

结项报告

项目信息

- 项目名称：优化Linux内核中的复杂头文件依赖
- 方案描述：
 1. 对于某个特定的源文件A，递归地查找其包含的所有头文件
 2. 计算所有头文件中被A使用到的函数声明/宏的个数，如果使用个数较少的话，则将其列入删除列表
 3. 考虑将删除列表内的头文件中被A用到的函数声明/宏移动到其他头文件中。若能够移动（不影响其他模块的编译），则移动并且删除源文件A对其的引用
 4. 人工检测确定改动合理之后发送补丁给Linux 内核
- 时间规划：
 1. 07/01-07/31 调研linux中复杂头文件依赖问题，调研，开发解决复杂依赖的算法
 2. 08/01-08/15 开发检测头文件依赖可优化点工具的prototype
 3. 08/16-09/30 完成优化头文件依赖的工具，并将优化结果提交给社区

项目总结

项目产出

1. 实现了检测内核头文件依赖的工具，它有两大功能

注：下列功能的使用方法/输出结果请参考gitee/gitlab上的使用文档

- 检测头文件的依赖关系并给出建议

1. 自动检测内核中不需要的头文件，并删除
2. 根据检测内容，自动提交，生成commit log和patch
3. 自动将patch发送给内核相应的maintainer等待审核
4. 对于一些无法删除的头文件，根据依赖状况给出建议。如源文件A只用到了头文件B的一个声明，同时内核中所有用到该声明的文件都包含了头文件C，则工具会建议将声明从头文件B移动到头文件C，以此取消A对于B的依赖
5. 对于不适合改动的头文件，输出其被使用到的宏和函数供程序员参考

- 输出文件依赖关系的信息

1. 对于某个特定的源文件/头文件A，输出以A为根节点的依赖树的图像，方便程序员理清依赖关系

2. 对于某个目录，输出目录下所有文件依赖关系的DAG
3. 对于某个源文件，输出其用到的所有函数/宏，方便程序员理清符号信息
4. 对于某个头文件，输出其所声明的所有函数/宏/自定义类型，方便程序员理清符号信息

2. Patch 的生成与接受情况

1. 目前生成patch大约**200**个（仅针对了部分模块，若针对整个linux内核，预估将会生成**1500**个以上的patch）
2. 目前提交patch 大约有**40**个（考虑到maintainer一次无法处理过多的patch，故采用分次提交的方法，同时部分patch也需要人工审核以确保无误）
3. 被接受的数量有**16**个(在log中已经显示的有11个，邮件确认接受的有5个）（由于项目的时间问题，截至2021/9/30，仍有部分patch处于被审核状态，预计到十月中旬会有更多被接受），如下为部分通过审核的截图（由于时间问题，某些patch还未在commit log中更新，故以maintainer回复的邮件替代）

```
oscar@DESKTOP-U066CHB ~/linux-next (master)> git log --oneline --author="Mianhan Liu"
75b1fd7dc887 include/linux/mm.h: move nr_free_buffer_pages from swap.h to mm.h
3d5f12d4ff78 net: ipv4: remove superfluous header files from fib_notifier.c
0eb10c0c6d61 Bluetooth: btrsi: remove superfluous header files from btrsi.c
2b73e209ba75 net/ipv4/tcp_nv.c: remove superfluous header files from tcp_nv.c
83688aec17bf net/ipv4/xfrm4_tunnel.c: remove superfluous header files from xfrm4_tunnel.c
07b855628c22 net/ipv4/sysctl_net_ipv4.c: remove superfluous header files from sysctl_net_ipv4.c
c595b120ebab net/ipv4/syncookies.c: remove superfluous header files from syncookies.c
bea714581a31 net/ipv4/udp_tunnel_core.c: remove superfluous header files from udp_tunnel_core.c
85c698863c15 net/ipv4/tcp_minisocks.c: remove superfluous header files from tcp_minisocks.c
222a31408ab0 net/ipv4/tcp_fastopen.c: remove superfluous header files from tcp_fastopen.c
ffa66f15e450 net/ipv4/route.c: remove superfluous header files from route.c
```

项目亮点

1. 全部使用自动化工具，工具完成后原则上不需要人为介入，即可自动完成 **发现可删除的依赖关系->删除头文件->检测编译是否通过->提交更改生成patch->发送给对应maintainer** 的整个流程
2. 使用了多线程技术，加快了代码运行速度，对于linux的某个大模块（如net模块），检测速度小于1分钟。在检测完成后，基本上每30S便可以生成一个删除头文件的patch（平均时间，依赖于模块本身的头文件依赖情况）
3. 使用了C++的新标准C++17,同时使用了一些相对新的特性（如filesystem等）
4. 代码使用了智能指针，同时进行了周密的内存管理，使用valgrind检测到并无内存泄漏，如下为检测情况

```
==5094==
==5094== HEAP SUMMARY:
==5094==    in use at exit: 0 bytes in 0 blocks
==5094== total heap usage: 62,930 allocs, 62,930 frees, 6,126,872 bytes allocated
==5094==
==5094== All heap blocks were freed -- no leaks are possible
==5094==
==5094== For lists of detected and suppressed errors, rerun with: -s
==5094== ERROR SUMMARY: 0 errors from 0 contexts (suppressed: 0 from 0)
```

