KubeEdge BP Test Documents

- Introduction
- Akarino Test Group Information
- Overall Test Architecture
    - Test Bed
    - Test Framework
    - Traffic Generator
- Test API description
    - KubeEdge CI/CD tests
        - Test Procedure
        - Test Results
    - BluVal Tests
        - Lynis
        - Vuls
        - Kube-Hunter
        - Conformance Test (Sonobuoy)
- Test Dashboards
- Additional Testing
- Bottlenecks/Errata
- Applied Fixes
    - Kube-Hunter
        - KHV002
        - CAP_NET_RAW
        - KHV050 https://aquasecurity.github.io/kube-hunter/kb/KHV050.html
- Bluval CI script
- HRDN-7220

Introduction

This wiki documents KubeEdge BP test environment set up and test results for release 4.

All test results and logs have been pushed to Akraino NEXUS:

https://nexus.akraino.org/content/sites/logs/futurewei/kubeedgees/36/

Akarino Test Group Information

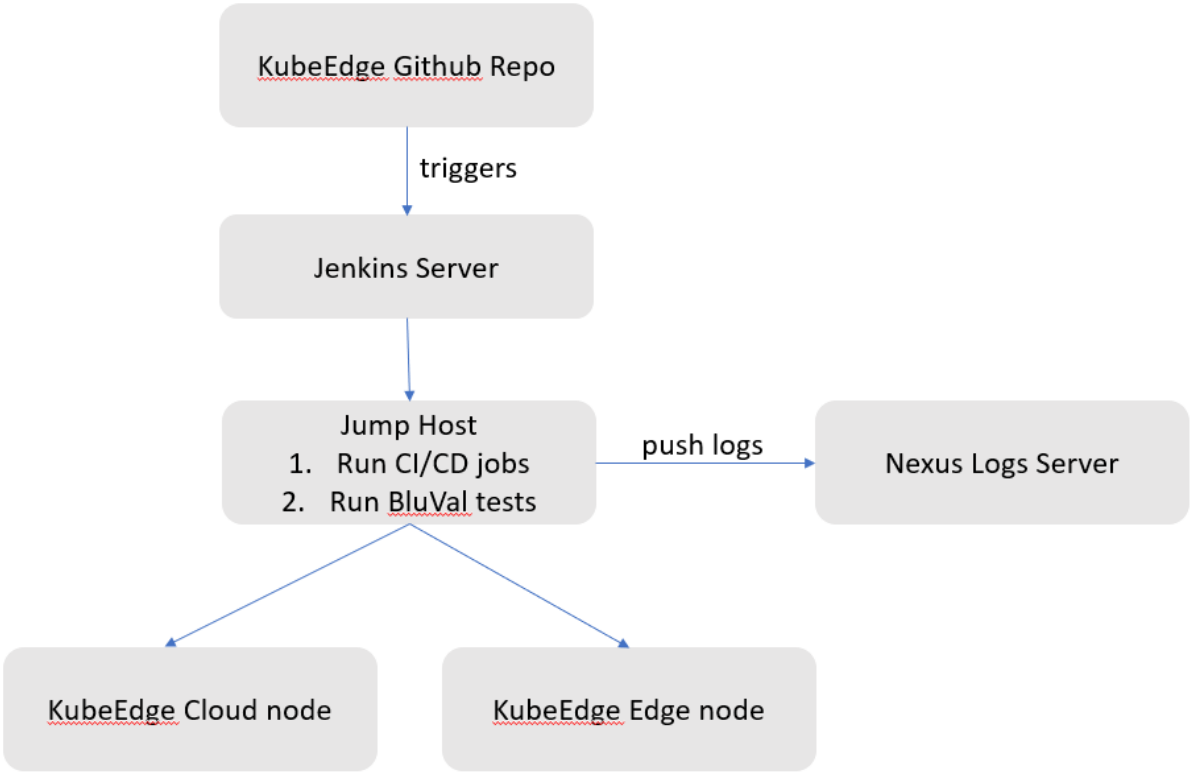
N/A

Overall Test Architecture

2 independent test labs have been set up:

1. A lab created exclusively within AWS
2. Akraino provided lab

They share the same high level test architecture:



Test Bed

In the AWS lab, all servers are EC2 instances:

Node Type	OS
Jump Host	Ubuntu 18.04
KubeEdge Cloud	Ubuntu 20.04
KubeEdge Edge	Ubuntu 20.04

Test Framework

Robot framework is used in BluVal.

Traffic Generator

N/A

Test API description

KubeEdge CI/CD tests

Test Procedure

1. Build for both x86 and ARM.
2. Unit tests.
3. Integration tests.
4. End to end tests.

Test Results

Pass

BluVal Tests

Lynis

Pass

Vuls

We used Ubuntu 20.04. All the packages have been updated or upgraded to latest version in the repo.

There are 4 CVEs with CVSS score > 9.0. These require upstream kernel patches, and exceptions have been requested here:

Akraino CVE Vulnerability Exception Request

Kube-Hunter

Fixed 3 vulnerabilities:

KHV002, KHV050 & CAP_NET_RAW.

No more to fix.

Conformance Test (Sonobuoy)

Sonobuoy does not apply to KubeEdge.

An exception has been granted:

Akraino BluVal Exception Request

Reason:

- Sonobuoy assumes the all nodes are within a Layer 2 network, which is the case for the standard Kubernetes environment.
- KubeEdge solves a different problem where typically the edge nodes are behind corporate firewalls. And cloud node do not have direct access to the edge nodes due to security and permission restrictions.

