LF *live*

MENTORSHIP SERIES

LF live MENTORSHIP SERIES

Speeding up Kernel Development with virtme-ng

Andrea Righi, Principal Software Engineer, NVIDIA

LF live MENTORSHIP SERIES

Disclaimer

- The views and opinions expressed in this talk are my own and do not necessarily reflects the official policy of positions of NVIDIA. Any content shared is based on my personal experience and perspective.

LF live MENTORSHIP SERIES

Introduction and Agenda

- What is virtme-ng?
- How does it work?
- What can I do with virtme-ng?

Scope and Expectations

- Lower the barrier of kernel development
- Speeding up kernel your development workflow

LF live MENTORSHIP SERIES

What is virtme-ng?

What is virtme-ng?

- A tool to quickly build and run kernels inside a virtualized snapshot of your live system
- Derived from virtme by Andy Lutomirski

What is **not virtme-ng?**

- Virtme-ng is not a virtualization manager (libvirt, Incus, docker, ...)
- It is not a platform to run services inside VMs

Why do I need virtme-ng?

- Testing kernel is painful and slow
- Lack of fast edit/compile/test cycle
- Lack of standard kernel development tooling

LF live MENTORSHIP SERIES

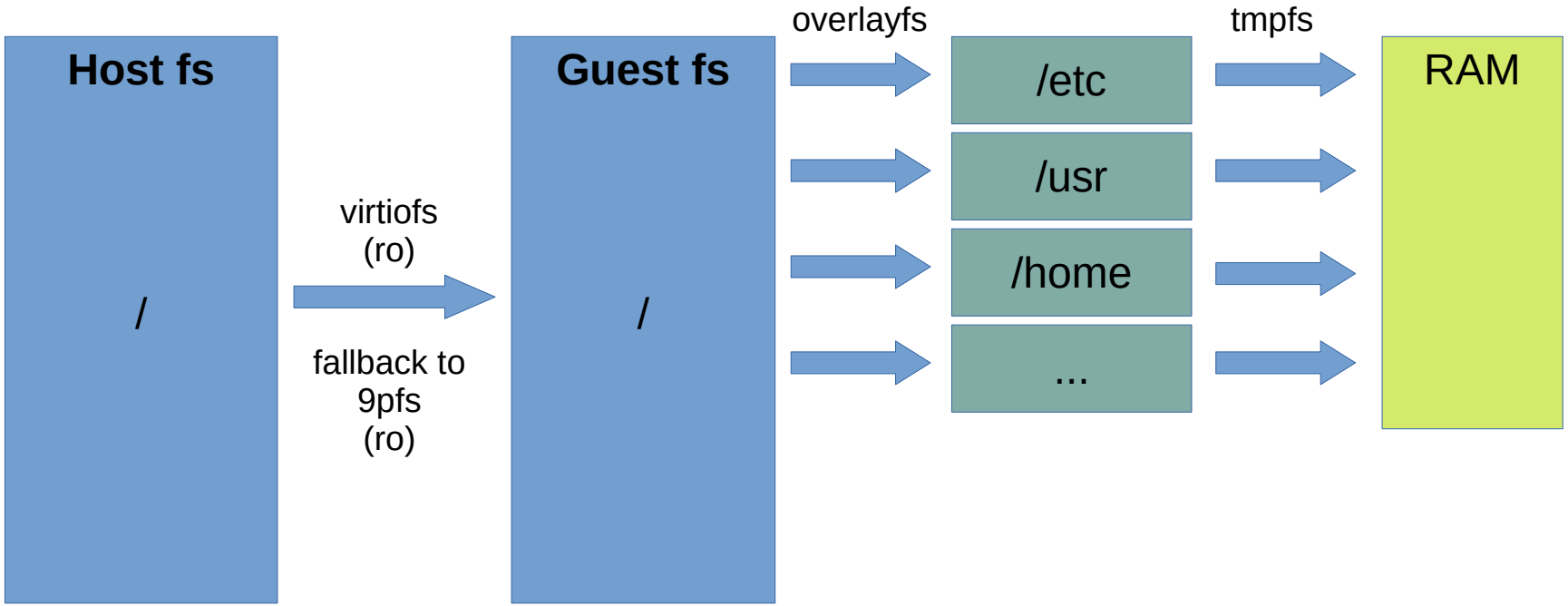
How does it work?

