LF ENERGY

SEAPATH

Virtualization for Real-time Power Grid Substation Automation

Savoir-faire LINUX®

20+

years in industrial product engineering in many areas

Savoir-faire Linux are experts in free and Open Source technologies in Canada and Europe.

Member of

LF ENERGY

THE LINUX FOUNDATION

yocto PROJECT SILVER MEMBER

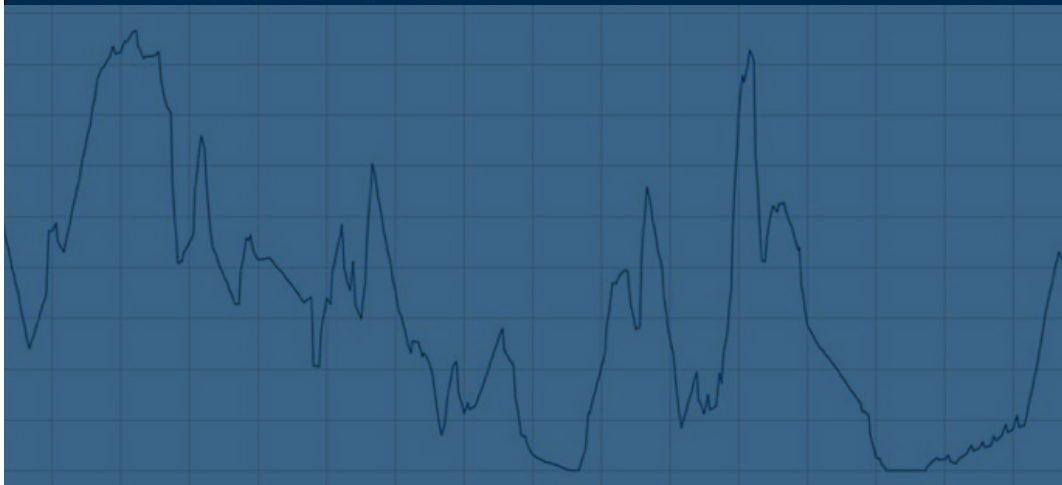
Context

Energy Transition drives change in power transmission and distribution grids

- Distributed renewable energy sources
- Demand response
- Electric mobility
- Smart services to the grids from a growing number of third-parties

Need to **swiftly adapt grid control architectures**

- Multiplication of distributed controls
- More dynamic and adaptive automation functions
- Increased data management needs



