ORACLE

Maintaining A Stable Real-time Kernel Patch Set

A Real-time Linux Collaborative Project

Joseph Salisbury
Consulting Member of Technical Staff
Oracle Corporation
September 18, 2024

Agenda

- Introduction
- Why Do We Need A RT Patch Set
- Patch Versions
- New Version Frequency
- RT Patch Examples
- Patch Creation Tools
- RT Patch Creation Workflow
- Testing
- Patch Distribution
- Conclusion
- Questions?

Introduction

"The Real Time Linux collaborative project was established to help coordinate the efforts around mainlining Preempt RT and ensuring that the maintainers have the ability to continue development work, long-term support and future research of RT." (RT wiki, 2024)

- Several Companies Participate In Patch Maintenance
- Project Hosted By The Linux Foundation
- Active community, that participates in bug fixing, testing automation and documentation.
- Mailing lists
- IRC Channel

Why Do We Need A RT Patch Set?

The RT patch set provides the logic for changing a generic vanilla kernel into a real-time operating system.

- Work began on RT Linux in ~2004. (Approaches to realtime linux, LWN 2004)
- “The main aim of the real-time preemption is to minimize the amount of kernel code that is non-preemptible.”(McKenney, LWN 2005)
- RT is partially available in mainline Linux.
- Merge into mainline started in 2021 with v5.15 (Preemption Locking).
  - E5e726f7bb9f (“Merge tag 'locking-core-2021-08-30' of git://git.kernel.org/pub/scm/linux/kernel/git/tip/tip”)
- Last hurdle is printk.

Patch Versions

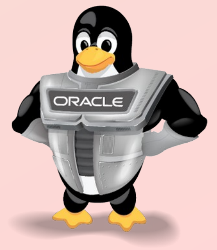
The PREEMPT_RT Patch Is Available For Mainline And LTS Kernels

6.11-rt Sebastian A. Siewior, development (Linutronix)
6.6-rt Clark Williams (Red Hat)
6.1-rt Clark Williams (Red Hat)
5.15-rt Joseph Salisbury (Oracle)
5.10-rt Luis Claudio R. Goncalves (Red Hat)
5.4-rt Tom Zanussi (Intel)
4.19-rt Daniel Wagner (Suse)

New Version Frequency

Upstream Stable Updates Are Primary Trigger

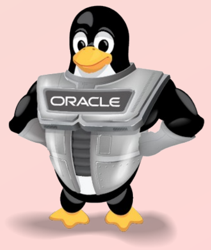
- Patch Maintainers Check for Upstream Stable Updates Regularly
- Maintainer Decides How Often To Generate New Version
  - Example Criteria:
    - At Least Once a Month – Depending on Stable Updates
    - At Least Every Four Stable Kernel Releases
    - If There Is A Critical CVE
    - Where There Are RT Specific Bug Fixes



RT Patch Examples

RT Patches Are Applied To Many Areas of The Kernel

- The real-time patches may be needed in the scheduler, drivers, tools, etc.
  - [PATCH 002/180] sched: Introduce migratable()
    - Introduce a helper which checks whether the current task can be migrated. This will allow us to keep preemption enabled and instead disable migration.
  - [PATCH 020/180] efi: Disable runtime services on RT
    - The EFI functions get_time, set_time take around 10ms, which is far too long. This patch disables EFI's runtime wrappers on PREEMPT_RT.
  - [PATCH 080/180] lockdep: Make it RT aware
    - Softirqs on PREEMPT_RT are always invoked within the context of a threaded interrupt handler or within ksoftirqd. This patch teaches lockdep that we don't really do softirqs on RT.
  - [PATCH 090/180] scsi/fcoe: Make RT aware.



