Hardening Guide - Rancher v2.3.3+

Contents

Hardening Guide for Rancher 2.3.3+ with Kubernetes 1.16

Click here to download a PDF version of this document

Overview

This document provides prescriptive guidance for hardening a production installation of Rancher v2.3.3+ with Kubernetes v1.14, v1.15 and v1.16. It outlines the configurations and controls required to address Kubernetes benchmark controls from the Center for Information Security (CIS).

For more detail about evaluating a hardened cluster against the official CIS benchmark, refer to the CIS Benchmark Rancher Self-Assessment Guide - Rancher v2.3.3+.

Profile Definitions

The following profile definitions agree with the CIS benchmarks for Kubernetes.

A profile is a set of configurations that provide a certain amount of hardening. Generally, the more hardened an environment is, the more it affects performance.

Level 1

Items in this profile intend to:

- offer practical advice appropriate for the environment;
- deliver an obvious security benefit; and
- not alter the functionality or utility of the environment beyond an acceptable margin

Level 2

Items in this profile extend the "Level 1" profile and exhibit one or more of the following characteristics:

- are intended for use in environments or use cases where security is paramount
- act as a defense in depth measure
- may negatively impact the utility or performance of the technology

---

2

1.1 - Rancher RKE Kubernetes cluster host configuration

(See Appendix A. for full ubuntu `cloud-config` example)

1.1.1 - Configure default sysctl settings on all hosts

Profile Applicability

- Level 1

Description

Configure sysctl settings to match what the kubelet would set if allowed.

Rationale

We recommend that users launch the kubelet with the `--protect-kernel-defaults` option. The settings that the kubelet initially attempts to change can be set manually.

This supports the following control:

- 2.1.7 - Ensure that the `--protect-kernel-defaults` argument is set to true (Scored)

Audit

- Verify `vm.overcommit_memory = 1`

```
sysctl vm.overcommit_memory
```

- Verify `vm.panic_on_oom = 0`

```
sysctl vm.panic_on_oom
```

- Verify `kernel.panic = 10`

```
sysctl kernel.panic
```

- Verify `kernel.panic_on_oops = 1`

```
sysctl kernel.panic_on_oops
```

- Verify `kernel.keys.root_maxkeys = 1000000`

```
sysctl kernel.keys.root_maxkeys
```

- Verify `kernel.keys.root_maxbytes = 25000000`

```
sysctl kernel.keys.root_maxbytes
```

Remediation

- Set the following parameters in `/etc/sysctl.d/90-kubelet.conf` on all nodes:

```
vm.overcommit_memory=1
vm.panic_on_oom=0
kernel.panic=10
kernel.panic_on_oops=1
kernel.keys.root_maxkeys=1000000
kernel.keys.root_maxbytes=25000000
```

- Run `sysctl -p /etc/sysctl.d/90-kubelet.conf` to enable the settings.

1.4.11 Ensure that the etcd data directory permissions are set to 700 or more restrictive

Profile Applicability

- Level 1

Description

Ensure that the etcd data directory has permissions of 700 or more restrictive.

Rationale

etcd is a highly-available key-value store used by Kubernetes deployments for persistent storage of all of its REST API objects. This data directory should be protected from any unauthorized reads or writes. It should not be readable or writable by any group members or the world.

Audit

On the etcd server node, get the etcd data directory, passed as an argument `--data-dir` , from the below command:

```
ps -ef | grep etcd
```

Run the below command (based on the etcd data directory found above). For example,

```
stat -c %a /var/lib/etcd
```

Verify that the permissions are `700` or more restrictive.

Remediation

Follow the steps as documented in 1.4.12 remediation.

1.4.12 - Ensure that the etcd data directory ownership is set to `etcd:etcd`

Profile Applicability

- Level 1

Description

Ensure that the etcd data directory ownership is set to `etcd:etcd`.

Rationale

etcd is a highly-available key-value store used by Kubernetes deployments for persistent storage of all of its REST API objects. This data directory should be protected from any unauthorized reads or writes. It should be owned by `etcd:etcd`.

Audit

On a etcd server node, get the etcd data directory, passed as an argument `--data-dir`, from the below command:

```
ps -ef | grep etcd
```

Run the below command (based on the etcd data directory found above). For example,

```
stat -c %U:%G /var/lib/etcd
```

Verify that the ownership is set to `etcd:etcd`.

Remediation

- On the etcd server node(s) add the `etcd` user:

```
useradd -c "Etcd user" -d /var/lib/etcd etcd
```

Record the uid/gid:

```
id etcd
```

- Add the following to the RKE `cluster.yml` etcd section under `services`:

```
services:
  etcd:
    uid: <etcd user uid recorded previously>
    gid: <etcd user gid recorded previously>
```

2.1 - Rancher HA Kubernetes Cluster Configuration via RKE

(See Appendix B. for full RKE `cluster.yml` example)

2.1.1 - Configure kubelet options

Profile Applicability

- Level 1

Description

Ensure Kubelet options are configured to match CIS controls.

Rationale

To pass the following controls in the CIS benchmark, ensure the appropriate flags are passed to the Kubelet.

- 2.1.1 - Ensure that the `--anonymous-auth` argument is set to false (Scored)
- 2.1.2 - Ensure that the `--authorization-mode` argument is not set to `AlwaysAllow` (Scored)
- 2.1.6 - Ensure that the `--streaming-connection-idle-timeout` argument is not set to 0 (Scored)
- 2.1.7 - Ensure that the `--protect-kernel-defaults` argument is set to true (Scored)
- 2.1.8 - Ensure that the `--make-iptables-util-chains` argument is set to true (Scored)
- 2.1.10 - Ensure that the `--event-qps` argument is set to 0 (Scored)
- 2.1.13 - Ensure that the `RotateKubeletServerCertificate` argument is set to true (Scored)
- 2.1.14 - Ensure that the Kubelet only makes use of Strong Cryptographic Ciphers (Not Scored)

Audit

Inspect the Kubelet containers on all hosts and verify that they are running with the following options:

- `--streaming-connection-idle-timeout=<duration greater than 0>`
- `--authorization-mode=Webhook`
- `--protect-kernel-defaults=true`
- `--make-iptables-util-chains=true`
- `--event-qps=0`
- `--anonymous-auth=false`
- `--feature-gates="RotateKubeletServerCertificate=true"`
- `--tls-cipher-suites="TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256,TLS_ECDHE_RSA_WITH_AES_128`

Remediation

- Add the following to the RKE `cluster.yml` kubelet section under `services`:

```
services:
  kubelet:
    generate_serving_certificate: true
    extra_args:
      feature-gates: "RotateKubeletServerCertificate=true"
      protect-kernel-defaults: "true"
      tls-cipher-suites: "TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256,TLS_ECDHE_RSA_WITH_AES_128
```

Where `<duration>` is in a form like `1800s`.

- Reconfigure the cluster:

```
rke up --config cluster.yml
```

2.1.2 - Configure kube-api options

Profile Applicability

- Level 1

Description

Ensure the RKE configuration is set to deploy the `kube-api` service with the options required for controls.