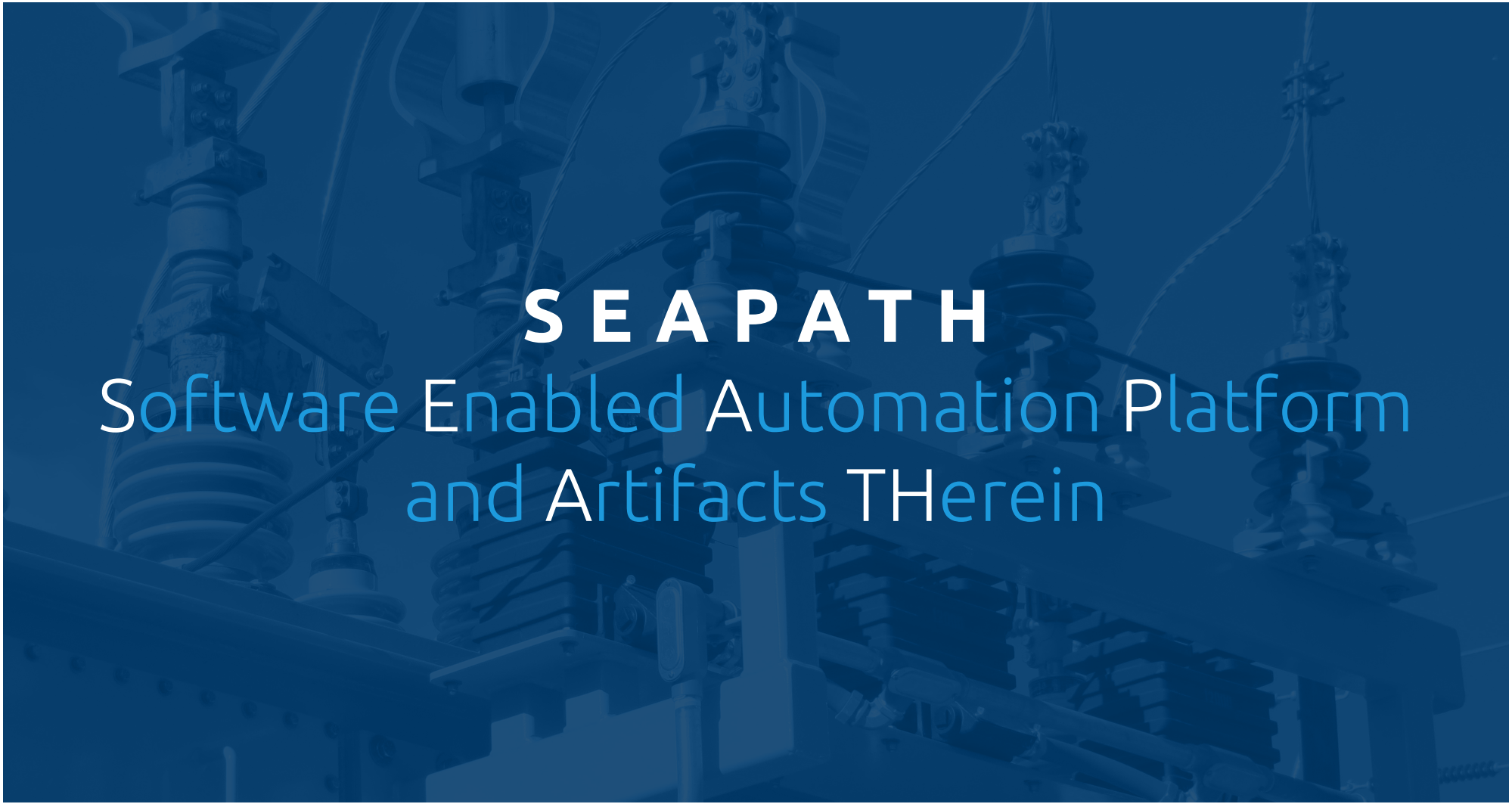
Vision

- Moving to **software-centric power automation** systems is a necessity to reach the required flexibility and scalability as well as substantial O&M cost savings

- An **Open Source** development model is needed to succeed

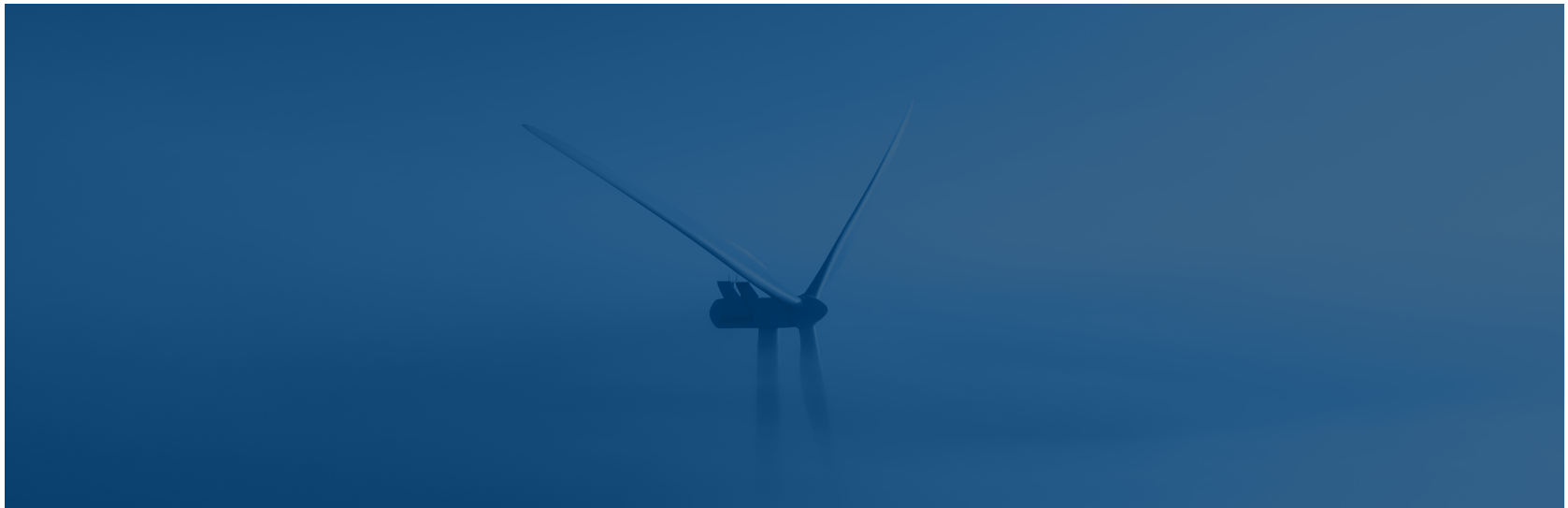
- Inspiration came from the telecommunication network industry

