KubeCon | CloudNativeCon

North America 2022

BUILDING FOR THE ROAD AHEAD

DETROIT 2022

Windows Operational Readiness

Amim Knabben (VMware)

Xinqi Li (VMware)

Speaker introductions

KubeCon | CloudNativeCon

North America 2022

BUILDING FOR THE ROAD AHEAD

DETROIT 2022

October 24-28, 2022

Amim Knabben

Sr. Software Engineer

VMware

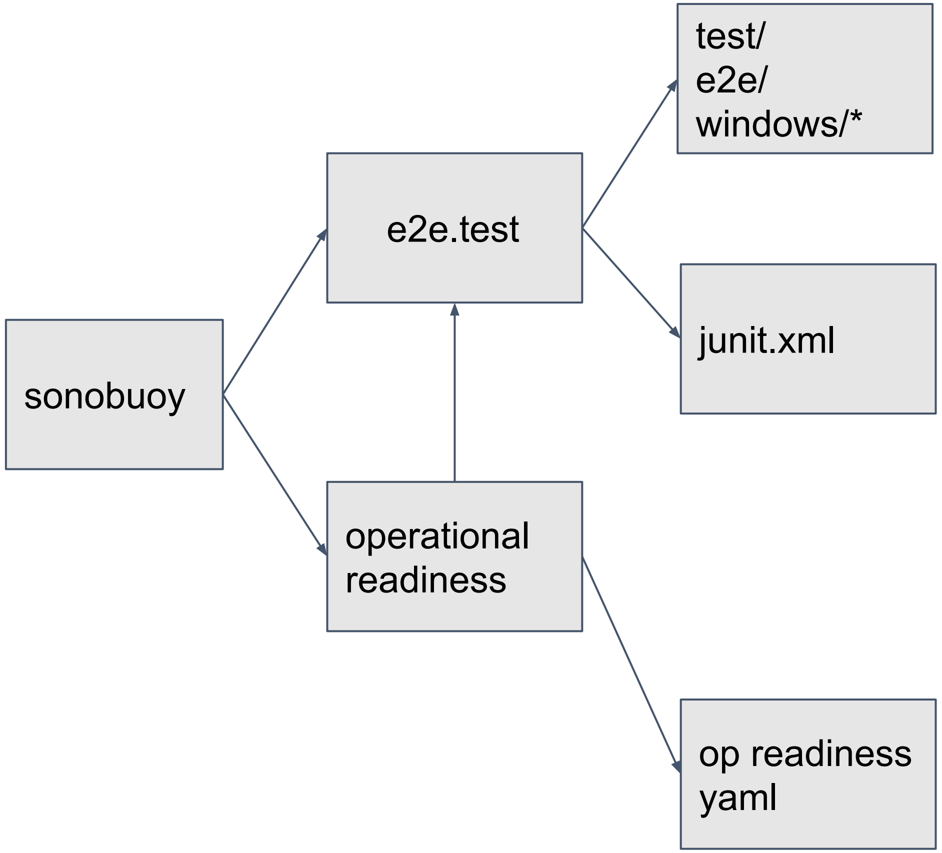
Xinqi Li

Software Engineer

VMware

Agenda

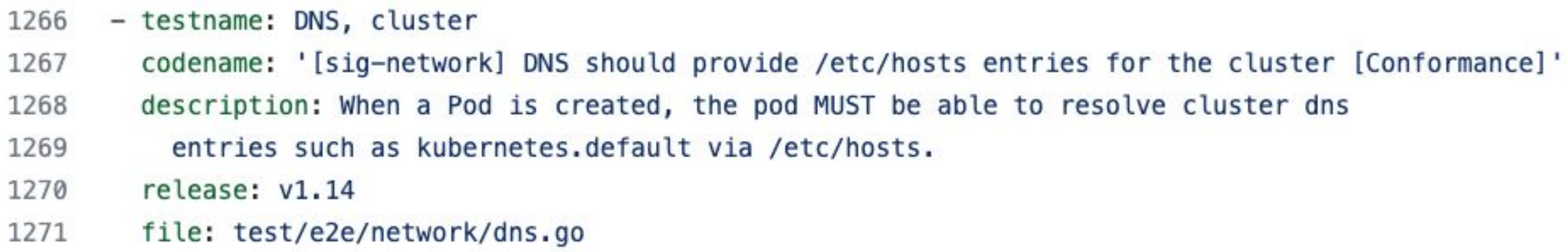
- Introduction
- KEP implementation
- The sonobuoy plugin
- Project usage
- Wrap up



Introduction

Conformance tests on Linux

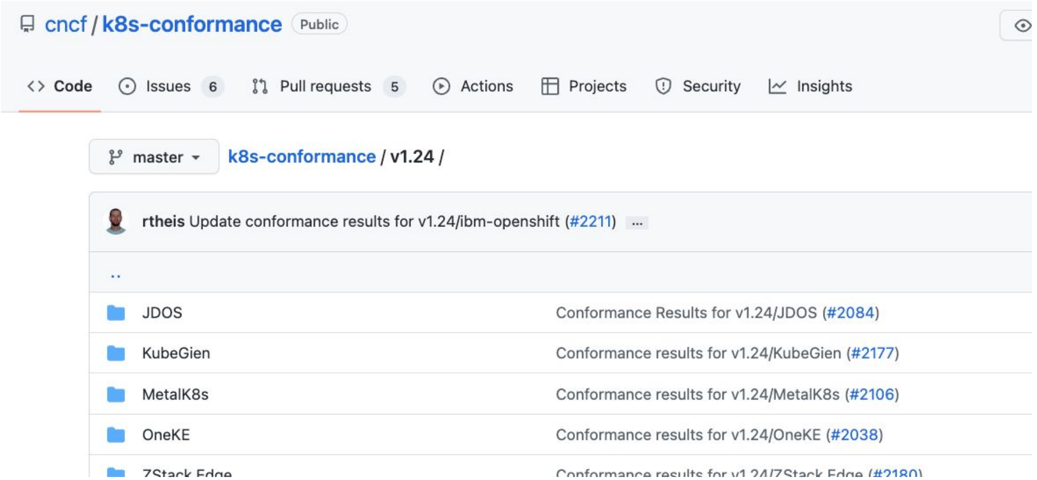
- A suite of tests is defined as Linux conformance test in Kubernetes e2e with [Conformance] tag.



```
- testname: DNS, cluster
  codename: '[sig-network] DNS should provide /etc/hosts entries for the cluster [Conformance]'
  description: When a Pod is created, the pod MUST be able to resolve cluster dns
    entries such as kubernetes.default via /etc/hosts.
  release: v1.14
  file: test/e2e/network/dns.go
```

- Each provider is encouraged to submit the conformance test result to prove the cluster behave as expected.

conformance.yaml



rtheis Update conformance results for v1.24/ibm-openshift (#2211) ...

..

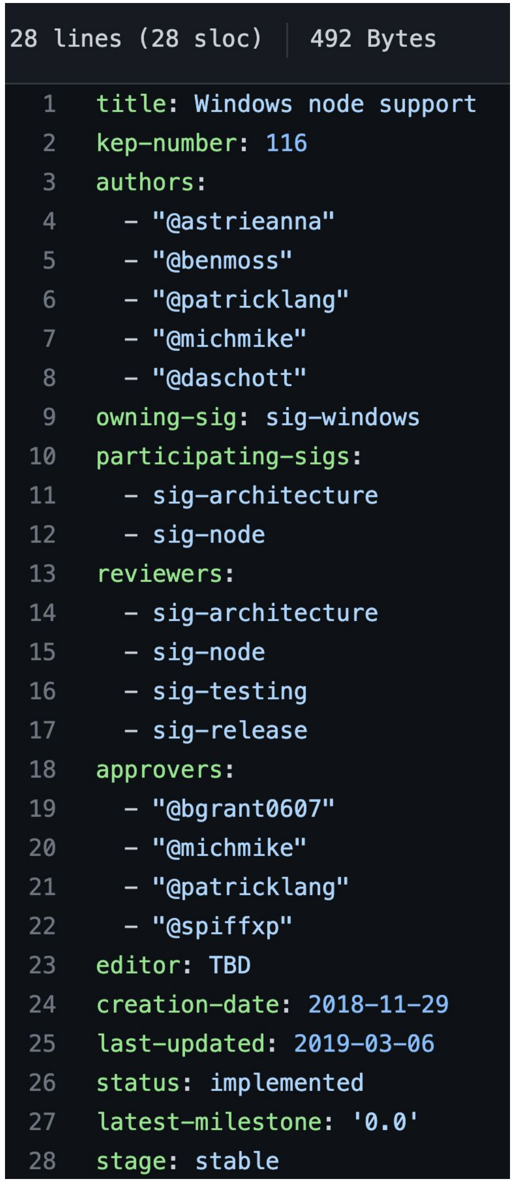
JDOS	Conformance Results for v1.24/JDOS (#2084)
KubeGien	Conformance results for v1.24/KubeGien (#2177)
MetalK8s	Conformance results for v1.24/MetalK8s (#2106)
OneKE	Conformance results for v1.24/OneKE (#2038)
ZStack Edge	Conformance results for v1.24/ZStack Edge (#2180)

No Conformance tests on Windows

- Windows is not required by official conformance testing.
- Is still important to verify that the behavior of the Windows nodes match the ones tested in the conformance suite as much as possible.
- It raises a question, for the enterprise what is considered “*as much as possible*”?