RT Patch Creation Tools

The Kernel Uploaded for kernel.org (Kup)

- Kup is a file upload utility for kernel.org.(Kup Documentation).
- It is designed to only accept cryptographically verified uploads from pre-authorized, trusted members.
- SRT needs kup installed and working.
  - This will mean you have write access to some parts of kernel.org
  - Have a kernel.org trusted gpg key and account
- Easy to install:
  - Clone repo and from within the repo run:
  - **sudo pip3 install kup**
  - Also packaged in some Distros
    - **yum install kup**

RT Patch Creation Tools

Kup Can Be Used For More Than Uploading

- Can be used for file operations on kernel.org
- List the patch directory for 5.15:

  -
    ```
    kup ls pub/linux/kernel/projects/rt/5.15/
    ```

  -
    ```
    kup ls pub/linux/kernel/projects/rt/5.15/older
    ```
- Remove uploaded files:

  -
    ```
    kup rm pub/linux/kernel/projects/rt/5.15/patches-5.15.158-rt76.tar.xz
    ```
- Move files around:

  -
    ```
    kup mv pub/linux/kernel/projects/rt/5.15/older/patches-5.15.158-
    rt76.tar.xz pub/linux/kernel/projects/rt/5.15/
    ```

RT Patch Creation Tools

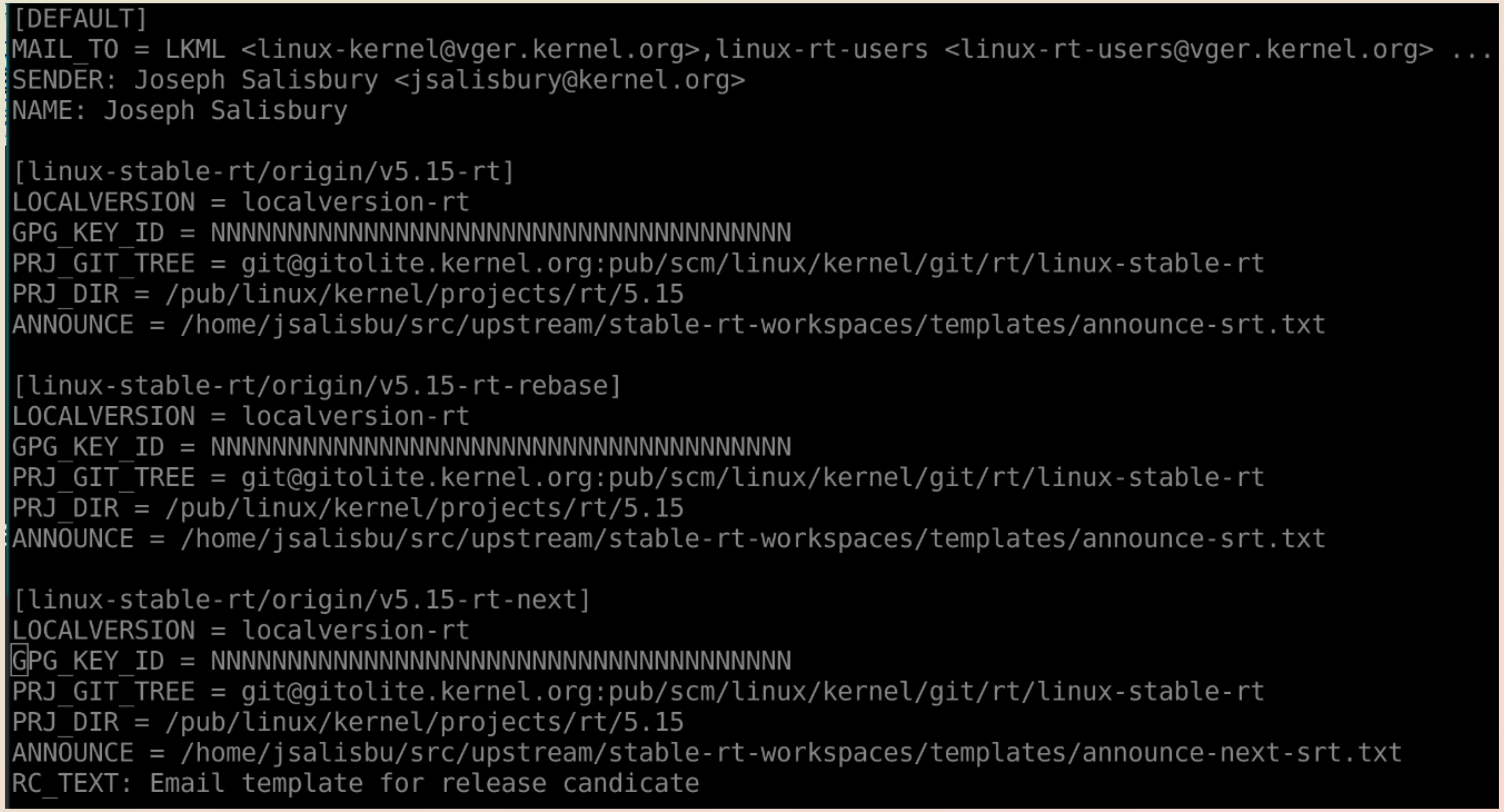
The stable-rt-tool (SRT) Project Performs The Heavy Lifting