工具的重要性和独特性

1. 由于 `#include` 指令会直接将头文件内的所有内容插入到源文件中，同时，内核中的头文件往往体积较大，因此引入不需要的依赖关系会极大地拖慢编译速度，还会造成中间文件较大占用过多空间。综上，减少依赖关系非常重要。
2. 内核中的依赖关系错综复杂，头文件数量众多，同时由于项目过大，很难使用常见的IDE功能（如查看定义等），因此靠肉眼检查依赖关系既慢又不准确
3. 现有的头文件检测工具（如 `iwyu`, `deheader`）对于Linux内核的适配程度不高，容易误报，容易漏掉某些依赖关系，同时检测速度较慢，不适合内核的检测

遇到的困难与解决方案

1. 多线程中shared variable有data race问题。解决方案：使用mutex或者atomic variable
2. 阅读代码缺少注释，同时代码量太大出现困难。解决方案：仅阅读happy path，忽略一些corner case的处理。同时配合文档阅读
3. 递归操作依赖树所需要的栈空间太大 解决方案：部分操作改为非递归
4. 对于内核中的一些特殊情况缺乏适配，导致工具检测错误，如一些少用的修饰符（`__always_inline` 等）。解决方案：增加错误处理模块，对于一些无法处理的模块或者明显错误的结果，继续执行，同时将相应的信息输出到错误日志中，方便日后对代码进行进一步的修正
5. 部分maintainer希望源文件用到的函数声明所在的头文件都被直接地 `#include`，而非通过间接的依赖关系引入。解决方案：修改代码，仅删去完全没有被使用过的头文件。截图如下

On Mon, 20 Sep 2021 21:42:00 +0800 Mianhan Liu wrote:

```
> sysctl_net_ipv4.c hasn't use any macro or function declared in mm.h, module.h,  
> igmp.h, inetdevice.h, swap.h, slab.h, nsproxy.h, snmp.h, route.h and inet_frag.h.  
> Thus, these files can be removed from sysctl_net_ipv4.c safely without affecting  
> the compilation of the net module.
```

How did you arrive at this conclusion? It certainly uses linux/slab.h as it calls `kmalloc()`. Please don't introduce dependencies on second-order includes. If file uses a function, define, struct etc. from a header it should directly include that header. Just because the code still compiles doesn't mean the change is for good.

Re: Re: [PATCH -next] net/ipv4/sysctl_net_ipv4.c: remove superfluous header files from sysctl_net_ipv4.c ★

liumh1@shanghaitech.edu.cn

发给 Jakub Kicinski, David S. Miller, Hideaki YOSHIFUJI, David Ahern, netdev@vger.kernel.org, linux-kernel@vger.kernel.org

2021-09-21 09:47 隐藏信息

发件人: liumh1@shanghaitech.edu.cn <liumh1@shanghaitech.edu.cn>

收件人: Jakub Kicinski <kuba@kernel.org>

抄送: David S. Miller <davem@davemloft.net>, Hideaki YOSHIFUJI <yoshifuji@linux-ipv6.org>, David Ahern <dsahern@kernel.org>, netdev@vger.kernel.org <netdev@vger.kernel.org>, linux-kernel@vger.kernel.org <linux-kernel@vger.kernel.org>

时间: 2021年9月21日 (周二) 09:47

大小: 9 KB

I am writing a headerfile tool which will try to remove the superfluous inclusion. The attempt to remove slab.h is that if you remove slab.h and preprocess the file with `make net/ipv4/sysctl_net_ipv4.i`, it turns out that `sysctl_net_ipv4.i` has already included slab.h indirectly. Thus, even `sysctl_net_ipv4.c` use `kmalloc` it can be removed safely. However, i agree with that if files use a declaration, it should directly include that header. Simply remove the header because it won't affect the compilation is unwise. I will take your advice, modify my tool and commit a new patch.Thank you.

其它信息

- 方案进度：已全部完成
- 项目完成质量：优秀
- 与导师沟通及反馈情况：积极沟通，平均一周开一次视频会议，日常经常在微信上沟通开发状况
- gitee地址：<https://gitee.com/openeuler-competition/summer2021-56>
- gitlab地址：<https://gitlab.summer-ospp.ac.cn/summer2021/210010078>