手把手教你如何向 Linux 内核提交代码

开源的力量

说到开源大家都会想到黑客和极客，开源的概念最早也是在极客们推出和推崇的。开源的提倡旨在开放源代码使之更方便自由的使用和再创作。随着这一思想的发展，衍生出诸多的开源协议，比如有GPL,BSD,MIT等。关于开源的一些故事推荐杜玉杰的 chat 文章《开源纵横谈：谷歌与开源那些事儿》。豪不夸张的说开源的传播已经在颠覆传统软件的开发模式，推动整个 IT 的进步，围绕着开源的社区文化也在这个新的时代发光发热。

值得关注的开源项目实在太多，相信每个软件行业每个软件模块都有自己的开源项目，今天不对开源话题进行交流探讨，我们选择 linux 的内核来手把手教你如何在内核社区提交自己的 patch，让你轻松迈入开源社区的第一步，找到高薪工作，从此走向人生巅峰，迎娶……。

在开源社区提交的每一笔代码都有可能对别人影响深远，所以要细心，细心，再细心。在提交代码之前端正态度很重要：

1. 细心
2. 声誉
3. 谦虚

你提交的每一笔代码都有可能给别人带来巨大的贡献或者潜在的风险，所以要细心对待你的每一笔 patch。开源社区圈子其实是比较小的，开源给你带来的价值不仅仅是代码本身，相信在工作中也会潜移默化的给你带去机会，所以你在开源社区的声誉就变的尤为重要，你的每一笔代码就像做事一样都体现着做人的本质。另外社区里的大牛很多，在你向 maintainer 提出问题的时候，要向学生请教老师问题一样抱着谦虚的心态，一般来讲社区里的 maintainer 态度都很好，人都很 nice。以上三点是我在开源社区的经验，希望对初学者有所帮助。

准备工作

工欲善其事，必先利其器。进入正题之前先准备下需要安装和配置的环境和工具。首先要安装 linux 操作系统，有很多发行版，比如 ubuntu,centos，看个人兴趣去选择，本人比较喜欢 ubuntu，这里主要以 ubuntu 操作系统为例，给大家演示准备工作。

1. 安装和配置 msmtp

点击左边栏的“软件中心”，在搜索框中输入“msmtp”，选择安装即可。然后按照如下步骤配置 msmtp：

```
root@-VirtualBox:~# cd ~/
root@-VirtualBox:~# vi .msmtp.log    //创建log文件，然后直接退出
root@-VirtualBox:~# vi .msmtprc       //键入如下内容

defaults
logfile ~/.msmtp.log
# gitchat
account gitchat
protocol smtp
host smtp.gitchat.com
from peter.liu@gitchat.com
user peter.liu@gitchat.com
password xxx  //由于single密码3个月一换，注意同步更新此处
port 25
auth plain
tls off
#tls_starttls on
#tls_trust_file /etc/ssl/certs/ca-certificates.crt
#syslog LOG_MAIL
# Set a default account
account default : gitchat
```


2. 安装和配置git环境

默认的 linux 系统一般都已经安装好 git。如果没有，随便找一本git的书都可以，这里不详述。比较好的git资料有：http://git.oschina.net/progit/

在配置用户名的时候，请注意社区朋友习惯用英语沟通，也就是名在前，姓在后。这一点会影响社区邮件讨论，因此需要留意。在配置邮箱时，也要注意。社区会将国内某些著名的邮件服务器屏蔽。因此建议你申请一个gmail邮箱。举例配置如下：

```
git config --global user.name 'Peter Liu'
$git config --global user.email ' peter.liu@gitchat.com'
$git config --global sendemail.chainreplyto false
$git config --global sendemail.smtpserver /usr/bin/msmtp
查看配置结果的话如下：
cat ~/.gitconfig
[user]
    name = Changbing Xiong
    email = peter.liu@gitchat.com
[color]
    ui = auto
[core]
    editor = vim
[sendemail]
```

```
chainreplyto = false
smtpserver = /usr/bin/msmtp
```