virtme-ng architecture

- qemu/KVM
  - virtme-ng is a python script on top of qemu/kvm
- virtiofs + overlayfs
  - CoW live snapshot of the host filesystem inside the guest
- qemu/KVM microVM
  - Lightweight virtual platform (optimized for boot time and memory footprint)
- virtme-ng-init
  - Lightweight custom init written in Rust

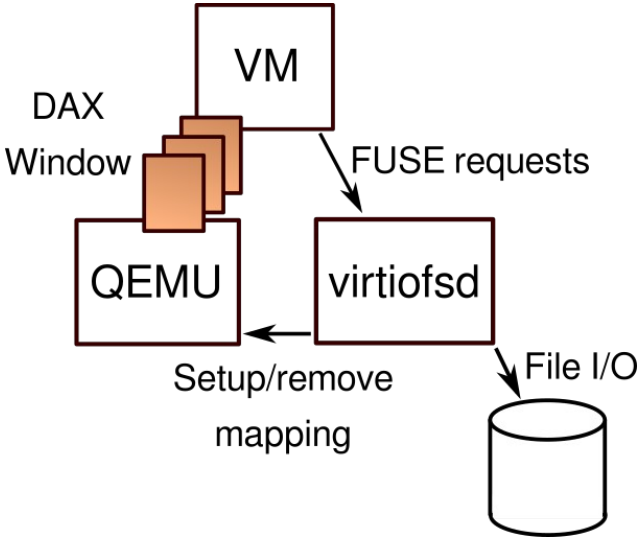
LF Live MENTORSHIP SERIES

virtme-ng filesystem overview



viritiofs

- Shared filesystem that lets VMs access a directory tree on the host using FUSE / vhost-user



https://virtio-fs.gitlab.io/design.html

virtme-ng filesystem performance (9pfs vs virtiofs)

```
$ time git diff
   9pfs:      284.5 sec
   virtiofs:    1.7 sec
```

