ALIBABA CLOUD

Alibaba Cloud

Container Service for Kubernetes

Bulletin

Document Version: 20220119

Alibaba Cloud

Legal disclaimer

Alibaba Cloud reminds you to carefully read and fully understand the terms and conditions of this legal disclaimer before you read or use this document. If you have read or used this document, it shall be deemed as your total acceptance of this legal disclaimer.

1. You shall download and obtain this document from the Alibaba Cloud website or other Alibaba Cloud-authorized channels, and use this document for your own legal business activities only. The content of this document is considered confidential information of Alibaba Cloud. You shall strictly abide by the confidentiality obligations. No part of this document shall be disclosed or provided to any third party for use without the prior written consent of Alibaba Cloud.

2. No part of this document shall be excerpted, translated, reproduced, transmitted, or disseminated by any organization, company or individual in any form or by any means without the prior written consent of Alibaba Cloud.

3. The content of this document may be changed because of product version upgrade, adjustment, or other reasons. Alibaba Cloud reserves the right to modify the content of this document without notice and an updated version of this document will be released through Alibaba Cloud-authorized channels from time to time. You should pay attention to the version changes of this document as they occur and download and obtain the most up-to-date version of this document from Alibaba Cloud-authorized channels.

4. This document serves only as a reference guide for your use of Alibaba Cloud products and services. Alibaba Cloud provides this document based on the "status quo", "being defective", and "existing functions" of its products and services. Alibaba Cloud makes every effort to provide relevant operational guidance based on existing technologies. However, Alibaba Cloud hereby makes a clear statement that it in no way guarantees the accuracy, integrity, applicability, and reliability of the content of this document, either explicitly or implicitly. Alibaba Cloud shall not take legal responsibility for any errors or lost profits incurred by any organization, company, or individual arising from download, use, or trust in this document. Alibaba Cloud shall not, under any circumstances, take responsibility for any indirect, consequential, punitive, contingent, special, or punitive damages, including lost profits arising from the use or trust in this document (even if Alibaba Cloud has been notified of the possibility of such a loss).

5. By law, all the contents in Alibaba Cloud documents, including but not limited to pictures, architecture design, page layout, and text description, are intellectual property of Alibaba Cloud and/or its affiliates. This intellectual property includes, but is not limited to, trademark rights, patent rights, copyrights, and trade secrets. No part of this document shall be used, modified, reproduced, publicly transmitted, changed, disseminated, distributed, or published without the prior written consent of Alibaba Cloud and/or its affiliates. The names owned by Alibaba Cloud shall not be used, published, or reproduced for marketing, advertising, promotion, or other purposes without the prior written consent of Alibaba Cloud. The names owned by Alibaba Cloud include, but are not limited to, "Alibaba Cloud", "Aliyun", "HiChina", and other brands of Alibaba Cloud and/or its affiliates, which appear separately or in combination, as well as the auxiliary signs and patterns of the preceding brands, or anything similar to the company names, trade names, trademarks, product or service names, domain names, patterns, logos, marks, signs, or special descriptions that third parties identify as Alibaba Cloud and/or its affiliates.

6. Please directly contact Alibaba Cloud for any errors of this document.

Document conventions

Style	Description	Example
Danger	A danger notice indicates a situation that will cause major system changes, faults, physical injuries, and other adverse results.	**Danger:** Resetting will result in the loss of user configuration data.
Warning	A warning notice indicates a situation that may cause major system changes, faults, physical injuries, and other adverse results.	**Warning:** Restarting will cause business interruption. About 10 minutes are required to restart an instance.
Notice	A caution notice indicates warning information, supplementary instructions, and other content that the user must understand.	**Notice:** If the weight is set to 0, the server no longer receives new requests.
Note	A note indicates supplemental instructions, best practices, tips, and other content.	**Note:** You can use Ctrl + A to select all files.
>	Closing angle brackets are used to indicate a multi-level menu cascade.	Click **Settings**> **Network**> **Set network type**.
Bold	Bold formatting is used for buttons , menus, page names, and other UI elements.	Click **OK**.
`Courier font`	Courier font is used for commands	Run the `cd /d C:/window` command to enter the Windows system folder.
Italic	Italic formatting is used for parameters and variables.	`bae log list --instanceid` *Instance_ID*
[] or [a|b]	This format is used for an optional value, where only one item can be selected.	`ipconfig [-all|-t]`
{} or {a|b}	This format is used for a required value, where only one item can be selected.	`switch {active|stand}`

Table of Contents

1.Announcement overview

This topic lists the announcements of Container Service for Kubernetes (ACK). The announcements are classified into the following types: release notes, product changes, Kubernetes versions, component upgrades, common vulnerabilities and exposures (CVE), and system restoration.

◉Release Notes○Product Changes○Kubernetes Versions○Component Upgrades○CVE○System Restoration

Announcement	Released at
[CVE] Vulnerability CVE-2021-25741 in Kubernetes	September 17, 2021
[Product Changes] ACK API enhances user authentication	August 10, 2021
[Kubernetes Versions] ACK ends support for Kubernetes 1.12.6 and earlier	April 24, 2021
[System Restoration] ECS OpenAPI errors on February 23, 2021 are fixed	February 23, 2021
[Component Upgrades] Upgrade the CCM	January 7, 2021
[Product Changes] The navigation pane in the ACK console is improved	December 16, 2020
[Product Changes] Alibaba Cloud starts charging for professional Kubernetes clusters	November 26, 2020
[Product Changes] ACK ends support for Kubernetes Dashboard	October 23, 2020
[Component Upgrades] Upgrade CoreDNS to 1.6.2 or later	October 23, 2020
[Component Upgrades] Upgrade Metrics Server	August 21, 2020

Announcement	Released at
[Product Changes] ACK API enhances user authentication	August 10, 2021
[Product Changes] The navigation pane in the ACK console is improved	December 16, 2020
[Product Changes] Alibaba Cloud starts charging for professional Kubernetes clusters	November 26, 2020
[Product Changes] ACK ends support for Kubernetes Dashboard	October 23, 2020
[Product Changes] ACK reduces the permissions of worker RAM roles in managed Kubernetes clusters	April 27, 2020
[Product Changes] ACK ends support for Swarm	July 23, 2019
[Product Changes] ACK upgrades security policies	May 8, 2019
[Product Changes] Alibaba Cloud starts charging for elastic container instances that are used in ASK clusters	May 8, 2019

Announcement	Released at
[Kubernetes Versions] ACK ends support for Kubernetes 1.12.6 and earlier	April 24, 2021
[Kubernetes Versions] Kubernetes version support policy	July 24, 2019

Announcement	Released at
[Component Upgrades] Upgrade the CCM	January 7, 2021
[Component Upgrades] Upgrade CoreDNS to 1.6.2 or later	October 23, 2020
[Component Upgrades] Upgrade Helm Tiller from V2 to V3	March 7, 2020
[Component Upgrades] Upgrade Terway	July 23, 2019

Announcement	Released at
[CVE] Vulnerability CVE-2021-25741 in Kubernetes	September 17, 2021
[CVE] Vulnerability CVE-2021-25740 in Kubernetes	July 23, 2021
[CVE] Vulnerability CVE-2021-25738 in the Kubernetes Java client	June 10, 2021
[CVE] Vulnerability CVE-2021-25737 in the Kubernetes API server	June 10, 2021
[CVE] Vulnerability CVE-2021-30465 in runC	June 2, 2021
[CVE] Vulnerability CVE-2021-3121 in GoGo Protobuf	May 25, 2021
[CVE] Vulnerability CVE-2020-8562 in the Kubernetes API server	May 25, 2021
[CVE] Vulnerability CVE-2021-25735 in the Kubernetes API server	April 15, 2021
[CVE] Vulnerability CVE-2021-1056 in NVIDIA GPU drivers	April 9, 2021
[CVE] Vulnerability CVE-2020-8554 in Kubernetes	December 8, 2020
[CVE] Vulnerability CVE-2020-15257 in the networking namespace	December 2, 2020
[CVE] Vulnerabilities CVE-2020-8564 in kubelet, CVE-2020-8565 in kube-apiserver, and CVE-2020-8566 in kube-controller-manager	November 2, 2020
[CVE] Vulnerability CVE-2020-14386 in the Linux kernel	September 18, 2020
[CVE] Vulnerability CVE-2020-8559 in the Kubernetes API server	July 15, 2020
[CVE] Vulnerability CVE-2020-8557 in kubelet	July 15, 2020
[CVE] Vulnerability CVE-2020-8558 in kube-proxy	July 8, 2020
[CVE] Vulnerability CVE-2020-8555 in kube-controller-manager	June 1, 2020

[CVE] Vulnerability CVE-2020-13401 in Docker Engine	June 1, 2020
[CVE] Vulnerability CVE-2019-11253 in the Kubernetes API server	November 14, 2019
[CVE] Vulnerability CVE-2019-16276 in Golang	November 15, 2019
[CVE] Vulnerability CVE-2019-11249 in the kubectl cp command	August 6, 2019
[CVE] Vulnerability CVE-2019-11246 in the kubectl cp command	July 1, 2019
[CVE] Vulnerability CVE-2019-1002101 in the kubectl cp command	May 8, 2019
[CVE] Vulnerability CVE-2019-5736 in runC	February 12, 2019
[CVE] Vulnerability CVE-2018-18264 in Kubernetes Dashboard	January 7, 2019
[CVE] Vulnerability CVE-2018-1002105 in Kubernetes	December 5, 2018

Announcement	Released at
[System Restoration] ECS OpenAPI errors on February 23, 2021 are fixed	February 23, 2021
[System Restoration] The issue where data disks fail to be mounted to a multi-zone ACK cluster is fixed	August 5, 2019

2.[Product Changes] ACK API enhances user authentication

Beginning August 18, 2021, Container Service for Kubernetes (ACK) implements enhanced authentication when Resource Access Management (RAM) users and roles make API calls. To prevent authentication errors due to unauthorized API calls, you must check the RAM policies that are attached to the RAM users and roles within your Alibaba Cloud account and add the required permissions based on your needs.

Impact

After enhanced authentication is used, if a RAM user or role attempts to perform an unauthorized operation, the ACK console or API returns an error message that contains the following content: `RAM policy Forbidden` or `STSToken policy Forbidden`. The RAM action that is required to perform the operation is also included in the error message.

The following error message contains a RAM action named cs:DescribeEvents:

```
RAM policy Forbidden for action cs:DescribeEvents
```

The following table lists API operations and the RAM actions that are required to call the API operations. If your RAM user or role is unauthorized to call the API operations in the following table, log on to the RAM console and grant the required permissions to the RAM user or role.

