

Continuous integration on SEAPATH

A continuous integration (CI) process has been implemented in order to deploy and test a custom cluster before merging any pull requests. Source code can be found in <https://github.com/seapath/ci>.

The CI is based on GitHub action and has been completely dockerized in order to guarantee its reproducibility and scalability.

Description

The CI operate on a SEAPATH operating system. To avoid many problems, all machines are already flashed with SEAPATH and the CI will only configure the systems using the Ansible playbooks. This implies the iso creation and the boot process can't be tested.

All machines are linked to a self-hosted runner, on which the GitHub Action for the CI will run.

The CI job is divided into different stages :

- Fetch the sources of the pull request to test
- Launch the Debian configuration and hardening using Ansible playbooks
- Deploy Cukinia and launch tests
- Generate and upload the test report

Tests

The CI use [Cukinia](https://github.com/seapath/cukinia-tests/tree/main), a system-level validation framework as testing tool, all tests launched on Debian are available at <https://github.com/seapath/cukinia-tests/tree/main>.

The CI performed the following tests:

- check regression on SEAPATH Ansible configuration by calling these Ansible playbooks as it will be in a regular SEAPATH setting up
- functional system tests, which check the OS configuration to ensure all machines are well configured and there are no regression
- hypervisor tests, to verify we can deploy VMs on hypervisors
- cluster tests, to check the cluster are well setup and the share storage works
- security tests, to ensure security hardening is correctly applied
- network connections tests, which verify that all network interface and all network works by making connections among them
- Quick IEC 61850 Sample values receptions and latency measurement tests
- PTP tests, to verify PTP clock is well synchronized in the host and the VM

No Real-time, complex IEC 61850, complex cluster tests or long times latency tests are run on the CI for now. These tests are too long to be run on each GitHub pull request, so it is planned to integrate these tests at every release of the SEAPATH project.

Report

The CI can fail in three ways :

- The configuration of SEAPATH is wrong and the associated Ansible playbook fail
- The tests fail
- The tests on VM fail

The failure of the configuration can be observed in the CI logs.

After the configuration, all tests will be gathered in a test report. All the test contained in the report must pass for the pull request to be merged.

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	PASS
SEAPATH-00034	/etc/gshadow does not include extra group	PASS
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-00047	/etc/passwd is consistent	PASS
SEAPATH-00048	/etc/passwd does not include extra user	PASS
SEAPATH-00046	/etc/shadow is consistent	PASS
SEAPATH-00015	Vulnerabilities sysfs entry exist	PASS
SEAPATH-00017	System is not vulnerable to : meltdown	PASS
SEAPATH-00017	System is not vulnerable to : ll t f	PASS
SEAPATH-00017	System is not vulnerable to : spectre_v1	PASS
SEAPATH-00017	System is not vulnerable to : spectre_v2	PASS
SEAPATH-00012	admin user exists	PASS
SEAPATH-00013	admin has a password	PASS

- number of tests: 17
- number of failures: 0

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 <i>iommu</i> : INTEL_IOMMU is enabled	PASS
SEAPATH-00050	Linux kernel <i>iommu</i> : AMD_IOMMU is enabled	PASS
SEAPATH-00050	Linux kernel <i>iommu</i> : AMD_IOMMU_V2 is enabled	PASS
SEAPATH-00050	Linux kernel <i>iommu</i> : IOMMU_IOVA is enabled	PASS

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


Using the CI


The CI is actually running on the [debian-main](#) branch and on the [main](#) branch of the Ansible repository
Every pull request need to pass the tests to be merged.

After opening a pull request, wait for an approval of a member of SEAPATH. This will trigger the CI as a GitHub Action. The logs of the CI are visible on your pull request page, either in the "Conversation" tab or the "Checks" tab.
All running actions are also visible in the Actions tab on the repository.

The complete CI takes about 15 minutes. At the end, the test report is available through a link given in the log of the GitHub Action, at the end of the step "Launch test" step.



Access the logs of the CI on the pull request "Conversation" tab



**Some checks were not successful**

1 failing and 1 successful checks



[Hide all checks](#)

**CI debian / CI (pull_request)**

Failing after 22m


Required

Details

**DCO — DCO**

Required

Details

**This pull request is still a work in progress**

Draft pull requests cannot be merged.

Ready for review

Rebase and merge

or view [command line instructions](#).

Location of the link of the test report

Summary

Jobs

Run details

Usage

Workflow file

CI

succeeded

6 minutes ago in 22m 13s

Search logs

Launch test

1m 20s

----- 0.42s

168 Use Bash as Cukinia's shell

----- 0.35s

169 Patch cukinia: CAT

----- 0.33s

170 Create temporary Cukinia directory

----- 0.32s

171 Create /usr/share/cukinia/includes

----- 0.28s

172 Create /usr/share/testdata

----- 0.27s

173 Patch cukinia: zcat

----- 0.25s

174 Patch cukinia: _cukinia_prepare

----- 0.25s

175 Patch cukinia: ikconfig test

----- 0.25s

176 Test tools deployed successfully

177 sha256:c5febbcfda3d343aaa4f2cea8ed579aca2ac55848eab2a3dd1ad4c532195dcf2

178 All tests pass

179 See test Report at https://github.com/seapath/ci/blob/reports/docs/reports/PR-128/test-report_4074699338_2_2023-02-02_14h44m31.pdf

> Clean

0s

> Complete job

0s

Ansible's configuration has fail, the tests were not launched, and the link was not given.