- Created By Daniel Wagner
- Based on Python 3 and hosted in Github (stable-rt-tool repo).
- SRT is well documented (SRT Documentation).
- Easy to install:
  - Clone repo and from within the repo run:
  - **sudo pip3 install stable-rt-tools**
- Depends on Kernel Uploader (KUP repo).

RT Patch Creation Tools

SRT Configuration

- Located in ~/USER/.config/srt.conf

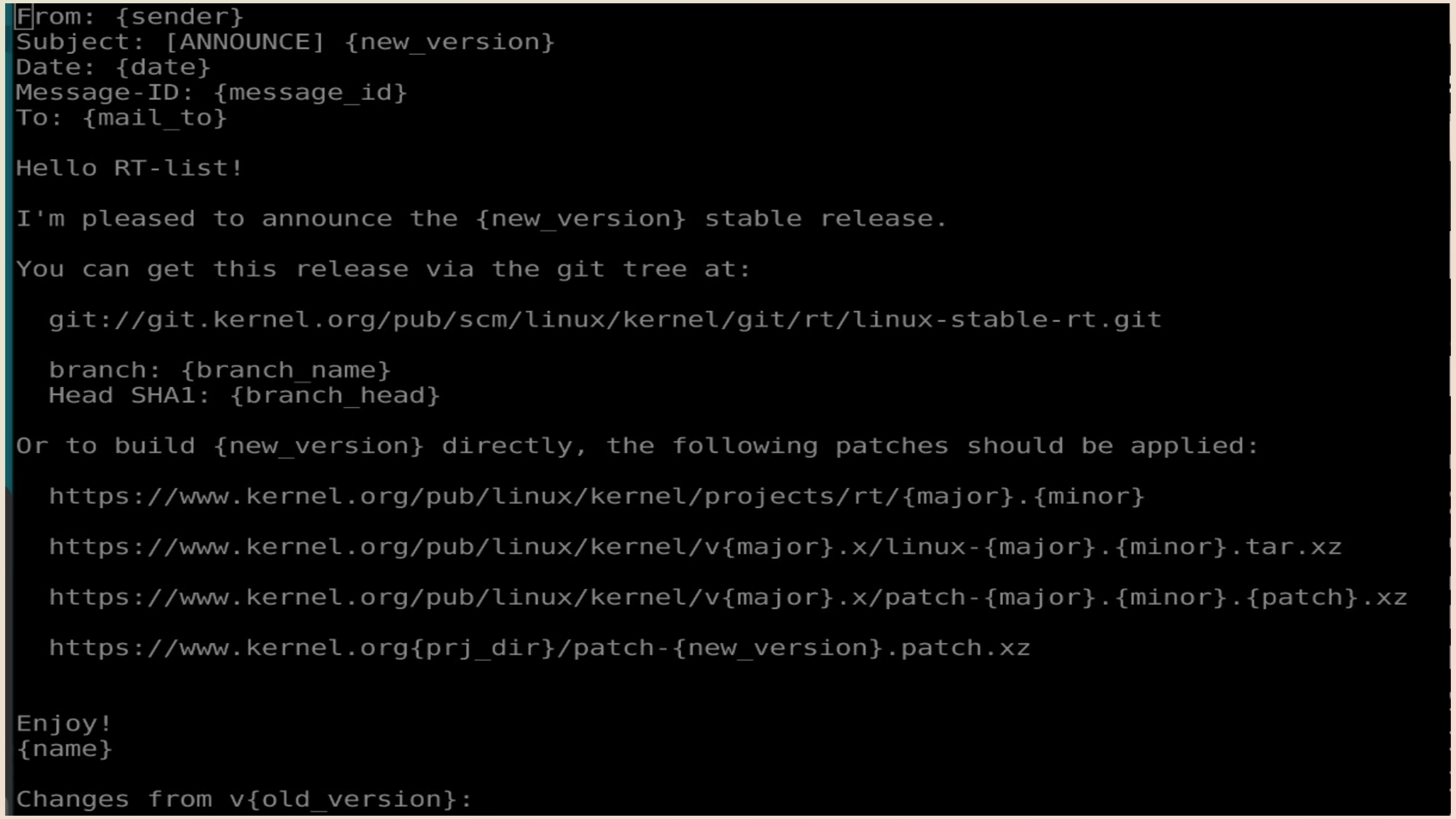


```
[DEFAULT]
MAIL_TO = LKML <linux-kernel@vger.kernel.org>,linux-rt-users <linux-rt-users@vger.kernel.org> ...
SENDER: Joseph Salisbury <jsalisbury@kernel.org>
NAME: Joseph Salisbury

[linux-stable-rt/origin/v5.15-rt]
LOCALVERSION = localversion-rt
GPG_KEY_ID = NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN
PRJ_GIT_TREE = git@gitolite.kernel.org:pub/scm/linux/kernel/git/rt/linux-stable-rt
PRJ_DIR = /pub/linux/kernel/projects/rt/5.15
ANNOUNCE = /home/jsalisbu/src/upstream/stable-rt-workspaces/templates/announce-srt.txt

[linux-stable-rt/origin/v5.15-rt-rebase]
LOCALVERSION = localversion-rt
GPG_KEY_ID = NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN
PRJ_GIT_TREE = git@gitolite.kernel.org:pub/scm/linux/kernel/git/rt/linux-stable-rt
PRJ_DIR = /pub/linux/kernel/projects/rt/5.15
ANNOUNCE = /home/jsalisbu/src/upstream/stable-rt-workspaces/templates/announce-srt.txt

[linux-stable-rt/origin/v5.15-rt-next]
LOCALVERSION = localversion-rt
GPG_KEY_ID = NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN
PRJ_GIT_TREE = git@gitolite.kernel.org:pub/scm/linux/kernel/git/rt/linux-stable-rt
PRJ_DIR = /pub/linux/kernel/projects/rt/5.15
ANNOUNCE = /home/jsalisbu/src/upstream/stable-rt-workspaces/templates/announce-next-srt.txt
RC_TEXT: Email template for release candicate
```

