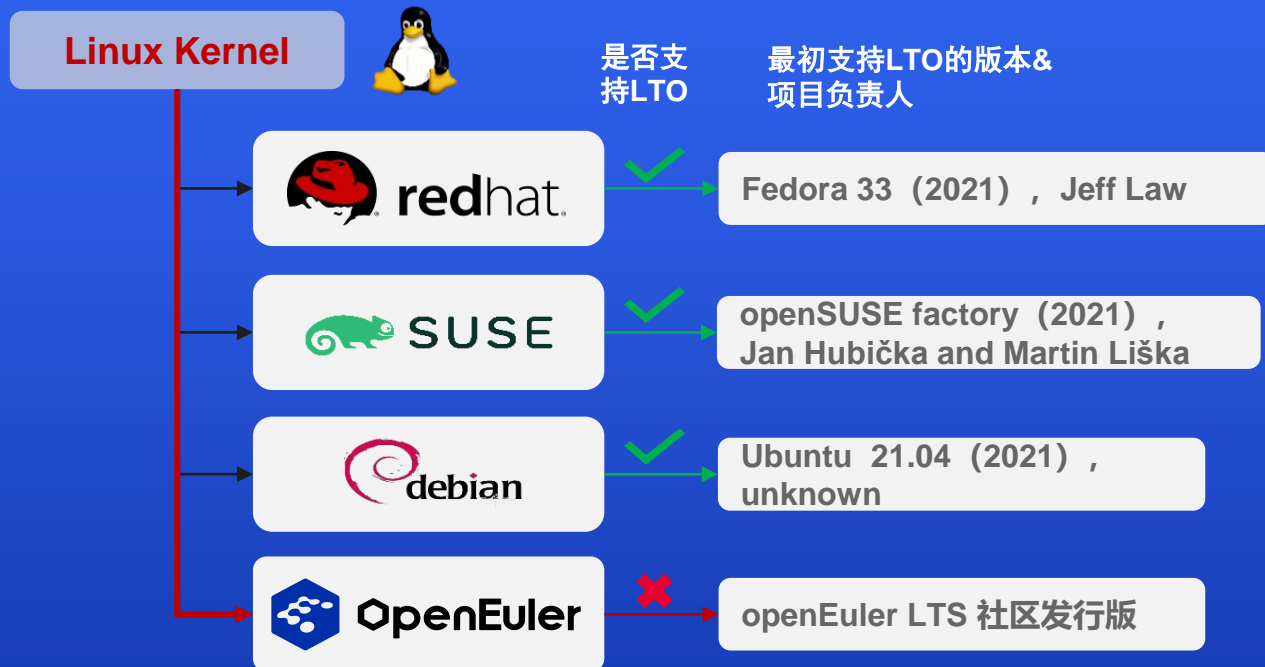


优化机会:

1. 提供跨编译模块之间的优化机会，增强其他优化的优化能力。
2. 全程序分析，舍弃不必要的（不被调用）的代码，直接减小二进制体积。

社区趋势

发展趋势



技术挑战

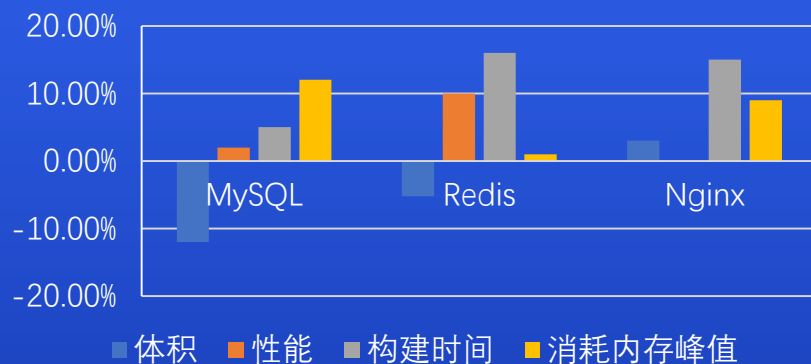
- LTO比正常链接需要更多的内存和更长时间
- LTO可能导致更多未定义行为错误暴露
- LTO不能保证一定产生性能和体积优势
- 已知其他发行版有数百个包遇到无法解决的问题