The screenshot displays a web interface for a CI/CD pipeline. On the left, a sidebar contains a 'Summary' section with a home icon, a 'Jobs' section with a list of jobs including 'CI' (marked with a red 'x'), and a 'Run details' section with links for 'Usage' and 'Workflow file'. The main panel shows the details for the 'CI' job, which failed 1 hour ago in 21m 3s. It includes a 'Search logs' input field and a list of steps: 'Set up job' (0s), 'Initialize sources' (38s), 'Configure Debian' (20m 22s, failed), 'Launch test' (0s), 'Clean' (0s), and 'Complete job' (0s). The 'Configure Debian' step is highlighted with a red 'x' icon.

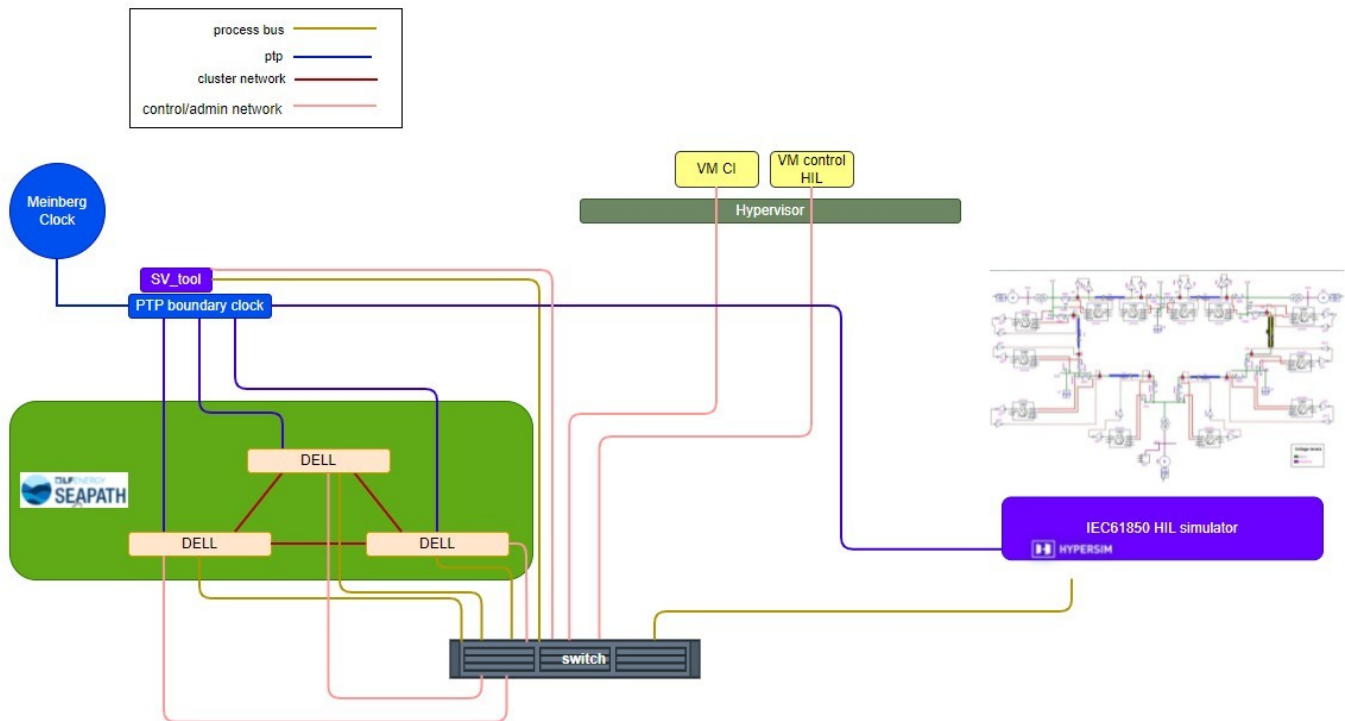
Host your own CI

Preparing a cluster

Hosting your own CI on SEAPATH first required having a cluster up and ready.

You need:

- a SEAPATH cluster (all instructions for cluster configuration can be found on the [SEAPATH architecture repository](#))
- a machine (or virtual machine) to host the GitHub runner
- a PTP clock
- a machine time synchronized with PTP clock with docker to host SV Tools (an IEC 61850 Sample Value simulator)



i IEC61850 HIL simulator is not needed because it is not used for the moment.

Preparing the CI

The runner used to configure the machines will be used for the CI. Docker and [cqfd](#) are the only required software to launch it.

If you choose GitHub Actions to trigger the CI, you can follow [this guide](#) to link your self-hosted runner with your repository. Otherwise, another trigger has to be configured for the chosen CI solution.

The CI script launched with the CI is `launch.sh` on the [SEAPATH CI](#) repository. It will have to be adapted to fit with your repository links and your inventories and keys on your runner.

This section will be expanded in the future for a better understanding.