How to run a weekly tests?

Weekly tests are daily tests + additional tests.

These additional tests are

- internal certificate (infrastructure-healthcheck)
- versions (security)
- basic_vm_stability and basic_cds_stability stability tests
- cpu hog, memory hog, node_ejection (resiliency)
- tern scan (sbo management and license scanning created by Alexander Mazuruk replaced by scancode.io)

Internal certificates, versions and tern are described in https://docs.onap.org/projects/onap-integration/en/latest/usecases/release_automated_usecases.html

Stability and resiliency tests are documented in the official documentation: https://docs.onap.org/projects/onap-integration/en/latest/integration-tests.html#stability-resiliency-tests

If you want to run a full weekly chain, you can use the existing chained-ci schedule.

It will deploy a weekly lab then execute the test.

Please note that the tests executed in weekly chains could be executed also in daily, it is just by configuration.

The configuration can be at 2 levels:

- the testing level: xtesting definition (if so the test must be initiated but due to constraint on env var not executed)
- the CI level: xtesting-onap gtlab-ci files that we decide whether we execute the tests. See https://gitlab.com/Orange-OpenSource/lfn/onap/xtesting-onap/-/blob/master/gitlab-ci/base.yml

At the moment all the constraints are done at the CI level

The core (daily test), versions (weekly defined at the CI level), are defined as follows:

```yaml
ore:
  <<: *core
  <<: *trigger_rules

security_versions:
  <<: *security_versions
  <<: *security_rules
  <<: *weekly_rules

weekly_rules: &weekly_rules
  rules:
  - if: '$pod =~ /^onap_weekly.*/'
    when: always
```

The weekly rules consist in looking for weekly in the pod name

```yaml
weekly_rules: &weekly_rules
  rules:
  - if: '$pod =~ /^onap_weekly.*/'
    when: always
```

Even if a stability and resiliency stage have been created, the management is the same (via the weekly rules)

```yaml
onap_stability:
  <<: *onap_stability
  <<: *weekly_rules

onap_resiliency:
  <<: *onap_resiliency
  <<: *weekly_rules
```

If the pod does not included weekly in its name, the stages are not triggered.

internal_certificate_check (kubernetes jobs) and versions (docker) are run exactly as any daily, see How to re-run a test on a daily platform?.

Stability, resiliency and tern testing require both some pre-installation steps performed by ansible roles.

- onap-chaos-tests: https://gitlab.com/Orange-OpenSource/lfn/onap/xtesting-onap/-/tree/master/roles/onap-chaos-tests

For example onap-chaos-tests install the litmus chaos engine on the namespace.
Some scripts are also deployed through the ansible scripts:


Executing these tests consist then to execute these scripts that are deployed respectively in /home/debian/run_stability_tests.sh, /tmp/resiliency/run_chaos_tests.sh and /home/debian/run_tern.sh

The dashboard (https://gitlab.com/Orange-OpenSource/lfn/onap/xtesting-onap/-/blob/master/doc/generate_status.py) must also be adapted accordingly.

```python
if ARGS.pod is not None:
    PODS = [ARGS.pod]
    # adapt tests according to the type of tests: daily/weekly/gating
    if "weekly" in ARGS.pod:
        INFRA_HEALTHCHECK['tests'].add('versions')
        PERIOD = 7
    if "gating" in ARGS.pod:
        SECURITY_USECASES['tests'].remove('kube_hunter')
    # adapt test according to the version: guilin / honolulu / master
    if "guilin" in ARGS.pod:
        HEALTHCHECK['tests'].remove('dcaemod')
        HEALTHCHECK['tests'].remove('cps-healthcheck')
        SMOKE_USECASES['tests'].remove('basic_clamp')
        SMOKE_USECASES['tests'].remove('cds_resource_resolution')
        SMOKE_USECASES['tests'].remove('basic_cnf_macro')
    if "honolulu" in ARGS.pod:
        HEALTHCHECK['tests'].remove('cps-healthcheck')
        SMOKE_USECASES['tests'].remove('cds_resource_resolution')
        SMOKE_USECASES['tests'].remove('basic_cnf_macro')
    if "istanbul" in ARGS.pod:
        SMOKE_USECASES['tests'].remove('cds_resource_resolution')
        SMOKE_USECASES['tests'].remove('basic_cnf_macro')
    # Exclude Cloudify based use cases in Master (after istanbul)
    # TO BE updated as it is possible to perform gating on old versions
    if "master" in ARGS.pod or "gating" in ARGS.pod:
        HEALTHCHECK['tests'].remove('dcaemod')
        SMOKE_USECASES['tests'].remove('5gbulkpm')
        SMOKE_USECASES['tests'].remove('cmpv2')
        SMOKE_USECASES['tests'].remove('basic_clamp')
```