Operation	RAM Action	Description
DescribeEvents	cs:DescribeEvents	Queries user events
StartAlert	cs:StartAlert	Enables an alert rule
StopAlert	cs:StopAlert	Disables an alert rule
DeleteAlertContact	cs:DeleteAlertContact	Deletes an alert contact
DeleteAlertContactGroup	cs:DeleteAlertContactGroup	Deletes an alert contact group
OpenAckService	cs:OpenAckService	Activates ACK
DescribeClusterResources	cs:DescribeClusterResources	Queries all resources in a cluster by cluster ID
DescribeUserQuota	cs:DescribeUserQuota	Queries resource quotas
DescribeClustersV1	cs:DescribeClustersV1	Queries the details about all clusters
DescribeExternalAgent	cs:DescribeExternalAgent	Queries a cluster registration proxy by cluster ID
DescribeKubernetesVersionMetadata	cs:DescribeKubernetesVersionMetadata	Queries the supported Kubernetes versions

Operation	RAM Action	Description
DescribeClusterAddonUpgradeStatus	cs:DescribeClusterAddonUpgradeStatus	Queries the upgrade progress of cluster add-ons
DescribeClusters	cs:DescribeClusters	Queries all clusters within the account, including Kubernetes clusters and Swarm clusters
DescribeClusterNamespaces	cs:DescribeClusterNamespaces	Queries the namespaces in a cluster
ModifyCluster	cs:ModifyCluster	Modifies the cluster configurations by cluster ID
MigrateCluster	cs:MigrateCluster	Migrates a cluster
UpdateK8sClusterUserConfigExpire	cs:UpdateK8sClusterUserConfigExpire	Updates the expiration time of custom configurations
DescribeClusterNodes	cs:DescribeClusterNodes	Queries the details about all nodes in a cluster by cluster ID
DescribeClusterAttachScripts	cs:DescribeClusterAttachScripts	Queries the script that is used to add instances to a cluster
GetUpgradeStatus	cs:GetUpgradeStatus	Queries the upgrade progress of a cluster by cluster ID
UpgradeCluster	cs:UpgradeCluster	Upgrades a cluster by cluster ID
PauseClusterUpgrade	cs:PauseClusterUpgrade	Pauses the upgrade of a cluster
CancelClusterUpgrade	cs:CancelClusterUpgrade	Cancels the upgrade of a cluster
CreateTemplate	cs:CreateTemplate	Creates an orchestration template
DescribeTemplates	cs:DescribeTemplates	Queries the details about all orchestration templates
DescribeTemplateAttribute	cs:DescribeTemplateAttribute	Queries the details about an orchestration template by template ID
UpdateTemplate	cs:UpdateTemplate	Updates an orchestration template by template ID
DeleteTemplate	cs:DeleteTemplate	Deletes an orchestration template by template ID
CreateKubernetesTrigger	cs:CreateKubernetesTrigger	Creates a trigger for an application

Operation	RAM Action	Description
GetKubernetesTrigger	cs:GetKubernetesTrigger	Queries the triggers of an application by application name
DeleteKubernetesTrigger	cs:DeleteKubernetesTrigger	Deletes a trigger by trigger ID
InstallClusterAddons	cs:InstallClusterAddons	Installs components in a cluster
DescribeAddons	cs:DescribeAddons	Queries the details about all supported components
DescribeClusterAddonsUpgradeStatus	cs:DescribeClusterAddonsUpgradeStatus	Queries the upgrade progress of a component by component name
DescribeClusterAddonsVersion	cs:DescribeClusterAddonsVersion	Queries the details about all components in a cluster by cluster ID
ModifyClusterConfiguration	cs:ModifyClusterConfiguration	Applies only to managed clusters
UpgradeClusterAddons	cs:UpgradeClusterAddons	Upgrades a component to a specified version by component name
PauseComponentUpgrade	cs:PauseComponentUpgrade	Pauses the upgrade of a component
ResumeComponentUpgrade	cs:ResumeComponentUpgrade	Resumes the upgrade of a component
CancelComponentUpgrade	cs:CancelComponentUpgrade	Cancels the upgrade of a component
UnInstallClusterAddons	cs:UnInstallClusterAddons	Uninstalls a component by component name
CreateAutoscalingConfig	cs:CreateAutoscalingConfig	Configures auto scaling

Modify a RAM policy

The following example shows how to modify the RAM policy that is attached to a RAM user or role. For more information about RAM authorization, see Create a custom RAM policy.

Scenario 1: A RAM user can perform only the cs:Get* action on a cluster and requires permissions on all read-only operations related to the cluster

The following code block shows the RAM policy when a RAM user can perform only the `cs:Get*` action on a cluster:

```
{
    "Statement": [
        {
            "Action": "cs:Get*",
            "Effect": "Allow",
            "Resource": [
                "acs:cs:*:*:cluster/c2e63856bcd714197****"
            ]
        }
    ],
    "Version": "1"
}
```

If the RAM user requires permissions on all read-only operations related to the cluster, modify the RAM policy as shown in the following code block:

```
{
    "Statement": [
        {
            "Action": [
                "cs:Get*",
                "cs:List*",
                "cs:Describe*"
            ],
            "Effect": "Allow",
            "Resource": [
                "acs:cs:*:*:cluster/c2e63856bcd714197****"
            ]
        }
    ],
    "Version": "1"
}
```

Note The `cs:Get*` action does not include all read-only operations. To grant the RAM user permissions on all read-only operations, you must add the `cs:List*` and `cs:Describe*` actions to the RAM policy.

Scenario 2: Grant a RAM user the permissions on an individual operation related to a cluster

To grant a RAM user the permissions on an individual operation related to a cluster, you need only to add the RAM action that corresponds to the operation in the RAM policy.

The following code block shows the current RAM policy:

```
{
    "Statement": [
        {
            "Action": [
                "cs:Get*",
                "cs:List*",
                "cs:Describe*"
            ],
            "Effect": "Allow",
            "Resource": [
                "acs:cs:*:*:cluster/c2e63856bcd714197****"
            ]
        }
    ],
    "Version": "1"
}
```

To grand the permissions on the `ModifyCluster` operation, you must add the corresponding RAM action `cs:ModifyCluster` to the RAM policy, as shown in the following code block:

```
{
    "Statement": [
        {
            "Action": [
                "cs:Get*",
                "cs:List*",
                "cs:Describe*",
                "cs:ModifyCluster"
            ],
            "Effect": "Allow",
            "Resource": [
                "acs:cs:*:*:cluster/c2e63856bcd714197****"
            ]
        }
    ],
    "Version": "1"
}
```

Scenario 3: Grant a RAM user permissions on operations that are not specific to individual clusters

Some API operations are not specific to individual clusters, such as `CreateCluster`, `DescribeClusters`, and `DescribeEvents`. To grant a RAM user permissions on these operations, you must not specify cluster IDs in the `Resource` section.

The following code block shows the current RAM policy:

```
{
    "Statement": [
        {
            "Action": [
                "cs:Get*",
                "cs:List*",
                "cs:Describe*"
            ],
            "Effect": "Allow",
            "Resource": [
                "acs:cs:*:*:cluster/c2e63856bcd714197****"
            ]
        }
    ],
    "Version": "1"
}
```

To grant the permissions on the `DescribeEvents` operation, you must add the corresponding RAM action `cs:DescribeEvents` to the RAM policy, as shown in the following code block:

```
{
    "Statement": [
        {
            "Action": [
                "cs:DescribeEvents"
            ],
            "Effect": "Allow",
            "Resource": [
              "*"
            ]
        },
        {
            "Action": [
                "cs:Get*",
                "cs:List*",
                "cs:Describe*"
            ],
            "Effect": "Allow",
            "Resource": [
                "acs:cs:*:*:cluster/c2e63856bcd714197****"
            ]
        }
    ],
    "Version": "1"
}
```

3.[System Restoration] ECS OpenAPI errors on February 23, 2021 are fixed

Background information

Due to the errors in Elastic Compute Service (ECS) OpenAPI on February 23, 2021, IP addresses allocated to pods that were created on this day may be invalid. As a result, the pods may become inaccessible. Only ACK clusters that run Terway in exclusive ENI mode or inclusive ENI mode are affected. We recommend that you perform the following steps to check for pods that have this issue and then fix the issue:

Procedure

Step 1: Run a script to scan nodes

Run the following script on each node:

```
#!/bin/bash
set -e
err(){
    echo "error at line $1"
}
trap 'err $LINENO' ERR
check(){
    cid=$1
    pid=$(docker inspect $cid -f '{{.State.Pid}}')
    if [ -z "$pid" ]; then
        echo 'cannot get pid from container $cid'
        return 1
    fi
    nsenter -t $pid -n curl -s --connect-timeout 4 100.100.100.200 -o /dev/null
}
for line in $(docker ps|grep -v k8s_POD|awk '$NF~/^k8s_/{print $1"_"$NF}'|awk -F_ '{print $
1"_"$3"_"$4"_"$5}')
do
    IFS=_ read cid cname pod namespace <<< $line
    if ! check $cid; then
        echo "pod $namespace/$pod has connectivity issues"
    fi
    if [[ "$cname" == "terway" && "$namespace" == "kube-system"  && "$pod" =~ ^(terway-|ter
way-eniip-|terway-eni-) ]]; then
        terway_container=$cid
    fi
done
if [ -n "$terway_container" ]; then
   for pod in $(docker exec -it $terway_container terway-cli mapping|sed -r "s/\x1B\[([0-9]
{1,3}(;[0-9]{1,2})?)?[mGK]//g"|awk '$3=="X"{print $1}')
   do
        echo "pod $pod on this host has connectivity issues"
   done
fi
```

Expected output:

```
pod *** has connectivity issues
```

If the preceding output is returned, it indicates that the pod may be assigned an invalid IP address.

Step 2: Recreate the pods that cannot connect to the network

1. Recreate the pods on nodes where Terway is installed.

   ```
   kubectl -n kube-system delete pod -l app=terway
   kubectl -n kube-system delete pod -l app=terway-eniip
   kubectl -n kube-system delete pod -l app=terway-eni
   ```

2. Recreate the pods that cannot connect to the network.
   - If the pod is created from a Deployment or DaemonSet, you can directly delete the pod. Then, the system recreates the pod.
   - If the pod is manually created, you must delete the pod and recreate it.

> **Note** The preceding script is executed to reload the Terway plug-in on nodes where the pods cannot connect to the network. After the script is executed, you can perform the operations in **Step 1: Run a script to scan nodes** again to check whether the cluster works as expected. If the issue persists, Submit a ticket.

4.[Product Changes] Alibaba Cloud charges for professional Kubernetes clusters

Starting from 00:00 (UTC+8) October 23, 2020, professional Kubernetes clusters are charged for commercial purposes.

Pricing

For more information about the pricing and billing of professional Kubernetes clusters, see Billing.

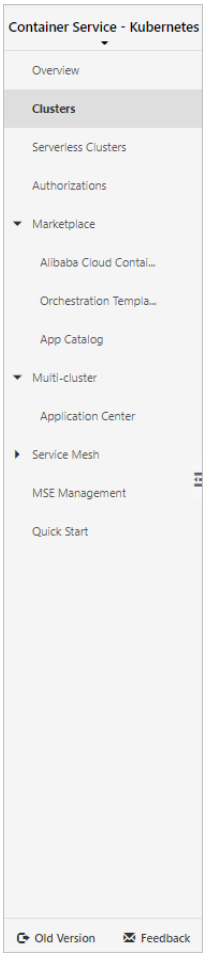
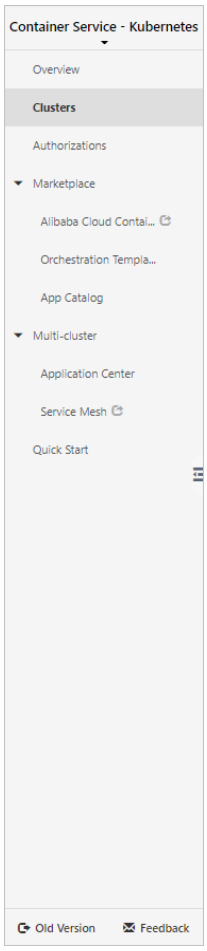
Features

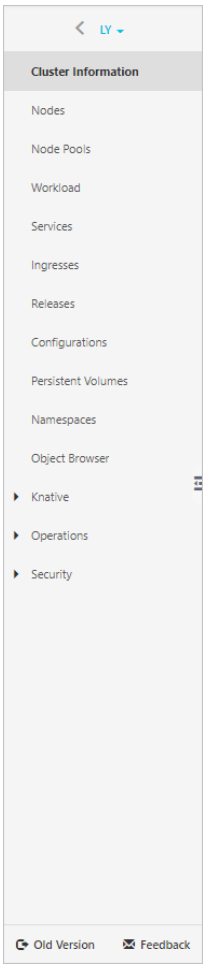
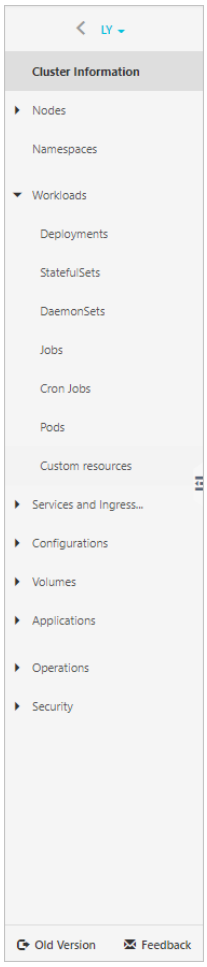
Professional Kubernetes clusters are developed based on managed Kubernetes clusters. Professional Kubernetes clusters are covered by the service-level agreement (SLA) that includes compensation clauses. This type of cluster is suitable for enterprise users that require higher stability and security for production environments. For more information, see Introduction to professional managed Kubernetes clusters.