- Open source projects such as ONAP allowed carriers to transition to software-centric telecommunication networks enabling them to increase tenfold the scalability of the network to meet new demands, such as 5G, while improving operational performance.


SEAPATH
Software Enabled Automation Platform
and Artifacts THerein

Mission statement

The mission of the **SEAPATH** project is to **develop a “reference design” and “industrial grade” open source platform** that can run virtualized automation and protection applications requiring **real-time performance** (for the power grid industry in the first place and potentially beyond). This platform is intended to host multi-provider applications.



Our needs

High performance required

- Real-time
- Low latency

Adaptable

- Cybersecurity
- Customizable
- Hardware agnostic
- Updatable

Following the state of the art

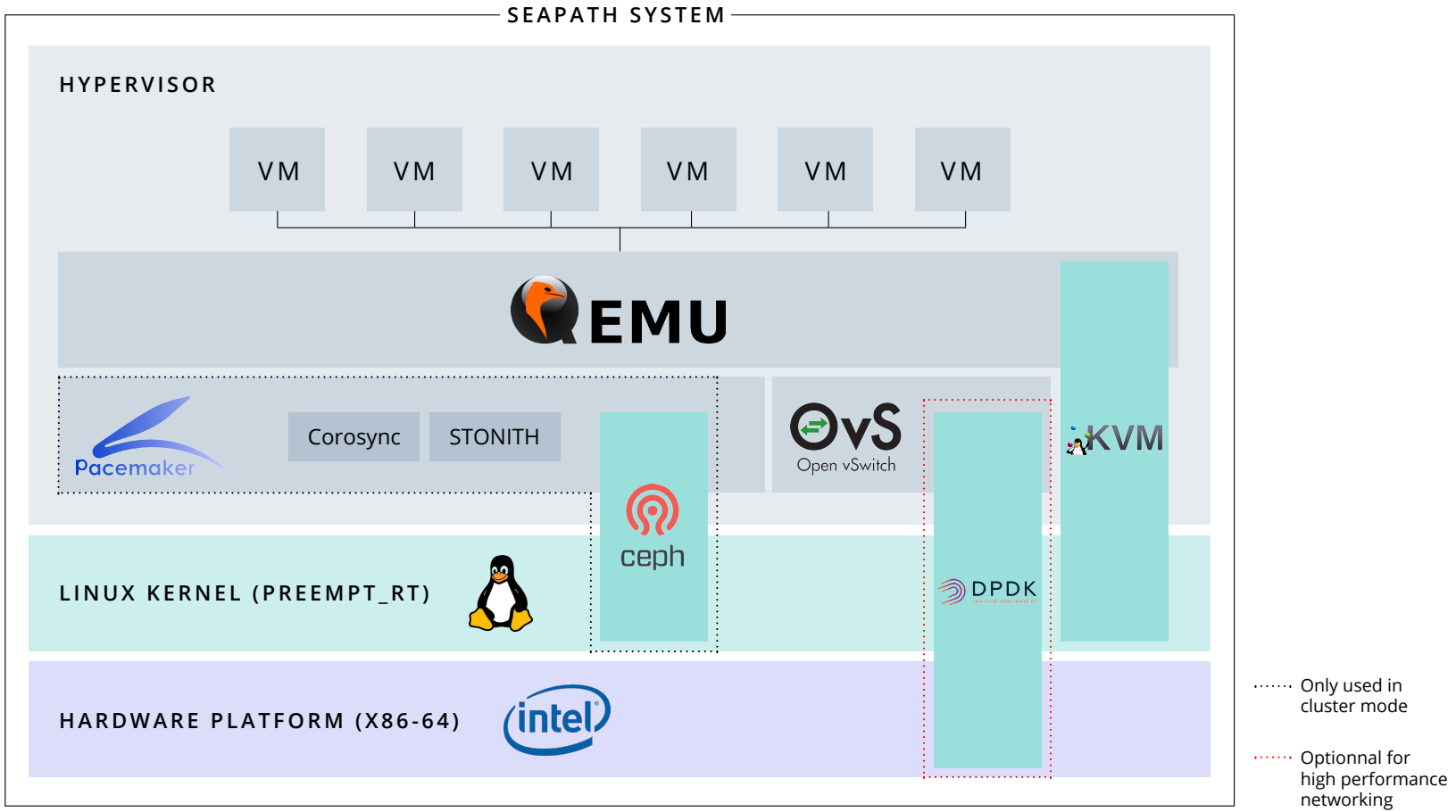
- Integrate innovative Open-source components