LTO 优化收益

性能收益

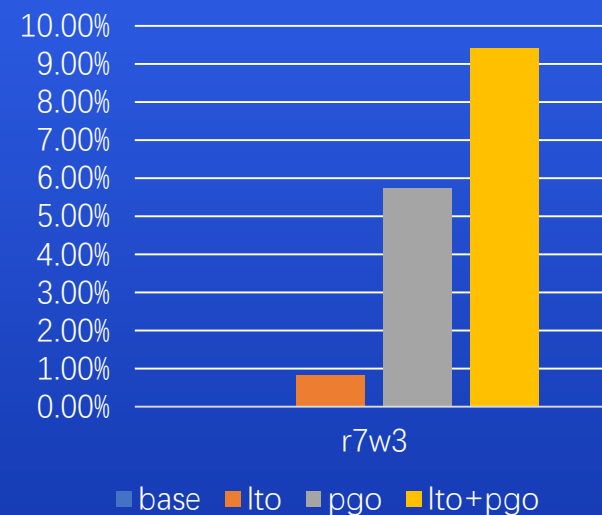
LTO在性能上的收益

单包 LTO 构建效果对各指标的影响



更小的体积，更优的性能、更长的构建时间与更高的编译峰值内存

LTO + PGO 组合优化可以达到 $1 + 1 > 2$ 的效果
以 rocksdb r7w3 随机读写场景为例：



体积削减

版本应用构建中，链接产物体积变化

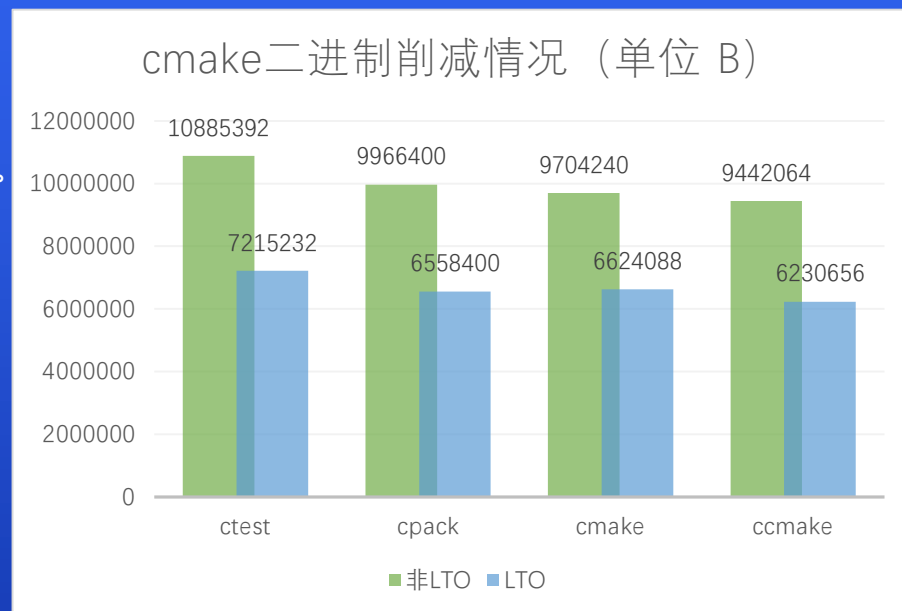
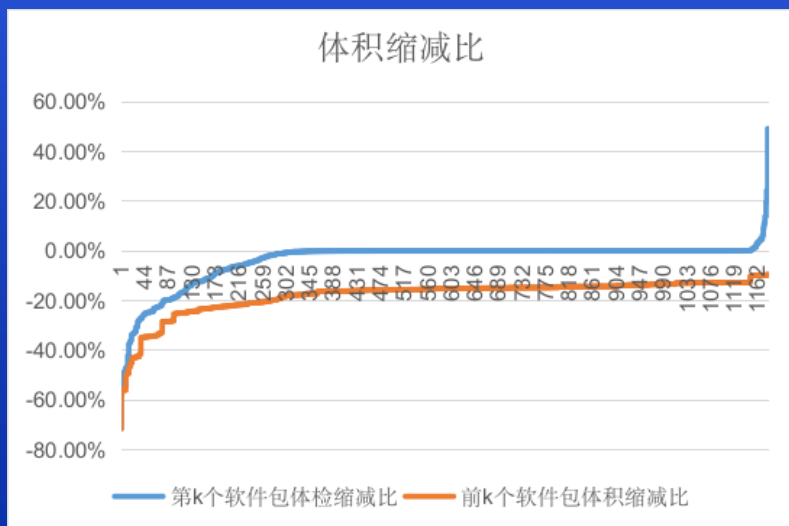
统计范围：openEuler版本构建中 1182 个依赖gcc构建的软件包