3. 订阅mailinglist

Linux 开源分支要求开发者上传 patch 或者 driver 时，需要将邮件抄送给mailinglist、maintainer 和其他人，首先需要订阅相关子系统的 mailinglist。

订阅邮件列表 mailing list：//改成自己所要提交代码所在子系统的 mailing list，详见 linux 代码根目录下的 MAINTAINERS 文档

```
To: majordomo@vger.kernel.org
邮件内容：
subscribe linux-media
取消订阅邮件列表mailing list：
To: majordomo@vger.kernel.org
邮件内容：
unsubscribe linux-media
```

注意：订阅mailinglist的邮件不需要标题，请参考如下方式：

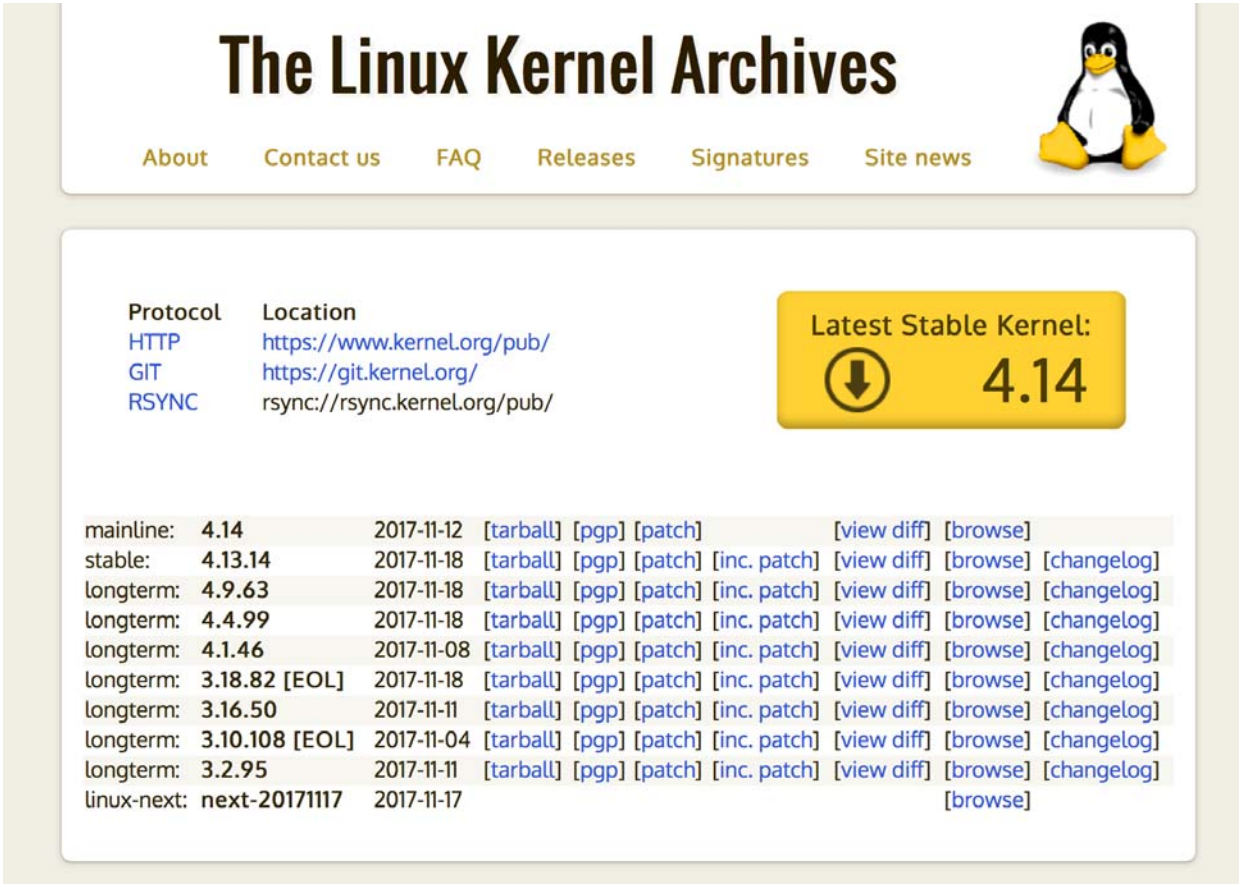

```
# vi subscrible
Subject: //冒号后面保留一个空格
//必须空一行，并且该行不要有空格
subscribe linux-media  //在这里加个回车，然后退出并保存文件

#git send-email --from peter.liu@gitchat.com --to
    peter.liu@gitchat.com ./subscrible  //验证一下
#git send-email --from peter.liu@gitchat.com --to
    majordomo@vger.kernel.org ./subscrible
```