RT Patch Creation Tools

SRT Template For Announcements

- Location Defined in SRT Configuration File

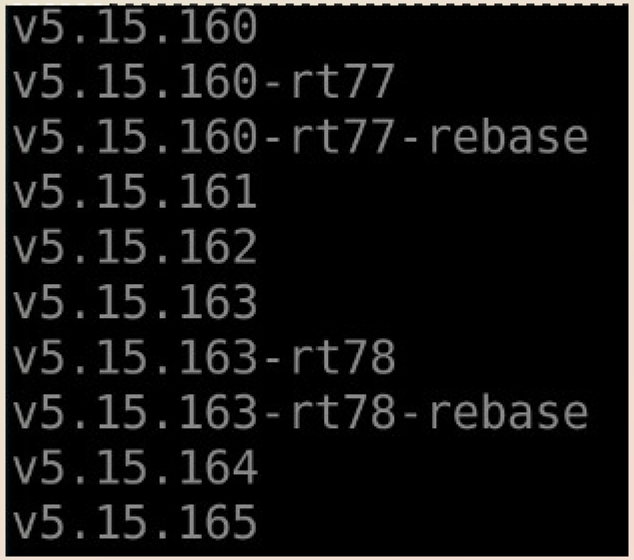


```
From: {sender}
Subject: [ANNOUNCE] {new_version}
Date: {date}
Message-ID: {message_id}
To: {mail_to}

Hello RT-list!

I'm pleased to announce the {new_version} stable release.

You can get this release via the git tree at:

  git://git.kernel.org/pub/scm/linux/kernel/git/rt/linux-stable-rt.git

  branch: {branch_name}
  Head SHA1: {branch_head}

Or to build {new_version} directly, the following patches should be applied:

  https://www.kernel.org/pub/linux/kernel/projects/rt/{major}.{minor}

  https://www.kernel.org/pub/linux/kernel/v{major}.x/linux-{major}.{minor}.tar.xz

  https://www.kernel.org/pub/linux/kernel/v{major}.x/patch-{major}.{minor}.{patch}.xz

  https://www.kernel.org{prj_dir}/patch-{new_version}.patch.xz


Enjoy!
{name}

Changes from v{old_version}:
```