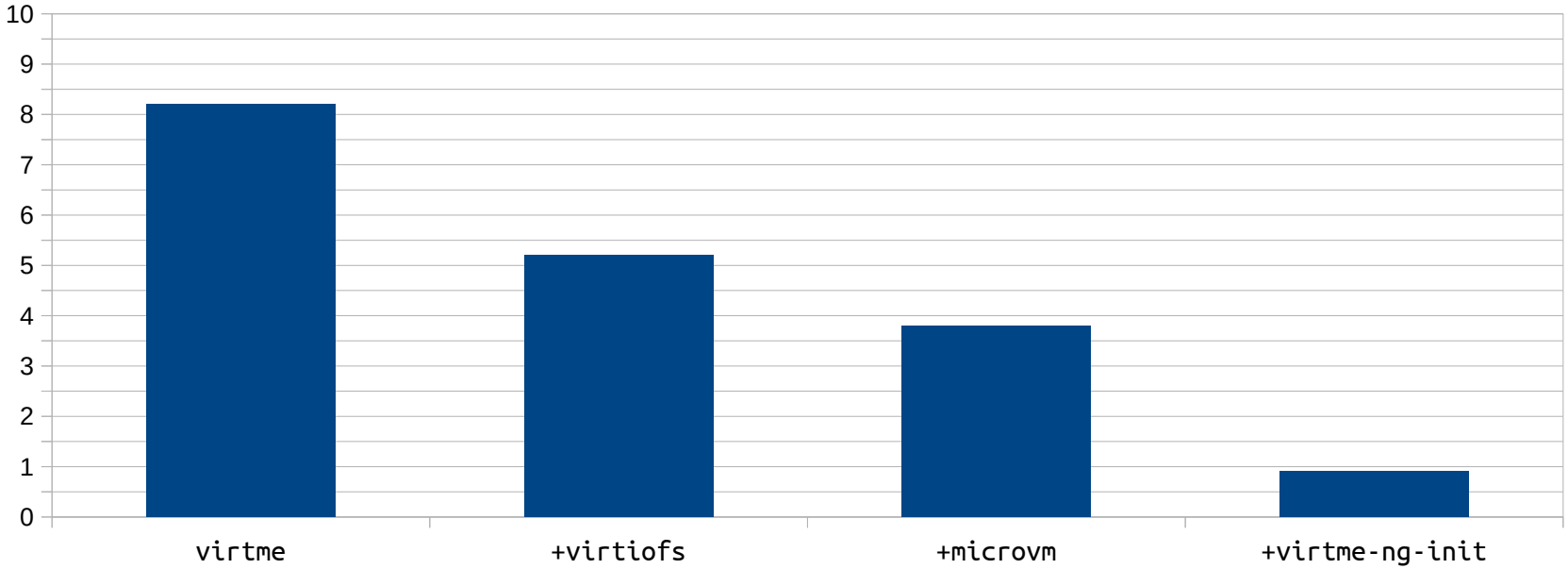
QEMU microVM

- Virtual platform (derived from firecracker)
- Minimalist machine type (no PCI / ACPI)
- Optimized for boot time and memory footprint

virtme-ng-init

- Lightweight init implemented in Rust
- Specifically designed for virtme-ng
- Replaced the original virtme init script in bash

virtme-ng boot time



LF live MENTORSHIP SERIES

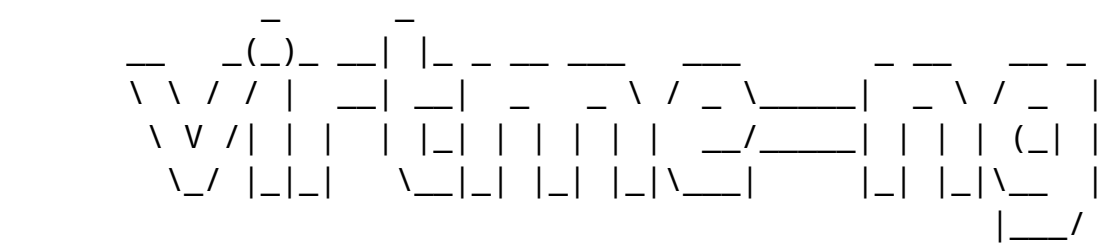
What can I do with virtme-ng?

“make localmodconfig” for KVM

```
arighi@gpd3~/s/linux (master)> time vng -b --build-host nv-builder
...
_____________________________________________________________________
Executed in   85.01 secs    fish           external
   usr time    1.20 secs  185.00 micros    1.20 secs
   sys time    0.83 secs   79.00 micros    0.83 secs
```

Run the recompiled kernel

```
arighi@gpd3~/s/linux (master)> vng
```



```
kernel version: 6.11.0-rc4-virtme x86_64
(CTRL+d to exit)

arighi@virtme-ng~/s/linux (master)> uname -r
6.11.0-rc4-virtme
```

Boot is pretty fast

```
arighi@gpd3~/s/linux (master)> time vng -- uname -r
6.11.0-rc6-virtme
```

```
_____________________________________________________________________
Executed in  985.58 millis    fish           external
   usr time  979.04 millis  221.00 micros  978.82 millis
   sys time  687.36 millis  131.00 micros  687.23 millis
```

Run kselftests

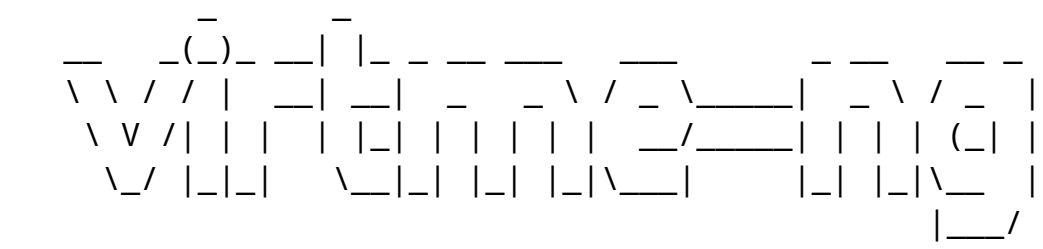
```
arighi@gpd3~/s/linux (master)> vng -- make -C tools/testing/selftests TARGETS=futex run_tests
...
# ok 1 futex_waitv private
# ok 2 futex_waitv shared
# ok 3 futex_waitv without FUTEX_32
# ok 4 futex_waitv with an unaligned address
# ok 5 futex_waitv NULL address in waitv.uaddr
# ok 6 futex_waitv NULL address in *waiters
# ok 7 futex_waitv invalid clockid
# # Totals: pass:7 fail:0 xfail:0 xpass:0 skip:0 error:0
ok 1 selftests: futex: run.sh
...

_____________________________________________________________
Executed in   11.57 secs    fish           external
   usr time    2.25 secs    0.40 millis    2.25 secs
   sys time    2.04 secs    1.05 millis    2.04 secs
```