NOTE:

Enabling the `AlwaysPullImages` admission control plugin can cause degraded performance due to overhead of always pulling images. Enabling the `DenyEscalatingExec` admission control plugin will prevent the 'Launch kubectl' functionality in the UI from working.

Rationale

To pass the following controls for the kube-api server ensure RKE configuration passes the appropriate options.

- 1.1.1 - Ensure that the `--anonymous-auth` argument is set to false (Scored)
- 1.1.8 - Ensure that the `--profiling` argument is set to false (Scored)
- 1.1.11 - Ensure that the admission control plugin `AlwaysPullImages` is set (Scored)
- 1.1.12 - Ensure that the admission control plugin `DenyEscalatingExec` is set (Scored)
- 1.1.14 - Ensure that the admission control plugin `NamespaceLifecycle` is set (Scored)
- 1.1.15 - Ensure that the `--audit-log-path` argument is set as appropriate (Scored)
- 1.1.16 - Ensure that the `--audit-log-maxage` argument is set as appropriate (Scored)
- 1.1.17 - Ensure that the `--audit-log-maxbackup` argument is set as appropriate (Scored)
- 1.1.18 - Ensure that the `--audit-log-maxsize` argument is set as appropriate (Scored)
- 1.1.23 - Ensure that the `--service-account-lookup` argument is set to true (Scored)

- 1.1.24 - Ensure that the admission control plugin `PodSecurityPolicy` is set (Scored)
- 1.1.30 Ensure that the API Server only makes use of Strong Cryptographic Ciphers (Not Scored)
- 1.1.34 - Ensure that the `--encryption-provider-config` argument is set as appropriate (Scored)
- 1.1.35 - Ensure that the encryption provider is set to `aescbc` (Scored)
- 1.1.36 - Ensure that the admission control plugin `EventRateLimit` is set (Scored)
- 1.1.37 - Ensure that the `AdvancedAuditing` argument is not set to `false` (Scored)

Audit

- On nodes with the `controlplane` role inspect the `kube-apiserver` containers:

bash   docker inspect kube-apiserver

- Look for the following options in the command section of the output:

```
--anonymous-auth=false
--profiling=false
--service-account-lookup=true
--enable-admission-plugins=ServiceAccount,NamespaceLifecycle,LimitRanger,PersistentVolumeLab
--encryption-provider-config=/etc/kubernetes/ssl/encryption.yaml
--admission-control-config-file=/etc/kubernetes/admission.yaml
--audit-log-path=/var/log/kube-audit/audit-log.json
--audit-log-maxage=30
--audit-log-maxbackup=10
--audit-log-maxsize=100
--audit-log-format=json
--audit-policy-file=/etc/kubernetes/audit-policy.yaml
--tls-cipher-suites=TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256,TLS_ECDHE_RSA_WITH_AES_128_GCM_S
```

- In the `volume` section of the output ensure the bind mount is present:

/var/log/kube-audit:/var/log/kube-audit

Remediation

- In the RKE `cluster.yml` add the following directives to the `kube-api` section under `services`:

```
services:
  kube_api:
    always_pull_images: true
    pod_security_policy: true
    service_node_port_range: 30000-32767
    event_rate_limit:
      enabled: true
```

```
    audit_log:
      enabled: true
    secrets_encryption_config:
      enabled: true
    extra_args:
      anonymous-auth: "false"
      enable-admission-plugins: "ServiceAccount,NamespaceLifecycle,LimitRanger,PersistentVol
      profiling: "false"
      service-account-lookup: "true"
      tls-cipher-suites: "TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256,TLS_ECDHE_RSA_WITH_AES_128
    extra_binds:
      - "/opt/kubernetes:/opt/kubernetes"
```

For k8s 1.14 `enable-admission-plugins` should be

```
      enable-admission-plugins: "ServiceAccount,NamespaceLifecycle,LimitRanger,PersistentV
```

- Reconfigure the cluster:

```
rke up --config cluster.yml
```

NOTE:

Files that are placed in `/opt/kubernetes` need to be mounted in using the `extra_binds` functionality in RKE.

2.1.3 - Configure scheduler options

Profile Applicability

- Level 1

Description

Set the appropriate options for the Kubernetes scheduling service.

NOTE: Setting `--address` to `127.0.0.1` will prevent Rancher cluster monitoring from scraping this endpoint.

Rationale

To address the following controls on the CIS benchmark, the command line options should be set on the Kubernetes scheduler.

- 1.2.1 - Ensure that the `--profiling` argument is set to `false` (Scored)
- 1.2.2 - Ensure that the `--address` argument is set to `127.0.0.1` (Scored)

Audit

- On nodes with the `controlplane` role: inspect the `kube-scheduler` containers:

```
docker inspect kube-scheduler
```

- Verify the following options are set in the `command` section.

```
--profiling=false
--address=127.0.0.1
```

Remediation

- In the RKE `cluster.yml` file ensure the following options are set:

```yaml
services:
  scheduler:
    extra_args:
      profiling: "false"
      address: "127.0.0.1"
```

- Reconfigure the cluster:

```
rke up --config cluster.yml
```

2.1.4 - Configure controller options

Profile Applicability

- Level 1

Description

Set the appropriate arguments on the Kubernetes controller manager.

5*NOTE:** Setting `--address` to `127.0.0.1` will prevent Rancher cluster monitoring from scraping this endpoint.

Rationale

To address the following controls the options need to be passed to the Kubernetes controller manager.

- 1.3.1 - Ensure that the `--terminated-pod-gc-threshold` argument is set as appropriate (Scored)
- 1.3.2 - Ensure that the `--profiling` argument is set to false (Scored)
- 1.3.6 Ensure that the RotateKubeletServerCertificate argument is set to true (Scored)
- 1.3.7 - Ensure that the `--address` argument is set to 127.0.0.1 (Scored)

Audit

- On nodes with the `controlplane` role inspect the `kube-controller-manager` container:

```
docker inspect kube-controller-manager
```

- Verify the following options are set in the `command` section:

```
--terminated-pod-gc-threshold=1000
--profiling=false
--address=127.0.0.1
--feature-gates="RotateKubeletServerCertificate=true"
```

Remediation

- In the RKE `cluster.yml` file ensure the following options are set:

```yaml
services:
  kube-controller:
    extra_args:
      profiling: "false"
      address: "127.0.0.1"
      terminated-pod-gc-threshold: "1000"
      feature-gates: "RotateKubeletServerCertificate=true"
```

- Reconfigure the cluster:

```
rke up --config cluster.yml
```

2.1.5 - Configure addons and PSPs

Profile Applicability

- Level 1

Description

Configure a restrictive pod security policy (PSP) as the default and create role bindings for system level services to use the less restrictive default PSP.

Rationale

To address the following controls, a restrictive default PSP needs to be applied as the default. Role bindings need to be in place to allow system services to still function.

