



会员

周边

众包

新闻

博问

闪存

赞助商

Chat2DB

代码改变世界



注册 登录

云水

MyGitee - <https://gitee.com/lsgx/>
MyGithub - <https://github.com/lsgxeva/>
Download - <https://d.serctl.com>
Proxy - <tcp://frpgz1.idcfengye.com:10000>

博客园

首页

新随笔

联系

订阅

管理

随笔 - 1779 文章 - 0 评论 - 145 阅读 - 615万

使用 GitHub Actions 云编译 OpenWrt

使用 GitHub Actions 云编译 OpenWrt

来源 <https://p3terx.com/archives/build-openwrt-with-github-actions.html>

前言

Github Actions 是 Microsoft 收购 GitHub 后推出的 CI/CD 服务，它提供了性能配置非常不错的虚拟服务器环境（E5 2vCPU/7G RAM），基于它可以进行构建、测试、打包、部署项目。对于公开仓库可免费无时间限制的使用，且单次使用时长长达 6 个小时，这对于编译 OpenWrt 来说是非常充足的。不过 GitHub Actions 有一定的使用门槛，首先要了解如何编写 workflow 文件。不过不用担心，博主已经编写好了相关的 workflow 文件模版，只需要按照教程的步骤来操作即可。

教程更新

- 2020-04-25 更新 DIY 脚本说明、添加自定义 feeds 配置文件说明
- 2020-04-09 新增上传固件到 WeTransfer
- 2020-03-30 新增上传固件到奶牛快传
- 2020-02-01 新图文教程
- 2019-12-10 新增 macOS 编译方案使用说明
- 2019-12-06 添加 tmate 网页终端链接说明
- 2019-12-05 优化基础使用教程，添加 @lietxia 大佬的图文教程链接
- 2019-12-04 新增云menuconfig使用方法
- 2019-12-03 新增并发编译使用方法
- 2019-11-30 新增自定义源码编译使用方法
- 2019-11-14 全网独家首发

方案特点

- 免费
- 一键快速编译
- 定时自动编译
- 客制化编译
- 并发编译（可同时进行20+5个编译任务）
- 无需搭建编译环境（在线 `make menuconfig` 生成配置文件）

公告

昵称: lsgxeva
园龄: 9年4个月
粉丝: 405
关注: 2
[+加关注](#)

2025年3月						
日	一	二	三	四	五	六
23	24	25	26	27	28	1
2	3	4	5	6	7	8
9	10	11	12	13	14	15
16	17	18	19	20	21	22
23	24	25	26	27	28	29
30	31	1	2	3	4	5

搜索

🔍

常用链接

[我的随笔](#)
[我的评论](#)
[我的参与](#)
[最新评论](#)
[我的标签](#)

随笔分类

[a10\(4\)](#)
[AI\(2\)](#)
[algorithm\(1\)](#)
[Android\(41\)](#)
[ans\(43\)](#)
[bitcoin\(4\)](#)
[c++\(111\)](#)
[c++11\(39\)](#)
[ccna\(50\)](#)
[cmake\(4\)](#)
[cnns\(17\)](#)
[cocos\(1\)](#)
[CSharp\(29\)](#)
[DataStructure\(6\)](#)
[DesignPattern\(39\)](#)
[更多](#)

- 无需消耗自己的计算机与服务器的计算资源（性感E5在线编译）
- 无需担心磁盘空间不足（近60G磁盘空间）
- 无需使用清理文件（内核更新不怕 boom）
- 编译速度快（编译时间1-2小时）
- 编译成功率提升200%（万兆自由网络环境）
- 全新环境（杜绝编译环境不干净导致编译失败）