注意：发送完subscribe邮件后，你会收到一封确认邮件，比如我的确认邮件标题为“Confirmationfor subscribe linux-media”，里面有认证信息，请按照邮件内容，再发一个认证邮件给majordomo@vger.kernel.org，如下：

```
# vi subscrible_auth
Subject:
auth xxxxxxxx subscribe linux-media peter.liu@gitchat.com
# git send-email --from peter.liu@gitchat.com --to
majordomo@vger.kernel.org ./subscrible_auth
```

4. 下载 linux 源代码

首先打开以下网页选取你想要工作的分支：https://www.kernel.org



从下载的代码里选取感兴趣的模块，你可以在内核源码目录\MAINTAINERS文件中，找一下相应文件的维护者，及其git地址。例如，watchdog模块的信息如下：


```
WATCHDOGDEVICE DRIVERS
  M:      Wim Van Sebroeck <wim@iguana.be>
  R:      Guenter Roeck <linux@roeck-us.net>
  L:      linux-watchdog@vger.kernel.org
  W:      http://www.linux-watchdog.org/
  T:      gitgit://www.linux-watchdog.org/linux-watchdog.git
  S:      Maintained
  F:      Documentation/devicetree/bindings/watchdog/
  F:      Documentation/watchdog/
  F:      drivers/watchdog/
  F:      include/linux/watchdog.h
  F:      include/uapi/linux/watchdog.h
```

其中，git://www.linux-watchdog.org/linux-watchdog.git是其git地址。你可以用如下命令拉取watchdog代码到本地：

```
gitremote add watchdog git://www.linux-watchdog.org/linux-watchdog.git
gitfetch --tags watchdog
```

当然，这里友情提醒一下，MAINTAINERS里面的信息可能不一定准确，这时候你可能需要借助google，或者问一下社区的朋友，或者直接问一下作者本人。不过，一般情况下，基于linux-next分支工作不会有太大的问题。实在有问题再去打扰作者本人。

5. 阅读Documentation/SubmittingPatches，这很重要。

制作补丁

1. 补丁描述

补丁第一行是标题，比较重要。它首先应当是模块名称。

举个例子，怎么找到drivers/clk/samsung/clk-s3c2412.c文件属于哪个模块？可以试试下面这个命令，看看drivers/clk/samsung/clk-s3c2412.c文件的历史补丁：

```
# git log drivers/clk/samsung/clk-s3c2412.c
commit 02c952c8f95fd0adf1835704db95215f57cfc8e6
Author:Martin Kaiser <martin@kaiser.cx>
Date:   Wed Jan 25 22:42:25 2017 +0100
clk: samsung:mark s3c...._clk_sleep_init() as __init
```

可以看出模块名称是“clk:samsung”。下面是我为这个补丁添加的描述，其中第一行是标题：


```
clk: samsung: mark symbols static where possible for s3c2410
We get 1 warnings when building kernel withW=1:
/dimsum/git/kernel.next/drivers/clk/samsung/clk-
s3c2410.c:363:13:warning: no previous prototype for
's3c2410_common_clk_init'[-Wmissing-prototypes]
 void __init s3c2410_common_clk_init(struct device_node *np, unsigned
long xti_f,

In fact, this function is only used in thefile in which they are
declared and don't need a declaration, butcan be made static.
So this patch marks these functions with 'static'.
```