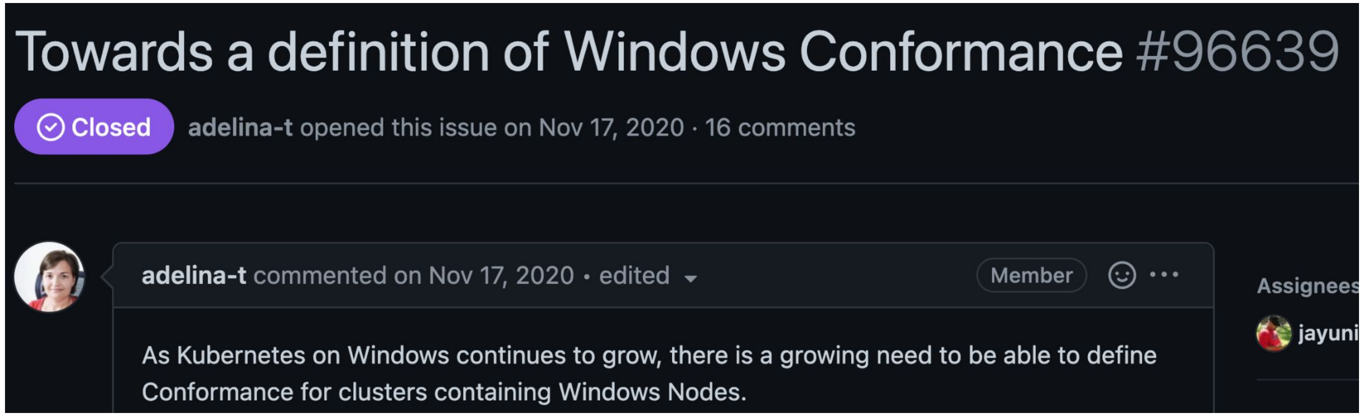
https://github.com/kubernetes/community/blob/master/contributors/devel/sig-architecture/conformance-tests.md#windows--linux-considerations

History of Windows Testing



28 lines (28 sloc) | 492 Bytes

```yaml
title: Windows node support
kep-number: 116
authors:
  - "@astrieanna"
  - "@benmoss"
  - "@patricklang"
  - "@michmike"
  - "@daschott"
owning-sig: sig-windows
participating-sigs:
  - sig-architecture
  - sig-node
reviewers:
  - sig-architecture
  - sig-node
  - sig-testing
  - sig-release
approvers:
  - "@bgrant0607"
  - "@michmike"
  - "@patricklang"
  - "@spiffxp"
editor: TBD
creation-date: 2018-11-29
last-updated: 2019-03-06
status: implemented
latest-milestone: '0.0'
stage: stable
```

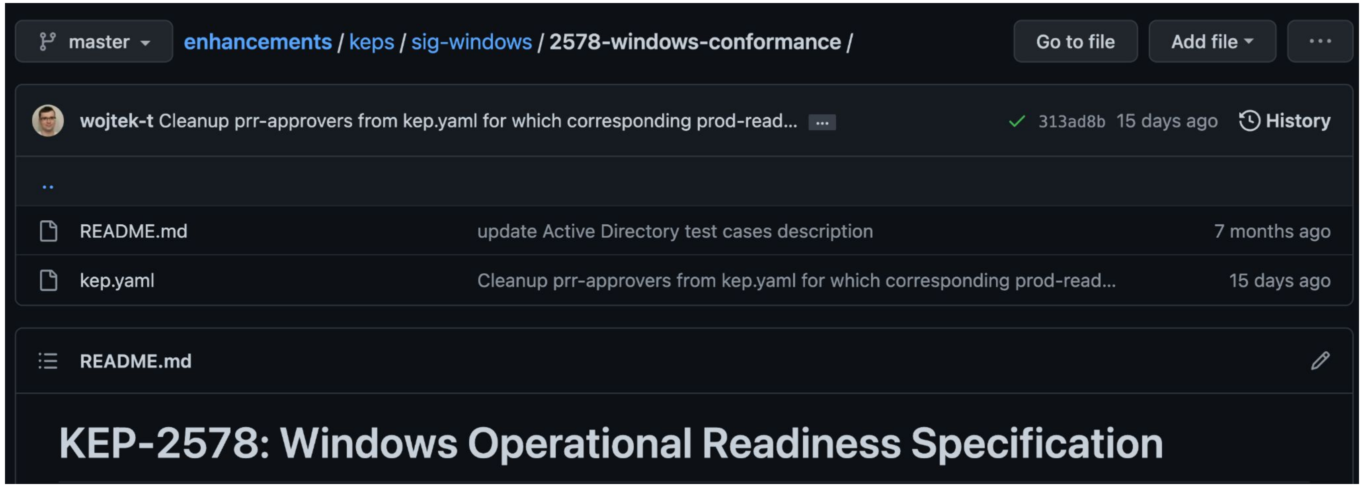


Towards a definition of Windows Conformance #96639

Closed adelina-t opened this issue on Nov 17, 2020 · 16 comments

adelina-t commented on Nov 17, 2020 • edited Member

As Kubernetes on Windows continues to grow, there is a growing need to be able to define Conformance for clusters containing Windows Nodes.



enhancements / keps / sig-windows / 2578-windows-conformance /

wojtek-t Cleanup prr-approvers from kep.yaml for which corresponding prod-read... ✓ 313ad8b 15 days ago

..

README.md	update Active Directory test cases description	7 months ago
kep.yaml	Cleanup prr-approvers from kep.yaml for which corresponding prod-read...	15 days ago

README.md

KEP-2578: Windows Operational Readiness Specification

KEP-2578

- **NOT** a CNCF official “conformance” standard.
- SIG-Windows definition of the MINIMUM Windows end-to-end TESTS required to pass in order to be considered operational.
- Architecture and definition for a testing tool implementation.

https://github.com/kubernetes/enhancements/tree/master/keps/sig-windows/2578-windows-conformance#kep-2578-windows-operational-readiness-specification

KEP-2578 – KEP categories

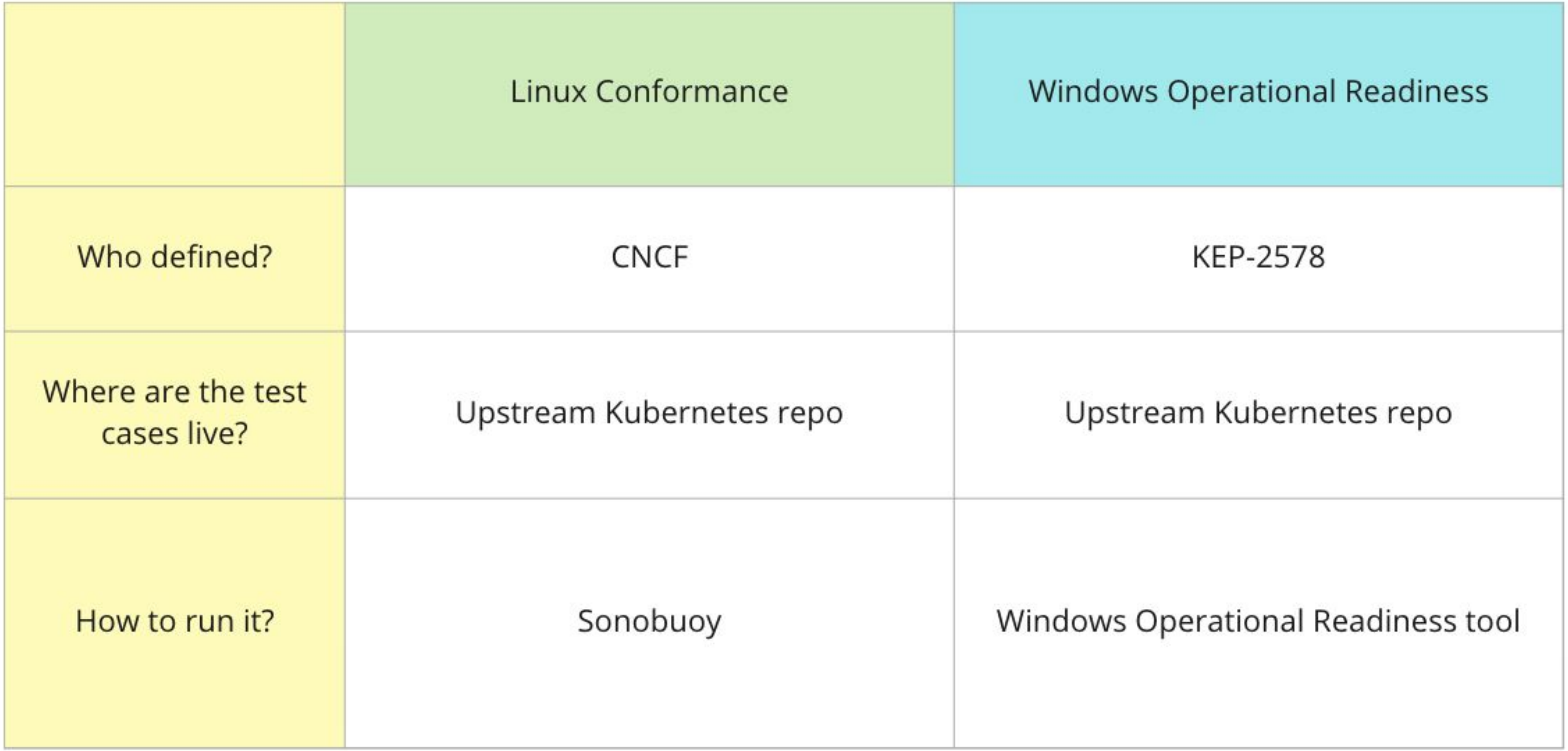
- **Required**
  - Basic networking (*Core.Network*)
  - Basic storage (*Core.Storage*)
  - Basic scheduling (*Core.Scheduling*)
  - Basic concurrent functionality (*Core.Concurrent*)