本解决方案是一个开放平台，任何人都可以基于此打造自己专属的编译方案。

项目地址

https://github.com/P3TERX/Actions-OpenWrt

支持项目请随手点个 star，让更多的人发现、使用并受益。

准备工作

- 注册 GitHub 账号
- 搭建编译环境，用于生成 .config 文件。（可选）

TIPS: 关于编译环境的搭建，推荐去看我之前写的相关文章，Windows 10 可以使用 WSL，macOS、Linux 可以使用 Docker。

基础使用

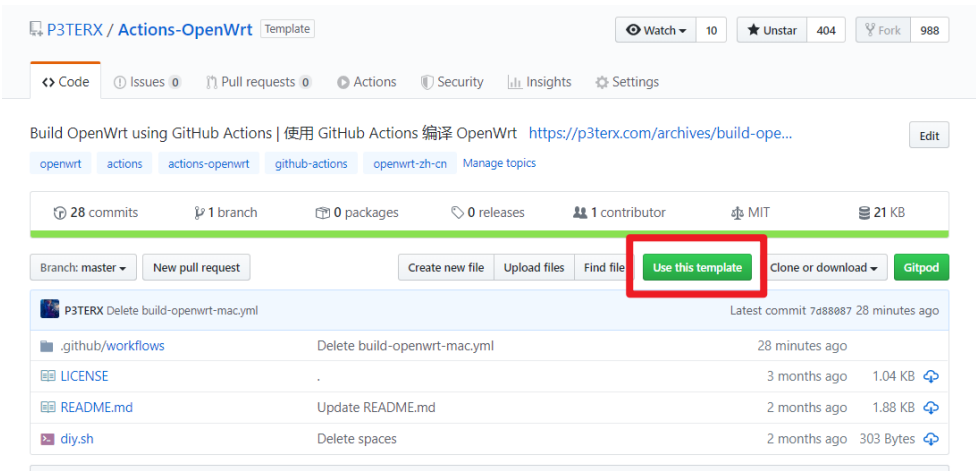
首先你必须要熟悉整个 OpenWrt 的编译过程，这会让你非常容易的理解如何使用 GitHub Actions 进行编译，即使你没有成功过。因为实际上本地编译近 90% 失败的原因是因为网络问题导致的，中国大陆特色，咱也不敢多说。而使用 GitHub Actions 编译成功率至少提升 200%，为什么这样说呢？因为 Actions 服务器由 Microsoft Azure 提供，在自由的美利坚，拥有万兆带宽。

首次编译

- 在自己搭建编译环境中使用 Lean's OpenWrt 源码生成 .config 文件。（或使用后面进阶玩法中的云 menuconfig，直接 SSH 到 Actions 进行操作）

TIPS: 方案默认引用 Lean 的源码，因为他的 README 影响了我开始学习编译，也就有了这个项目，而且他的源码非常的优秀。有其它需求可自行修改 workflow 文件，方法后面的进阶使用中有说明。

- 进入 P3TERX/Actions-OpenWrt 项目页面，点击页面中的 Use this template（使用这个模版）按钮。



- 填写仓库名称，然后点击 Create repository from template（从模板创建储存库）按钮。

随笔档案

- 2025年3月(5)
- 2025年2月(3)
- 2025年1月(5)
- 2024年12月(9)
- 2024年11月(4)
- 2024年10月(4)
- 2024年9月(2)
- 2024年7月(1)
- 2024年6月(2)
- 2024年5月(5)
- 2024年4月(3)
- 2024年3月(10)
- 2024年2月(11)
- 2024年1月(5)
- 2023年12月(8)
- 更多

阅读排行榜

- 1. Docker 创建镜像、修改、上传镜像(239417)
- 2. Git Submodule使用完整教程(140211)
- 3. IPV6地址格式分析(67095)
- 4. Windows WMIC命令使用详解(附实例)(64585)
- 5. Qt基本控件及三大布局(62592)

评论排行榜

- 1. Git Submodule使用完整教程(10)
- 2. c++11 类默认函数的控制: "=default" 和 "=delete"函数(7)
- 3. GCC 中 -L、-rpath和-rpath-link的区别(6)
- 4. babel从入门到入门(6)
- 5. libuv 简单使用(5)

推荐排行榜

- 1. Git Submodule使用完整教程(17)
- 2. Docker 创建镜像、修改、上传镜像(10)
- 3. javaee, javaweb和javase的区别以及各自的知识体系(9)
- 4. c++11 类默认函数的控制: "=default" 和 "=delete"函数(9)
- 5. 关于qt中的tr () 函数(7)

最新评论

- 1. Re:DocGuarder
如何联系到博主啊，有关于docguarder方在的问题请教
--leocheng
- 2. Re:GNU Emacs命令速查表
多谢
--苦旅人生
- 3. Re:卫星转发器资源选择的考虑
太棒了，请问如果我想了解更多，有什么途径吗？
--何加一
- 4. Re:Nftables - 数据包流和 Netfilter 钩子
这是哪里的机翻文章
--一只青皮袖
- 5. Re:IP协议之TOS字段说明
不错
--昨夜寒蝉不住鸣

Create a new repository from Actions-OpenWrt

The new repository will start with the same files and folders as [P3TERX/Actions-OpenWrt](#).