- 1.7.1 - Do not admit privileged containers (Not Scored)
- 1.7.2 - Do not admit containers wishing to share the host process ID namespace (Not Scored)
- 1.7.3 - Do not admit containers wishing to share the host IPC namespace (Not Scored)
- 1.7.4 - Do not admit containers wishing to share the host network namespace (Not Scored)

- 1.7.5 - Do not admit containers with `allowPrivilegeEscalation` (Not Scored)
- 1.7.6 - Do not admit root containers (Not Scored)
- 1.7.7 - Do not admit containers with dangerous capabilities (Not Scored)

Audit

- Verify that the `cattle-system` namespace exists:

```
kubectl get ns |grep cattle
```

- Verify that the roles exist:

```
kubectl get role default-psp-role -n ingress-nginx
kubectl get role default-psp-role -n cattle-system
kubectl get clusterrole restricted-clusterrole
```

- Verify the bindings are set correctly:

```
kubectl get rolebinding -n ingress-nginx default-psp-rolebinding
kubectl get rolebinding -n cattle-system default-psp-rolebinding
kubectl get clusterrolebinding restricted-clusterrolebinding
```

- Verify the restricted PSP is present.

```
kubectl get psp restricted-psp
```

Remediation

- In the RKE `cluster.yml` file ensure the following options are set:

```yaml
addons: |
  apiVersion: rbac.authorization.k8s.io/v1
  kind: Role
  metadata:
    name: default-psp-role
    namespace: ingress-nginx
  rules:
  - apiGroups:
    - extensions
    resourceNames:
    - default-psp
    resources:
    - podsecuritypolicies
    verbs:
    - use
  ---
  apiVersion: rbac.authorization.k8s.io/v1
  kind: RoleBinding
  metadata:
    name: default-psp-rolebinding
```

```yaml
  namespace: ingress-nginx
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: Role
  name: default-psp-role
subjects:
- apiGroup: rbac.authorization.k8s.io
  kind: Group
  name: system:serviceaccounts
- apiGroup: rbac.authorization.k8s.io
  kind: Group
  name: system:authenticated
---
apiVersion: v1
kind: Namespace
metadata:
  name: cattle-system
---
apiVersion: rbac.authorization.k8s.io/v1
kind: Role
metadata:
  name: default-psp-role
  namespace: cattle-system
rules:
- apiGroups:
  - extensions
  resourceNames:
  - default-psp
  resources:
  - podsecuritypolicies
  verbs:
  - use
---
apiVersion: rbac.authorization.k8s.io/v1
kind: RoleBinding
metadata:
  name: default-psp-rolebinding
  namespace: cattle-system
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: Role
  name: default-psp-role
subjects:
- apiGroup: rbac.authorization.k8s.io
  kind: Group
```

```yaml
    name: system:serviceaccounts
- apiGroup: rbac.authorization.k8s.io
  kind: Group
  name: system:authenticated
---
apiVersion: extensions/v1beta1
kind: PodSecurityPolicy
metadata:
  name: restricted-psp
spec:
  requiredDropCapabilities:
  - NET_RAW
  privileged: false
  allowPrivilegeEscalation: false
  defaultAllowPrivilegeEscalation: false
  fsGroup:
    rule: RunAsAny
  runAsUser:
    rule: MustRunAsNonRoot
  seLinux:
    rule: RunAsAny
  supplementalGroups:
    rule: RunAsAny
  volumes:
  - emptyDir
  - secret
  - persistentVolumeClaim
  - downwardAPI
  - configMap
  - projected
---
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRole
metadata:
  name: restricted-clusterrole
rules:
- apiGroups:
  - extensions
  resourceNames:
  - restricted-psp
  resources:
  - podsecuritypolicies
  verbs:
  - use
---
```

```yaml
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRoleBinding
metadata:
  name: restricted-clusterrolebinding
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: ClusterRole
  name: restricted-clusterrole
subjects:
- apiGroup: rbac.authorization.k8s.io
  kind: Group
  name: system:serviceaccounts
- apiGroup: rbac.authorization.k8s.io
  kind: Group
  name: system:authenticated
```

- Reconfigure the cluster:

```
rke up --config cluster.yml
```

3.1 - Rancher Management Control Plane Installation

3.1.1 - Disable the local cluster option

Profile Applicability

- Level 2

Description

When deploying Rancher, disable the local cluster option on the Rancher Server.

NOTE: This requires Rancher v2.1.2 or above.

Rationale

Having access to the local cluster from the Rancher UI is convenient for troubleshooting and debugging; however, if the local cluster is enabled in the Rancher UI, a user has access to all elements of the system, including the Rancher management server itself. Disabling the local cluster is a defense in depth measure and removes the possible attack vector from the Rancher UI and API.

Audit

- Verify the Rancher deployment has the `--add-local=false` option set.

```
kubectl get deployment rancher -n cattle-system -o yaml |grep 'add-local'
```

- In the Rancher UI go to *Clusters* in the *Global* view and verify that no `local` cluster is present.

Remediation

- While upgrading or installing Rancher 2.3.3 or above, provide the following flag:

```
--set addLocal="false"
```

3.1.2 - Enable Rancher Audit logging

Profile Applicability

- Level 1

Description

Enable Rancher's built-in audit logging capability.

Rationale

Tracking down what actions were performed by users in Rancher can provide insight during post mortems, and if monitored proactively can be used to quickly detect malicious actions.

Audit

- Verify that the audit log parameters were passed into the Rancher deployment.

```
kubectl get deployment rancher -n cattle-system -o yaml | grep auditLog
```

- Verify that the log is going to the appropriate destination, as set by `auditLog.destination`
- `sidecar`:
    1. List pods:
       ```
       kubectl get pods -n cattle-system
       ```
    2. Tail logs:
       ```
       kubectl logs <pod> -n cattle-system -c rancher-audit-log
       ```
- `hostPath`
    1. On the worker nodes running the Rancher pods, verify that the log files are being written to the destination indicated in `auditlog.hostPath`.

Remediation

Upgrade the Rancher server installation using Helm, and configure the audit log settings. The instructions for doing so can be found in the reference section below.

Reference

- https://rancher.com/docs/rancher/v2.x/en/installation/ha/helm-rancher/chart-options/#advanced-options

3.2 - Rancher Management Control Plane Authentication

3.2.1 - Change the local admin password from the default value

Profile Applicability

- Level 1

Description

The local admin password should be changed from the default.

Rationale

The default admin password is common across all Rancher installations and should be changed immediately upon startup.

Audit

Attempt to login into the UI with the following credentials: - Username: admin - Password: admin

The login attempt must not succeed.

Remediation

Change the password from `admin` to a password that meets the recommended password standards for your organization.

3.2.2 - Configure an Identity Provider for Authentication

Profile Applicability

- Level 1

Description

When running Rancher in a production environment, configure an identity provider for authentication.

Rationale

Rancher supports several authentication backends that are common in enterprises. It is recommended to tie Rancher into an external authentication system to simplify user and group access in the Rancher cluster. Doing so assures that access control follows the organization's change management process for user accounts.