5.[Product Changes] Optimize the navigation pane in the ACK console

To improve the user experience of the Container Service for Kubernetes (ACK) console, ACK has optimized the home navigation pane and the navigation pane on the cluster details page.

Module	Before adjustment	After adjustment	Description

Module	Before adjustment	After adjustment	Description
Home navigation pane			• **Serverless Clusters** is moved to **Clusters**. • **Service Mesh** is moved to the **Multi-cluster** menu and redirects to the Alibaba Cloud Service Mesh (ASM) console. • **MSE Management** is moved to the **Applications** menu.

Module	Before adjustment	After adjustment	Description
Navigation pane on the cluster details page			The following list describes each menu and the related submenus in the navigation pane after the adjustments: • Nodes: node pools and nodes. • Workloads: Deployments, StatefulSets, DaemonSets, Jobs, CronJobs, pods, and custom resources. • Services and Ingresses: Services and Ingresses. • Configurations: ConfigMaps and Secrets. • Volumes: persistent volume claims (PVCs), persistent volumes (PVs), and StorageClasses. • Applications: Helm, canary release (public preview), service mesh, Knative, and workflows. **Note** The **Workflows** menu is available only to users in the whitelist. • Operations: events, Prometheus monitoring, add-ons, cluster check, cluster upgrading, and runtime upgrading. • Security: authorization, cluster auditing, policy management, inspections, and runtime security.

6.[Product Changes] ACK ends support for Kubernetes Dashboard

Container Service for Kubernetes (ACK) clusters of Kubernetes 1.18 and later no longer support Kubernetes Dashboard. To use Kubernetes Dashboard, we recommended that you deploy the kubernetes-dashboard application on the App Catalog page. Log on to the ACK console to deploy kubernetes-dashboard.

7.[Kubernetes Versions] ACK ends support for Kubernetes 1.12.6 and earlier

Container Service for Kubernetes (ACK) no longer supports Kubernetes 1.12.6 and earlier based on the support policy for Kubernetes versions. ACK also ends technical support for Kubernetes 1.12.6 and earlier. We recommend that you upgrade Kubernetes versions for your ACK clusters as soon as possible. For more information about how to upgrade a cluster, see Upgrade the Kubernetes version of an ACK cluster.

For more information about the support policy for Kubernetes versions, see Kubernetes versions.

8.[Component Upgrades] Upgrade the cloud controller manager

To improve the stability of Container Service for Kubernetes (ACK) clusters, we recommend that you upgrade the cloud control manager to V1.9.3.340. This topic describes how to manually upgrade the cloud controller manager.

Context

Starting from December 15, 2020, ACK will upgrade the cloud controller manager to the latest version for your clusters where the installed versions are earlier than V1.9.3.340. This improves the stability of your clusters. If you manually upgrade the cloud controller manager before December 15, 2020, ACK skips the upgrade plan for your clusters. You can also Submit a ticket to skip the upgrade and continue using the current versions. We recommend that you join the upgrade plan or manually upgrade the cloud controller manager.

Manually upgrade the cloud controller manager

1. Log on to the ACK console.
2. In the left-side navigation pane of the ACK console, click **Clusters**.
3. On the **Clusters** page, find the cluster that you want to manage and click the name of the cluster or click **Details** in the **Actions** column. The details page of the cluster appears.
4. In the left-side navigation pane of the details page, choose **Operations > Add-ons**.
5. On the **Add-ons** page, click the **Core Components** tab, find the **Cloud Control Manager** section, and then click **Upgrade**.

   > **Note** If no Upgrade button is displayed in the **Cloud Control Manager** section, it indicates that the installed cloud controller manager is the latest version.

6. In the **Note** message, click **OK**.

9.[Component Upgrades] Upgrade CoreDNS

To improve the stability of DNS resolution in Container Service for Kubernetes (ACK) clusters, we recommend that you upgrade CoreDNS to the latest version. This topic describes how to upgrade CoreDNS.

Context

The following issues exist in CoreDNS versions earlier than 1.7.0 and may affect the stability of DNS resolution in ACK clusters:

- If connectivity exceptions occur between CoreDNS and the API server, such as network jitters, API server restarts, or API server migrations, CoreDNS pods may be restarted because error logs cannot be written. For more information, see Set klog's logtostderr flag.
- CoreDNS occupies extra memory resources during the initialization process. In this process, the default memory limit may cause out of memory (OOM) errors in large-scale clusters. If this situation intensifies, CoreDNS pods may be restarted repetitively but fail to be started. For more information, see CoreDNS uses a lot memory during initialization phase.
- CoreDNS has issues that may affect the domain name resolution of headless Services and requests from outside the cluster. For more information, see plugin/kubernetes: handle tombstones in default processor and Data is not synced when CoreDNS reconnects to kubernetes api server after protracted disconnection.
- Some earlier CoreDNS versions are configured with default toleration rules that may cause CoreDNS pods to fail to be automatically evicted when exceptions occur on the host node. This may lead to domain name resolution errors in the cluster.

Update methods

Before you upgrade CoreDNS, we recommend that you read CoreDNS release notes and CoreDNS community changelog to learn the upgrade details and notes.

You can upgrade CoreDNS automatically or manually:

- Automatically upgrade CoreDNS: Go to the **Add-ons** page of the ACK console and find the CoreDNS component. If the Upgrade button appears on the page, it indicates that CoreDNS is upgradable. For more information, see Automatically upgrade CoreDNS.
- Manually upgrade CoreDNS: If no Upgrade button is displayed on the **Add-ons** page of the ACK console and the current CoreDNS version is outdated, it means that you cannot automatically upgrade CoreDNS for your cluster. In this case, you can manually upgrade CoreDNS. For more information, see Manually upgrade CoreDNS.

Automatically upgrade CoreDNS

You can upgrade CoreDNS on the Add-ons page of the ACK console. For more information about the precautions for upgrading CoreDNS, see Precautions for upgrading CoreDNS. For more information about how to automatically upgrade CoreDNS, see Manage system components.

For ACK clusters of Kubernetes 1.14.8 and later, you can upgrade CoreDNS to the latest version. For ACK clusters whose Kubernetes versions are earlier than 1.14.8, you can upgrade CoreDNS to 1.6.2 or earlier.

If you used to manually upgrade CoreDNS, you must check the CoreDNS configuration file and make sure that the ready plug-in is enabled before you can configure to automatically upgrade CoreDNS. If the ready plug-in is not specified in the configuration file, you must enable the ready plug-in before you configure to automatically upgrade CoreDNS. For more information about how to enable the ready plug-in, see Enable the ready plug-in.

Manually upgrade CoreDNS

1. Check the compatibility between the Kubernetes version and the CoreDNS version.

   Check the Kubernetes version of the ACK cluster. Make sure that the Kubernetes version is compatible with CoreDNS 1.6.2. The following table lists the Kubernetes versions that are compatible with CoreDNS 1.6.2. Kubernetes 1.11, 1.12, 1.14, and 1.16 are compatible with CoreDNS 1.6.2.

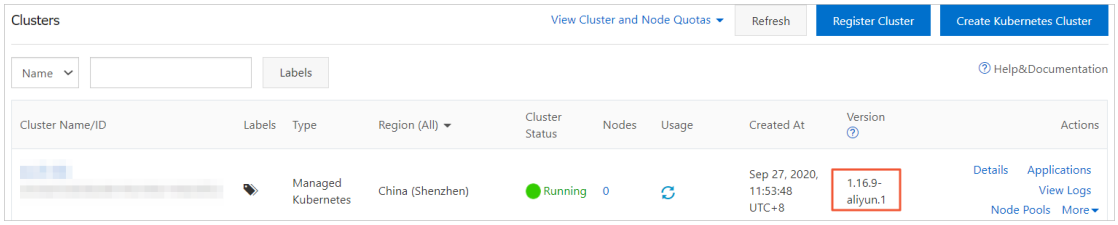
Item	Compatible version			
Kubernetes version	1.11	1.12	1.14	1.16
CoreDNS	1.6.2	1.6.2	1.6.2	1.6.2

   Perform the following steps to check the Kubernetes version of an ACK cluster:

   i. Log on to the ACK console.

   ii. In the left-side navigation pane of the ACK console, click **Clusters**.

   iii. On the Clusters page, find the cluster that you want to manage and check the Kubernetes version in the Version column.



2. Check the CoreDNS version.

   - You can check the CoreDNS version in the ACK console.

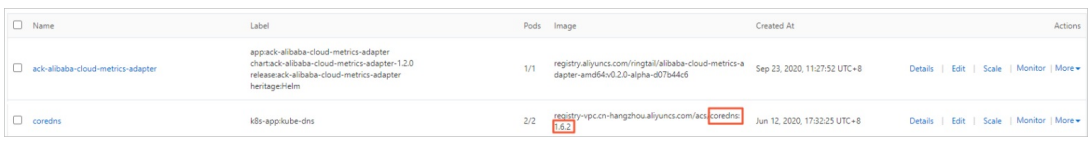
     a. Log on to the Container Service for Kubernetes (ACK) console.

     b. In the left-side navigation pane of the ACK console, click **Clusters**.

     c. On the **Clusters** page, find the cluster that you want to manage. Then, click the name of the cluster or click **Details** in the **Actions** column.

     d. In the left-side navigation pane of the details page, choose **Workloads > Deployments**.

     e. On the **Deployments** page, set **Namespace** to kube-system and check the CoreDNS version in the Image column.



   - You can also run the following **kubectl** command to query the CoreDNS version:

```
kubectl get deployment coredns -n kube-system -o jsonpath="{.spec.template.spec.containers[0].image}"
```

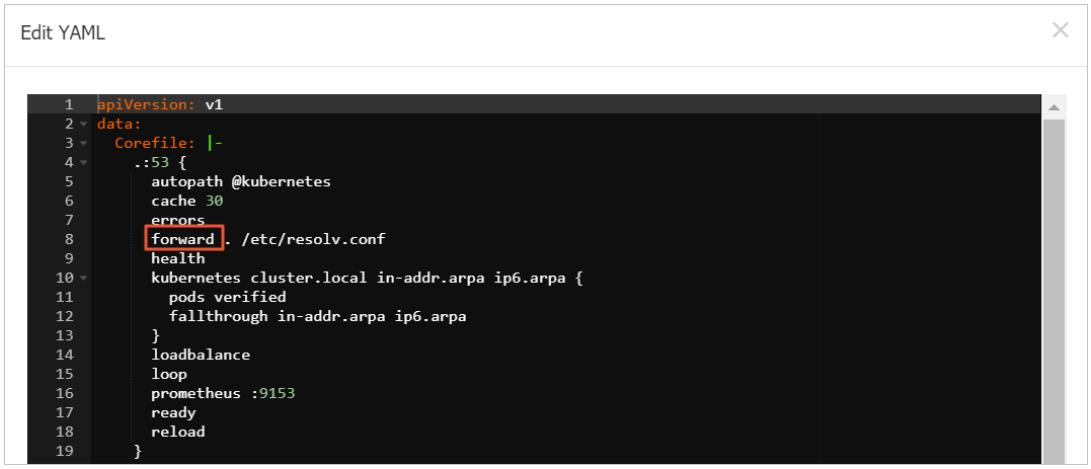
Expected output:

```
registry-vpc.cn-hangzhou.aliyuncs.com/acs/coredns:1.3.1
```

3. Modify the coredns ConfigMap.

   In CoreDNS 1.6.2, the proxy plug-in is replaced by the forward plug-in. You must replace proxy with forward in the **coredns** ConfigMap in the kube-system namespace.