RT Patch Creation Workflow

Using SRT To Create and Upload Patches

- First step is to check latest upstream stable version
  - cd ~/stable-rt-workspace/v5.15-rt
  - git fetch --all
  - git tag -l 'v5\.15\.*' --sort=v:refname | tail



```
v5.15.160
v5.15.160-rt77
v5.15.160-rt77-rebase
v5.15.161
v5.15.162
v5.15.163
v5.15.163-rt78
v5.15.163-rt78-rebase
v5.15.164
v5.15.165
```

RT Patch Creation Workflow

Repositories Are Updated By A Merge and Rebase

- There are three branches for a RT stable version
  - A branch that has stable updates merged in.
    - Provides a code base that can be used for bisects, custom kernels, etc
  - A branch that has stable updates rebased.
    - Provides a tree with the RT patches ‘on top’.
    - This is how the RT patches are created for distribution.
  - A ‘-next’ tree that is used for release candidates.
    - Used to update patch sets when there are RT specific fixes.

RT Patch Creation Workflow

Merging Stable Updates

- Use tag identified in prior step for merge
  - **git merge v5.15.165**
  - It most cases the merge is clean.
  - Sometimes there can be conflicts.
- Once merge is complete, use SRT to commit and tag
  - **srt commit** (Performs a ‘git add’ and ‘git commit’ on proper branch.
  - **srt tag** (Performs a ‘git tag -s -y’ using the defined GPG_KEY_ID).

RT Patch Creation Workflow

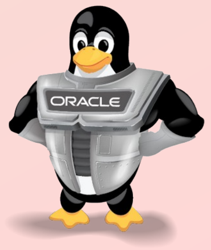
Rebase Stable Updates

- Use tag identified in prior step for rebase
  - **git rebase -i v5.15.165**
  - It most cases the rebase is clean.
  - Sometimes there can be conflicts.
- Once rebase is complete, use SRT to commit and tag
  - **srt commit** (Performs a ‘git add’ and ‘git commit’ on proper branch.
  - **srt tag** (Performs a ‘git tag -s -y’ using the defined GPG_KEY_ID).

RT Patch Creation Workflow

Compare Merge and Rebase Branches

- Once the merge and rebase trees are done, compare them.
  - Branches should be the same except where patches are commited.
  - git diff v5.15-rt && echo "no differences"
  - If there are “no differences” continue, otherwise see what went wrong.



RT Patch Creation Workflow

Create The Patches

- Use SRT to create the patches.
  - **srt create v5.15.163-rt78 v5.15.165-rt79**
    - Under the covers SRT is running: “git format-patch” based on rebase.
  - Patches will be created in rebase tree at: ~/patches/v5.15.165-rt67
  - Under that directory will be:
    - A patches directory with actual patches
    - A compressed single patch file
    - A tar file containing all the patches.

RT Patch Creation Workflow

Perform Test Builds / Local Testing Prior To Upload

- Test build will vary based on what distro the maintainer uses.
  - Debian/Ubuntu
    - fakeroot debian/rules
  - RHEL Derivatives
    - make -j$(nproc) binrpm-pkg
- Perform boot test.
- Run cyclictest and compare against baseline.