统计方式：对这些软件包使能 LTO 构建，下载并解压生成的 rpm 包，

统计其中可执行文件与共享库（链接产物，除.a .o外的 ELF 文件）的体积变化。

统计结果（以下体积均只包含可执行文件与共享库）：

1. 233 个应用的体积减小 5% 以上，12 个应用的体积增大5%以上，1182个应用整体体积减小 9.43%。
2. 233 个减小 5% 以上的应用整体减少 21.03% 。
3. 一些众所周知的应用：git -54.69%，cmake -33.5%。



cmake 构建产生的四个可执行文件 ctest cpack cmake ccmake 均有明显的体积削减

LTO 在 openEuler 上的应用

使能方案

全局默认使能方案（以 Mysql 为例）

- 修改 openEuler-rpm-config 以添加全局选项，并增量构建新的 openEuler-rpm-config。

```
-%_general_options      -O2 -g -grecord-gcc-switches -pipe -fstack-protector-strong %[ "%{toolchain}" == "clang" ? "-fgcc-compatible" : "" ]
+%_lto_cflags -flto=auto -ffat-lto-objects
+%_general_options      -O2 %[_lto_cflags] -g -grecord-gcc-switches -pipe -fstack-protector-strong %[ "%{toolchain}" == "clang" ? "-fgcc-compatible" : "" ]
```

- 通过 EulerMaker 构建软件包时，在日志中可以找到对应选项

```
2024-07-16 02:56:29 cd /root/rpmbuild/BUILD/mysql-8.0.37/build/libservices && /usr/bin/cc -DBOOST_NO_CXX98_FUNCTION_BASE -DHAVE_CONFIG_H -DHAVE_TLSv13 -DLZ4_DISABLE_DEPRECATED_WARNINGS -
D_FILE_OFFSET_BITS=64 -D_GNU_SOURCE -D_USE_MATH_DEFINES -D__STDC_FORMAT_MACROS -D__STDC_LIMIT_MACROS -I/root/rpmbuild/BUILD/mysql-8.0.37/build -I/root/rpmbuild/BUILD/mysql-8.0.37/build/include -
I/root/rpmbuild/BUILD/mysql-8.0.37 -I/root/rpmbuild/BUILD/mysql-8.0.37/include -isystem /usr/include/editline -fno-omit-frame-pointer -ffp-contract=off -O2 -flto=auto -ffat-lto-objects -g -grecord-gcc-
switches -pipe -fstack-protector-strong -Wall -Werror=format-security -Wp,-D_FORTIFY_SOURCE=2 -Wp,-D_GLIBCXX_ASSERTIONS -specs=/usr/lib/rpm/generic-hardened-cc1 -fasynchronous-unwind-tables -fstack-clash-
protection -Wall -Wextra -Wformat-security -Wvla -Wundef -Wmissing-format-attribute -Wwrite-strings -Wjump-misses-init -Wstringop-truncation -Wmissing-include-dirs -ffunction-sections -fdata-sections -O2 -
g -DNDEBUG -fPIC -Wshadow=local -MD -MT libservices/CMakeFiles/mysqlservices.dir/mysql_password_policy_service.c.o -MF CMakeFiles/mysqlservices.dir/mysql_password_policy_service.c.o.d -o
CMakeFiles/mysqlservices.dir/mysql_password_policy_service.c.o -c /root/rpmbuild/BUILD/mysql-8.0.37/libservices/mysql_password_policy_service.c
```