Owner

P3TERX

Repository name *

OpenWrt-x86_64-firmware

Great repository names are short and memorable. Need inspiration? How about [psychic-train](#)?

Description (optional)

☒ Public

Anyone can see this repository. You choose who can commit.

☐ Private

You choose who can see and commit to this repository.

Create repository from template

1

2

- 经过几秒钟的等待，页面会跳转到新建的仓库，内容和我的项目是相同的。然后点击 [Create new file](#)（创建新文件）按钮。

P3TERX / OpenWrt-x86_64-firmware

generated from P3TERX/Actions-OpenWrt

Unwatch

1

Star

0

Fork

0

Code

Issues

Pull requests

Actions

Projects

Wiki

Security

Insights

Settings

No description, website, or topics provided.

Manage topics

1 commit

1 branch

0 packages

0 releases

1 contributor

MIT

0 Bytes

Branch: master

New pull request

Create new file

Upload files

Find file

Clone or download

Gitpod

P3TERX Initial commit

Latest commit 4927eFc 1 minute ago

.github/workflows

Initial commit

1 minute ago

LICENSE

Initial commit

1 minute ago

1.04 KB

README.md

Initial commit

1 minute ago

1.88 KB

diy.sh

Initial commit

1 minute ago

303 Bytes

README.md

- 文件名填写为 `.config`，把生成的 `.config` 文件的内容复制粘贴到下面的文本框中。

P3TERX / OpenWrt-x86_64-firmware

generated from P3TERX/Actions-OpenWrt

Unwatch

1

Star

0

Fork

0

Code

Issues

Pull requests

Actions

Projects

Wiki

Security

Insights

Settings

OpenWrt-x86_64-firmware