- **Optional**
  - Windows HostProcess Operational Readiness (*Extend.HostProcess*)
  - Active Directory (*Extend.ActiveDirectory*)
  - Network Policy (*Extend.NetworkPolicy*)
  - Windows Advanced Networking / Service Proxy (*Extend.ServiceProxy*)
  - Windows worker configuration

KEP-2578 – KEP categories Example

- **Basic Networking**
  - Ability to access Linux container IPs *by service IP* from Windows containers
  - Ability to access Windows container IPs *by service IP* from Linux containers
  - Ability to access Linux container IPs *by NodePort IP* from Windows containers
  - Ability to access Windows container IPs *by NodePort IP* from Linux containers
  - Ability to access Linux container IPs *by pod IP* from Windows containers
  - Ability to access Windows container IPs *by pod IP* from Linux containers
  - Ability to schedule multiple containers, with distinct IP addresses, on the same node.
  - Ability to delete and recreate services in such a way that load balancing rules for pods are recovered by whatever chosen service proxy is being utilized.
  - Ability to delete and recreate pods for StatefulSets which preserve their ability to serve as routed endpoints for services, while also having unchanging IP addresses.
  - Ability to access internal pods by internal Kubernetes DNS service endpoints.
  - Ability to access external services by Kubernetes DNS services endpoints (for non-airgapped clusters).
  - Ability to access internal pods through custom DNS records, injected by the Kubernetes pod Specification.
  - Ability to route services from pods from the EndpointSlice API.

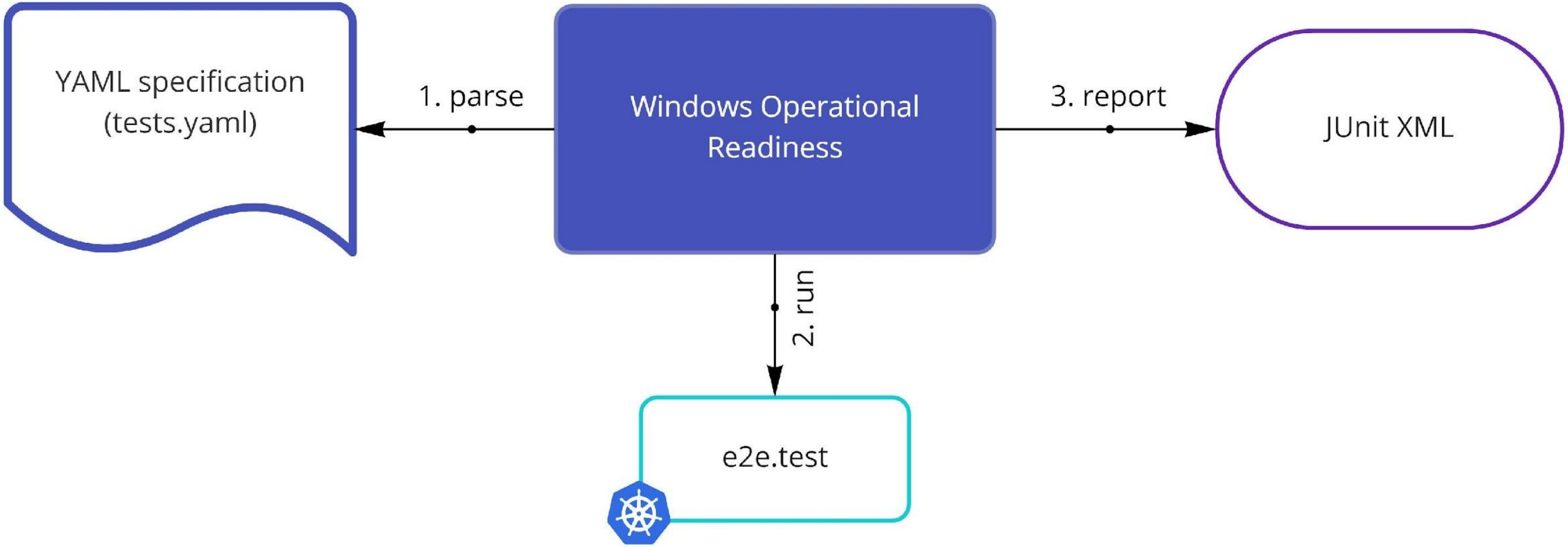
Linux Conformance VS. Windows Op-Readiness



	Linux Conformance	Windows Operational Readiness
Who defined?	CNCF	KEP-2578
Where are the test cases live?	Upstream Kubernetes repo	Upstream Kubernetes repo
How to run it?	Sonobuoy	Windows Operational Readiness tool