Audit

- In the Rancher UI, select *Global*
- Select *Security*
- Select *Authentication*
- Ensure the authentication provider for your environment is active and configured correctly

Remediation

Configure the appropriate authentication provider for your Rancher installation according to the documentation found at the link in the reference section below.

Reference

- https://rancher.com/docs/rancher/v2.x/en/admin-settings/authentication/

3.3 - Rancher Management Control Plane RBAC

3.3.1 - Ensure that administrator privileges are only granted to those who require them

Profile Applicability

- Level 1

Description

Restrict administrator access to only those responsible for managing and operating the Rancher server.

Rationale

The `admin` privilege level gives the user the highest level of access to the Rancher server and all attached clusters. This privilege should only be granted to a few people who are responsible for the availability and support of Rancher and the clusters that it manages.

Audit

The following script uses the Rancher API to show users with administrator privileges:

```bash
#!/bin/bash
for i in $(curl -sk -u 'token-<id>:<secret>' https://<RANCHER_URL>/v3/users|jq -r .data[].li

curl -sk -u 'token-<id>:<secret>' $i| jq '.data[] | "\(.userId) \(.globalRoleId)"'

done
```

The `admin` role should only be assigned to users that require administrative privileges. Any role that is not `admin` or `user` should be audited in the RBAC section of the UI to ensure that the privileges adhere to policies for global access.

The Rancher server permits customization of the default global permissions. We recommend that auditors also review the policies of any custom global roles.

Remediation

Remove the `admin` role from any user that does not require administrative privileges.

3.4 - Rancher Management Control Plane Configuration

3.4.1 - Ensure only approved node drivers are active

Profile Applicability

- Level 1

Description

Ensure that node drivers that are not needed or approved are not active in the Rancher console.

Rationale

Node drivers are used to provision compute nodes in various cloud providers and local IaaS infrastructure. For convenience, popular cloud providers are enabled by default. If the organization does not intend to use these or does not allow users to provision resources in certain providers, the drivers should be disabled. This will prevent users from using Rancher resources to provision the nodes.

Audit

- In the Rancher UI select *Global*
- Select *Node Drivers*
- Review the list of node drivers that are in an *Active* state.

Remediation

If a disallowed node driver is active, visit the *Node Drivers* page under *Global* and disable it.

4.1 - Rancher Kubernetes Custom Cluster Configuration via RKE

(See Appendix C. for full RKE template example)

4.1.1 - Configure kubelet options

Profile Applicability

- Level 1

Description

Ensure Kubelet options are configured to match CIS controls.

Rationale

To pass the following controls in the CIS benchmark, ensure the appropriate flags are passed to the Kubelet.

- 2.1.1 - Ensure that the `--anonymous-auth` argument is set to false (Scored)
- 2.1.2 - Ensure that the `--authorization-mode` argument is not set to `AlwaysAllow` (Scored)
- 2.1.6 - Ensure that the `--streaming-connection-idle-timeout` argument is not set to 0 (Scored)
- 2.1.7 - Ensure that the `--protect-kernel-defaults` argument is set to true (Scored)
- 2.1.8 - Ensure that the `--make-iptables-util-chains` argument is set to true (Scored)
- 2.1.10 - Ensure that the `--event-qps` argument is set to 0 (Scored)
- 2.1.13 - Ensure that the `RotateKubeletServerCertificate` argument is set to true (Scored)
- 2.1.14 - Ensure that the Kubelet only makes use of Strong Cryptographic Ciphers (Not Scored)

Audit

Inspect the Kubelet containers on all hosts and verify that they are running with the following options:

- `--streaming-connection-idle-timeout=<duration greater than 0>`
- `--authorization-mode=Webhook`
- `--protect-kernel-defaults=true`
- `--make-iptables-util-chains=true`
- `--event-qps=0`
- `--anonymous-auth=false`
- `--feature-gates="RotateKubeletServerCertificate=true"`
- `--tls-cipher-suites="TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256,TLS_ECDHE_RSA_WITH_AES_128`

Remediation

- Add the following to the RKE `cluster.yml` kubelet section under `services`:

```
services:
  kubelet:
    generate_serving_certificate: true
    extra_args:
      feature-gates: "RotateKubeletServerCertificate=true"
      protect-kernel-defaults: "true"
      tls-cipher-suites: "TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256,TLS_ECDHE_RSA_WITH_AES_128
```

Where `<duration>` is in a form like `1800s`.

- Reconfigure the cluster:

```
rke up --config cluster.yml
```

4.1.2 - Configure kube-api options

Profile Applicability

- Level 1

Description

Ensure the RKE configuration is set to deploy the `kube-api` service with the options required for controls.

NOTE:

Enabling the `AlwaysPullImages` admission control plugin can cause degraded performance due to overhead of always pulling images. Enabling the `DenyEscalatingExec` admission control plugin will prevent the 'Launch kubectl' functionality in the UI from working.

Rationale

To pass the following controls for the kube-api server ensure RKE configuration passes the appropriate options.

- 1.1.1 - Ensure that the `--anonymous-auth` argument is set to false (Scored)
- 1.1.8 - Ensure that the `--profiling` argument is set to false (Scored)
- 1.1.11 - Ensure that the admission control plugin `AlwaysPullImages` is set (Scored)
- 1.1.12 - Ensure that the admission control plugin `DenyEscalatingExec` is set (Scored)
- 1.1.14 - Ensure that the admission control plugin `NamespaceLifecycle` is set (Scored)

- 1.1.15 - Ensure that the `--audit-log-path` argument is set as appropriate (Scored)
- 1.1.16 - Ensure that the `--audit-log-maxage` argument is set as appropriate (Scored)
- 1.1.17 - Ensure that the `--audit-log-maxbackup` argument is set as appropriate (Scored)
- 1.1.18 - Ensure that the `--audit-log-maxsize` argument is set as appropriate (Scored)
- 1.1.23 - Ensure that the `--service-account-lookup` argument is set to true (Scored)
- 1.1.24 - Ensure that the admission control plugin `PodSecurityPolicy` is set (Scored)
- 1.1.30 Ensure that the API Server only makes use of Strong Cryptographic Ciphers (Not Scored)
- 1.1.34 - Ensure that the `--encryption-provider-config` argument is set as appropriate (Scored)
- 1.1.35 - Ensure that the encryption provider is set to `aescbc` (Scored)
- 1.1.36 - Ensure that the admission control plugin `EventRateLimit` is set (Scored)
- 1.1.37 - Ensure that the `AdvancedAuditing` argument is not set to `false` (Scored)

Audit

- On nodes with the `controlplane` role inspect the `kube-apiserver` containers:

```bash
docker inspect kube-apiserver
```