   - You can modify the **coredns** ConfigMap in the ACK console.
     a. Log on to the Container Service for Kubernetes (ACK) console.
     b. In the left-side navigation pane of the ACK console, click **Clusters**.
     c. On the **Clusters** page, find the cluster that you want to manage. Then, click the name of the cluster or click **Details** in the **Actions** column.
     d. In the left-side navigation pane of the details page, choose **Configurations > ConfigMaps**.
     e. On the top of the **ConfigMap** page, set **Namespace** to kube-system. Find the coredns ConfigMap and click **Edit YAML** in the **Actions** column.
     f. In the **View in YAML** panel, replace **proxy** with **forward** and click **OK**.



   - You can also run a **kubectl** command to modify the coredns ConfigMap.

```
# Modify the coredns ConfigMap.
kubectl edit configmap/coredns -n kube-system
# Replace proxy with forward.
# Save the change and exit.
```

4. Print the log of a CoreDNS pod to check whether the new configuration is loaded. It requires about 30 seconds for hot loading to complete.

i. Run the following command to query the status of CoreDNS pods in the cluster:

```
kubectl get pods -n kube-system | grep coredns
```

Expected output:

```
coredns-78d4b8bd88-6g62w                             1/1     Running   0          9d
coredns-78d4b8bd88-n6wjm                             1/1     Running   0          9d
```

ii. Run the following command to query the log of a CoreDNS pod:

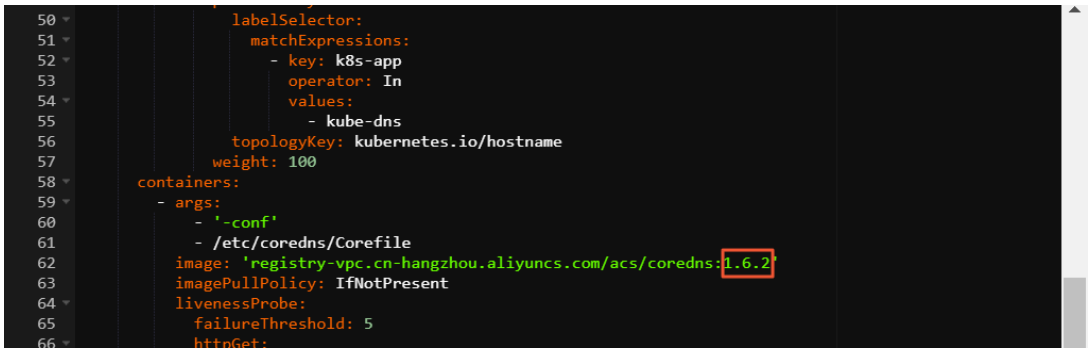
```
kubectl logs coredns-78d4b8bd88-n6wjm -n kube-system
```

Expected output:

```
.:53
[INFO] plugin/reload: Running configuration MD5 = 71c5f1ff539d304c630521f315dc2ac2
CoreDNS-1.6.7
linux/amd64, go1.13.6, da7f65b
[INFO] 127.0.0.1:48329 - 42313 "HINFO IN 1108347002237365533.4506541768939609094. u
dp 57 false 512" NXDOMAIN qr,rd,ra 132 0.008874794s
```

The output shows `plugin/reload`. This indicates that the new CoreDNS configuration is loaded.

5. Change the image version of the CoreDNS application to V1.6.2.
   - Change the image version in the ACK console.
     a. Log on to the Container Service for Kubernetes (ACK) console.
     b. In the left-side navigation pane of the ACK console, click **Clusters**.
     c. On the **Clusters** page, find the cluster that you want to manage. Then, click the name of the cluster or click **Details** in the **Actions** column.
     d. In the left-side navigation pane of the details page, choose **Workloads > Deployments**.
     e. On the top of the **Deployments** page, set **Namespace** to kube-system. Find **coredns** and choose **More > View in YAML** in the Actions column.
     f. In the **Edit YAML** dialog box, change the image version to **1.6.2**. Click **Update**.



   - Change the image version by using **kubectl**.

```
# Modify the coredns Deployment.
kubectl edit deployment/coredns -n kube-system
# Change the image version to 1.6.2.
# Save the change and exit.
```

6. Verify the result.

   Run the following command to check whether all CoreDNS pods in the cluster are in the Running state:

   ```
   kubectl get pods -n kube-system | grep coredns
   ```

   Expected output:

   ```
   coredns-78d4b8bd88-6g62w                            1/1     Running   0          9d
   coredns-78d4b8bd88-n6wjm                            1/1     Running   0          9d
   ```

Enable the ready plug-in

If you used to manually upgrade CoreDNS by following the guide in this topic and the current CoreDNS version is later than 1.5.0, confirm that the ready plug-in is enabled in the CoreDNS configuration file. Otherwise, CoreDNS fails to start.

1. Log on to the ACK console.
2. In the left-side navigation pane of the ACK console, click **Clusters**.
3. On the **Clusters** page, find the cluster that you want to manage and click the name of the cluster or click **Details** in the **Actions** column. The details page of the cluster appears.
4. In the left-side navigation pane of the details page, choose **Configurations > ConfigMaps**.
5. On the top of the **ConfigMap** page, set **Namespace** to kube-system. Find the coredns ConfigMap and click **Edit YAML** in the **Actions** column.
6. In the **View in YAML** panel, check whether the `ready` field exists. If not, add the `ready` field and click **OK**.

   ```
   apiVersion: v1
   data:
     Corefile: |
       .:53 {
           errors
           health {
               lameduck 15s
           }
           ready # Add this line and make sure that word "ready" is aligned with word "kub
   ernetes".
           kubernetes cluster.local in-addr.arpa ip6.arpa {
               pods verified
               fallthrough in-addr.arpa ip6.arpa
           }
           prometheus :9153
           forward . /etc/resolv.conf {
               max_concurrent 1000
           }
           cache 30
           loop
           log
           reload
           loadbalance
       }
   ```

7. Run the following command to print the log of a CoreDNS pod to check whether the new configuration is loaded. It requires about 30 seconds for hot loading to complete.

   ```
   kubectl logs coredns-78d4b8bd88-n6wjm -n kube-system
   ```

   The output shows `plugin/reload`. This indicates that the new CoreDNS configuration is loaded.

Related information

- CoreDNS

10.[Component Upgrades] Upgrade Helm V2 to V3

Container Service for Kubernetes (ACK) has upgraded Helm to V3 for newly created clusters. The Tiller server component for Helm V2 has known security issues among community users. Attackers can use Tiller to install unauthorized applications in the cluster. We recommend that you upgrade to Helm V3 at your earliest opportunity.

Impact scope

Run the **kubectl get deploy -n kube-system tiller-deploy** command to check whether a tiller Deployment exists. If a tiller Deployment exists, check the following conditions:

- Whether the kubeconfig file of your ACK cluster is provided to external users.
- Whether external access to the ACK console is allowed.
- Whether your ACK cluster is used in a multi-tenancy scenario and whether privilege isolation is enabled among users.

If one of the preceding conditions is met, we recommend that you upgrade to Helm V3.

Unaffected scenarios or scenarios where upgrade to Helm V3 is unavailable

If the scenario does not meet the preceding conditions or you cannot upgrade to Helm V3, we recommend that you manually upgrade Tiller of Helm V2 to the latest version for higher security. Perform the following steps to upgrade Tiller of Helm V2 to the latest version:

1. Run the following command:

   ```
   helm init --tiller-image registry.cn-hangzhou.aliyuncs.com/acs/tiller:v2.16.3 --upgrade
   ```

2. After Tiller passes the health check, you can run the **helm version** command to query the upgrade result.

   The preceding command upgrades only the server component of Helm. To download the client components for different operating systems, click the download link.

> **Note** If the scenario does not meet the preceding conditions or you cannot upgrade to Helm V3, you can upgrade Tiller to the latest version and skip the following steps. You can upgrade to Helm V3 later.

Precheck

Before you upgrade Helm V2, perform the following steps for a precheck.

1. Check whether Tiller is installed in your ACK cluster. Run the **kubectl get deploy -n kube-system tiller-deploy** command to check whether a tiller Deployment exists.
2. If a tiller Deployment exists, run the **helm ls -a** command to check whether applications are installed.
3. If applications are installed, you must first delete these applications due to the data incompatibility between Helm V2 and V3.

> **Notice** The Helm community provides a plug-in to migrate Helm V2 configurations and releases to Helm V3. To prevent data loss, proceed with caution when you use the plug-in. For more information about the plug-in, see helm-2to3.

Upgrade procedure

1. Make sure that you have passed the Precheck.
2. Run the **kubectl delete deploy tiller-deploy -n kube-system** command.
3. Download the Helm V3 client component to install new applications.

> **Notice**
>
> Before you install new applications, take note of the following items:
>
> - You must use Helm V3 to reinstall the applications that were installed by using Helm V2. Evaluate the impacts on your workloads.
> - When you reinstall an application, the original data will be lost. Back up your data in advance.

Related information

- Helm v3 Change log
- Differences between Helm V3 and Helm V2
- Migrate Helm V2 to V3

11.[Product Changes] ACK ends support for Swarm

Container Service for Kubernetes (ACK) will end technical support for Swarm at the end of 2019. We appreciate your support for Container Service for Swarm in the past years and will continue to provide more stable and reliable enterprise-class services with the support of ACK in the future. You can start to plan the migration of your services deployed in Container Service for Swarm clusters.

1. Starting from July 1, 2019, Container Service for Swarm clusters can no longer be created in the console. If you need to create Container Service for Swarm clusters in the console, submit a ticket.
2. Starting from December 31, 2019, the Container Service for Swarm documentation is removed and the Container Service for Swarm console is no longer in use. Technical support for Container Service for Swarm is also no longer provided. You can manage your Container Service for Swarm clusters by calling the API.

To help you migrate from Container Service for Swarm to ACK, we release a migration guide in the ACK documentation. For more information, see Overview. You can contact us if you have any questions.

12.[Component Upgrades] Upgrade Terway

- v1.0.9.15-g3957085-aliyun

  The issue that the Terway upgrade occasionally fails is fixed.

- v1.0.9.14-ga0346bb-aliyun
  - The issue that Terway fails to obtain the elastic network interface (ENI) information is fixed.
  - The issue that the **failed to move veth to host netns: file exists** error is reported during container creation is fixed.
  - Periodic scanning is supported to check the status of ENIs. ENIs that are abnormally released are periodically recycled.
  - Health checks are optimized. TCP port checks are performed instead of HTTP path checks.

You can upgrade Terway in the ACK console. The upgrade does not affect your workloads.

13.[Kubernetes Versions] Release the support policy for Kubernetes versions

To provide you with stable and reliable Kubernetes versions, we support four Kubernetes versions based on the support policy for Kubernetes versions. Each Kubernetes version will be maintained for one year. Make sure that you upgrade your Kubernetes version in a timely manner.

For more information, see Kubernetes versions.