.config

Cancel

Edit new file

Preview

Spaces

2

No wrap

1

CONFIG_TARGET_x86=y

2

CONFIG_TARGET_x86_64=y

3

CONFIG_TARGET_x86_64_Generic=y

4

CONFIG_ARIA2_ASYNC_DNS=y

5

CONFIG_ARIA2_BITTORRENT=y

6

CONFIG_ARIA2_COOKIE=y

7

CONFIG_ARIA2_LIBXML2=y

8

CONFIG_ARIA2_METALINK=y

9

CONFIG_ARIA2_OPENSSL=y

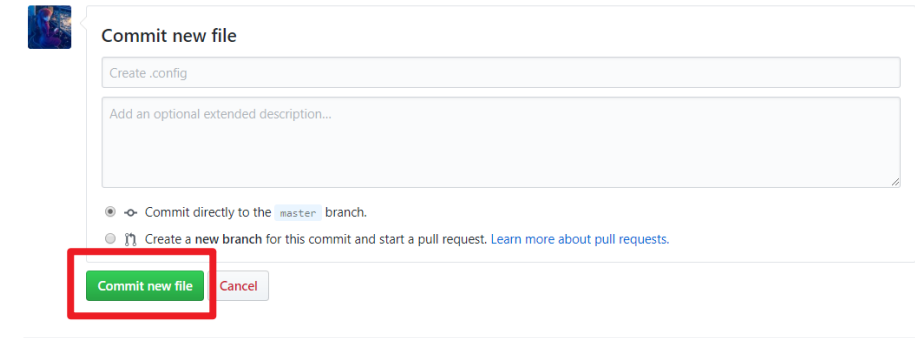
10

CONFIG_ARIA2_SFTP=y

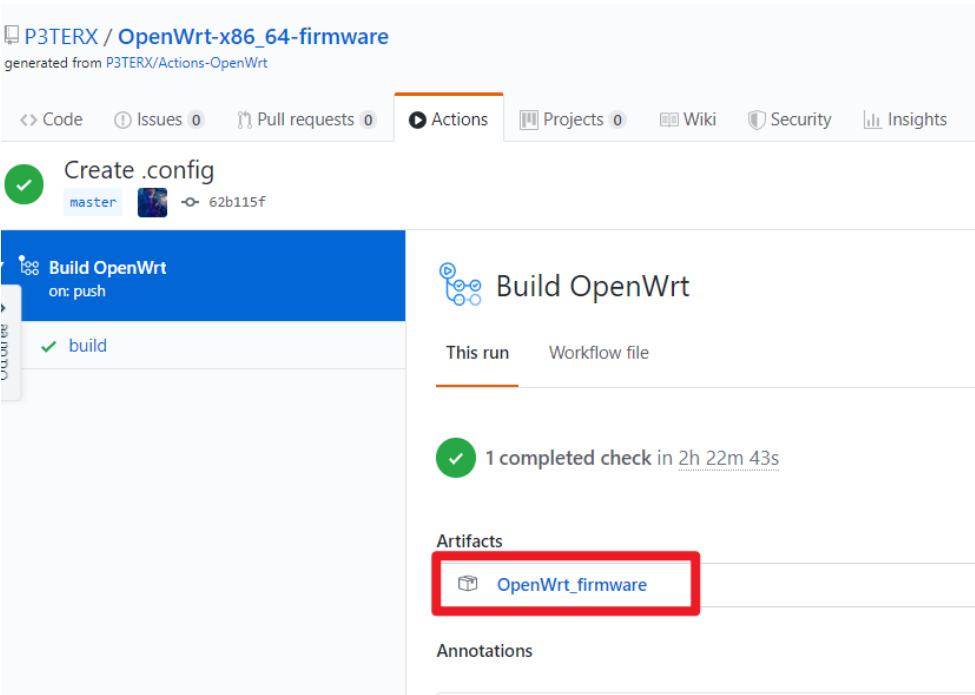
11

CONFIG_ARIA2_WEBSOCKET=y

- 翻到页面最下方，点击 [Commit new file](#)（提交新文件）按钮即可。后续编译工作会自动开始，你可以在 [Actions](#) 页面进行查看。



- 在等待编译完成的过程中，你可以进入[这个页面](#)点击右上角的 `star`，这是对博主最大的支持，而且还可以加快编译速度哦（雾
- 最后经过一两个小时的等待，不出意外你就可以在 Actions 页面看到已经打包好的固件目录压缩包。



TIPS: 如需 ipk 文件可以在进阶使用章节找到方法。因为大多数人只需要固件，而且总是有萌新问固件在哪，所以现在默认只上传固件。

再次编译

默认情况下触发编译工作流程有两种方式：

- 发布 release
- 修改 `.config` 文件

他们分别对应以下使用场景：

- 在编译配置没有修改的情况下，你发现大佬的仓库源码有更新，那么在 releases 页面发布一个 release 将直接触发编译的工作流程，使用最新源码进行编译。
- 如果你想修改配置，则生成船新的 `.config` 文件 push 到仓库来触发编译的工作流程。

其它触发方式你可以在后面的进阶使用中看到。

进阶使用

自定义环境变量与功能

打开 workflow 文件（`.github/workflows/build-openwrt.yml`），你会看到有如下一些环境变量，可按照自己的需求对这些变量进行定义。

```
env:
  REPO_URL: https://github.com/coolnowolf/lede
  REPO_BRANCH: master
  FEEDS_CONF: feeds.conf.default
```

```
CONFIG_FILE: .config
DIY_P1_SH: diy-part1.sh
DIY_P2_SH: diy-part2.sh
SSH_ACTIONS: false
UPLOAD_BIN_DIR: false
UPLOAD_FIRMWARE: true
UPLOAD_COWTRANSFER: false
UPLOAD_WETRANSFER: false
TZ: Asia/Shanghai
```