Implementation

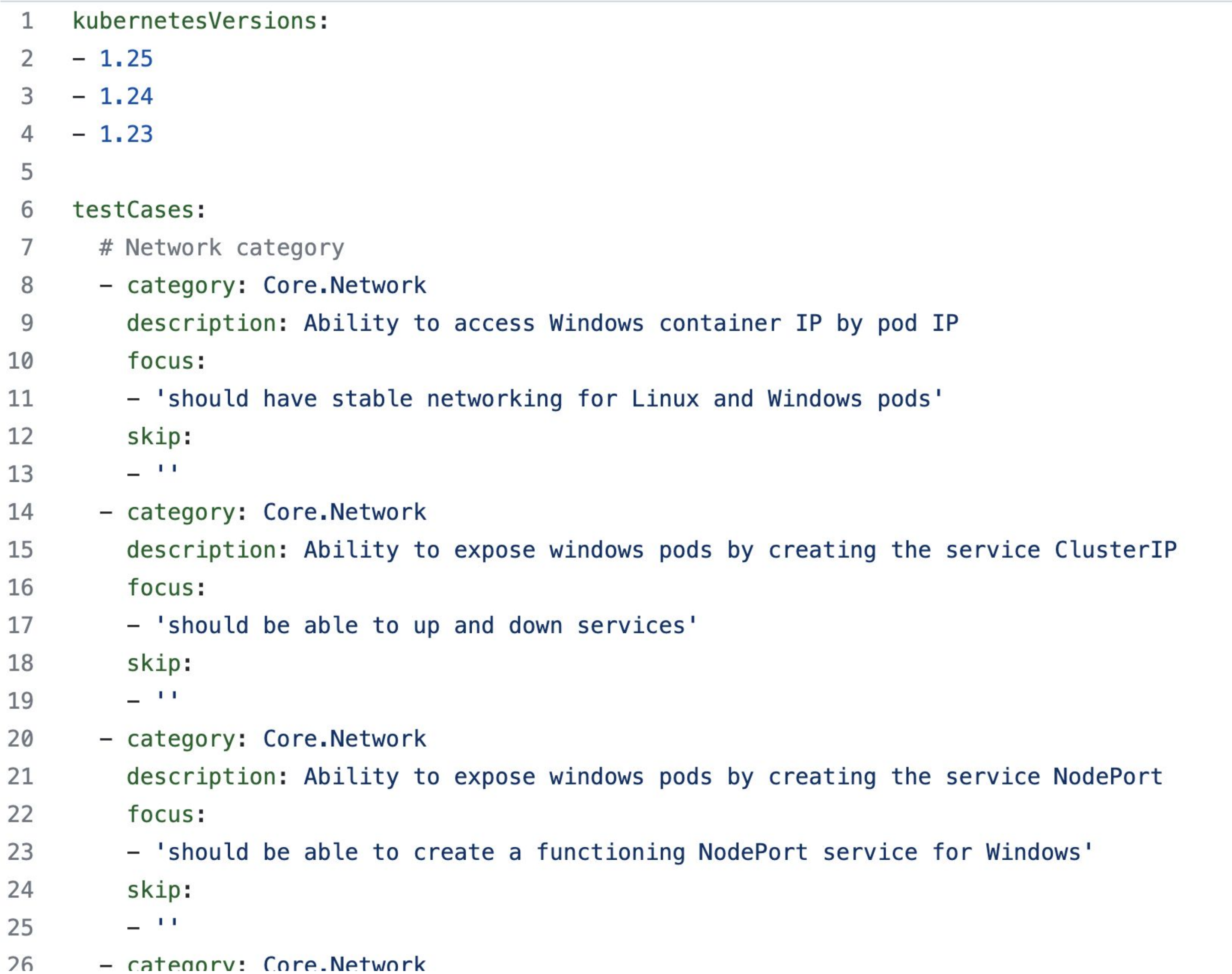
High level architecture



https://github.com/kubernetes-sigs/windows-operational-readiness

YAML definition

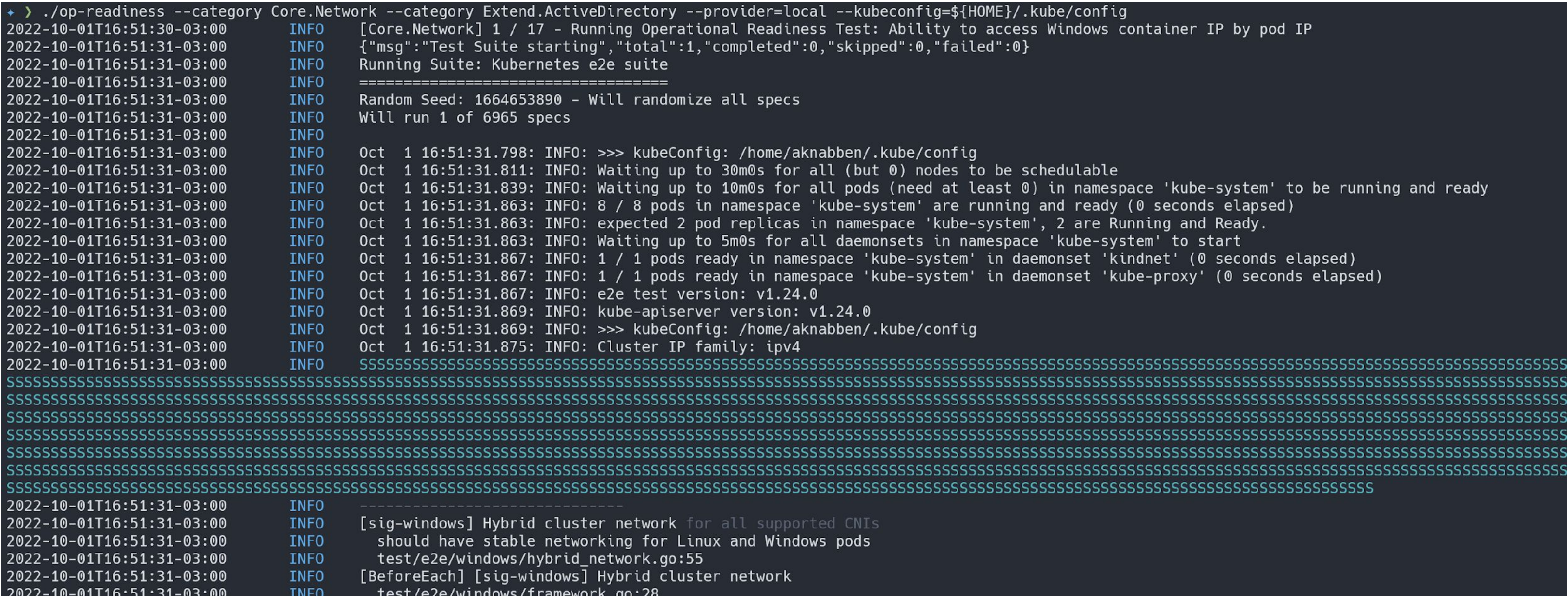
- **Category**: Represents the test group.
- **Description**: A verbose definition of the test.
- **Focus**: Ginkgo tests to focus in.
- **Skip:** Ginkgo tests to be skipped.
- **Kubernetes Version:** the kubernetes version to run this test suite.



```yaml
kubernetesVersions:
- 1.25
- 1.24
- 1.23

testCases:
  # Network category
  - category: Core.Network
    description: Ability to access Windows container IP by pod IP
    focus:
    - 'should have stable networking for Linux and Windows pods'
    skip:
    - ''
  - category: Core.Network
    description: Ability to expose windows pods by creating the service ClusterIP
    focus:
    - 'should be able to up and down services'
    skip:
    - ''
  - category: Core.Network
    description: Ability to expose windows pods by creating the service NodePort
    focus:
    - 'should be able to create a functioning NodePort service for Windows'
    skip:
    - ''
  - category: Core.Network
```

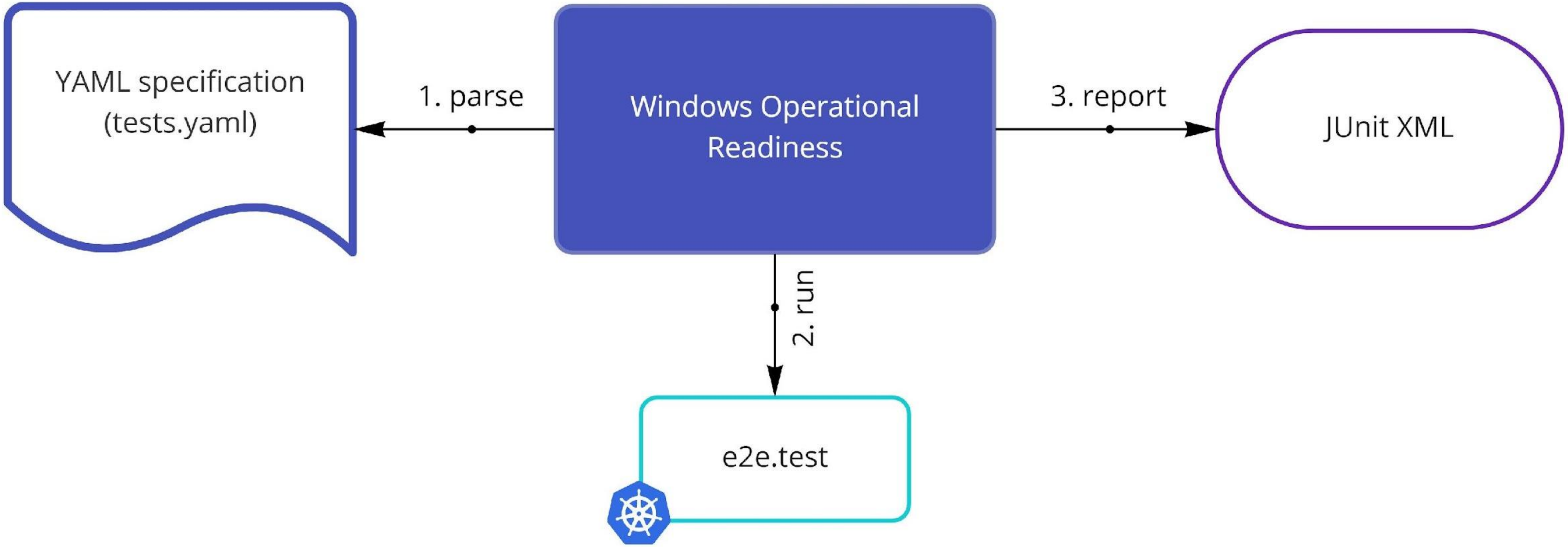
Choosing one or more categories

```
./op-readiness --provider=local --kubeconfig=<path-to-kubeconfig> --category=Core.Network --category=Extend.ActiveDirectory
```



```
✦ ❯ ./op-readiness --category Core.Network --category Extend.ActiveDirectory --provider=local --kubeconfig=${HOME}/.kube/config
2022-10-01T16:51:30-03:00       INFO    [Core.Network] 1 / 17 - Running Operational Readiness Test: Ability to access Windows container IP by pod IP
2022-10-01T16:51:31-03:00       INFO    {"msg":"Test Suite starting","total":1,"completed":0,"skipped":0,"failed":0}
2022-10-01T16:51:31-03:00       INFO    Running Suite: Kubernetes e2e suite
2022-10-01T16:51:31-03:00       INFO    ===================================
2022-10-01T16:51:31-03:00       INFO    Random Seed: 1664653890 - Will randomize all specs
2022-10-01T16:51:31-03:00       INFO    Will run 1 of 6965 specs
2022-10-01T16:51:31-03:00       INFO
2022-10-01T16:51:31-03:00       INFO    Oct  1 16:51:31.798: INFO: >>> kubeConfig: /home/aknabben/.kube/config
2022-10-01T16:51:31-03:00       INFO    Oct  1 16:51:31.811: INFO: Waiting up to 30m0s for all (but 0) nodes to be schedulable
2022-10-01T16:51:31-03:00       INFO    Oct  1 16:51:31.839: INFO: Waiting up to 10m0s for all pods (need at least 0) in namespace 'kube-system' to be running and ready
2022-10-01T16:51:31-03:00       INFO    Oct  1 16:51:31.863: INFO: 8 / 8 pods in namespace 'kube-system' are running and ready (0 seconds elapsed)
2022-10-01T16:51:31-03:00       INFO    Oct  1 16:51:31.863: INFO: expected 2 pod replicas in namespace 'kube-system', 2 are Running and Ready.
2022-10-01T16:51:31-03:00       INFO    Oct  1 16:51:31.863: INFO: Waiting up to 5m0s for all daemonsets in namespace 'kube-system' to start
2022-10-01T16:51:31-03:00       INFO    Oct  1 16:51:31.867: INFO: 1 / 1 pods ready in namespace 'kube-system' in daemonset 'kindnet' (0 seconds elapsed)
2022-10-01T16:51:31-03:00       INFO    Oct  1 16:51:31.867: INFO: 1 / 1 pods ready in namespace 'kube-system' in daemonset 'kube-proxy' (0 seconds elapsed)
2022-10-01T16:51:31-03:00       INFO    Oct  1 16:51:31.867: INFO: e2e test version: v1.24.0
2022-10-01T16:51:31-03:00       INFO    Oct  1 16:51:31.869: INFO: kube-apiserver version: v1.24.0
2022-10-01T16:51:31-03:00       INFO    Oct  1 16:51:31.869: INFO: >>> kubeConfig: /home/aknabben/.kube/config
2022-10-01T16:51:31-03:00       INFO    Oct  1 16:51:31.875: INFO: Cluster IP family: ipv4
2022-10-01T16:51:31-03:00       INFO    SSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSS
SSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSS
SSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSS
SSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSS
SSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSS
SSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSS
SSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSS
SSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSS
2022-10-01T16:51:31-03:00       INFO    ------------------------------
2022-10-01T16:51:31-03:00       INFO    [sig-windows] Hybrid cluster network for all supported CNIs
2022-10-01T16:51:31-03:00       INFO      should have stable networking for Linux and Windows pods
2022-10-01T16:51:31-03:00       INFO      test/e2e/windows/hybrid_network.go:55
2022-10-01T16:51:31-03:00       INFO    [BeforeEach] [sig-windows] Hybrid cluster network
2022-10-01T16:51:31-03:00       INFO      test/e2e/windows/framework.go:28
```