- Look for the following options in the command section of the output:

```
--anonymous-auth=false
--profiling=false
--service-account-lookup=true
--enable-admission-plugins=ServiceAccount,NamespaceLifecycle,LimitRanger,PersistentVolumeLab
--encryption-provider-config=/etc/kubernetes/ssl/encryption.yaml
--admission-control-config-file=/etc/kubernetes/admission.yaml
--audit-log-path=/var/log/kube-audit/audit-log.json
--audit-log-maxage=30
--audit-log-maxbackup=10
--audit-log-maxsize=100
--audit-log-format=json
--audit-policy-file=/etc/kubernetes/audit-policy.yaml
--tls-cipher-suites=TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256,TLS_ECDHE_RSA_WITH_AES_128_GCM_S
```

- In the `volume` section of the output ensure the bind mount is present:

```
/var/log/kube-audit:/var/log/kube-audit
```

Remediation

- In the RKE `cluster.yml` add the following directives to the `kube-api` section under `services`:

```
services:
  kube_api:
    always_pull_images: true
    pod_security_policy: true
    service_node_port_range: 30000-32767
    event_rate_limit:
      enabled: true
    audit_log:
      enabled: true
    secrets_encryption_config:
      enabled: true
    extra_args:
      anonymous-auth: "false"
      enable-admission-plugins: "ServiceAccount,NamespaceLifecycle,LimitRanger,PersistentVol
      profiling: "false"
      service-account-lookup: "true"
      tls-cipher-suites: "TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256,TLS_ECDHE_RSA_WITH_AES_128
    extra_binds:
      - "/opt/kubernetes:/opt/kubernetes"
```

For k8s 1.14 `enable-admission-plugins` should be

```
      enable-admission-plugins: "ServiceAccount,NamespaceLifecycle,LimitRanger,PersistentV
```

- Reconfigure the cluster:

```
rke up --config cluster.yml
```

NOTE:

Files that are placed in `/opt/kubernetes` need to be mounted in using the `extra_binds` functionality in RKE.

4.1.3 - Configure scheduler options

Profile Applicability

- Level 1

Description

Set the appropriate options for the Kubernetes scheduling service.

NOTE: Setting `--address` to `127.0.0.1` will prevent Rancher cluster monitoring from scraping this endpoint.

Rationale

To address the following controls on the CIS benchmark, the command line options should be set on the Kubernetes scheduler.

- 1.2.1 - Ensure that the `--profiling` argument is set to `false` (Scored)
- 1.2.2 - Ensure that the `--address` argument is set to `127.0.0.1` (Scored)

Audit

- On nodes with the `controlplane` role: inspect the `kube-scheduler` containers:

```
docker inspect kube-scheduler
```

- Verify the following options are set in the `command` section.

```
--profiling=false
--address=127.0.0.1
```

Remediation

- In the RKE `cluster.yml` file ensure the following options are set:

```yaml
services:
  scheduler:
    extra_args:
      profiling: "false"
      address: "127.0.0.1"
```

- Reconfigure the cluster:

```
rke up --config cluster.yml
```

4.1.4 - Configure controller options

Profile Applicability

- Level 1

Description

Set the appropriate arguments on the Kubernetes controller manager.

5*NOTE:** Setting `--address` to `127.0.0.1` will prevent Rancher cluster monitoring from scraping this endpoint.

Rationale

To address the following controls the options need to be passed to the Kubernetes controller manager.

- 1.3.1 - Ensure that the `--terminated-pod-gc-threshold` argument is set as appropriate (Scored)
- 1.3.2 - Ensure that the `--profiling` argument is set to false (Scored)

- 1.3.6 Ensure that the RotateKubeletServerCertificate argument is set to true (Scored)
- 1.3.7 - Ensure that the `--address` argument is set to 127.0.0.1 (Scored)

Audit

- On nodes with the `controlplane` role inspect the `kube-controller-manager` container:

```
docker inspect kube-controller-manager
```

- Verify the following options are set in the `command` section:

```
--terminated-pod-gc-threshold=1000
--profiling=false
--address=127.0.0.1
--feature-gates="RotateKubeletServerCertificate=true"
```

Remediation

- In the RKE `cluster.yml` file ensure the following options are set:

```yaml
services:
  kube-controller:
    extra_args:
      profiling: "false"
      address: "127.0.0.1"
      terminated-pod-gc-threshold: "1000"
      feature-gates: "RotateKubeletServerCertificate=true"
```

- Reconfigure the cluster:

```
rke up --config cluster.yml
```

4.1.5 - Check PSPs

Profile Applicability

- Level 1

Description

Configure a restrictive pod security policy (PSP) as the default and create role bindings for system level services to use the less restrictive default PSP.

Rationale

To address the following controls, a restrictive default PSP needs to be applied as the default. Role bindings need to be in place to allow system services to still function.

- 1.7.1 - Do not admit privileged containers (Not Scored)

- 1.7.2 - Do not admit containers wishing to share the host process ID namespace (Not Scored)
- 1.7.3 - Do not admit containers wishing to share the host IPC namespace (Not Scored)
- 1.7.4 - Do not admit containers wishing to share the host network namespace (Not Scored)
- 1.7.5 - Do not admit containers with `allowPrivilegeEscalation` (Not Scored)
- 1.7.6 - Do not admit root containers (Not Scored)
- 1.7.7 - Do not admit containers with dangerous capabilities (Not Scored)

Audit

- Verify that the `cattle-system` namespace exists:

```
kubectl get ns |grep cattle
```

- Verify that the roles exist:

```
kubectl get role default-psp-role -n ingress-nginx
kubectl get role default-psp-role -n cattle-system
kubectl get clusterrole restricted-clusterrole
```

- Verify the bindings are set correctly:

```
kubectl get rolebinding -n ingress-nginx default-psp-rolebinding
kubectl get rolebinding -n cattle-system default-psp-rolebinding
```

- Verify the restricted PSP is present.

```
kubectl get psp restricted-psp
```

Appendix A - Complete ubuntu `cloud-config` Example

`cloud-config` file to automate hardening manual steps on nodes deployment.

```
#cloud-config
bootcmd:
- apt-get update
- apt-get install -y apt-transport-https
apt:
  sources:
    docker:
      source: "deb [arch=amd64] https://download.docker.com/linux/ubuntu $RELEASE stable"
      keyid: 0EBFCD88
packages:
- [docker-ce, '5:19.03.5~3-0~ubuntu-bionic']
```

```
- jq
write_files:
# 1.1.1 - Configure default sysctl settings on all hosts
- path: /etc/sysctl.d/90-kubelet.conf
  owner: root:root
  permissions: '0644'
  content: |
    vm.overcommit_memory=1
    vm.panic_on_oom=0
    kernel.panic=10
    kernel.panic_on_oops=1
    kernel.keys.root_maxkeys=1000000
    kernel.keys.root_maxbytes=25000000
# 1.4.12 etcd user
groups:
  - etcd
users:
  - default
  - name: etcd
    gecos: Etcd user
    primary_group: etcd
    homedir: /var/lib/etcd
# 1.4.11 etcd data dir
runcmd:
  - chmod 0700 /var/lib/etcd
  - usermod -G docker -a ubuntu
  - sysctl -p /etc/sysctl.d/90-kubelet.conf
```

Appendix B - Complete RKE `cluster.yml` Example

Before apply, replace `rancher_kubernetes_engine_config.services.etcd.gid` and `rancher_kubernetes_engine_config.services.etcd.uid` with the proper etcd group and user ids that were created on etcd nodes.