这段描述是我从其他补丁中拷贝出来的，有几下几点需要注意：首先标题中故意添加了“for s3c2410”，以区别于另外两个补丁。其次“1 warnings”这个单词中，错误的使用了复数，这是因为复制的原因。

接着“/dimsum/git/kernel.next/”这个路径名与我的本地路径相关，不应当出现在补丁中。最后警告描述超过了80个字符，但是这是一个特例，这里允许超过80字符。

这些问题，如果不处理的话，Maintainer会不高兴的！如果Maintainer表示了不满，而你不修正的话，这个补丁就会被忽略。修正后的补丁描述如下：

```
clk: samsung: mark symbols static wherepossible for s3c2410
We get 1 warning when building kernel withW=1:
drivers/clk/samsung/clk-s3c2410.c:363:13:warning: no previous prototype
```

```
for 's3c2410_common_clk_init'[-Wmissing-prototypes]
void__init s3c2410_common_clk_init(struct device_node *np, unsigned long
xti_f,
In fact, this function is only used in thefile in which they are
declared and don't need a declaration, but can be made static.
So this patch marks these functions with 'static'.
```

我们的补丁描述一定要注意用词，不要出现将“unused”写为“no used”这样的错误。反复使用git add，git commit将补丁提交到git仓库。

2. 如何生成补丁

有很多的场景根据不同需求生成补丁，这里介绍两种工作中常用遇到的场景：

```
# git format-patch HEAD^
0001-au0828-fix-logic-of-tuner-disconnection.patch
# cat 0001-au0828-fix-logic-of-tuner-disconnection.patch
From cc4f6646ae5eb0d75d56cca62e2d60c1ac8cad66 Mon Sep 17 00:00:00 2001
From: Changbing Xiong <cb.xiong@samsung.com>
Date: Tue, 22 Apr 2014 16:10:29 +0800
Subject: [PATCH] au0828: fix logic of tuner disconnection //此处的
[PATCH]是工具自动加上的

The driver crashed when the tuner was disconnected while restart stream
。。。。。。。
restart stream operations has been released.

Change-Id: Iaa1b93f4d5b08652921069182cdd682aba151dbf //需要通过vim删除此行
Signed-off-by: peter liu <peter.liu@gitchat.com>
---
 drivers/media/usb/au0828/au0828-dvb.c |   13 ++++++++++++
。。。。。。。
```

上面是将最近一次的修改生成一个 patch，不过注意，如果 patch 中有 Change-Id 行，需要删除；

3. 检查补丁

在发送补丁前，我们需要用脚本检查一下补丁：

```
./scripts/checkpatch.pl 000*
----------------------------------------
0001-clk-samsung-mark-symbols-static-where-possible-for-s.patch
----------------------------------------
WARNING: Possible unwrapped commit description (prefer a maximum 75
chars per line)
#9:
```

```
 void__init s3c2410_common_clk_init(struct device_node *np, unsigned
long xti_f,

WARNING: line over 80 characters
#29: FILE:drivers/clk/samsung/clk-s3c2410.c:363:
+static void __init s3c2410_common_clk_init(struct device_node *np,
unsigned long xti_f,

total: 0 errors, 2 warnings, 8 lineschecked
```

请留意输出警告，其中第一个警告是说我们的描述中，有过长的语句。前面已经提到，这个警告可以忽略。但是第二个警告告诉我们代码行超过80个字符了。这是不能忽略的警告，必须处理。