TIPS: 修改时需要注意 **:** (冒号)后面有空格。

环境变量	功能
REPO_URL	源码仓库地址
REPO_BRANCH	源码分支
FEEDS_CONF	自定义feeds.conf.default文件名
CONFIG_FILE	自定义.config文件名
DIY_P1_SH	自定义diy-part1.sh文件名
DIY_P2_SH	自定义diy-part2.sh文件名
SSH_ACTIONS	SSH 连接 Actions 功能。默认false
UPLOAD_BIN_DIR	上传 bin 目录。即包含所有 ipk 文件和固件的目录。默认false
UPLOAD_FIRMWARE	上传固件目录。默认true
UPLOAD_COWTRANSFER	上传固件到奶牛快传。默认false
UPLOAD_WERANSFER	上传固件到 WeTransfer 。默认false
TZ	时区设置

DIY 脚本

仓库根目录目前有两个 DIY 脚本：**diy-part1.sh** 和 **diy-part2.sh**，它们分别在更新与安装 feeds 的前后执行，你可以把对源码修改的指令写到脚本中，比如修改默认 IP、主机名、主题、添加 / 删除软件包等操作。但不仅限于这些操作，发挥你强大的想象力，可做出更强大的功能。

TIPS: 脚本工作目录在源码目录，内附几个简单的例子供参考。

添加额外的软件包

- 在 DIY 脚本中加入对指定软件包源码的远程仓库的克隆指令。就像下面这样：

```
git clone https://github.com/P3TERX/xxx_package/xxx
```

- 本地 **make menuconfig** 生成 **.config** 文件时添加相应的软件包，如果你知道包名可以直接写到 **.config** 文件中。

TIPS: 如果额外添加的软件包与 OpenWrt 源码中已有的软件包同名的情况，则需要把 OpenWrt 源码中的同名软件包删除，否则会优先编译 OpenWrt 中的软件包。这同样可以利用到的 DIY 脚本，相关指令应写在 **diy-part2.sh**。

原理是把软件包源码放到 **package** 目录下，编译时会自动遍历，与本地编译是一样的。当然方法不止一种，其它方式请自行探索。

自定义 feeds 配置文件

把 **feeds.conf.default** 文件放入仓库根目录即可，它会覆盖 OpenWrt 源码目录下的相关文件。

Custom files（自定义文件）

俗称“files 大法”，在仓库根目录下新建 **files** 目录，把相关文件放入即可。有关详情请自行搜索了解。

定时自动编译

编辑 workflow 文件 (`.github/workflows/build-openwrt.yml`) 取消注释下面两行。

```
# schedule:
#   - cron: 0 8 * * 5
```

例子是北京时间每周五下午 4 点 (16 时) 开始编译 (周末下班回家直接下载最新固件开始折腾)。如需自定义则按照 cron 格式修改即可, GitHub Actions 的时区为 UTC, 注意按照自己所在地时区进行转换。

真·一键编译 (点击 star 开始编译)

点击自己仓库页面上的 Star 按钮开始编译, 为了防止产生垃圾记录, 所以这个功能默认没有开启。

编辑 workflow 文件 (`.github/workflows/build-openwrt.yml`) 取消注释下面两行, 后续点击自己仓库上的 star 即可开始编译。

```
# watch:
#   types: started
```

TIPS: 字段 `started` 并不是“开始了”的意思, 而是“已经点击 Star”。

吐槽: 官方并没有提供一个开始按钮, 通过搜索找到过很多奇怪的一键触发方式, 但都是通过 Webhook 来实现的。机智的我发现了可以通过点击 Star 来触发, 这样就相当于把 Star 当成开始按钮。这个 `started` 有种一句双关的意思了。

自定义源码编译

此方案默认引用的是 Lean 的源码, 如果你有编译其它源码的需求可以进行替换, 自由是本解决方案最大的特点。

编辑 workflow 文件 (`.github/workflows/build-openwrt.yml`), 修改下面的相关环境变量字段。

```
REPO_URL: https://github.com/coolsnowwolf/lede
REPO_BRANCH: master
```

比如修改为 OpenWrt 官方源码 19.07 分支

```
REPO_URL: https://github.com/openwrt/openwrt
REPO_BRANCH: openwrt-19.07
```

TIPS: 注意冒号后面有空格

并发编译 (同时编译多个固件)