{{% accordion id="cluster-1.14" label="RKE yaml for k8s 1.14" %}}

```
nodes:
- address: 18.191.190.205
  internal_address: 172.31.24.213
  user: ubuntu
  role: [ "controlplane", "etcd", "worker" ]
- address: 18.191.190.203
  internal_address: 172.31.24.203
  user: ubuntu
  role: [ "controlplane", "etcd", "worker" ]
- address: 18.191.190.10
```

```yaml
    internal_address: 172.31.24.244
    user: ubuntu
    role: [ "controlplane", "etcd", "worker" ]
addon_job_timeout: 30
authentication:
    strategy: x509
authorization: {}
bastion_host:
    ssh_agent_auth: false
cloud_provider: {}
ignore_docker_version: true
#
# # Currently only nginx ingress provider is supported.
# # To disable ingress controller, set `provider: none`
# # To enable ingress on specific nodes, use the node_selector, eg:
#     provider: nginx
#     node_selector:
#       app: ingress
#
ingress:
    provider: nginx
kubernetes_version: v1.14.9-rancher1-1
monitoring:
    provider: metrics-server
#
#    If you are using calico on AWS
#
#    network:
#      plugin: calico
#      calico_network_provider:
#        cloud_provider: aws
#
# # To specify flannel interface
#
#    network:
#      plugin: flannel
#      flannel_network_provider:
#      iface: eth1
#
# # To specify flannel interface for canal plugin
#
#    network:
#      plugin: canal
#      canal_network_provider:
#        iface: eth1
```

```
#
network:
  options:
    flannel_backend_type: vxlan
  plugin: canal
restore:
  restore: false
#
#     services:
#       kube-api:
#         service_cluster_ip_range: 10.43.0.0/16
#       kube-controller:
#         cluster_cidr: 10.42.0.0/16
#         service_cluster_ip_range: 10.43.0.0/16
#       kubelet:
#         cluster_domain: cluster.local
#         cluster_dns_server: 10.43.0.10
#
services:
  etcd:
    backup_config:
      enabled: true
      interval_hours: 12
      retention: 6
      safe_timestamp: false
    creation: 12h
    extra_args:
      election-timeout: '5000'
      heartbeat-interval: '500'
    gid: 1000
    retention: 72h
    snapshot: false
    uid: 1000
  kube-api:
    always_pull_images: true
    audit_log:
      enabled: true
    event_rate_limit:
      enabled: true
    extra_args:
      anonymous-auth: 'false'
      enable-admission-plugins: >-
        ServiceAccount,NamespaceLifecycle,LimitRanger,PersistentVolumeLabel,DefaultStorageCl
      profiling: 'false'
      service-account-lookup: 'true'
```

```yaml
      tls-cipher-suites: >-
        TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256,TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256,TLS_EC
    extra_binds:
      - '/opt/kubernetes:/opt/kubernetes'
    pod_security_policy: true
    secrets_encryption_config:
      enabled: true
    service_node_port_range: 30000-32767
  kube-controller:
    extra_args:
      address: 127.0.0.1
      feature-gates: RotateKubeletServerCertificate=true
      profiling: 'false'
      terminated-pod-gc-threshold: '1000'
  kubelet:
    extra_args:
      protect-kernel-defaults: 'true'
    fail_swap_on: false
    generate_serving_certificate: true
  kubeproxy: {}
  scheduler:
    extra_args:
      address: 127.0.0.1
      profiling: 'false'
ssh_agent_auth: false
```

{{% /accordion %}}

{{% accordion id="cluster-1.15" label="RKE yaml for k8s 1.15" %}}

```yaml
nodes:
- address: 18.191.190.205
  internal_address: 172.31.24.213
  user: ubuntu
  role: [ "controlplane", "etcd", "worker" ]
- address: 18.191.190.203
  internal_address: 172.31.24.203
  user: ubuntu
  role: [ "controlplane", "etcd", "worker" ]
- address: 18.191.190.10
  internal_address: 172.31.24.244
  user: ubuntu
  role: [ "controlplane", "etcd", "worker" ]
addon_job_timeout: 30
authentication:
  strategy: x509
```

```yaml
ignore_docker_version: true
#
# # Currently only nginx ingress provider is supported.
# # To disable ingress controller, set `provider: none`
# # To enable ingress on specific nodes, use the node_selector, eg:
#     provider: nginx
#     node_selector:
#       app: ingress
#
ingress:
  provider: nginx
kubernetes_version: v1.15.6-rancher1-2
monitoring:
  provider: metrics-server
#
#   If you are using calico on AWS
#
#     network:
#       plugin: calico
#       calico_network_provider:
#         cloud_provider: aws
#
# # To specify flannel interface
#
#     network:
#       plugin: flannel
#       flannel_network_provider:
#       iface: eth1
#
# # To specify flannel interface for canal plugin
#
#     network:
#       plugin: canal
#       canal_network_provider:
#         iface: eth1
#
network:
  options:
    flannel_backend_type: vxlan
  plugin: canal
#
#     services:
#       kube-api:
#         service_cluster_ip_range: 10.43.0.0/16
#       kube-controller:
```

```yaml
#         cluster_cidr: 10.42.0.0/16
#         service_cluster_ip_range: 10.43.0.0/16
#       kubelet:
#         cluster_domain: cluster.local
#         cluster_dns_server: 10.43.0.10
#
services:
  etcd:
    backup_config:
      enabled: true
      interval_hours: 12
      retention: 6
      safe_timestamp: false
    creation: 12h
    extra_args:
      election-timeout: 5000
      heartbeat-interval: 500
    gid: 1000
    retention: 72h
    snapshot: false
    uid: 1000
  kube_api:
    always_pull_images: true
    pod_security_policy: true
    service_node_port_range: 30000-32767
    event_rate_limit:
      enabled: true
    audit_log:
      enabled: true
    secrets_encryption_config:
      enabled: true
    extra_args:
      anonymous-auth: "false"
      enable-admission-plugins: "ServiceAccount,NamespaceLifecycle,LimitRanger,PersistentVol
      profiling: "false"
      service-account-lookup: "true"
      tls-cipher-suites: "TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256,TLS_ECDHE_RSA_WITH_AES_128
    extra_binds:
      - "/opt/kubernetes:/opt/kubernetes"
  kubelet:
    generate_serving_certificate: true
    extra_args:
      feature-gates: "RotateKubeletServerCertificate=true"
      protect-kernel-defaults: "true"
      tls-cipher-suites: "TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256,TLS_ECDHE_RSA_WITH_AES_128
```

```yaml
  kube-controller:
    extra_args:
      profiling: "false"
      address: "127.0.0.1"
      terminated-pod-gc-threshold: "1000"
      feature-gates: "RotateKubeletServerCertificate=true"
  scheduler:
    extra_args:
      profiling: "false"
      address: "127.0.0.1"
ssh_agent_auth: false
```