- 构建成功

| 构建历史 | | | | | | | | | | |
|--------|---------------------|---------|-----------------------|-------------|------|---------------|--------------------|----------|--------|----|
| spec名称 | 构建环境 | 架构 | Build ID | Job ID | 构建类型 | 执行机 | 创建时间 | 持续时间 | job状态 | 操作 |
| mysql | openEuler:24.03-LTS | aarch64 | 5aa9c0ec-42ac-11ef... | cbs.5696028 | 全量构建 | local-at40-10 | 2024/7/16 02:51:49 | 30.88min | ● 构建成功 | -- |
| mysql | openEuler:24.03-LTS | aarch64 | da268a02-3f1a-11ef... | cbs.5653383 | 全量构建 | local-at40-4 | 2024/7/11 08:29:42 | 31.47min | ● 构建成功 | -- |
| mysql | openEuler:24.03-LTS | aarch64 | 8ed0efe6-3ee0-11ef... | cbs.5648389 | 全量构建 | local-at32-7 | 2024/7/11 01:32:01 | 1.37h | ● 构建成功 | -- |
| mysql | openEuler:24.03-LTS | aarch64 | 001c334e-3e8d-11e... | cbs.5645707 | 增量构建 | local-at4-1 | 2024/7/10 16:02:34 | 27.13min | ● 构建成功 | -- |
| mysql | openEuler:24.03-LTS | aarch64 | b333d030-3ccc-11e... | cbs.5616969 | 增量构建 | local-at30-5 | 2024/7/9 02:08:08 | 23.83min | ● 构建成功 | -- |

- 该方案可以为 openEuler 上约 80%（1200+/1500+）依赖 gcc 构建的应用使能 LTO。

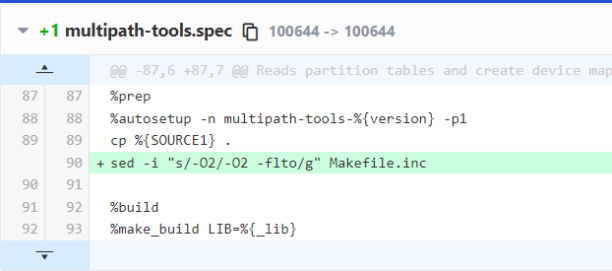
使能方案

独立使能案例（以 multipath-tools 为例）

- 现象：同样通过 openEuler-rpm-config 修改全局编译选项使能，并查看构建日志。另外 20% 的应用的日志中没有预期中的 -flto -ffat-lto-objects 存在。

```
2024-07-04 19:44:55 cc -D_FORTIFY_SOURCE=2 -DBIN_DIR=\"/usr/sbin\" -DMULTIPATH_DIR=\"/lib64/multipath\" -DRUNTIME_DIR=\"/run\" -DCONFIG_DIR=\"/etc/multipath/conf.d\" -DEXTRAVERSION=\"\" -MMD -MP -I. -I../libmultipath -D_LARGEFILE64_SOURCE -D_FILE_OFFSET_BITS=64 --std=gnu99 -O2 -g -fstack-protector-strong --param=ssp-buffer-size=4 -Werror -Wall -Wextra -Wformat=2 -Wformat-overflow=2 -Werror=implicit-int -Werror=implicit-function-declaration -Werror=format-security -Wno-clobbered -Wno-error=clobbered -Werror=cast-qual -Werror=discarded-qualifiers -pipe -fPIE -DPIE -c -o bsd.o bsd.c
2024-07-04 19:44:55 make[1]: Leaving directory '/home/lkp/rpmbuild/BUILD/multipath-tools-0.9.5/kpartx'
2024-07-04 19:44:55 make[1]: Entering directory '/home/lkp/rpmbuild/BUILD/multipath-tools-0.9.5/kpartx'
```

- 原因：部分 rpm 构建时有自定义的构建环境与参数，不受全局编译选项影响。
- 解决方案：可以在 .spec 文件中添加注入 -flto 选项的逻辑后重新构建。



- 以 multipath-tools 为例，替换Makefile.inc 文件中的 -O2 为 -O2 -flto 后，可以在构建日志中找到 -flto 选项，使能成功。

```
2024-07-17 14:31:55 make[1]: Entering directory '/home/lkp/rpmbuild/BUILD/multipath-tools-0.9.5/kpartx'
2024-07-17 14:31:55 building bsd.o because of bsd.c
2024-07-17 14:31:55 cc -D_FORTIFY_SOURCE=2 -DBIN_DIR=\"/usr/sbin\" -DMULTIPATH_DIR=\"/lib64/multipath\" -DRUNTIME_DIR=\"/run\" -DCONFIG_DIR=\"/etc/multipath/conf.d\" -DEXTRAVERSION=\"\" -MMD -MP -I. -I../libmultipath -D_LARGEFILE64_SOURCE -D_FILE_OFFSET_BITS=64 --std=gnu99 -O2 -flto -g -fstack-protector-strong --param=ssp-buffer-size=4 -Werror -Wall -Wextra -Wformat=2 -Wformat-overflow=2 -Werror=implicit-int -Werror=implicit-function-declaration -Werror=format-security -Wno-clobbered -Wno-error=clobbered -Werror=cast-qual -Werror=discarded-qualifiers -pipe -fPIE -DPIE -c -o bsd.o bsd.c
2024-07-17 14:31:55 make[1]: Leaving directory '/home/lkp/rpmbuild/BUILD/multipath-tools-0.9.5/kpartx'
```