Choosing Yocto or Debian

yocto PROJECT

- Create a custom Linux distribution
- Hardware agnostic
- Full control of packages and versions
- Easy tracking and patching of CVE

Utilizing existing technology



SEAPATH testing process

Meeting requirements

Seapath must provide many guarantees

- The CI is launched at every pull requests
- The build must succeed AND all the tests must pass
- Avoid regression and display future requirements to meet
- Visible for everyone on GitHub

Generating a test report

- Organizing more than 1500 tests
- Link all test to specific requirements
- Automatically separate non-regression part and future work
- Tests are visible on the Github repository

Tests hypervisoriommu for virtu-ci1

Test ID	Tests	Results
SEAPATH-00030	iommu enabled in passthrough mode	PASS
SEAPATH-00031	iommu is loaded	PASS
SEAPATH-00032	iommu is populated	PASS
SEAPATH-00050	Linux kernel *iommu* : INTEL_IOMMU is enabled	PASS
SEAPATH-00050	Linux kernel *iommu* : AMD_IOMMU is enabled	PASS
SEAPATH-00050	Linux kernel *iommu* : AMD_IOMMU_V2 is enabled	PASS
SEAPATH-00050	Linux kernel *iommu* : IOMMU_IOVA is enabled	PASS

- number of tests: 8
- number of failures: 0

Tests hypervisorsecurity for virtu-ci1

Test ID	Tests	Results
SEAPATH-00033	/etc/group is consistent	PASS
SEAPATH-00033	/etc/gshadow is consistent	PASS
SEAPATH-00034	/etc/group does not include extra group	FAIL
SEAPATH-00034	/etc/gshadow does not include extra group	FAIL
SEAPATH-00006	Audit subsystem is disabled on cmdline	FAIL
SEAPATH-00008	Slab merging is disabled on cmdline	PASS
SEAPATH-00009	Kernel Page Table Isolation is always enabled on cmdline	PASS
SEAPATH-00010	SLUB redzoning and sanity checking enabled on cmdline	PASS
SEAPATH-00004	libvirtd can not acquire new privileges	FAIL
SEAPATH-00005	libvirtd capabilities are bounded	FAIL
SEAPATH-00125	libvirtd system calls are filtered	FAIL
SEAPATH-00039	openvswitch user is created and locked	FAIL
SEAPATH-00040	openvswitch user is part of hugepages group	FAIL
SEAPATH-00041	openvswitch user is part of vfio-net group	FAIL
SEAPATH-00042	ovs-vswitchd is running as user openvswitch	FAIL
SEAPATH-00043	ovsdb-server is running as user openvswitch	FAIL
SEAPATH-00126	ovs-vswitchd system calls are filtered	PASS

Writing tests

System-level testing

- All customer code is in the virtual machines
- Functional testing is impossible here
- Unit testing to check the requirements

Check /etc/ssh/ssh_host_ed25519_key permissions
Check /etc/ssh/ssh_host_rsa_key permissions
root password was randomized at boot
root password is randomized at each boot
root password is encrypted with a cryto at least equivalent as sha512
bash timeout is set read-only to 300s
sshd forbids setting environment variables

Introducing cukinia

Linux firmware validation framework

- Offer abstraction
- Simple portability (require only sh)
- No compilation and easy installation
- Extract results in xml or csv
- Of course, open source