{{% /accordion %}}

{{% accordion id="cluster-1.16" label="RKE yaml for k8s 1.16" %}}

```yaml
nodes:
- address: 18.191.190.205
  internal_address: 172.31.24.213
  user: ubuntu
  role: [ "controlplane", "etcd", "worker" ]
- address: 18.191.190.203
  internal_address: 172.31.24.203
  user: ubuntu
  role: [ "controlplane", "etcd", "worker" ]
- address: 18.191.190.10
  internal_address: 172.31.24.244
  user: ubuntu
  role: [ "controlplane", "etcd", "worker" ]
addon_job_timeout: 30
authentication:
  strategy: x509
ignore_docker_version: true
#
# # Currently only nginx ingress provider is supported.
# # To disable ingress controller, set `provider: none`
# # To enable ingress on specific nodes, use the node_selector, eg:
#    provider: nginx
#    node_selector:
#      app: ingress
#
ingress:
  provider: nginx
kubernetes_version: v1.16.3-rancher1-1
monitoring:
  provider: metrics-server
```

33

```yaml
#
#    If you are using calico on AWS
#
#      network:
#        plugin: calico
#        calico_network_provider:
#          cloud_provider: aws
#
# # To specify flannel interface
#
#      network:
#        plugin: flannel
#        flannel_network_provider:
#        iface: eth1
#
# # To specify flannel interface for canal plugin
#
#      network:
#        plugin: canal
#        canal_network_provider:
#          iface: eth1
#
network:
  options:
    flannel_backend_type: vxlan
  plugin: canal
#
#      services:
#        kube-api:
#          service_cluster_ip_range: 10.43.0.0/16
#        kube-controller:
#          cluster_cidr: 10.42.0.0/16
#          service_cluster_ip_range: 10.43.0.0/16
#        kubelet:
#          cluster_domain: cluster.local
#          cluster_dns_server: 10.43.0.10
#
services:
  etcd:
    backup_config:
      enabled: true
      interval_hours: 12
      retention: 6
      safe_timestamp: false
    creation: 12h
```

```yaml
    extra_args:
      election-timeout: 5000
      heartbeat-interval: 500
    gid: 1000
    retention: 72h
    snapshot: false
    uid: 1000
  kube_api:
    always_pull_images: true
    pod_security_policy: true
    service_node_port_range: 30000-32767
    event_rate_limit:
      enabled: true
    audit_log:
      enabled: true
    secrets_encryption_config:
      enabled: true
    extra_args:
      anonymous-auth: "false"
      enable-admission-plugins: "ServiceAccount,NamespaceLifecycle,LimitRanger,PersistentVol
      profiling: "false"
      service-account-lookup: "true"
      tls-cipher-suites: "TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256,TLS_ECDHE_RSA_WITH_AES_128
    extra_binds:
      - "/opt/kubernetes:/opt/kubernetes"
  kubelet:
    generate_serving_certificate: true
    extra_args:
      feature-gates: "RotateKubeletServerCertificate=true"
      protect-kernel-defaults: "true"
      tls-cipher-suites: "TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256,TLS_ECDHE_RSA_WITH_AES_128
  kube-controller:
    extra_args:
      profiling: "false"
      address: "127.0.0.1"
      terminated-pod-gc-threshold: "1000"
      feature-gates: "RotateKubeletServerCertificate=true"
  scheduler:
    extra_args:
      profiling: "false"
      address: "127.0.0.1"
ssh_agent_auth: false
```

{{% /accordion %}}

Appendix C - Complete RKE Template Example

Before apply, replace `rancher_kubernetes_engine_config.services.etcd.gid` and `rancher_kubernetes_engine_config.services.etcd.uid` with the proper etcd group and user ids that were created on etcd nodes.

{{% accordion id="k8s-1.14" label="RKE template for k8s 1.14" %}}

```
#
# Cluster Config
#
answers: {}
default_pod_security_policy_template_id: restricted
docker_root_dir: /var/lib/docker
enable_cluster_alerting: false
enable_cluster_monitoring: false
enable_network_policy: false
local_cluster_auth_endpoint:
  enabled: false
name: test-35378
#
# Rancher Config
#
rancher_kubernetes_engine_config:
  addon_job_timeout: 30
  authentication:
    strategy: x509
  authorization: {}
  bastion_host:
    ssh_agent_auth: false
  cloud_provider: {}
  ignore_docker_version: true
#
# # Currently only nginx ingress provider is supported.
# # To disable ingress controller, set `provider: none`
# # To enable ingress on specific nodes, use the node_selector, eg:
#     provider: nginx
#     node_selector:
#       app: ingress
#
  ingress:
    provider: nginx
  kubernetes_version: v1.14.9-rancher1-1
  monitoring:
    provider: metrics-server
#
```

```
#    If you are using calico on AWS
#
#     network:
#       plugin: calico
#       calico_network_provider:
#         cloud_provider: aws
#
# # To specify flannel interface
#
#     network:
#       plugin: flannel
#       flannel_network_provider:
#       iface: eth1
#
# # To specify flannel interface for canal plugin
#
#     network:
#       plugin: canal
#       canal_network_provider:
#         iface: eth1
#
  network:
    options:
      flannel_backend_type: vxlan
    plugin: canal
  restore:
    restore: false
#
#     services:
#       kube-api:
#         service_cluster_ip_range: 10.43.0.0/16
#       kube-controller:
#         cluster_cidr: 10.42.0.0/16
#         service_cluster_ip_range: 10.43.0.0/16
#       kubelet:
#         cluster_domain: cluster.local
#         cluster_dns_server: 10.43.0.10
#
  services:
    etcd:
      backup_config:
        enabled: true
        interval_hours: 12
        retention: 6
        safe_timestamp: false
```

```yaml
      creation: 12h
      extra_args:
        election-timeout: '5000'
        heartbeat-interval: '500'
      gid: 1000
      retention: 72h
      snapshot: false
      uid: 1000
  kube-api:
    always_pull_images: true
    audit_log:
      enabled: true
    event_rate_limit:
      enabled: true
    extra_args:
      anonymous-auth: 'false'
      enable-admission-plugins: >-
        ServiceAccount,NamespaceLifecycle,LimitRanger,PersistentVolumeLabel,DefaultStorage
      profiling: 'false'
      service-account-lookup: 'true'
      tls-cipher-suites: >-
        TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256,TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256,TLS_
    extra_binds:
      - '/opt/kubernetes:/opt/kubernetes'
    pod_security_policy: true
    secrets_encryption_config:
      enabled: true
    service_node_port_range: 30000-32767
  kube-controller:
    extra_args:
      address: 127.0.0.1
      feature-gates: RotateKubeletServerCertificate=true
      profiling: 'false'
      terminated-pod-gc-threshold: '1000'
  kubelet:
    extra_args:
      protect-kernel-defaults: 'true'
    fail_swap_on: false
    generate_serving_certificate: true
  kubeproxy: {}
  scheduler:
    extra_args:
      address: 127.0.0.1
      profiling: 'false'
ssh_agent_auth: false
```

```
windows_prefered_cluster: false
```