14.[Product Changes] Update the security policies for cluster authorizations

Container Service for Kubernetes (ACK) will update the security policies for cluster authorizations a week later. After the security policies are updated, unauthorized Resource Access Management (RAM) users cannot access cluster resources. You can grant role-based access control (RBAC) permissions and RAM permissions to RAM users to manage your clusters. For more information, see Assign RBAC roles to RAM users. After the security policies are updated, RAM users are granted limited permissions only on clusters within the authorization domain. RAM users are no longer allowed to access clusters outside the authorization domain in compatibility mode.

15.[System Restoration] The issue where data disks fail to be mounted to a multi-zone ACK cluster is fixed

Alibaba Cloud has fixed the issue where data disks cannot be mounted to a multi-zone Container Service for Kubernetes (ACK) cluster. This issue is fixed for newly created multi-zone ACK clusters. In existing multi-zone ACK clusters, if the number of running applications or pulled images increases, the disk space may become insufficient on nodes where the Docker data directory is not mounted with data disks. ACK provides a solution to this issue. For more information, see Mount a data disk to a node. If you have further questions or require technical support, contact the ACK support team on DingTalk.

16.[Product Changes] Alibaba Cloud charges for elastic container instances used by ASK clusters

Starting from 10:00:00 (UTC+8), January 22, 2019, Alibaba Cloud charges for the elastic container instances that are used by serverless Kubernetes (ASK) clusters. For more information, see Pricing. If you have created ASK clusters and pods that run on elastic container instances, make sure that you have a sufficient budget for the costs of the elastic container instances. Elastic container instances are billed when they serve your workloads. If you no longer require elastic container instances, delete them. Otherwise, you may be charged extra fees. You can continue to use ASK clusters free of charge to serve your workloads.

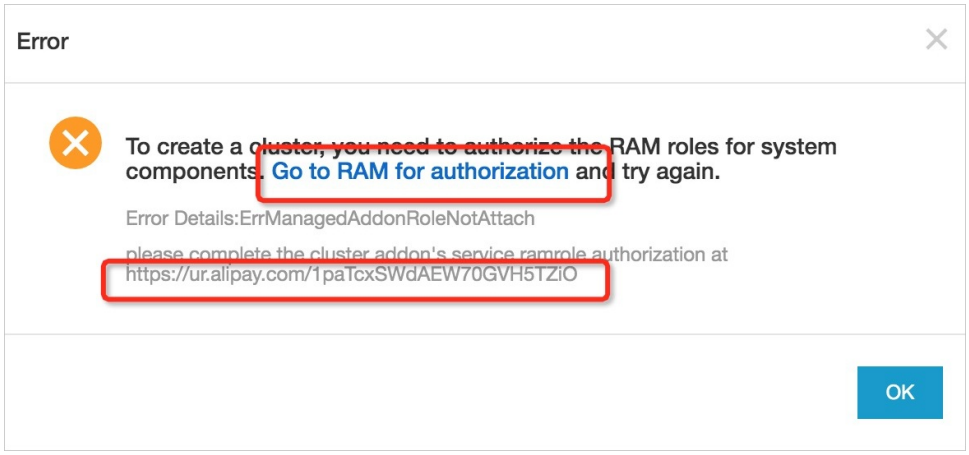
17.[Product Changes] ACK reduces the permissions of worker RAM roles in managed Kubernetes clusters

The default permission policy WorkerRolePolicy attached to worker roles in a managed Kubernetes cluster has excessive permissions. To improve data security and resource isolation in multi-tenancy scenarios, Container Service for Kubernetes (ACK) reduces the permissions of Resource Access Management (RAM) roles assigned to the worker nodes in managed Kubernetes clusters.

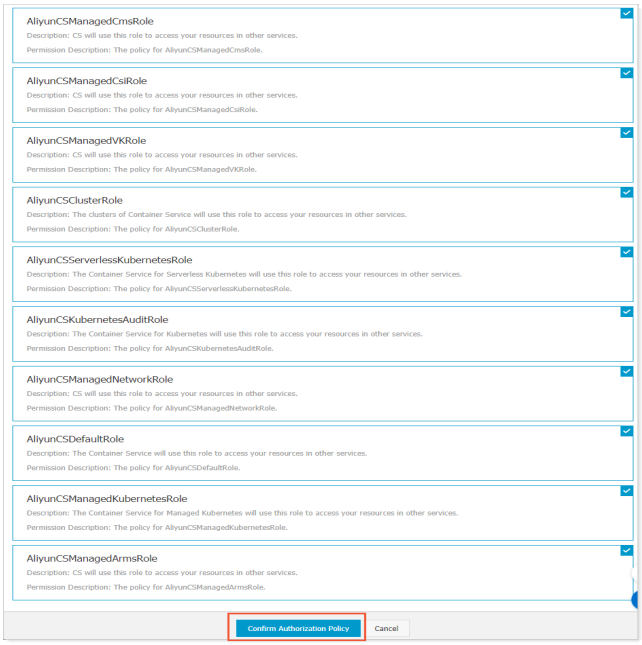
Assign RAM roles

ACK removes the permissions required for add-on management from worker RAM roles, and introduces new system roles to manage permissions on different add-ons. After the permissions are reduced, the following message appears when you create new managed Kubernetes clusters in the ACK console. You can log on with your Alibaba Cloud account or as a RAM user that is attached with the AliyunRAMFullAccess or AdministratorAccess policy and click **Go to RAM console** to perform the authorization.

> **Note** If you create ACK clusters by calling the API, click RAM access comtrol to assign the required RAM roles.



At the bottom of the Cloud Resource Access Authorization page, click **Confirm Authorization Policy**. Then, log on to the ACK console again and create ACK clusters.



The preceding operation assigns the following system roles to your account. These roles are required for add-on management by calling the API.

- AliyunCSManagedLogRole
- AliyunCSManagedCmsRole
- AliyunCSManagedCsiRole
- AliyunCSManagedVKRole
- AliyunCSManagedNetworkRole
- AliyunCSManagedArmsRole

The following code block shows the default RAM policy that is attached to the worker RAM roles of your ACK cluster after the permissions are reduced:

```
{
  "Version": "1",
  "Statement": [{
      "Action": [
        "ecs:DescribeInstanceAttribute",
        "ecs:DescribeInstanceTypesNew",
        "ecs:DescribeInstances"
      ],
      "Resource": [
        "*"
      ],
      "Effect": "Allow"
    },
    {
      "Action": [
        "log:GetProject",
        "log:GetLogStore",
        "log:GetConfig",
        "log:GetMachineGroup",
        "log:GetAppliedMachineGroups",
        "log:GetAppliedConfigs",
        "log:GetIndex",
        "log:GetSavedSearch",
        "log:GetDashboard",
        "log:GetJob"
      ],
      "Resource": [
        "*"
      ],
      "Effect": "Allow"
    },
    {
      "Action": [
        "cr:GetAuthorizationToken",
        "cr:ListInstanceEndpoint",
        "cr:PullRepository"
      ],
      "Resource": [
        "*"
      ],
      "Effect": "Allow"
     }
  ]
}
```

18.[CVE Securities] CVE vulnerability Fixes

18.1. Vulnerability fixed: CVE-2021-25742

Vulnerability CVE-2021-25742 was recently disclosed by Kubernetes. This vulnerability is related to the ingress-nginx component. This vulnerability can be exploited by attackers to use the custom snippets feature to create or modify Ingresses and obtain all Secrets in a cluster. This topic describes the impacts, affected ingress-nginx versions, and fixes for this vulnerability.

CVE-2021-25742 is rated as high severity and its Common Vulnerability Scoring System (CVSS) score is 7.6.

Affected ingress-nginx versions

The following ingress-nginx versions are affected:

- v1.0.0
- ingress-nginx 0.49.0 and earlier

This vulnerability is fixed in the following ingress-nginx versions:

- v1.0.1
- v0.49.1

For more information about this vulnerability, see CVE-2021-25742.

Impacts

If the permissions to create and modify Ingresses are granted to a non-administrator user in a multi-tenant cluster, the user can use the custom snippets feature to obtain all Secrets in the cluster. This may cause unauthorized access to other tenants or secret information in the cluster.

Mitigation

Run the following command to modify the nginx-configuration ConfigMap in the kube-system namespace:

```
kubectl edit configmap -nkube-system nginx-configuration
```

Set `allow-snippet-annotations` to `false`:

```
data:
  allow-snippet-annotations: "false"
```

18.2. Vulnerability fixed: CVE-2021-41103

Vulnerability CVE-2021-41103 was recently disclosed by the containerd community. This vulnerability is related to the containerd runtime. If the permissions on container root directories and system components are not limited, unprivileged Linux users can traverse the entire container file system and execute programs. This topic describes the impacts, affected containerd versions, and fixes for this vulnerability.

CVE-2021-41103 is rated as medium severity and its Common Vulnerability Scoring System (CVSS) score is 5.9.

Affected containerd versions

The following containerd versions are affected:

- <v1.4.11
- <v1.5.7

This vulnerability is fixed in the following containerd versions:

- v1.14.11
- v1.5.7

For more information about this vulnerability, see CVE-2021-41103.

Impacts

If a multi-tenant cluster has executable programs with extended permission bits (such as setuid), unprivileged Linux users may discover and execute these programs. When the UID of an unprivileged Linux user on the host collided with the file owner or group inside a container, the unprivileged Linux user on the host can discover, read, and modify these files.

Mitigation

1. Allow only trusted users to access cluster nodes. Do not grant access permissions to untrusted users.
2. Remove unnecessary extended permissions on container bundles directories.

18.3. Vulnerability CVE-2021-25741 in Kubernetes

The Kubernetes community discovered CVE-2021-25741, a vulnerability that can be exploited by attackers to access the host directories by using a symbolic link and creating a container that has a subPath volume mounted. This topic describes the impacts, affected Kubernetes versions, and fixes of this vulnerability.

CVE-2021-25741 is rated as high severity and its Common Vulnerability Scoring System (CVSS) score is 8.8.

Affected versions

kubelet that is installed in clusters of the following Kubernetes versions is affected by this vulnerability:

- v1.22.0~v1.22.1
- v1.21.0~v1.21.4
- v1.20.0~v1.20.10

- 1.19.14 and earlier

This vulnerability is fixed in the following Kubernetes versions:

- v1.22.2
- v1.21.5
- v1.20.11
- v1.19.15

For more information about the vulnerability, see #104980.

Impacts

In multi-tenant scenarios, attackers with the permissions to start containers as the root user can exploit this vulnerability to escape into the host file system and obtain the read and write permissions on sensitive directories of the host.

Mitigation

Upgrade to Kubernetes 1.20.11-aliyun.1. CVE-2021-25741 is fixed in Kubernetes 1.20.11-aliyun.1.

18.4. Vulnerability fixed: CVE-2021-25740

The Kubernetes community discovered CVE-2021-25740, a vulnerability that can be exploited by attackers to launch confused deputy attacks to access cluster services that they would otherwise be unable to access. This topic describes the impacts of the vulnerability, and how to detect and mitigate the vulnerability.

CVE-2021-25740 is rated as low severity and its Common Vulnerability Scoring System (CVSS) score is 3.0.

Affected versions

All Kubernetes versions. For more information about the vulnerability, see #103675.

Impacts

If an attacker has permissions to create or modify Endpoints or EndpointSlices, the attacker can call the Kubernetes API to modify the addresses of Endpoints. In this case, the attacker may use a LoadBalancer or Ingress to access backend IP addresses that the attacker is not supposed to access. Besides, if the network policy of the cluster already trusts the LoadBalancer or Ingress, the network policy cannot be used to prevent exposure from other namespaces.

Detection

Services for which no selectors are specified rely on custom Endpoints and are vulnerable to the preceding attack. We recommend that you run the following command to check all Services and their selectors in the cluster:

```
kubectl get svc --all-namespaces -o=custom-columns='NAME:metadata.name,NAMESPACE:metadata.namespace,SELECTOR:spec.selector'
```

Note If no selectors are specified for the `default/kubernetes Service`, the Endpoints of the Service are managed by the Kubernetes API Server. This is normal.