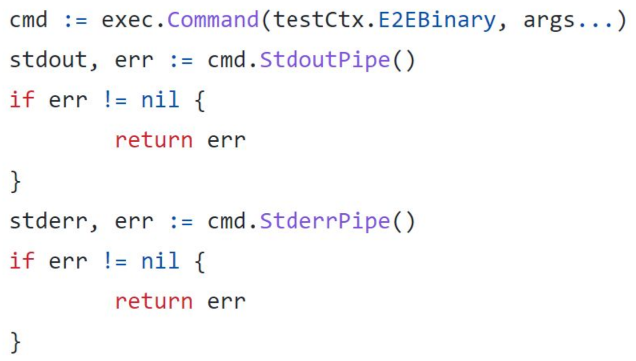
High level architecture



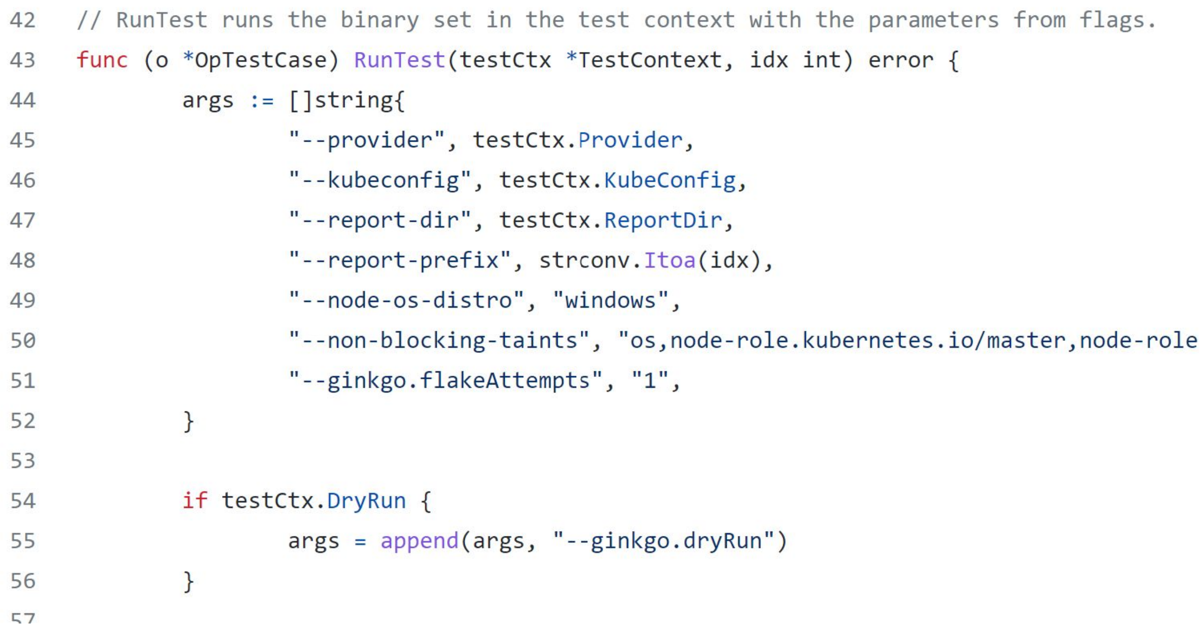
https://github.com/kubernetes-sigs/windows-operational-readiness

Calling upstream tests

- The ./e2e.test binary is called internally.
- Windows parameters are set by default.
- Filters and skip are applied per test.



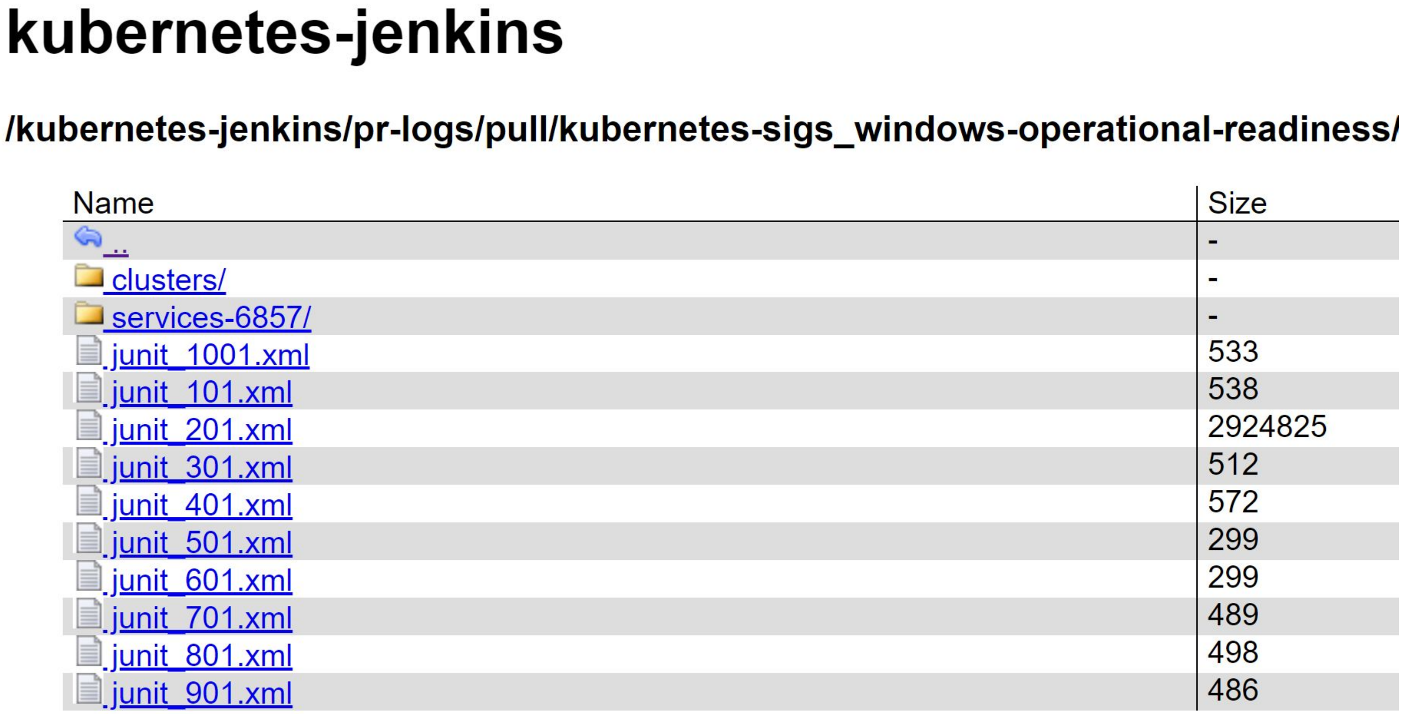
```go
cmd := exec.Command(testCtx.E2EBinary, args...)
stdout, err := cmd.StdoutPipe()
if err != nil {
	return err
}
stderr, err := cmd.StderrPipe()
if err != nil {
	return err
}
```



```go
// RunTest runs the binary set in the test context with the parameters from flags.
func (o *OpTestCase) RunTest(testCtx *TestContext, idx int) error {
	args := []string{
		"--provider", testCtx.Provider,
		"--kubeconfig", testCtx.KubeConfig,
		"--report-dir", testCtx.ReportDir,
		"--report-prefix", strconv.Itoa(idx),
		"--node-os-distro", "windows",
		"--non-blocking-taints", "os,node-role.kubernetes.io/master,node-role
		"--ginkgo.flakeAttempts", "1",
	}

	if testCtx.DryRun {
		args = append(args, "--ginkgo.dryRun")
	}
```

Report generated by JUnit

- If **–report-dir** or **ARTIFACTS** env var set.
- Junit reports will be saved and dumped.
- Can be parsed by dashboard.



kubernetes-jenkins

/kubernetes-jenkins/pr-logs/pull/kubernetes-sigs_windows-operational-readiness/

Name	Size
..	-
clusters/	-
services-6857/	-
junit_1001.xml	533
junit_101.xml	538
junit_201.xml	2924825
junit_301.xml	512
junit_401.xml	572
junit_501.xml	299
junit_601.xml	299
junit_701.xml	489
junit_801.xml	498
junit_901.xml	486

2 Ways to run Windows Op-readiness tests

- 1. Run the “windows operational readiness” binary directly

```
$ go run op-readiness
```