多 repository 方案

通过 [P3TERX/Actions-OpenWrt](#) 项目创建多个仓库来编译不同架构机型的 OpenWrt 固件。

多 workflow 方案

基于 GitHub Actions 可同时运行多个工作流程的特性, 最多可以同时进行至少 20 个编译任务。也可以单独选择其中一个进行编译, 这充分的利用到了 GitHub Actions 为每个账户免费提供的 20 个 Ubuntu 虚拟服务器环境。此外你还可以额外再使用 5 个 macOS 虚拟服务器环境进行编译, 开启方法在后面有说明。

假设有三台路由器的固件需要编译, 比如 K2P、x86_64 软路由、新路由 3。

- 生成它们的 `.config` 文件
- 分别将它们重命名为 `k2p.config`、`x64.config`、`d2.config` 放入本地仓库根目录。
- 复制多个 workflow 文件 (`.github/workflows/build-openwrt.yml`)。为了更好的区分可以对它进行重命名, 比如 `k2p.yml`、`x64.yml`、`d2.yml`。此外第一行 `name` 字段也可以进行相应的修改。
- 然后分别用上面修改的文件名替换对应 workflow 文件中下面两个位置的 `.config`, 不同的机型同样可以使用不同的 DIY 脚本。

```
...
paths:
```

```
- '.config'
...
    CONFIG_FILE: '.config'
    DIY_SH: 'diy.sh'
...
```

- 最后 push，此时这就触发了3个并行的编译工作流程。

云 menuconfig (SSH 连接到 Actions)

通过 tmate 连接到 GitHub Actions 虚拟服务器环境，可直接进行 `make menuconfig` 操作生成编译配置，或者任意的客制化操作。也就是说，你不需要再自己搭建编译环境了。这可能改变之前所有使用 GitHub Actions 的编译 OpenWrt 方式。

- 编辑 workflow 文件（`.github/workflows/build-openwrt.yml`），修改 `SSH_ACTIONS` 环境变量的值为 `true`。（或者也可以不修改，而是通过 `webhook` 方式发送带有 `ssh` 触发关键词的请求。）

```
SSH_ACTIONS: true
```

- 在触发工作流程后，在 Actions 页面等待执行到 `SSH connection to Actions` 步骤，会出现下面的信息。

```
To connect to this session copy-n-paste the following into a terminal or browser:

ssh Y26QeagDtsPXp2mT6me5cnMRd@nyc1.tmate.io

https://tmate.io/t/Y26QeagDtsPXp2mT6me5cnMRd
```

- 复制 SSH 连接命令粘贴到终端内执行，或者复制链接在浏览器中打开使用网页终端。（网页终端可能会遇到黑屏的情况，按 `Ctrl` + `C` 即可）
- `cd openwrt && make menuconfig`
- 完成后按快捷键 `Ctrl+D` 或执行 `exit` 命令退出，后续编译工作将自动进行。

TIPS: 固件目录下有个 `config.seed` 文件，如果你需要再次编译可以使用它。

WARNING: 默认连接30分钟后会断开并终止编译工作流程，防止资源浪费与封号风险。如果你想解除这个限制，可以根据提示操作，但导致的一切后果请自行承担。

macOS 虚拟机编译方案

GitHub Actions 的 macOS 虚拟机性能要高于 Ubuntu 虚拟机，所以使用它编译 OpenWrt 理论上速度会更快。博主经过几天时间的研究已经总结出了 [macOS 下的 OpenWrt 编译环境的搭建方法](#)，并编写出了适用于 macOS 虚拟环境的 OpenWrt 编译方案的 workflow 文件。

由于极少有开发者会考虑兼容 macOS 下的规范，所以使用 macOS 编译 OpenWrt 不可避免的会遇到非常多的问题，甚至 OpenWrt 官方源码也是。而且后续测试发现 macOS 虚拟机性能已大幅下降，故相关 workflow 文件已经移除。也不建议任何人使用 macOS 编译 OpenWrt。

上传固件到奶牛快传

[奶牛快传](#)是中国大陆的一款临时文件传输分享服务网盘，特点是不限速。因国情所致，中国大陆地区 GitHub 访问速度缓慢，有些小伙伴可能无法正常下载固件，上传固件到奶牛快传是个非常好的选择。