Mitigation

No patch is available for this vulnerability. It can be mitigated only by restricting access to the vulnerable features. To mitigate the exposure, we recommend that you run the following commands to update the `system:aggregate-to-edit` role and restrict write access to Endpoints and EndpointSlices. This revokes write access to Endpoints from the `admin` and `edit` roles.

```
# Allow kubectl auth reconcile to work
kubectl annotate --overwrite clusterrole/system:aggregate-to-edit rbac.authorization.kubernetes.io/autoupdate=true
# Test reconcile, then run for real if happy
kubectl auth reconcile --remove-extra-permissions -f aggregate_to_edit_no_endpoints.yaml.txt --dry-run
kubectl auth reconcile --remove-extra-permissions -f aggregate_to_edit_no_endpoints.yaml.txt
# Prevent autoreconciliation back to old state
kubectl annotate --overwrite clusterrole/system:aggregate-to-edit rbac.authorization.kubernetes.io/autoupdate=false
```

The following code block shows the content of the *aggregate_to_edit_no_endpoints.yaml.txt* file that is used in the preceding commands:

```
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRole
metadata:
  annotations:
    rbac.authorization.kubernetes.io/autoupdate: "true"
  creationTimestamp: null
  labels:
    kubernetes.io/bootstrapping: rbac-defaults
    rbac.authorization.k8s.io/aggregate-to-edit: "true"
  name: system:aggregate-to-edit
rules:
- apiGroups:
  - ""
  resources:
  - pods/attach
  - pods/exec
  - pods/portforward
  - pods/proxy
  - secrets
  - services/proxy
  verbs:
  - get
  - list
  - watch
- apiGroups:
  - ""
  resources:
```

```
  - serviceaccounts
  verbs:
  - impersonate
- apiGroups:
  - ""
  resources:
  - pods
  - pods/attach
  - pods/exec
  - pods/portforward
  - pods/proxy
  verbs:
  - create
  - delete
  - deletecollection
  - patch
  - update
- apiGroups:
  - ""
  resources:
  - configmaps
  - persistentvolumeclaims
  - replicationcontrollers
  - replicationcontrollers/scale
  - secrets
  - serviceaccounts
  - services
  - services/proxy
  verbs:
  - create
  - delete
  - deletecollection
  - patch
  - update
- apiGroups:
  - apps
  resources:
  - daemonsets
  - deployments
  - deployments/rollback
  - deployments/scale
  - replicasets
  - replicasets/scale
  - statefulsets
  - statefulsets/scale
  verbs:
  - create
  - delete
  - deletecollection
  - patch
  - update
- apiGroups:
  - autoscaling
  resources:
```

```
  - horizontalpodautoscalers
  verbs:
  - create
  - delete
  - deletecollection
  - patch
  - update
- apiGroups:
  - batch
  resources:
  - cronjobs
  - jobs
  verbs:
  - create
  - delete
  - deletecollection
  - patch
  - update
- apiGroups:
  - extensions
  resources:
  - daemonsets
  - deployments
  - deployments/rollback
  - deployments/scale
  - ingresses
  - networkpolicies
  - replicasets
  - replicasets/scale
  - replicationcontrollers/scale
  verbs:
  - create
  - delete
  - deletecollection
  - patch
  - update
- apiGroups:
  - policy
  resources:
  - poddisruptionbudgets
  verbs:
  - create
  - delete
  - deletecollection
  - patch
  - update
- apiGroups:
  - networking.k8s.io
  resources:
  - ingresses
  - networkpolicies
  verbs:
  - create
  - delete
  - deletecollection
```

```
  - deletecollection
  - patch
  - update
```

After you complete the preceding operations, new versions of Kubernetes cannot modify the default permissions of the `system:aggregate-to-edit` role. No new default permissions are added to this role in Kubernetes V1.14.0 and later versions. We recommend that you remove the `autoupdate=false` annotation immediately after Kubernetes provides a fix to this vulnerability.

If you require the permissions to modify Endpoints and EndpointSlices for certain use cases, we recommend that you create a new role with the desired permissions and use the role only for these cases.

Similar attacks can be launched by using Ingresses that can forward traffic to ExternalName Services. In this case, an attacker can forward network traffic to Services in other namespaces or sensitive Endpoints. If you are using the Ingress API, we recommend that you check whether your Ingress can forward traffic to ExternalName Services. If your Ingress cannot forward traffic to ExternalName Services, you are not affected by this attack. If your Ingress can forward traffic to ExternalName Services, you must temporarily disable the functionality.

18.5. Vulnerability CVE-2021-25738

A vulnerability has been discovered in the Kubernetes Java client. An attacker can use specific YAML templates to execute mailicious code. This topic describes the affected Kubernetes Java client versions and fixes of the vulnerability.

The Common Vulnerability Scoring System (CVSS) score of this vulnerability is 6.7.

Affected versions

The following Kubernetes Java client versions are affected by this vulnerability:

- Kubernetes Java Client=v12.0.0
- Kubernetes Java client ≤ v11.0.1
- Kubernetes Java client ≤ v10.0.1
- Kubernetes Java client ≤ v9.0.2

The vulnerability is fixed in the following Kubernetes Java client versions:

- Kubernetes Java client master: 1676
- Kubernetes Java client ≥ v12.0.1: 1691
- Kubernetes Java client ≥ v11.0.2: 1692

Fixes

Use a patched Kubernetes Java client version to access your cluster. For more information about this vulnerability, see 1698.

18.6. Vulnerability fixed: CVE-2021-25737

Vulnerability CVE-2021-25737 was recently disclosed by Kubernetes. This vulnerability is related to the **kube-apiserver** component. Kubernetes does not check whether the IP address used by the endpoint of an EndpointSlice is valid. This can be exploited by attackers to redirect traffic to the network in which cluster nodes are deployed. This topic describes vulnerability CVE-2021-25737, the impacts of, and fixes for this vulnerability.

Vulnerability CVE-2021-25737 is rated as low severity and the Common Vulnerability Scoring System (CVSS) score of the vulnerability is 2.7.

Symptom

You can list all EndpointSlices in your cluster by running kubectl commands or calling the API. If the IP address of the endpoint of an EndpointSlice falls within `127.0.0.0/8` or `169.254.0.0/16`, the cluster is exposed to attacks that exploit this vulnerability. For more information, see #101084.

Impacts

The following **kube-apiserver** versions are affected:

- v1.21.0
- V1.20.0 to V1.20.6
- V1.19.0 to V1.19.10
- V1.16.0 to V1.18.18

> **Note** By default, the EndpointSlice feature is disabled in kube-apiserver 1.16 to 1.18. Therefore, clusters that use default kube-apiserver settings are not affected.

This vulnerability is fixed in the following Kubernetes versions:

- v1.21.1
- v1.20.7
- v1.19.11
- v1.18.19

Fixes

You can deploy the **gatekeeper** component to function as a validating admission webhook. This can prohibit access to the cluster network from EndpointSlices whose endpoint IP addresses fall within `127.0.0.0/8` or `169.254.0.0/16`. For more information, see gatekeeper.

18.7. Vulnerability fixed: CVE-2021-30465

The Open Containers Initiative community has reported a vulnerability that is related to runC. An attacker can use a symlink and exploit race condition flaws to mount the host file system to a container by creating a malicious pod. This results in a container escape. This topic describes CVE-2021-30465 and the affected versions, impacts, and fixes of the vulnerability.

Description

In a Kubernetes cluster, an attacker can specify a mount target as a symlink in a volume that is mounted to the root directory of the host. For example, */var/lib/kubelet/pods/$MY_POD_UID/volumes/kubernetes.io~empty-dir* specifies the symlink of an emptyDir volume. This way, the attacker can obtain a mount target from the host. The mount source is a directory that is controlled by the attacker. Therefore, the attacker can use a symlink to link the subdirectories from the mount source to the root directory of the host. Then, the attacker can exploit a Time Of Check To Time Of Use (TOCTTOU) flaw to mount a specified directory of the malicious container to the root directory of the host.

The severity of CVE-2021-30465 is rated high and the Common Vulnerability Scoring System (CVSS) score of the vulnerability is 7.6. For more information, see Official announcements.

Impacts

When multiple containers are started in a pod, an attacker can exploit a race condition and create a malicious pod that contains a mount target with a symlink. Under certain circumstances, the attacker can escape a container and access the host file system except for the container `rootfs`.

Affected versions

runC 1.0.0-rc94 and earlier versions are affected by this vulnerability.

The vulnerability is fixed in runC 1.0.0-rc95. For more information, see runC 1.0.0-rc95.

Fixes

Manually upgrade containerd to the latest version. For information about the latest containerd version, see runC 1.0.0-rc95.

18.8. Vulnerability CVE-2021-3121

The Kubernetes community has disclosed the CVE-2021-3121 vulnerability. This vulnerability allows a remote attacker to send crafted protobuf messages, which cause panics and result in a denial of service. If an earlier version of the gogo protobuf compiler is used in your Kubernetes cluster, the compiler may be affected by this vulnerability. This topic describes the impacts, affected components, and fixes of this vulnerability.

Impacts

The system components of Kubernetes can automatically recover when a panic occurs. The crafted protobuf messages cannot cause service interruptions. Therefore, the Kubernetes system components are not affected by this vulnerability.

Programs are affected by this vulnerability if they need to accept and handle protobuf messages but their components cannot gracefully handle panics. The attacks may result in a denial of service.

Affected versions

The Kubernetes community has tested and verified that the API server is not affected by this vulnerability. However, the Kubernetes community has updated related protobuf files to fix the vulnerability. The vulnerability is fixed in the following protobuf versions:

- v1.21: 1.21.0 and 1.21.1.
- v1.20: 1.20.6 and 1.20.7.
- v1.19: 1.19.10 and 1.19.11.

- v1.18: 1.18.18 and 1.18.19.

Fixes

If your application uses the automatically generated protobuf messages and you find a process that exits with messages similar to the following, an attacker may be exploiting this vulnerability:

```
panic: runtime error: index out of range [-9223372036854775804]
goroutine 1 [running]:
v1.(*MessageName).Unmarshal(0xc000057ef8, 0xc0000161a0, 0xa, 0x10, 0xc000057ec8, 0x1)
        .../protofile.pb.go:250 +0xb86
```

If you are using components related to protobuf messages, we recommend that you upgrade the gogo protobuf compiler to a patched version (v1.3.2 or later) and regenerate affected protobuf messages with the updated protobuf compiler.

18.9. Vulnerability CVE-2020-8562

The Kubernetes community has disclosed the CVE-2020-8562 vulnerability. An attacker can bypass the proxy IP limit imposed by the API server and access the Kubernetes control plane components in the private network of a cluster. This results in unauthorized access. This topic describes the impacts, affected Kubernetes versions, and fixes of this vulnerability.

The CVE-2020-8562 vulnerability is rated low and the Common Vulnerability Scoring System (CVSS) score of the vulnerability is 2.2.

Impacts

For security reasons, the Kubernetes community has forbidden user-driven connections to the internal proxy of Services, pods, nodes, and StorageClasses. When Kubernetes forbids the connections, Kubernetes performs a DNS resolution and checks whether the IPs requested by the proxy are within the link-local (169.254.0.0/16) or localhost (127.0.0.1/8) range. Then, Kubernetes performs a second DNS resolution without validating the requested IP addresses. If a non-standard DNS server returns different non-cached responses, an attacker can bypass the proxy IP limit by using a race condition and access the control plane components in the private network.

An attacker can bypass the proxy IP limit on the condition that the attacker can create or modify nodes and the attacker can access nodes by using the proxy. The attacker can also create or modify StorageClasses and access the log of kube-controller-manager.