RT Patch Creation Workflow

Time To Sign and Upload Patches

- All done from the rebase tree.
- Sign with SRT:
  - **srt sign v5.15.163-rt78 v5.15.165-rt79**
    - SRT signs all the newly created files using gpg.
- Upload with SRT:
  - **srt upload v5.15.163-rt78 v5.15.165-rt79**
    - This is where the Kup tool now comes in.
      - Kup removes the prior version patch files.
      - Kup performs a ‘put’ of new patch files.

RT Patch Creation Workflow

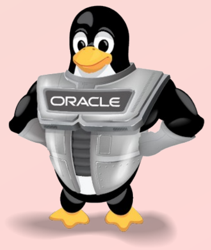
Push The Merge Tree

- All done from the merged tree.
- Use SRT to push updated repo:
  - **srt push v5.15.163-rt78 v5.15.165-rt79**
  - SRT performs a ‘git push’ to kernel.org
- The push and upload use the srt.conf file for parameters such as:
  - PRJ_DIR
  - GPG_KEY_ID
  - PRJ_GIT_TREE

RT Patch Creation Workflow

Announce New Release

- SRT uses template to create mail message.
- **srt announce v5.15.160-rt77 v5.15.165-rt78 > ../announce-rt**
- I use mutt msmtp to send email.
  - cat ../announce-rt | msmtp -t --
- Sends email message to LKML, ,linux-rt-users, RT maintiners.



RT Patch Creation Workflow

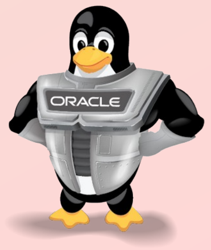
What About That ‘-next’ Branch

- Rarely Stable RT patches require an update.
- Updates to the patches require more care.
- A release candidate (RC) process is used.
- Only RT fixes are made – No stable updates.
- Similar workflow to stable update release.
  - Major difference is a RC is created and announced.
  - After a week or two, the official patch set is created.
- In the merge branch:
  - **git merge --ff-only v5.15-rt-next**
- In the rebase branch:
  - Apply RT patches and rebase to the same prior stable release.

RT Patch Testing

Manual Tests

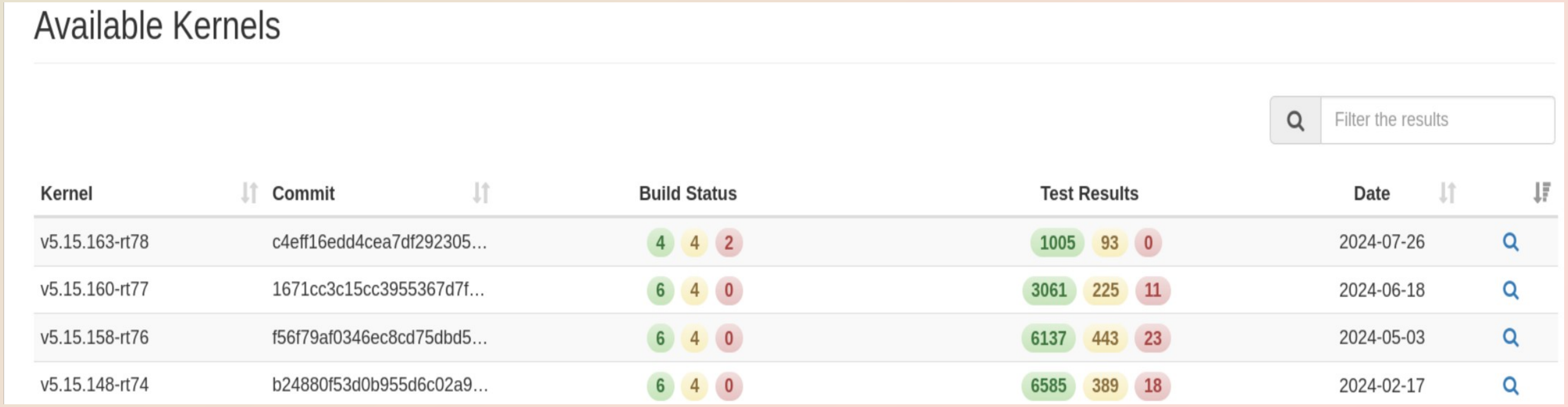
- Compile test.
- Boot test.
- Run of Cyclictest.
  - Compare to baseline.
  - Rule of thumb is <100us.
- RT-Tests
  - Test suite that contains various tests for real-time.(RT-Tests)
- RTEval
  - Runs hackbench and a kernel compile while cyclictest is running.(RTEval)