4. 补丁的提交

在第一次commit时使用-s，后面修改下面内容时，用—amend即可。

```
au0828: fix logic of tuner disconnection                          //标题，带上
模块名称，如au0828

//此处必须空一行
The driver crashed when the tuner was disconnected while restart stream
operations are still being performed. Fixed by adding a flag in struct
au0828_dvb to indicate whether restart stream operations can be
performed.

If the stream gets misaligned, the work of restart stream operations are
 usually scheduled for many times in a row. If tuner is disconnected at
this time and some of restart stream operations are still not flushed,
then the driver crashed due to accessing the resource which used in
restart stream operations has been released.

//此处必须空一行
Signed-off-by: Changbing Xiong <cb.xiong@samsung.com>    //通过git
commit -s 自动生成

# Please enter the commit message for your changes. Lines starting  //从
这往下都是自动生成，勿动
# with '#' will be ignored, and an empty message aborts the commit.
# On branch tizen
# Your branch is ahead of 'origin/tizen' by 1 commit.
#
# Changes to be committed:
#   (use "git reset HEAD^1 <file>..." to unstage)
#
# modified:   drivers/media/usb/au0828/au0828-dvb.c
# modified:   drivers/media/usb/au0828/au0828.h
```

5. 发送补丁

生成正确的补丁后，请再次用checkpatch.pl检查补丁正确性。确保无误后，可以准备将它发送给Maintainer了。但是应该将补丁发给谁？这可以用get_maintainer.pl来查看：

```
#./scripts/get_maintainer.pl 000*
Kukjin Kim <kgene@kernel.org>(maintainer:ARM/SAMSUNG EXYNOS ARM
ARCHITECTURES)
Krzysztof Kozlowski <krzk@kernel.org>(maintainer:ARM/SAMSUNG EXYNOS ARM
ARCHITECTURES)
Sylwester Nawrocki<s.nawrocki@samsung.com> (supporter:SAMSUNG SOC CLOCK
DRIVERS)
Tomasz Figa <tomasz.figa@gmail.com>(supporter:SAMSUNG SOC CLOCK DRIVERS)
Chanwoo Choi <cw00.choi@samsung.com>(supporter:SAMSUNG SOC CLOCK
DRIVERS)
Michael Turquette<mturquette@baylibre.com> (maintainer:COMMON CLK
FRAMEWORK)
Stephen Boyd <sboyd@codeaurora.org>(maintainer:COMMON CLK FRAMEWORK)
linux-arm-kernel@lists.infradead.org(moderated list:ARM/SAMSUNG EXYNOS
ARM ARCHITECTURES)
linux-samsung-soc@vger.kernel.org (moderatedlist:ARM/SAMSUNG EXYNOS ARM
ARCHITECTURES)
linux-clk@vger.kernel.org (open list:COMMONCLK FRAMEWORK)
linux-kernel@vger.kernel.org (open list)
```


接下来，可以用git send-email命令发送补丁了：

```
git send-email 000* --
tokgene@kernel.org,krzk@kernel.org,s.nawrocki@samsung.com,tomasz.figa@gm
ail.com,cw00.choi@samsung.com,mturquette@baylibre.com,sboyd@codeaurora.o
rg--cc linux-arm-kernel@lists.infradead.org,linux-samsung-
soc@vger.kernel.org,linux-clk@vger.kernel.org,linux-
kernel@vger.kernel.org
```

注意哪些人应当作为邮件接收者，哪些人应当作为抄送者。在本例中，补丁是属于实验性质的，可以不抄送给邮件列表帐户。提醒：你应当将补丁先发给自己，检查无误后再发出去。如果你有朋友在社区有较高的威望，也可以抄送给他，必要的时候，也许他能给你一些帮助。这有助于将补丁顺利的合入社区。重要提醒：本文讲述的，主要是实验性质的补丁，用于打开社区大门。真正重要的补丁，可能需要经过反复修改，才能合入社区。

工具介绍

本文重在讲述向内核提交代码的方法论，主要是实验性质的补丁，用于打开社区大门。真正重要的补丁，可能需要经过反复修改，才能合入社区。最后给大家介绍下常用工具的链接：

- https://www.kernel.org/pub/software/scm/git/docs/git-send-email.html
- https://www.kernel.org/pub/software/scm/git/docs/git-format-patch.html