- 编辑 workflow 文件（`.github/workflows/build-openwrt.yml`），将环境变量 `UPLOAD_COWTRANSFER` 的值修改为 `true`：

```
UPLOAD_COWTRANSFER: true
```

编译完成后你可以在 `Upload firmware to cowtransfer` 步骤的日志中找到下载链接。

CLI 上传工具来自 [Mikubill/transfer](#)，特此感谢。

上传固件到 WeTransfer

[WeTransfer](#) 是荷兰的一款临时文件传输分享服务网盘，前面提到的奶牛快传实际上师从自它，二者的网站都非常相似。WeTransfer 使用的是 Amazon S3 存储并通过 Amazon CloudFront CDN 全球加速，它在中国大陆的下载体验完全不输奶牛快传，甚至某些情况下要更好。

- 编辑 workflow 文件（`.github/workflows/build-openwrt.yml`），将环境变量 `UPLOAD_WERANSFER` 的值修改为 `true`：

```
UPLOAD_WERANSFER: true
```

编译完成后你可以在 `Upload firmware to WeTransfer` 步骤的日志中找到下载链接。

CLI 上传工具来自 [Mikubill/transfer](#)，特此感谢。

上传固件到 release

不建议任何人发布并上传到 release，因为 release 的文件是永久保存的，日积月累会给 GitHub 带来很多大的存储空间浪费。故不直接方法不提供方法，有能力且有正常需求的小伙伴请自行研究学习，相关问题不会解答。

题外话：几年前博主曾亲眼目睹著名 Android 开源项目 Open GApps 正常发布但过于频繁导致直接封号。前一秒还在下载，后一秒仓库就没了。

尾巴

希望大家合理使用免费的资源，必要时再编译，过度占用资源虽然使用者不会得到任何实质性的惩罚，但会为中国抹黑，造成很多国际争端。只有让开发者来充分利用才能产生更多更好的软件，这样大家才能受益。最后感谢 Microsoft 为我们提供 GitHub Actions 这样强大的工具。

===== End


分类: `openwrt`

好文要顶

关注我

收藏该文

微信分享



lsgxeva

粉丝 - 405 关注 - 2

+加关注

1

推荐

0


反对

升级成为会员

« 上一篇：[OpenWrt 编译步骤与命令详解](#)
» 下一篇：[OpenWrt 介绍](#)

posted @ 2020-09-28 07:43 lsgxeva 阅读(15862) 评论(0) 编辑 收藏 举报

[刷新页面](#) [返回顶部](#)

 登录后才能查看或发表评论，立即 [登录](#) 或者 [逛逛](#) 博客园首页

- 【推荐】还在用 ECharts 开发大屏？试试这款永久免费的开源 BI 工具！
- 【推荐】国内首个AI IDE，深度理解中文开发场景，立即下载体验Trae
- 【推荐】编程新体验，更懂你的AI，立即体验豆包MarsCode编程助手
- 【推荐】抖音旗下AI助手豆包，你的智能百科全书，全免费不限次数
- 【推荐】轻量又高性能的 SSH 工具 IShell：AI 加持，快人一步



编辑推荐：

- 10年+ .NET Coder 心语，封装的思维：从隐藏、稳定开始理解其本质意义
- .NET Core 中如何实现缓存的预热?
- 从 HTTP 原因短语缺失研究 HTTP/2 和 HTTP/3 的设计差异
- AI与.NET技术实操系列：向量存储与相似性搜索在 .NET 中的实现
- 基于Microsoft.Extensions.AI核心库实现RAG应用

阅读排行：

- 10年+ .NET Coder 心语 —— 封装的思维：从隐藏、稳定开始理解其本质意义
- 地球OL攻略 —— 某应届生求职总结
- 提示词工程——AI应用必不可少的技术
- Open-Sora 2.0 重磅开源！
- 字符编码：从基础到乱码解决

历史上的今天：

- 2017-09-28 showmemory.c 和 hello.s 源码
- 2017-09-28 Linux库函数制作(静态库、动态库)
- 2017-09-28 GCC(警告,优化以及调试选项)