Symptom

Check whether address not allowed exists in the audit log of the API server. If a large number of entries are found, the Kubernetes cluster may be affected by this vulnerability.

> **Note** If the API server is not affected by this vulnerability, the audit log of the API server does not contain address not allowed.

Affected versions

kube-apiserver that is installed in the following Kubernetes versions is affected by this vulnerability:

- Kubernetes ≤ v1.21.0
- Kubernetes ≤ v1.20.6

- Kubernetes ≤ v1.19.10
- Kubernetes ≤ v1.18.18

The Kubernetes community has not released the patched Kubernetes versions.

Fixes

Follow the principle of least privilege and control the permissions to create and modify nodes and StorageClasses.

18.10. Vulnerability fixed: CVE-2021-25735

A vulnerability has been found in kube-apiserver of Kubernetes that may allow node updates to bypass a validating admission webhook in some scenarios. This topic describes the kube-apiserver versions that are affected by this vulnerability. This topic also describes the impacts and fixes of this vulnerability.

The CVE-2021-25735 vulnerability is rated medium and the Common Vulnerability Scoring System (CVSS) score of the vulnerability is 3.0.

Affected versions

Only Kubernetes clusters that use a validating admission webhook are affected. The validating admission webhook relies on the original values of certain fields before node updates.

The following kube-apiserver versions are affected by this vulnerability:

- kube-apiserver v1.20.0 to v1.20.5
- kube-apiserver v1.19.0 to v1.19.9
- kube-apiserver<=v1.18.17

This vulnerability is fixed in the following kube-apiserver versions:

- kube-apiserver v1.21.0
- kube-apiserver v1.20.6
- kube-apiserver v1.19.10
- kube-apiserver v1.18.18

Impacts

> **Note**
> - If your cluster uses the default settings and no new validating admission webhook is used, your cluster is not affected by this vulnerability.
> - The default NodeRestriction admission plug-in is not affected by this vulnerability.

If a webhook that uses the validating admission mechanism for node updates exists in your cluster and the admission of the webhook relies on the original values of certain fields before node updates, an attacker can bypass the validating admission webhook and modify node properties.

Fixes

If the preceding validating admission webhook exists in your cluster, the validating admission mechanism cannot be secured before you upgrade the cluster to a patched version. Control the permissions on node updates by managing the role-based access control (RBAC) permissions. For more information, see Assign RBAC roles to RAM users.

18.11. Vulnerability fixed: CVE-2021-1056 in NVIDIA GPU drivers

NVIDIA has announced the discovery of CVE-2021-1056, a vulnerability that exploits NVIDIA GPU drivers. The default GPU drivers that are installed by Container Service for Kubernetes (ACK) are also exposed to this vulnerability. This topic describes the background information, impact, and fixes of this vulnerability.

Context

NVIDIA has announced the discovery of a vulnerability that exploits the device isolation capabilities of NVIDIA GPU drivers. This vulnerability allows an attacker to gain access to all GPU devices on a node by creating character device files in non-privileged containers that run on this node.

For more information about this vulnerability, see CVE-2021-1056.

Affected versions

The affected ACK cluster versions are:

- ACK 1.12.6-aliyun.1 (By default, the NVIDIA driver of version 410.79 is installed.)
- ACK 1.14.8-aliyun.1 (By default, the NVIDIA driver of version 410.79 is installed.)
- ACK 1.16.9-aliyun.1 (By default, the NVIDIA driver of version 418.87.01 is installed.)
- ACK 1.18.8-aliyun.1 (By default, the NVIDIA driver of version 418.87.01 is installed.)

If you selected a custom NVIDIA driver version, check whether your NVIDIA driver is affected by this vulnerability in the following figure. For more information, see the official NVIDIA website.

CVE IDs Addressed	Software Product	Operating System	Driver Branch	Affected Versions	Updated Driver Version
CVE-2021-1052 CVE-2021-1053	GeForce	Linux	R460	All versions prior to 460.32.03	460.32.03
			R450	All versions prior to 450.102.04	450.102.04
	NVIDIA RTX/Quadro, NVS	Linux	R460	All versions prior to 460.32.03	460.32.03
			R450	All versions prior to 450.102.04	450.102.04
	Tesla	Linux	R460	All versions prior to 460.32.03	460.32.03
			R450	All versions prior to 450.102.04	450.102.04
CVE-2021-1056	GeForce	Linux	R460	All versions prior to 460.32.03	460.32.03
			R450	All versions prior to 450.102.04	450.102.04
	NVIDIA RTX/Quadro, NVS	Linux	R460	All versions prior to 460.32.03	460.32.03
			R450	All versions prior to 450.102.04	450.102.04
			R390	All version prior to 390.141	390.141
	Tesla	Linux	R460	All versions prior to 460.32.03	460.32.03
			R450	All versions prior to 450.102.04	450.102.04
			R418	All versions prior to 418.181.07	418.181.07

Notice When you upgrade the NVIDIA driver for a node, the node must be restarted. This disrupts the services that are deployed on the node.

Fix

Upgrade the NVIDIA driver based on the preceding figure.

- If your NVIDIA driver belongs to the R390 branch, upgrade it to version 390.141.

- If your NVIDIA driver belongs to the R418 branch, upgrade it to version 418.181.07.
- If your NVIDIA driver belongs to the R450 branch, upgrade it to version 450.102.04.
- If your NVIDIA driver belongs to the R460 branch, upgrade it to version 460.32.03.

For more information about how to upgrade the NVIDIA driver, see Use a node pool to upgrade the NVIDIA driver for a node, Manually upgrade the NVIDIA driver of a node, and Use a node pool to create a node with a custom NVIDIA driver version.

Related information

- Use a node pool to upgrade the NVIDIA driver for a node
- Manually upgrade the NVIDIA driver of a node
- Use a node pool to create a node with a custom NVIDIA driver version

18.12. Vulnerability fixed: CVE-2020-8554

Vulnerability CVE-2020-8554 is disclosed by the Kubernetes community. Attackers can exploit this vulnerability to perform man-in-the-middle (MITM) attacks by using load balancers or external IP addresses. Attackers can create Services and then set the status field or other fields to intercept traffic from desired pods in a multi-tenant cluster.

The Common Vulnerability Scoring System (CVSS) score of this vulnerability is 3.0. The severity level is medium.

Impact scope

Clusters of all Kubernetes versions are affected. This vulnerability is caused by security defects in the design of Kubernetes. For more information about the fixing plan, see issue 97110.

If your ACK cluster is a multi-tenant cluster or runs applications that are deployed by untrusted users, your cluster is vulnerable to attacks. To check whether your cluster is under attack, check the audit log of the API server and search for events of patching to the status field of Services in your cluster. For more information, see View detailed log data.

Description

In a multi-tenant cluster, attackers can intercept traffic that is forwarded to a desired Service in the following ways:

- Create a ClusterIP type Service and set the externalIP field in the spec parameter of the Service to a desired IP address. This way, traffic that is sent to this IP address is intercepted and forwarded to the created Service.
- Set the spec.loadBalancerIP field and patch the status.loadBalancer.ingress.ip field of a LoadBalancer type Service. This way, traffic that is sent to the original IP address is intercepted and forwarded to the modified IP address.

Prevention and mitigation

The Kubernetes community has not released any solution to fix this vulnerability because great changes to the design of Kubernetes may be required. It may take a longer period of time to release the solution. To prevent attacks caused by this vulnerability, we recommend that you implement the following measures:

- Limit the use of external IP addresses:

- Use admission webhooks to authenticate and authorize the use of external IP addresses. For more information about the source code and deployment of the admission webhook that is provided by Kubernetes, see externalip-webhook.
- Use Open Policy Agent (OPA) Gatekeeper to limit the range of external IP addresses that can be used. For more information, see gatekeeper. For more information about the sample templates of ConstraintTemplate and Constraint, see externalip.

- Limit the use of IP addresses by LoadBalancer type Services:
  - Reduce the permissions to the minimum extent on patching the status field of Services.
  - You can also use admission webhooks or OPA Gatekeeper to limit the external IP addresses that can be used by LoadBalancer type Services.

18.13. Vulnerability fixed: CVE-2020-15257

The containerd community disclosed the GHSA-36xw-fx78-c5r4 vulnerability. The Common Vulnerabilities and Exposures (CVE) identifier of the vulnerability is CVE-2020-15257. If a container shares the same networking namespace with the host and the UID of the container is 0, attackers can use the containerd-shim API to control containerd-shim processes in the host and launch attacks with elevated privileges. This topic describes the impacts, causes, and preventive measures of the vulnerability.

The severity of the CVE-2020-15257 vulnerability is medium and the Common Vulnerability Scoring System (CVSS) score of the vulnerability is 5.2.

Impacts

The containerd community has fixed the vulnerability in containerd 1.3.9 and 1.4.3. All clusters of Container Service for Kubernetes (ACK) are affected by this vulnerability. To query pods that use host networking, run the following kubectl command:

```
kubectl get pods -A -o json |
  jq -c '.items[] | select(.spec.hostNetwork==true) |[.metadata.namespace, .metadata.name]'
```

Vulnerability description

containerd and containerd-shim communicate with each other by using abstract sockets. If a container is in the same networking namespace as the host with a UID of 0, attackers can access containerd-shim processes in the host as a root user. Consequently, the attackers can escape the container and use the containerd-shim API to launch attacks with elevated privileges.

> **Note**
>
> - As a runtime of Kubernetes clusters, containerd manages underlying runC containers. containerd includes a daemon and exposes gRPC service interfaces through on-premises UNIX sockets. This way, containerd can manage container lifecycle.
> - containerd-shim is a component of containerd. It is used to isolate the daemon of containerd and container processes. You can call runC interfaces through containerd-shim to manage your containers.

Preventive measures

To minimize the probability of privilege escalation attacks, you must run your applications in the following five core namespaces: Net, Mount, IPC, PID, and UTS. The more you share the namespace of the host with containers, the higher the probability of being attacked. Avoid using the host networking mode in a pod. You can restrict the use of the host networking mode in the following ways:

- Enable the Pod Security Policy (PSP) feature. You can set the hostNetwork parameter in a PSP to prevent pods in a specified namespace from using host networking. ACK allows you to configure PSPs in the ACK console. For more information, see Configure and enforce pod security policies.
- Install the gatekeeper component. For more information about the component, see gatekeeper. For more information about how to install the component, see Example of open policy agent.

If you need to use the host networking mode, we recommend that you start your containers as a non-root user by setting the securityContext parameter in the pod. Then, set the allowPrivilegeEscalation parameter to *false*. The following code block is an example:

```
hostNetwork: true   #Your containers must use host networking due to business requirements.
containers:
- name: foo
  securityContext:
    runAsUser: 12345
    allowPrivilegeEscalation: false
```

18.14. Vulnerability fix: CVE-2018-18264 for Kubernetes dashboard

Alibaba Cloud Container Service for Kubernetes has fixed dashboard vulnerability *CVE-2018-18264*. This topic describes the dashboard versions affected by the vulnerability and how to fix the vulnerability. The Kubernetes dashboards that are built in Alibaba Cloud Container Service for Kubernetes are not affected by this vulnerability because they work in the hosted form and their security settings were upgraded before the vulnerability occurred.

Background information

A security vulnerability, that is, *CVE-2018-18264*, was discovered in Kubernetes dashboards of V1.10 and earlier versions. This vulnerability allowed attackers to bypass identity authentication and read secrets within the cluster by using the dashboard logon account.

The Kubernetes dashboards that are built in Alibaba Cloud Container Service for Kubernetes are not affected by this vulnerability because they work in the hosted form and their security settings were upgraded before the vulnerability occurred.