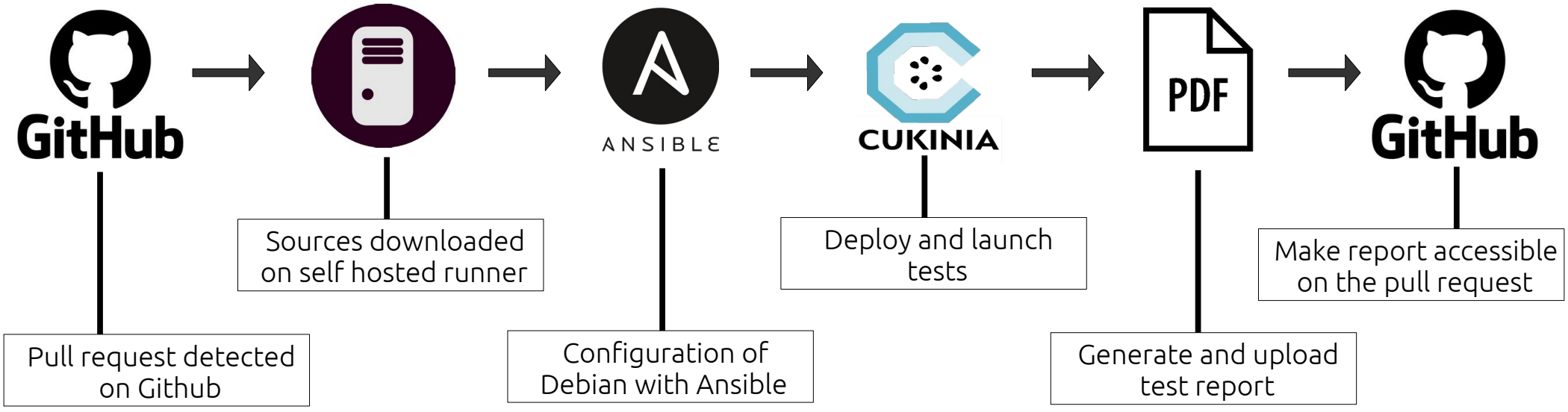

```
logging prefix "cukinia: "

cukinia_user www-data
cukinia_run_dir /etc/cukinia/tests.d
cukinia_process Xorg
cukinia_python_pkg math
not cukinia_python_pkg math
cukinia_mount sysfs /sys sysfs rw

not cukinia_mount /dev/nonex /nodir noopt

cukinia_log "result: $cukinia_failures failure(s)"
```

The complete CI



Implementation
1/2

debian

Use an already deployed Debian

- Already set up Debian on all machines
- No compilation needed
- Avoid the big problem of flashing the machines

ANSIBLE

LVM

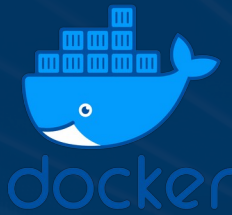
Control default state

- Configuration through Ansible
- Use idem-potency to control changes between different CI launches
- LVM will be configured for a true rollback mechanism

Implementation
2/2

A more complex CI than usual

- The Ci have to launch tests, gather results and build the test report
- Goal is to set up as few things as possible in the runner



Avoid downloading and configuring packages

- The CI code is re-download every time through Docker
- Ansible commands and report generation each have a specific Docker container

CQFD

- Use cqfd, a command line docker wrapper
- Useful to launch commands inside a docker container

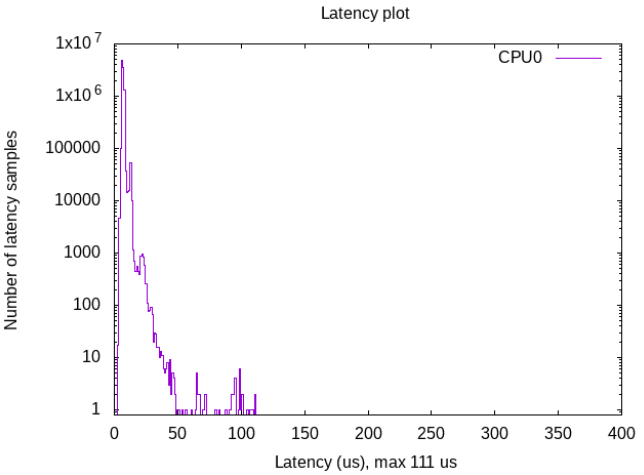
Future works

Deploy a CI for the Yocto version

- Implies deploying **other runners** for the compilation
- Handle concurrency problems
- All machines need to be **flashed** on every launch. It can be done either with an **update mechanism** or with **usb gadget**

Run long-term tests

- Real-time tests
- Cyclictests in virtual machines
- Launch at **every release** to certify that it meet the requirements



Thank you for your attention

https://github.com/seapath/

LF ENERGY