- 2. Run as a Sonobuoy plugin

```
$ sonobuoy run
```

https://github.com/kubernetes-sigs/windows-operational-readiness

Sonobuoy plugin

Sonobuoy plugin

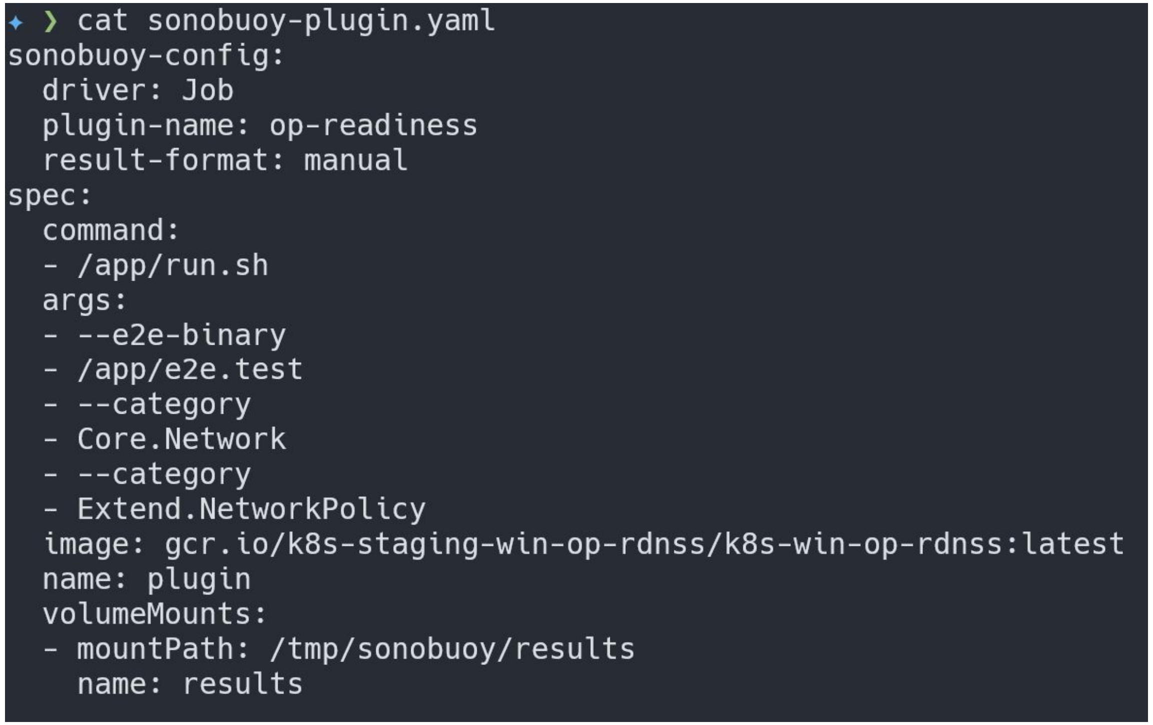
SONOBUOY

- Advantages of running as a Pod having in-cluster access.
- Tooling and support around report extracting, parsing and publishing.
- Sonobuoy is the **STANDARD** tool for running and submitting Conformance tests.

Running Operational Readiness via Sonobuoy

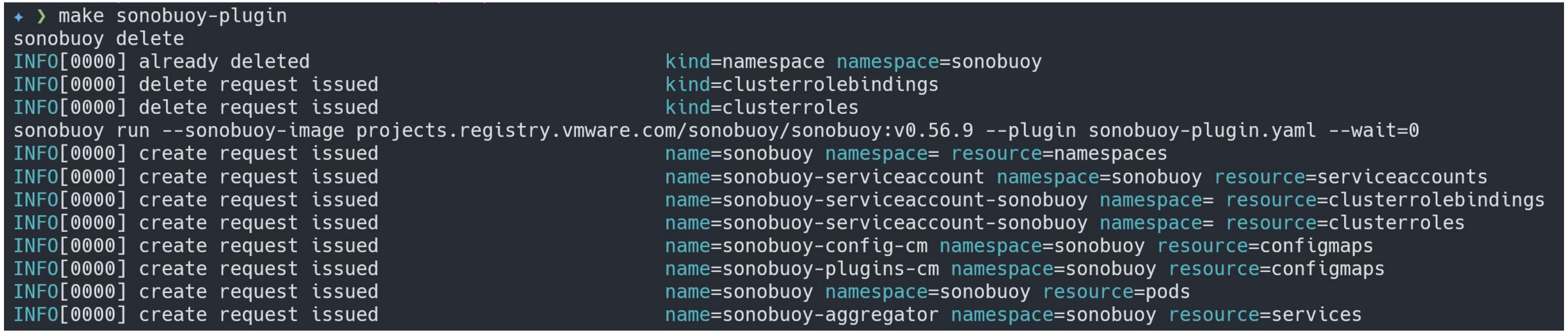
- Define the op readiness configuration (sonobuoy plugin):
  - Core.Network
  - Extend.NetworkPolicy

- Change binary parameters if needed (spec field)
- A new image is published on master commit, get the latest version at gcr.io/k8s-staging-win-op-rdnss/k8s-win-op-rdnss:latest



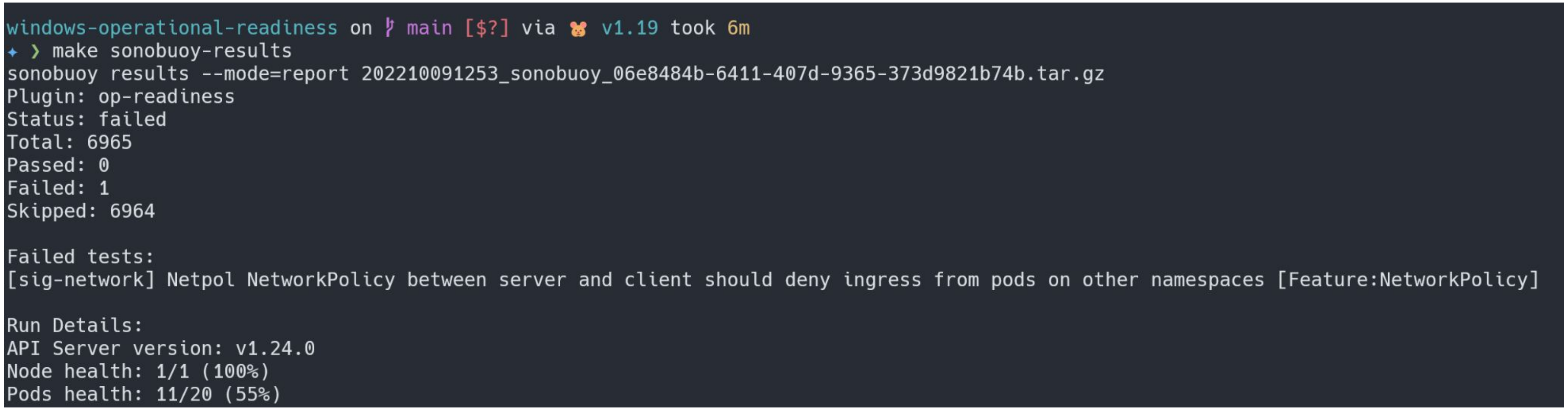
```
✦ ❯ cat sonobuoy-plugin.yaml
sonobuoy-config:
  driver: Job
  plugin-name: op-readiness
  result-format: manual
spec:
  command:
  - /app/run.sh
  args:
  - --e2e-binary
  - /app/e2e.test
  - --category
  - Core.Network
  - --category
  - Extend.NetworkPolicy
  image: gcr.io/k8s-staging-win-op-rdnss/k8s-win-op-rdnss:latest
  name: plugin
  volumeMounts:
  - mountPath: /tmp/sonobuoy/results
    name: results
```

Running the plugin



```
✦ ❯ make sonobuoy-plugin
sonobuoy delete
INFO[0000] already deleted                               kind=namespace namespace=sonobuoy
INFO[0000] delete request issued                         kind=clusterrolebindings
INFO[0000] delete request issued                         kind=clusterroles
sonobuoy run --sonobuoy-image projects.registry.vmware.com/sonobuoy/sonobuoy:v0.56.9 --plugin sonobuoy-plugin.yaml --wait=0
INFO[0000] create request issued                         name=sonobuoy namespace= resource=namespaces
INFO[0000] create request issued                         name=sonobuoy-serviceaccount namespace=sonobuoy resource=serviceaccounts
INFO[0000] create request issued                         name=sonobuoy-serviceaccount-sonobuoy namespace= resource=clusterrolebindings
INFO[0000] create request issued                         name=sonobuoy-serviceaccount-sonobuoy namespace= resource=clusterroles
INFO[0000] create request issued                         name=sonobuoy-config-cm namespace=sonobuoy resource=configmaps
INFO[0000] create request issued                         name=sonobuoy-plugins-cm namespace=sonobuoy resource=configmaps
INFO[0000] create request issued                         name=sonobuoy namespace=sonobuoy resource=pods
INFO[0000] create request issued                         name=sonobuoy-aggregator namespace=sonobuoy resource=services
```

- Use **make sonobuoy-plugin** to run in-cluster.
- Wait and run **make sonobuoy-results** will summarize the failed tests.