RT Patch Testing

Automated CI/CD Testing On kernelci.org

- “To ensure the quality, stability and long-term maintenance of the Linux kernel by maintaining an open ecosystem around test automation practices and principles.“(kernelci.org mission-objectives)
- Code hosted on Github(kernelci.org github)
- RT-Stable Dashboard - https://linux.kernelci.org/job/rt-stable/
- LTP and many other tests against a variety of hardware.



Available Kernels

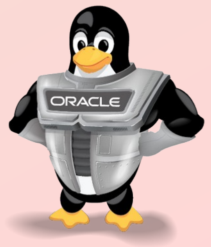
Filter the results

Kernel	Commit	Build Status	Test Results	Date
v5.15.163-rt78	c4eff16edd4cea7df292305...	4 4 2	1005 93 0	2024-07-26
v5.15.160-rt77	1671cc3c15cc3955367d7f...	6 4 0	3061 225 11	2024-06-18
v5.15.158-rt76	f56f79af0346ec8cd75dbd5...	6 4 0	6137 443 23	2024-05-03
v5.15.148-rt74	b24880f53d0b955d6c02a9...	6 4 0	6585 389 18	2024-02-17

RT Patch Distribution

Once Uploaded and Pushed, Patches are available from kernel.org

- **Patches**
- https://cdn.kernel.org/pub/linux/kernel/projects/rt/
  - Links to all current and prior kernel versions.
    - Each kernel version release provides access to all prior patch sets.
- **GIT Repo**
  - git://git.kernel.org/pub/scm/linux/kernel/git/rt/linux-stable-rt.git
    - Can be cloned to be built or modified.



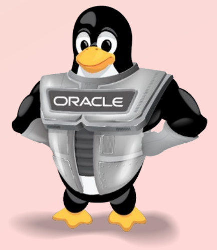
Conclusion

The Real-time Linux Project is An Example of How A Project Can Be Maintained Until Merged Into Mainline - With The Help of The Linux Foundation!

- Promising emails suggest RT could be in mainline as soon as v6.12
  - Subject [PATCH 0/3] Allow to enable PREEMPT_RT (https://lkml.org/lkml/2024/9/6/773)
- RT patch set EOL dates:

Release	EOL
Linux-6.6-rt	Dec 2026
Linux-6.1-rt	Dec 2026
Linux-5.15-rt	Oct 2026
Linux-5.10-rt	Dec 2026
Linux-5.4-rt	Dec 2025
Linux-4.19-rt	Dec 2024

Questions?



References

- *“Approaches to Realtime Linux.” Approaches to Realtime Linux [LWN.Net], lwn.net/Articles/106010/.*
- *“A Realtime Preemption Overview.” A Realtime Preemption Overview [LWN.Net], lwn.net/Articles/146861/.*
- “*LF Real-time Wiki*.”, wiki.linuxfoundation.org/realtime/start.
- “*Kernelci.org mission-objectives*”, https://kernelci.org/mission-objectives/.
- “*KernelCI Github Repo*”, https://github.com/kernelci/kernelci-core.
- “*Kup repo*.”, https://github.com/mricon/kup.
- “*Kup Documentation*”, https://korg.docs.kernel.org/kup.html.
- “*RTEval*”, https://wiki.linuxfoundation.org/realtime/documentation/howto/tools/rteval
- “*RT-Tests*”, https://wiki.linuxfoundation.org/realtime/documentation/howto/tools/rt-tests
- “*SRT Github Repo*.”, https://github.com/igaw/stable-rt-tools.
- “*SRT Documentation*.”, https://stable-rt-tools.readthedocs.io/en/latest/index.html.