My LKML workflow

```
arighi@gpd3~/s/linux (master)> b4 shazam \
                          https://lore.kernel.org/lkml/87zfod54wq.fsf@linux.ibm.com/T/#t
...
Applying: workqueue: Clear worker->pool in the worker thread context
...
Executed in  933.19 millis    fish           external

arighi@gpd3~/s/linux (master)> vng -vb --build-host nv-builder
...
Executed in   96.06 secs    fish           external

arighi@gpd3~/s/linux (master)> time vng -- uname -r
6.11.0-rc6-virtme
...
Executed in  984.35 millis    fish           external
```

Simulate CPU topology

```
arighi@gpd3~> vng -r --cpu 4,sockets=1,cores=2,threads=2
```




```
arighi@virtme-ng~> lscpu -e
CPU NODE SOCKET CORE L1d:L1i:L2:L3 ONLINE
  0    0      0    0 0:0:0:0          yes
  1    0      0    0 0:0:0:0          yes
  2    0      0    1 1:1:1:0          yes
  3    0      0    1 1:1:1:0          yes
```

Simulate memory topology

```
arighi@gpd3~> vng -r -m 4G \
                  --numa 1G,cpus=0-1,cpus=3 \
                  --numa 3G,cpus=2,cpus=4-7 \
                  -- numactl -H
available: 2 nodes (0-1)
node 0 cpus: 0 1 3
node 0 size: 1006 MB
node 0 free: 933 MB
node 1 cpus: 2 4 5 6 7
node 1 size: 2913 MB
node 1 free: 2770 MB
node distances:
node   0   1
  0:  10  20
  1:  20  10
```

stdin/stdout pipeline

```
arighi@gpd3~/s/linux (master)> echo "lscpu -e" | \
                                    vng --cpu 4,sockets=1,cores=2,threads=2 -- bash | \
                                    cowsay -n
 ___________________________________________
/ CPU NODE SOCKET CORE L1d:L1i:L2:L3 ONLINE \
|   0    0      0    0 0:0:0:0          yes |
|   1    0      0    0 0:0:0:0          yes |
|   2    0      0    1 1:1:1:0          yes |
\   3    0      0    1 1:1:1:0          yes /
 -------------------------------------------
        \   ^__^
         \  (oo)\_______
            (__)\       )\/\
                ||----w |
                ||     ||
```

Multi-kernel testing pipeline!

```
arighi@gpd3~/s/linux (master)> time true | vng -r v6.11-rc1 -- "cat -; uname -r" | vng -r v6.11-rc2 -- "cat
-; uname -r" | vng -r v6.11-rc3 -- "cat -; uname -r" | vng -r v6.11-rc4 -- "cat -; uname -r" | vng -r v6.11-
rc5 "cat -; uname -r" | vng -r v6.11-rc6 "cat -; uname -r" | vng -r v6.11-rc7 "cat -; uname -r" | cowsay -d
 _____________________________
/ 6.11.0-061100rc1-generic \
| 6.11.0-061100rc2-generic |
| 6.11.0-061100rc3-generic |
| 6.11.0-061100rc4-generic |
| 6.11.0-061100rc5-generic |
| 6.11.0-061100rc6-generic |
\ 6.11.0-061100rc7-generic /
 -----------------------------
        \   ^__^
         \  (xx)\_______
            (__)\       )\/\
             U  ||----w |
                ||     ||


________________________________________________________
Executed in    4.71 secs    fish           external
   usr time   22.22 secs  508.00 micros   22.22 secs
   sys time    8.00 secs  151.00 micros    8.00 secs
```