{{% /accordion %}}

{{% accordion id="k8s-1.15" label="RKE template for k8s 1.15" %}}

```
#
# Cluster Config
#
default_pod_security_policy_template_id: restricted
docker_root_dir: /var/lib/docker
enable_cluster_alerting: false
enable_cluster_monitoring: false
enable_network_policy: false
local_cluster_auth_endpoint:
  enabled: true
#
# Rancher Config
#
rancher_kubernetes_engine_config:
  addon_job_timeout: 30
  authentication:
    strategy: x509
  ignore_docker_version: true
#
# # Currently only nginx ingress provider is supported.
# # To disable ingress controller, set `provider: none`
# # To enable ingress on specific nodes, use the node_selector, eg:
#    provider: nginx
#    node_selector:
#      app: ingress
#
  ingress:
    provider: nginx
  kubernetes_version: v1.15.6-rancher1-2
  monitoring:
    provider: metrics-server
#
#   If you are using calico on AWS
#
#    network:
#      plugin: calico
#      calico_network_provider:
#        cloud_provider: aws
#
# # To specify flannel interface
```

```
#
#    network:
#      plugin: flannel
#      flannel_network_provider:
#      iface: eth1
#
# # To specify flannel interface for canal plugin
#
#    network:
#      plugin: canal
#      canal_network_provider:
#        iface: eth1
#
  network:
    options:
      flannel_backend_type: vxlan
    plugin: canal
#
#    services:
#      kube-api:
#        service_cluster_ip_range: 10.43.0.0/16
#      kube-controller:
#        cluster_cidr: 10.42.0.0/16
#        service_cluster_ip_range: 10.43.0.0/16
#      kubelet:
#        cluster_domain: cluster.local
#        cluster_dns_server: 10.43.0.10
#
  services:
    etcd:
      backup_config:
        enabled: true
        interval_hours: 12
        retention: 6
        safe_timestamp: false
      creation: 12h
      extra_args:
        election-timeout: 5000
        heartbeat-interval: 500
      gid: 1000
      retention: 72h
      snapshot: false
      uid: 1000
    kube_api:
      always_pull_images: true
```

```
      pod_security_policy: true
      service_node_port_range: 30000-32767
      event_rate_limit:
        enabled: true
      audit_log:
        enabled: true
      secrets_encryption_config:
        enabled: true
      extra_args:
        anonymous-auth: "false"
        enable-admission-plugins: "ServiceAccount,NamespaceLifecycle,LimitRanger,PersistentV
        profiling: "false"
        service-account-lookup: "true"
        tls-cipher-suites: "TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256,TLS_ECDHE_RSA_WITH_AES_1
      extra_binds:
        - "/opt/kubernetes:/opt/kubernetes"
    kubelet:
      generate_serving_certificate: true
      extra_args:
        feature-gates: "RotateKubeletServerCertificate=true"
        protect-kernel-defaults: "true"
        tls-cipher-suites: "TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256,TLS_ECDHE_RSA_WITH_AES_1
    kube-controller:
      extra_args:
        profiling: "false"
        address: "127.0.0.1"
        terminated-pod-gc-threshold: "1000"
        feature-gates: "RotateKubeletServerCertificate=true"
    scheduler:
      extra_args:
        profiling: "false"
        address: "127.0.0.1"
  ssh_agent_auth: false
windows_prefered_cluster: false
```

{{% /accordion %}}

{{% accordion id="k8s-1.16" label="RKE template for k8s 1.16" %}}

```
#
# Cluster Config
#
default_pod_security_policy_template_id: restricted
docker_root_dir: /var/lib/docker
enable_cluster_alerting: false
enable_cluster_monitoring: false
```

```yaml
enable_network_policy: false
local_cluster_auth_endpoint:
  enabled: true
#
# Rancher Config
#
rancher_kubernetes_engine_config:
  addon_job_timeout: 30
  authentication:
    strategy: x509
  ignore_docker_version: true
#
# # Currently only nginx ingress provider is supported.
# # To disable ingress controller, set `provider: none`
# # To enable ingress on specific nodes, use the node_selector, eg:
#     provider: nginx
#     node_selector:
#       app: ingress
#
  ingress:
    provider: nginx
  kubernetes_version: v1.16.3-rancher1-1
  monitoring:
    provider: metrics-server
#
#    If you are using calico on AWS
#
#     network:
#       plugin: calico
#       calico_network_provider:
#         cloud_provider: aws
#
# # To specify flannel interface
#
#     network:
#       plugin: flannel
#       flannel_network_provider:
#       iface: eth1
#
# # To specify flannel interface for canal plugin
#
#     network:
#       plugin: canal
#       canal_network_provider:
#         iface: eth1
```

```yaml
#
  network:
    options:
      flannel_backend_type: vxlan
    plugin: canal
#
#    services:
#      kube-api:
#        service_cluster_ip_range: 10.43.0.0/16
#      kube-controller:
#        cluster_cidr: 10.42.0.0/16
#        service_cluster_ip_range: 10.43.0.0/16
#      kubelet:
#        cluster_domain: cluster.local
#        cluster_dns_server: 10.43.0.10
#
  services:
    etcd:
      backup_config:
        enabled: true
        interval_hours: 12
        retention: 6
        safe_timestamp: false
      creation: 12h
      extra_args:
        election-timeout: 5000
        heartbeat-interval: 500
      gid: 1000
      retention: 72h
      snapshot: false
      uid: 1000
    kube_api:
      always_pull_images: true
      pod_security_policy: true
      service_node_port_range: 30000-32767
      event_rate_limit:
        enabled: true
      audit_log:
        enabled: true
      secrets_encryption_config:
        enabled: true
      extra_args:
        anonymous-auth: "false"
        enable-admission-plugins: "ServiceAccount,NamespaceLifecycle,LimitRanger,PersistentV
        profiling: "false"
```

```yaml
        service-account-lookup: "true"
        tls-cipher-suites: "TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256,TLS_ECDHE_RSA_WITH_AES_1
      extra_binds:
        - "/opt/kubernetes:/opt/kubernetes"
    kubelet:
      generate_serving_certificate: true
      extra_args:
        feature-gates: "RotateKubeletServerCertificate=true"
        protect-kernel-defaults: "true"
        tls-cipher-suites: "TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256,TLS_ECDHE_RSA_WITH_AES_1
    kube-controller:
      extra_args:
        profiling: "false"
        address: "127.0.0.1"
        terminated-pod-gc-threshold: "1000"
        feature-gates: "RotateKubeletServerCertificate=true"
    scheduler:
      extra_args:
        profiling: "false"
        address: "127.0.0.1"
  ssh_agent_auth: false
windows_prefered_cluster: false
```

{{% /accordion %}}