- 不同的应用有不同的构建方式，需要开发者根据实际情况选择使能方式。

使能方案

测试 与 单包主动规避

- 由于 lto 在架构与优化上涉及甚广，使能 lto 之后，应用需要有相关的功能和性能测试，以防止出现功能问题或者性能衰退。
- 在当前使能构建成功的 1182 个软件仓中，仅有 586 个仓在构建时执行了 %check 段。

```
360 %check
361 #To avoid some test failed which need root permission
362 NMTST_FORCE_REAL_ROOT=true %make_build check
```

spec文件中的%check段测试脚本

```
2024-07-09 16:47:12 + /usr/lib/rpm/brp-python-hardlink
2024-07-09 16:47:12 Executing(%check): /bin/sh -e /var/tmp/rpm-tmp.7sm2NG
2024-07-09 16:47:12 + umask 022
2024-07-09 16:47:12 + cd /home/lkp/rpmbuild/BUILD
2024-07-09 16:47:12 + cd NetworkManager-1.44.2
2024-07-09 16:47:12 + NMTST_FORCE_REAL_ROOT=true
2024-07-09 16:47:12 + /usr/bin/make -O -j48 V=1 VERBOSE=1 check
2024-07-09 16:47:12 /usr/bin/make check-recursive
2024-07-09 16:47:12 Making check in .
2024-07-09 16:47:12 make[2]: Entering directory '/home/lkp/rpmbuild/BUILD/NetworkManager-1.44.2'
2024-07-09 16:47:12 /usr/bin/xsltproc --output man/nm-settings-keyfile.5 --path man --xinclude --nonet --st
man.authors.section.enabled 0 --stringparam man.copyright.section.enabled 0 --stringparam man.th.title.max.
2024-07-09 16:47:12 make[2]: Leaving directory '/home/lkp/rpmbuild/BUILD/NetworkManager-1.44.2'
```

在构建时自动执行%check中编写的测试脚本

- 各软件仓的也依赖各 owner 的维护。
- 对于有短期内无法解决的问题/性能衰退的包，可在 .spec 文件中添加 '%define _lto_cflags %{nil}' 规避 LTO。

```
@@ -20,6 +20,7 @@
20 20 %bcond_without slowdebug
21 21 # Enable release builds by default on relevant arches.
22 22 %bcond_without release
23 23 + %define _lto_cflags %{nil}
24 24
25 25 # The -g flag says to use strip -g instead of full strip on DSOs or EXEs.
26 26 # This fixes detailed NMT and other tools which need minimal debug info.
```

应用适配

构建失败案例（以 libaio & NetworkManager 为例）

- libaio 与 NetworkManager 使能 lto 后构建失败，原因为 gcc lto 不支持 `__asm__(.symver)` 控制符号版本。
- 上游社区已为 gcc 添加 lto 支持的符号版本控制方式 `__attribute__((__symver__()))`,

```
#if __has_attribute(__symver__)
#define SYMVER(compat_sym, orig_sym, ver_sym) \
+ extern __typeof(compat_sym) compat_sym \
+ __attribute__((__symver__(SYMSTR(orig_sym) "@LIBAIO_" SYMSTR(ver_sym))))
#define DEFSYMVER(compat_sym, orig_sym, ver_sym) \
+ extern __typeof(compat_sym) compat_sym \
+ __attribute__((__symver__(SYMSTR(orig_sym) "@LIBAIO_" SYMSTR(ver_sym))))
#else
#define SYMVER(compat_sym, orig_sym, ver_sym) \
__asm__(".symver " SYMSTR(compat_sym) ", " SYMSTR(orig_sym) "@LIBAIO_" SYMSTR(ver_sym));

#define DEFSYMVER(compat_sym, orig_sym, ver_sym) \
__asm__(".symver " SYMSTR(compat_sym) ", " SYMSTR(orig_sym) "@LIBAIO_" SYMSTR(ver_sym));
#endif
```