Kernel debugging

```
arighi@gpd3~/s/linux (master)> vng -v --debug
...
arighi@gpd3~/s/linux (master)> echo c | sudo tee /proc/sysrq-trigger

[ On another shell ]

arighi@gpd3~/s/linux (master)> vng --gdb
kernel version = 6.11.0-rc6-virtme
Reading symbols from vmlinux...
Remote debugging using localhost:1234
0xffffffff81cc1590 in rdtsc_ordered () at ./arch/x86/include/asm/msr.h:230
230         asm volatile(ALTERNATIVE_2("rdtsc",
(gdb) bt
#0  0xffffffff81cc1590 in rdtsc_ordered ()
    at ./arch/x86/include/asm/msr.h:230
#1  delay_tsc (cycles=2918423) at arch/x86/lib/delay.c:72
#2  0xffffffff8113c727 in panic (
    fmt=fmt@entry=0xffffffff8237346b "sysrq triggered crash\n")
    at kernel/panic.c:474
```

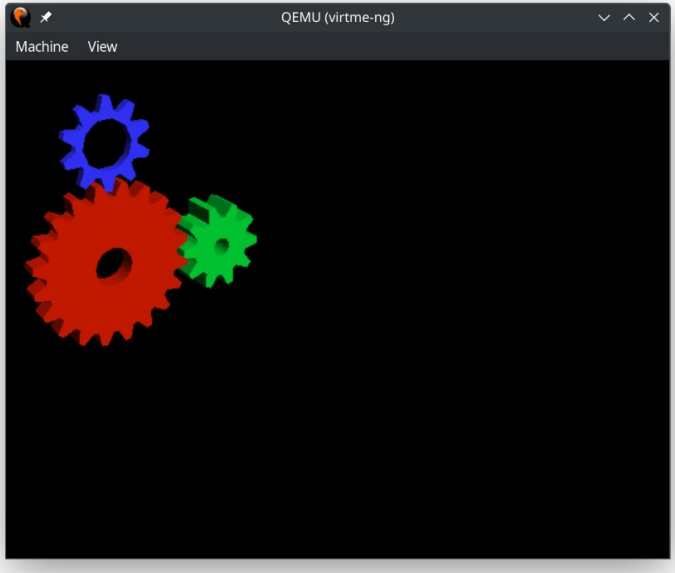
Kernel dump + drgn

```
arighi@gpd3~/s/linux (master)> vng -v --debug
...

[ On another shell ]

arighi@gpd3~/s/linux (master)> vng --dump /tmp/vmcore.img
arighi@gpd3~/s/linux (master)> echo "print(prog['jiffies'])" | \
                                          drgn -q -s vmlinux -c /tmp/vmcore.img
drgn 0.0.27 (using Python 3.12.6, elfutils 0.191, with libkdumpfile)
For help, type help(drgn).
>>> import drgn
>>> from drgn import FaultError, NULL, Object, cast, container_of, execscript, offsetof,
reinterpret, sizeof, stack_trace
>>> from drgn.helpers.common import *
>>> from drgn.helpers.linux import *
>>> (volatile unsigned long)4294678457
```

LF live MENTORSHIP SERIES

Graphic mode

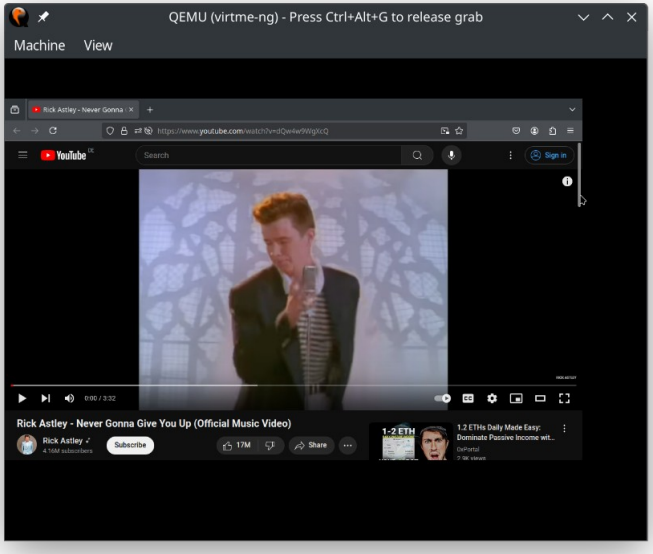
```
arighi@gpd3~/s/linux (master)> vng -v -g glxgears
...
11040 frames in 5.0 seconds = 2203.060 FPS
11932 frames in 5.0 seconds = 2386.394 FPS
12297 frames in 5.0 seconds = 2459.279 FPS
11461 frames in 5.0 seconds = 2290.772 FPS
```