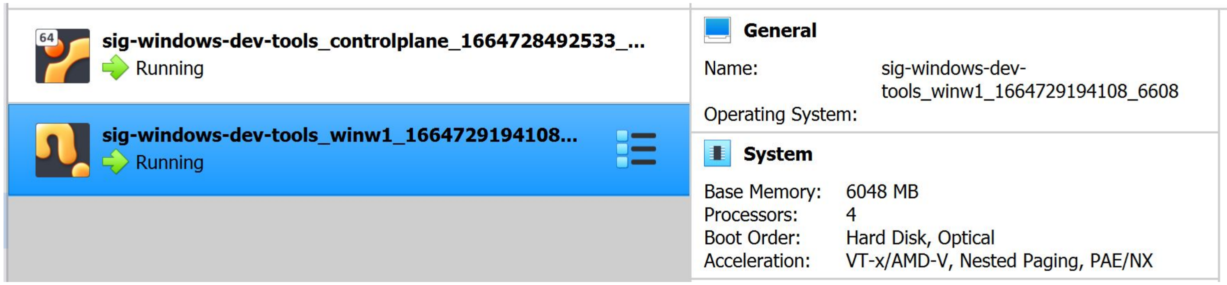
```
windows-operational-readiness on ᚠ main [$?] via 🐹 v1.19 took 6m
✦ ❯ make sonobuoy-results
sonobuoy results --mode=report 202210091253_sonobuoy_06e8484b-6411-407d-9365-373d9821b74b.tar.gz
Plugin: op-readiness
Status: failed
Total: 6965
Passed: 0
Failed: 1
Skipped: 6964

Failed tests:
[sig-network] Netpol NetworkPolicy between server and client should deny ingress from pods on other namespaces [Feature:NetworkPolicy]

Run Details:
API Server version: v1.24.0
Node health: 1/1 (100%)
Pods health: 11/20 (55%)
```

Project usage

Windows dev tools

- The easiest way to learn internals of Kubernetes on Windows locally, no cloud required !!!
- Run a Kubernetes Windows cluster locally.
- Use vagrant to create a Hybrid OS cluster.
- All is required is a robust PC and Virtualbox
- Easy to start with **make**.
- Easy to run the windows operational readiness.

https://sigs.k8s.io/sig-windows-dev-tools



sig-windows-dev-tools_controlplane_1664728492533_...
Running

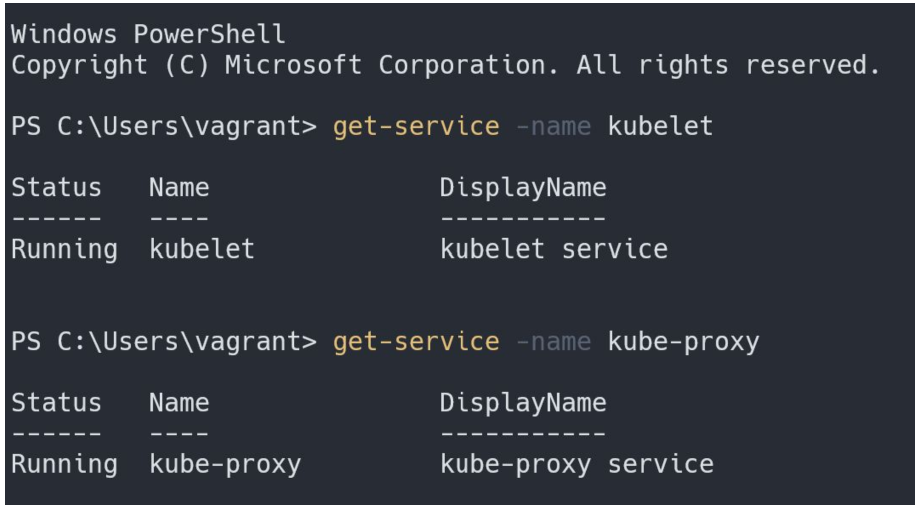
sig-windows-dev-tools_winw1_1664729194108...
Running

General

Name: sig-windows-dev-tools_winw1_1664729194108_6608
Operating System:

System

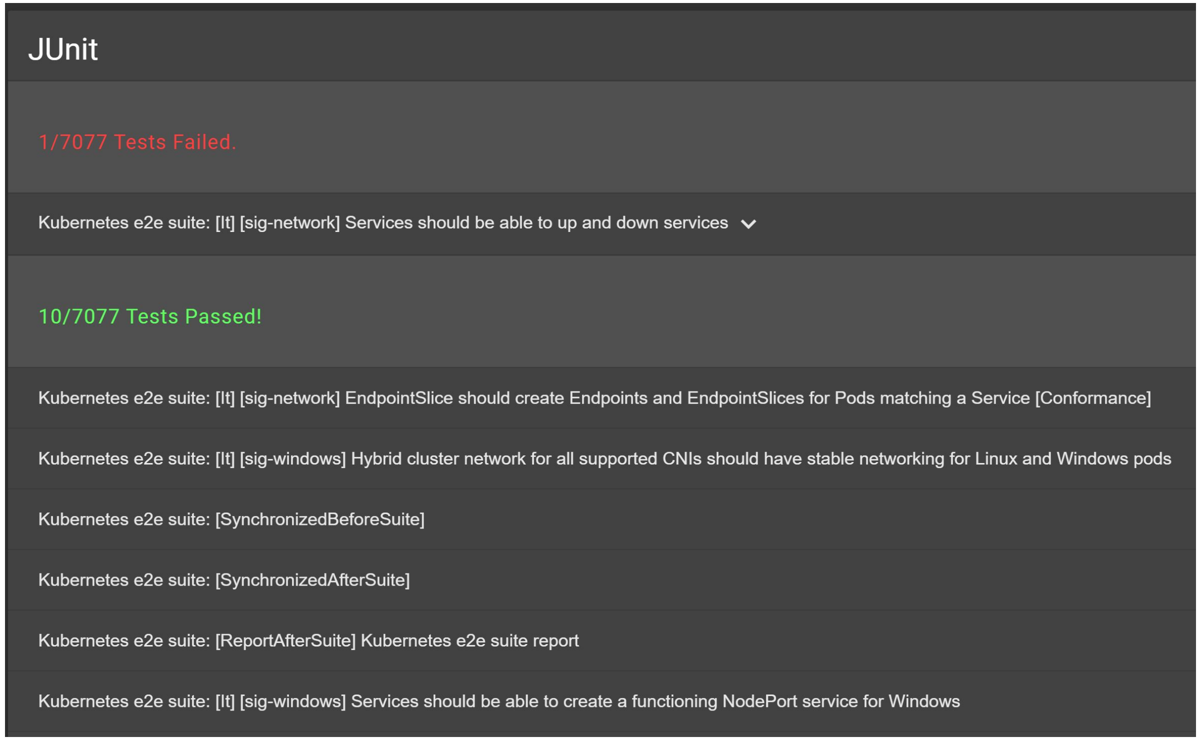
Base Memory: 6048 MB
Processors: 4
Boot Order: Hard Disk, Optical
Acceleration: VT-x/AMD-V, Nested Paging, PAE/NX



```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

PS C:\Users\vagrant> get-service -name kubelet

Status   Name               DisplayName
------   ----               -----------
Running  kubelet            kubelet service


PS C:\Users\vagrant> get-service -name kube-proxy

Status   Name               DisplayName
------   ----               -----------
Running  kube-proxy         kube-proxy service
```

Prow jobs

- Added a new test and want to test upstream?
- Run it on ClusterAPI on Azure.
- The project has a presubmit job.
- The job is NOT enabled by default.
- Run manually via Prow bot.



JUnit

1/7077 Tests Failed.

Kubernetes e2e suite: [It] [sig-network] Services should be able to up and down services

10/7077 Tests Passed!

Kubernetes e2e suite: [It] [sig-network] EndpointSlice should create Endpoints and EndpointSlices for Pods matching a Service [Conformance]

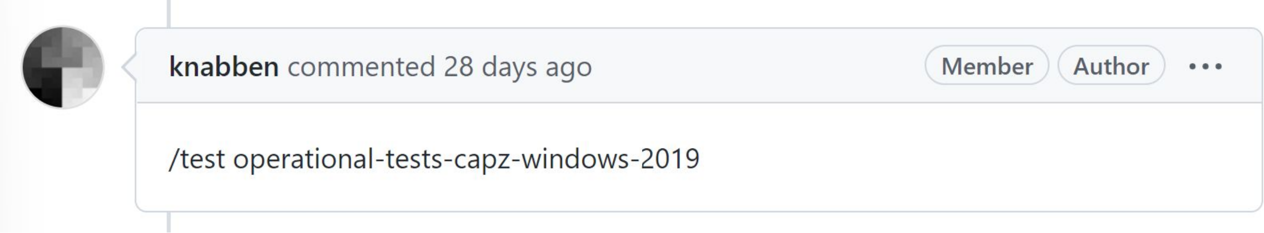
Kubernetes e2e suite: [It] [sig-windows] Hybrid cluster network for all supported CNIs should have stable networking for Linux and Windows pods

Kubernetes e2e suite: [SynchronizedBeforeSuite]

Kubernetes e2e suite: [SynchronizedAfterSuite]