Test Dashboards

Single pane view of how the test score looks like for the Blue print.

Total Tests	Test Executed	Pass	Fail	In Progress
2	2	2	0	0

Test results and logs are all pushed to Akraino NEXUS:

https://nexus.akraino.org/content/sites/logs/futurewei/kubeedgees/36/

Additional Testing

Bottlenecks/Errata

Applied Fixes

Kube-Hunter

KHV002

https://aquasecurity.github.io/kube-hunter/kb/KHV002.html

Solution:

Change the default ClusterRole system:public-info-viewer

```
kubectl replace -f - <<EOF
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRole
metadata:
  annotations:
    rbac.authorization.kubernetes.io/autoupdate: "false"
  labels:
    kubernetes.io/bootstrapping: rbac-defaults
  name: system:public-info-viewer
rules:
- nonResourceURLs:
  - /healthz
  - /livez
  - /readyz
  verbs:
  - get
EOF
```

CAP_NET_RAW

Docker runtime enables Linux "NET_RAW" capability by default. Docker daemon does not have an option to disable "NET_RAW":

https://docs.docker.com/engine/reference/commandline/dockerd/#daemon-configuration-file

So we have to turn to K8s provided "PodSecurityPolicy" for help.

Solution:

Use PodSecurityPolicy

https://kubernetes.io/docs/concepts/policy/pod-security-policy/

High level steps:

1. Create a PodSecurityPolicy to drop the Linux capability "NET_RAW".
2. Create an RBAC role to allow use of the PodSecurityPolicy created in step 1.
3. Bind the RBAC role to serviceaccount "default".

Exact PodSecurityPolicy Spec we are using:

```
spec:
  allowPrivilegeEscalation: true
  fsGroup:
    rule: RunAsAny
  hostIPC: true
  hostNetwork: true
  hostPID: true
  hostPorts:
  - max: 65535
    min: 0
  privileged: true
  requiredDropCapabilities:
  - NET_RAW
  runAsUser:
    rule: RunAsAny
  seLinux:
    rule: RunAsAny
  supplementalGroups:
    rule: RunAsAny
  volumes:
  - '*'
```

KHV050
https://aquasecurity.github.io/kube-hunter/kb/KHV050.html

Solution:

```
kubectl replace -f - <<EOF
apiVersion: v1
kind: ServiceAccount
metadata:
  name: default
  namespace: default
automountServiceAccountToken: false
EOF
```

Bluval CI script

Bluval

```
set -euo pipefail
cwd=$(pwd)
results_dir=$cwd/results
sudo rm -fr $results_dir
sudo rm -f $cwd/results.zip
mkdir -p $results_dir

blueprint=kubeedgees

info ()  {
    logger -s -t "run_bluval.info" "$*"
}

error () {
    logger -s -t "run_bluval.error" "$*"
    exit 1
}

cwd=$cwd/validation
cd $cwd


# update information in volumes yaml
sed -i \
    -e "/ssh_key_dir/{n; s@local: ''@local: '$SSH_KEY_DIR'@}" \
    -e "/kube_config_dir/{n; s@local: ''@local: '$K8S_CONFIG_DIR'@}" \
    -e "/custom_variables_file/{n; s@local: ''@local: '$cwd/tests/variables.yaml'@}" \
    -e "/blueprint_dir/{n; s@local: ''@local: '$cwd/bluval/'@}" \
    -e "/results_dir/{n; s@local: ''@local: '$results_dir'@}" \
    "$cwd/bluval/volumes.yaml"

sed -i \
    -e "s/host: [0-9]*.[0-9]*.[0-9]*.[0-9]*/host: $CLUSTER_MASTER_IP/" \
    -e "s/username: [A-Za-z0-9_]* /username: $SSH_USER/" \
    -e "s/password: [A-Za-z0-9_]* /password: /" \
    -e "s|ssh_keyfile: [A-Za-z0-9_]* |ssh_keyfile: /root/.ssh/id_rsa|" \
    "$cwd/tests/variables.yaml"

cat >"$cwd/bluval/bluval-kubeedgees.yaml" <<EOF
blueprint:
    name: kubeedgees
    layers:
        - os
        - k8s

    os: &os
        -
            name: lynis
            what: lynis
            optional: "False"
        -
            name: vuls
            what: vuls
            optional: "False"

    k8s: &k8s
        -
            name: kube-hunter
            what: kube-hunter
            optional: "False"
EOF

$cwd/bluval/blucon.sh $blueprint

if [ $? -ne 0 ]; then
    sudo chown -R $(id -u):$(id -g) "$results_dir"
```

```
    error "blucon.sh exited with return code $?"
fi

sudo chown -R $(id -u):$(id -g) "$results_dir"

echo $BLUEPRINT_BUILD_VERSION

source ~/.lftools/bin/activate
NEXUS_PATH="${LAB_SILO}/$blueprint/$BLUEPRINT_BUILD_VERSION"
cd "$results_dir/.." && zip -r results.zip ./results
lftools deploy nexus-zip https://nexus.akraino.org/ logs "$NEXUS_PATH" results.zip
```

HRDN-7220

Check if compilers like gcc, g++ is installed. If yes, remove them. For example on ubuntu:

sudo apt remove gcc g++