LF live MENTORSHIP SERIES

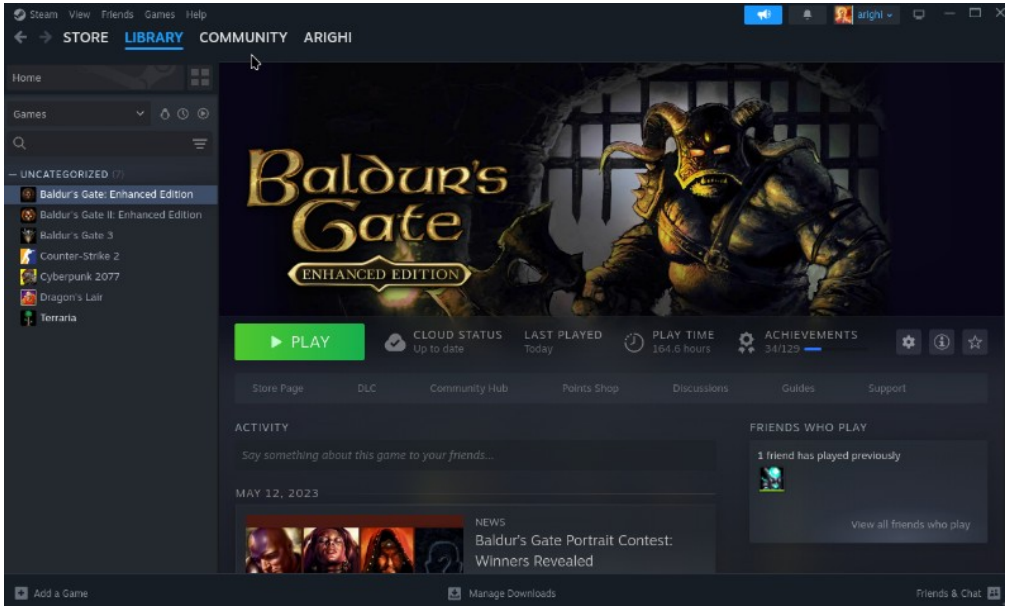
Watch YouTube

```
arighi@gpd3~/s/linux (master)> vng --disable-microvm \
                               -m 4G --sound --net user \
                               -g "firefox https://www.youtube.com/watch?v=dQw4w9WgXcQ"
```



Play games!

```
arighi@gpd3~/s/linux (master)> vng -r --disable-microvm \
                               -m 4G --sound --net user -g steam
```



Use in CI/CD

- Use virtme-ng in your github workflow to test your app with any kernel
- Real world CI/CD examples
  - sched_ext
  - Linux netdev
  - Mutter
  - ...

LF live MENTORSHIP SERIES

Conclusion

virtme-ng goals

- Lower the barrier of kernel development
- Fast edit/compile/test iterations
- Standard tool for kernel devs
- Powerful tool for CI/CD

Future plans

- Multiple vsock consoles
- Support systemd
- Boot kernel using qcow2 images
- GPU passthrough
- Confidential computing
- Secure boot

Reference

- virtme-ng github page
  - https://github.com/arighi/virtme-ng
- Faster kernel testing with virtme-ng
  - https://lwn.net/Articles/951313/
- Eco-friendly Linux kernel development: minimizing energy consumption during CI/CD
  - https://lwn.net/Articles/935180/

LF live MENTORSHIP SERIES

Thank you for joining us today!

We hope it will be helpful in your journey to learning more about effective and productive participation in open source projects. We will leave you with a few additional resources for your continued learning:

- The LF Mentoring Program is designed to help new developers with necessary skills and resources to experiment, learn and contribute effectively to open source communities.
- Outreachy remote internships program supports diversity in open source and free software
- Linux Foundation Training offers a wide range of free courses, webinars, tutorials and publications to help you explore the open source technology landscape.
- Linux Foundation Events also provide educational content across a range of skill levels and topics, as well as the chance to meet others in the community, to collaborate, exchange ideas, expand job opportunities and more. You can find all events at events.linuxfoundation.org.