libaio

```
diff --git a/src/libnm-glib-aux/nm-macros-internal.h b/src/libnm-glib-aux/nm-macros-internal.h
index 9972dc4..8c83530 100644
--- a/src/libnm-glib-aux/nm-macros-internal.h
+++ b/src/libnm-glib-aux/nm-macros-internal.h
@@ -1075,8 +1075,8 @@ nm_g_variant_equal(GVariant *a, GVariant *b)
     return orig_func args;
 }
 return_type orig_func args_typed;
-__asm__(".symver " G_STRINGIFY(versioned_func) ", " G_STRINGIFY(orig_func) "@" G_STRINGIFY(
- version))
+extern __typeof(versioned_func) versioned_func
+__attribute__((__symver__(G_STRINGIFY(orig_func) "@" G_STRINGIFY(version))))

#define NM_BACKPORT_SYMBOL(version, return_type, func, args_typed, args) \
    _NM_BACKPORT_SYMBOL_IMPL(version, return_type, func, _##func##_##version, args_typed, args)
--
2.43.0
```

NetworkManager

- 类似的问题，比如一些不支持 LTO 的工具(objcopy)/特性(--start-lib --end-lib)，一些跟符号表强相关的测试项，需要应用开发者从源码/构建逻辑方面尽量规避。

当前缺陷

版本应用使能 LTO 构建中暴露的问题

不支持的特性/功能:

`__asm__(.symver)`

`--wrap-func`

`--start-lib --end-lib (gold)`

`objcopy`

手写汇编

优化引起的问题:

更多的inline导致可能出现inline函数定义丢失问题

修改符号表导致一些与符号表强相关的测试项失败

测试套件不足:

仅 50% 的软件仓有 `%check` 段

未来规划

默认使能+自动退出机制

GCC 默认在允许使用 LTO 的场景下默认开启 LTO。

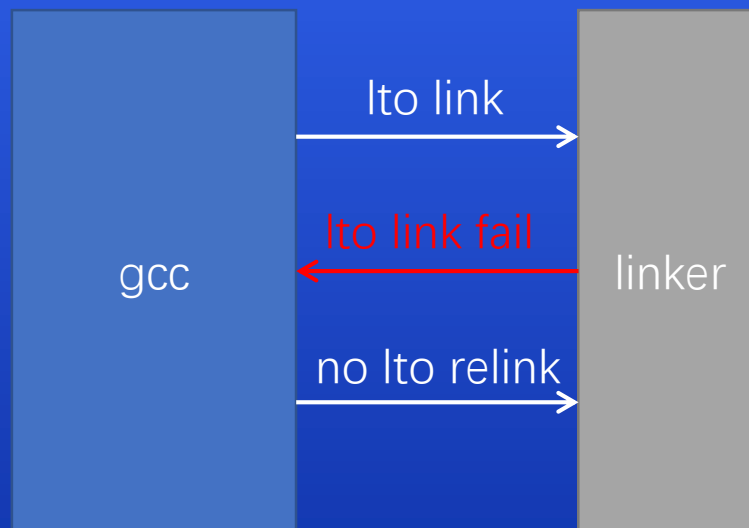
GCC 可以识别到编译命令/构建脚本中与 LTO 不兼容的特性与功能，自动规避 LTO。

GCC LTO 允许在 compile 阶段生成同时包含 LTO 信息段（gimple IR）和汇编段的 fat-lto-object 胖中间文件。

因此在链接阶段，同时具备执行 LTO 优化与不执行 LTO 优化的条件。



路径1：提前识别不支持的特性，利用 fat-lto-object 汇编段链接。



路径2：链接失败时利用 fat-lto-object 汇编段重链接。

未来规划

openEuler全版本使能LTO 与 相关问题修复

根据目前分析情况，构建中遇到的问题可以分为以下几类

| | | | |
|------------------------------|------|-------------|----------------------|
| openEuler 版本构建 默认使能LTO | 构建失败 | 工具链不支持 | gcc/binutils 上游社区 |
| | | 特性不支持 | |
| | | 编译器bug | |
| | | 源码/测试用例需要调整 | |
| | 构建成功 | 功能与性能测试 | 应用owner |
| | 疑难杂症 | 单包规避 | |
| | 未使能 | 构建环境需定制使能方案 | |

联合上游社区和应用 owner，逐步解决全版本 LTO 编译构建中遇到的问题，扩大 LTO 使能场景范围。
在 openEuler 构建版本应用时，默认使能 LTO，以获取更小的软件包体积，与更优的开箱性能。

THANKS