Kubernetes e2e suite: [ReportAfterSuite] Kubernetes e2e suite report

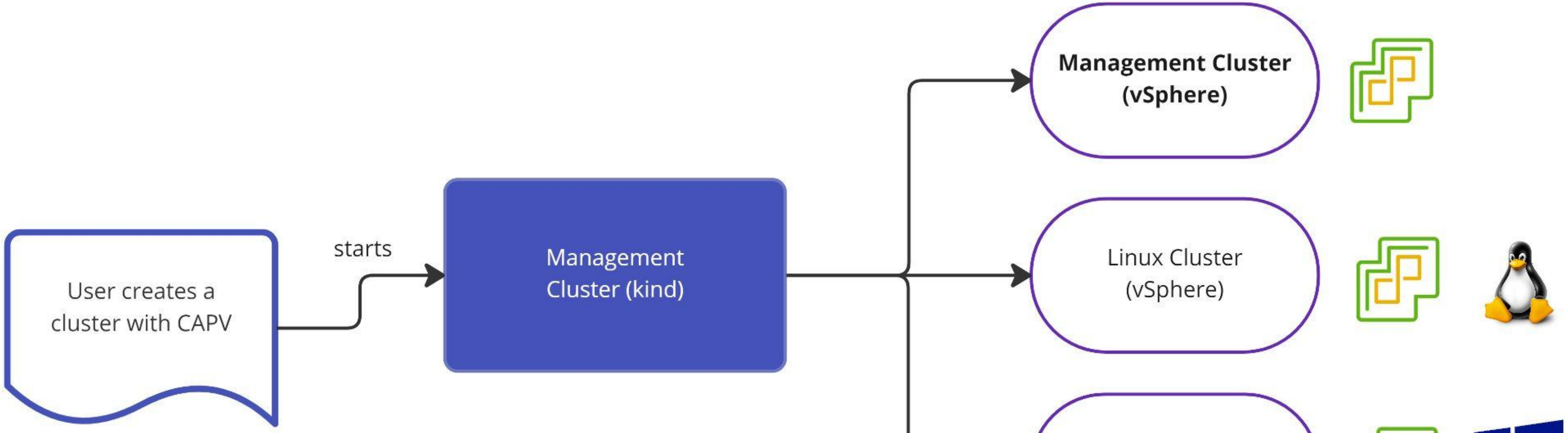
Kubernetes e2e suite: [It] [sig-windows] Services should be able to create a functioning NodePort service for Windows



knabben commented 28 days ago

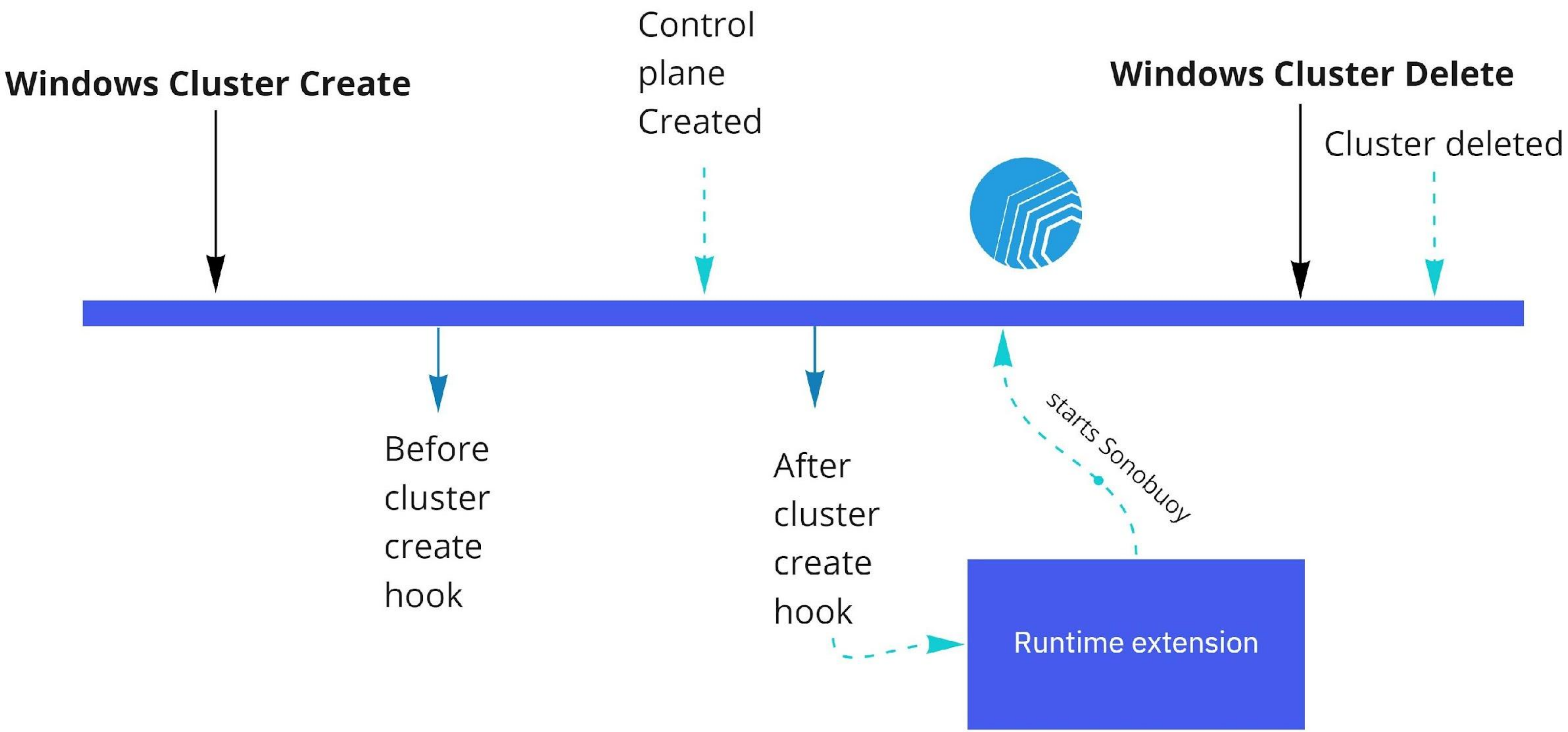
/test operational-tests-capz-windows-2019

ClusterAPI (vSphere)


User creates a cluster with CAPV
starts
Management Cluster (kind)
Management Cluster (vSphere)
Linux Cluster (vSphere)
Windows Cluster (vSphere)

- Kubernetes project to bring declarative, Kubernetes-style APIs to cluster creation, configuration, and management.

ClusterAPI - Runtime Extensions (exp)



https://github.com/kubernetes-sigs/windows-operational-readiness/tree/main/exp

Wrap up

Special thanks

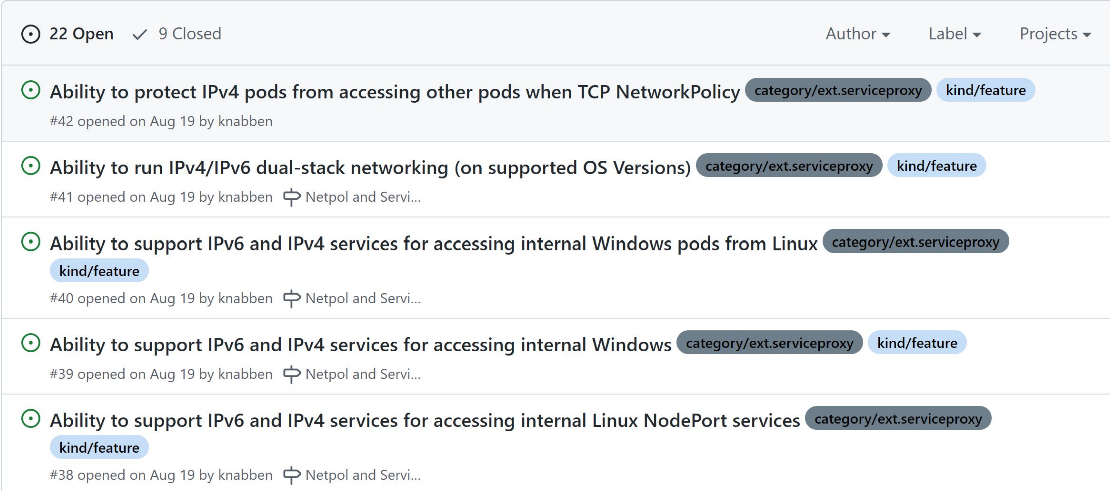
- SIG-Windows + SIG-Testing + SIG-Network
  - **Jay Vyas (VMware)** - for all mentoring and leadership.
  - **Mark Rossetti (Microsoft)** and **James Sturtevant (Microsoft) -** for mentoring and guidance on Windows + Kubernetes.
  - **Aravindh Puthiyaparambil (RedHat), Claudiu Belu** (**Cloudbase**) and **Luther Monson (Spectro Cloud), Douglas Landgraf (RedHat)** for working on new Windows features and always ready to help the community.
  - **Adelina** and **Ben the Elder** for early discussions on this topic !!!
  - **Antonio Ojea** for helping with networking e2e tests.

How To Contribute

- Right now **~40%** of the tests are implemented
- There are **~20** issues open for the missing tests
- First comers are welcome.
- We are happy to mentor and help onboarding!

- https://sigs.k8s.io/windows-operational-readiness
- Join us on **#sig-windows** at Kubernetes Slack!

help wanted good first issue



⊙ 22 Open ✓ 9 Closed Author ▾ Label ▾ Projects ▾

⊙ Ability to protect IPv4 pods from accessing other pods when TCP NetworkPolicy category/ext.serviceproxy kind/feature
#42 opened on Aug 19 by knabben

⊙ Ability to run IPv4/IPv6 dual-stack networking (on supported OS Versions) category/ext.serviceproxy kind/feature
#41 opened on Aug 19 by knabben Netpol and Servi...

⊙ Ability to support IPv6 and IPv4 services for accessing internal Windows pods from Linux category/ext.serviceproxy kind/feature
#40 opened on Aug 19 by knabben Netpol and Servi...

⊙ Ability to support IPv6 and IPv4 services for accessing internal Windows category/ext.serviceproxy kind/feature
#39 opened on Aug 19 by knabben Netpol and Servi...

⊙ Ability to support IPv6 and IPv4 services for accessing internal Linux NodePort services category/ext.serviceproxy kind/feature
#38 opened on Aug 19 by knabben Netpol and Servi...

Interested in learning K8s on windows? Try out

https://github.com/kubernetes-sigs/sig-windows-dev-tools

Thank you

Amim Knabben (VMware)

Xinqi Li (VMware)

Please scan the QR Code above to leave feedback on this session