For more information about security vulnerability *CVE-2018-18264*, see:

- Fix for unauthenticated secret access
- Security fix (CVE-2018-18264)
- v1.10.1

Conditions required to determine that a Kubernetes dashboard is vulnerable

Your dashboard is vulnerable if you have independently deployed Kubernetes dashboard V1.10 or earlier versions (V1.7.0 to V1.10.0) that supports the logon function in your Kubernetes cluster, and you have used custom certificates.

Resolution

- If you do not need a dashboard that is deployed independently, run the following command to remove the Kubernetes dashboard from your cluster:

  ```
  kubectl --namespace kube-system delete deployment kubernetes-dashboard
  ```

- If you need an independently deployed dashboard, upgrade your dashboard to V1.10.1. For more information, see dashboard.
- If you use the dashboard hosted by Alibaba Cloud Container Service for Kubernetes, you can continue to use your dashboard in the Container Service console because the dashboard was upgraded before the vulnerability occurred.

18.15. Vulnerability fixed: CVE-2018-1002105 in Kubernetes

Alibaba Cloud has fixed vulnerability *CVE-2018-1002105* for Container Service for Kubernetes (ACK). This topic describes the impact and how to fix the vulnerability.

Background

Vulnerability *CVE-2018-1002105* is discovered by the Kubernetes community. Kubernetes users can send requests to the API Server of a Kubernetes cluster through established connections and perform privilege escalation to access backend services. Alibaba Cloud has fixed this vulnerability at the earliest opportunity. You can log on to the ACK console and upgrade the Kubernetes version for your clusters.

For more information about vulnerability *CVE-2018-1002105*, see CVE-2018-1002105.

Affected versions

- Kubernetes v1.0.x-1.9.x
- Kubernetes v1.10.0-1.10.10 (fixed in v1.10.11)
- Kubernetes v1.11.0-1.11.4 (fixed in v1.11.5)
- Kubernetes v1.12.0-1.12.2 (fixed in v1.12.3)

Affected cluster configurations

- ACK clusters where an extension API Server is set up and the extension API Server can directly connect to kube-apiserver.
- ACK clusters that expose the pod exec/attach/portforward interface to users. Attackers can exploit the vulnerability to gain full permissions on the kebelet API.

ACK cluster configurations

- By default, role based access control (RBAC) is enabled for API Servers of ACK clusters. Anonymous users that are not authorized by Alibaba Cloud accounts are prohibited to call certain APIs. In addition, `anonymous-auth=false` is added to the startup parameters of kubelet to control external access.

- Resource Access Management (RAM) users of multi-tenant ACK clusters can perform unauthorized access through the pod exec/attach/portforward interface. You do not need to be concerned if your clusters have only administrator accounts.
- By default, RAM users that are not authorized by Alibaba Cloud accounts cannot access the Aggregation API.

Fixes

Log on to the ACK console and upgrade your clusters. For more information, see Upgrade the Kubernetes version of an ACK cluster.

- If your clusters use Kubernetes 1.11.2, upgrade to Kubernetes 1.11.5.
- If your clusters use Kubernetes 1.10.4, upgrade to Kubernetes 1.10.11 or 1.11.5.
- If your clusters use Kubernetes 1.9 or earlier, upgrade to Kubernetes 1.10.11 or 1.11.5. When you upgrade Kubernetes 1.9 to 1.10 or 1.11, you must first upgrade FlexVolume in the ACK cluster if cloud disks are mounted to your cluster.

> **Note** In the ACK console, select the cluster for which you want to upgrade FlexVolume. In the navigation pane, choose **More > Upgrade System Component**. On the **Upgrade System Component** page, select **flexvolume** and click **Upgrade**.

The security of serverless Kubernetes (ASK) clusters has been reinforced before this vulnerability is introduced. Therefore, ASK clusters are not affected.

18.16. Announcement about fixing the runC vulnerability CVE-2019-5736

Alibaba Cloud has fixed the runC vulnerability CVE-2019-5736 in Container Service for Kubernetes (ACK). This topic describes the impacts and how to fix the vulnerability in earlier versions.

Background information

A vulnerability is found in the runC runtime that is used for Docker, containerd, or other runC-based containers. An attacker can overwrite the host runC binary and consequently obtain host root access by abusing the ability to execute a command as root within a specific container. Such a container can be attached with docker exec and therefore the attacker has the write permissions on the container.

For more information, see CVE-2019-5736.

Impacts

- ACK:

  Docker Swarm clusters and Kubernetes clusters that use Docker versions earlier than 18.09.2 are affected.

- User-defined Docker/Kubernetes runtimes:

  Runtimes that use Docker versions earlier than 18.09.2 or runC versions earlier than 1.0-rc6 are affected.

Fixes

Alibaba Cloud has fixed this vulnerability in Docker versions used by ACK clusters 1.11 or 1.12. For other clusters, you can use the following methods to fix the vulnerability:

- Upgrade Docker. Upgrade the version of Docker to 18.09.2 or later. This may cause container disconnection and business disruption.
- Upgrade runC only (for Docker 17.06). To avoid business disruption caused by Docker engine upgrades, take the following steps to upgrade the runC binary on each cluster node:
  i. Run the following command to locate docker-runc. In most cases, docker-runc is located in the */usr/bin/docker-runc* path.

     ```
     which docker-runc
     ```

  ii. Run the following command to back up the current runC:

     ```
     mv /usr/bin/docker-runc /usr/bin/docker-runc.orig.$(date -Iseconds)
     ```

  iii. Run the following command to download the fixed runC:

     ```
     curl -o /usr/bin/docker-runc -sSL https://acs-public-mirror.oss-cn-hangzhou.aliyuncs.com/runc/docker-runc-17.06-amd64
     ```

  iv. Run the following command to make docker-runc executable:

     ```
     chmod +x /usr/bin/docker-runc
     ```

  v. Run the following command to test whether runC works as expected:

     ```
     docker-runc -v
     # runc version 1.0.0-rc3
     # commit: fc48a25bde6fb041aae0977111ad8141ff396438
     # spec: 1.0.0-rc5
     docker run -it --rm ubuntu echo OK
     ```

  vi. (Optional)For GPU nodes in an ACK cluster, you must take the following steps to install nvidia-runtime:
     a. Run the following command to locate nvidia-container-runtime. In most cases, nvidia-container-runtime is located in the */usr/bin/nvidia-container-runtime* path.

        ```
        which nvidia-container-runtime
        ```

     b. Run the following command to back up the current nvidia-container-runtime:

        ```
        mv /usr/bin/nvidia-container-runtime /usr/bin/nvidia-container-runtime.orig.$(date -Iseconds)
        ```

     c. Run the following command to download the fixed nvidia-container-runtime:

        ```
        curl -o /usr/bin/nvidia-container-runtime -sSL https://acs-public-mirror.oss-cn-hangzhou.aliyuncs.com/runc/nvidia-container-runtime-17.06-amd64
        ```

     d. Run the following command to make nvidia-container-runtime executable:

        ```
        chmod +x /usr/bin/nvidia-container-runtime
        ```

e. Run the following command to test whether nvidia-container-runtime works as expected:

```
nvidia-container-runtime -v
#  runc version 1.0.0-rc3
#  commit: fc48a25bde6fb041aae0977111ad8141ff396438-dirty
#  spec: 1.0.0-rc5
docker run -it --rm -e NVIDIA_VISIBLE_DEVICES=all ubuntu nvidia-smi -L
#  GPU 0: Tesla P100-PCIE-16GB (UUID: GPU-122e199c-9aa6-5063-0fd2-da009017e6dc)
```

> **Note** In this topic, the test is run on nodes that use GPU P100. The test method varies based on the GPU model.

18.17. Vulnerability fix: CVE-2019-11246 related to kubectl cp

Kubernetes has announced the vulnerability CVE-2019-11246 related to the kubectl cp. This vulnerability may allow attackers to exploit the kubectl cp command and write malicious files from the TAR package of a container to a path on the host of the container by using path traversal. This process is limited by only local system permissions.

Background information

The effects of this vulnerability are similar to those of the CVE-2019-1002101 vulnerability. For more information about the CVE-2019-1002101 vulnerability, see CVE-2019-1002101: kubectl fix potential directory traversal #75037.

The kubectl cp command is used to copy files between containers and hosts. When you copy a file from a container to your host by running the kubectl cp command, Kubernetes performs the following steps: creates a TAR file in the container, sends the package file to your host, and then decompresses the package file on your host.

If an attacker has permissions to run the kubectl cp command, the attacker can send a malicious TAR file to perform a path traversal attack on your host.

For more information about the Privileges Required (PR) of the CVE-2019-11246 vulnerability, see CVE-2019-11246: Clean links handling in cp's tar code#76788.

Affected Kubernetes versions

- kubectl V1.11.x and earlier
- kubectl V1.12.1 to V1.12.8 (fixed in V1.12.9)
- kubectl V1.13.1 to v1.13.5 (fixed in V1.13.6)
- kubectl V1.14.1 (fixed in V1.14.2)

> **Note** You can run the `kubectl version --client` command to check your kubectl version.

Solution

Upgrade the kubectl and confirm the kubectl version. For more information, see Install and set up kubectl.

- If your kubectl version is V1.12.x, upgrade it to V1.12.9.
- If your kubectl version is V1.13.x, upgrade it to V1.13.6.
- If your kubectl version is V1.14.x, upgrade it to V1.14.2.
- If your kubectl version is V1.11 or earlier, upgrade it to V1.12.9, V1.13.6, or V1.14.2.

18.18. Announcement about fixing the CVE-2019-11249 vulnerability

The Kubernetes community disclosed the vulnerability CVE-2019-11249 that is related to the kubectl cp command. Attackers can exploit this vulnerability to write malicious files into any paths other than the destination paths on your host through directory traversal. The malicious files are saved in TAR packages in containers. This process is restricted only by system permissions of the current user.

Background information

The kubectl cp command is used to copy files between containers and hosts. When you copy a file from a container to your host by running the kubectl cp command, Kubernetes performs the following three steps: runs the **tar** command to create a TAR package in the container, sends the package to your host, and decompresses the package on your host.

If an attacker has permissions to run the kubectl cp command, they can send a malicious TAR package through directory traversal.

To fix this vulnerability, the kubectl cp command is required to perform a more rigorous verification on the destination paths of all files during TAR package decompression. The command must disallow copying decompressed files to any paths other than the destination paths. This prevents malicious attacks during `TAR package decompression`.

For more information, see Kubernetes announcements.

For more information about the pull requests for fixing this vulnerability, see CVE-2019-11249.

Impacts

You can run the `kubectl version --client` command to check your kubectl version.

The following kubectl versions are affected:

- kubectl 1.0.x-1.12.x
- Kubectl 1.13.0 to 1.13.8 (fixed in v1.13.9)
- Kubectl 1.14.0 to 1.14.4 (fixed in v1.14.5)
- Kubectl 1.15.0 to 1.15.1 (fixed in v1.15.2)

Fixes

You can upgrade the kubectl version to fix this vulnerability. For more information, see install and set up kubectl. Check the kubectl version after kubectl is installed.

- If the kubectl version is 1.13.x, upgrade it to 1.13.9.
- If the kubectl version is 1.14.x, upgrade it to 1.14.5.
- If the kubectl version is 1.15.x, upgrade it to 1.15.2.
- If your kubectl version is 1.12.x or an earlier version, upgrade it to 1.13.9, 1.14.5, or 1.15.2.

18.19. Announcement about fixing the Kubernetes vulnerability CVE-2019-11253

Alibaba Cloud has fixed the Kubernetes vulnerability *CVE-2019-11253* in Container Service for Kubernetes (ACK). This topic describes the impacts and how to fix this vulnerability in earlier versions.

Background information

The Kubernetes vulnerability *CVE-2019-11253* was disclosed by the Kubernetes community. Kubernetes users can send POST requests with forged YAML files to launch Denial-of-Service (DoS) attacks against Kubernetes clusters. Alibaba Cloud has fixed this vulnerability in ACK at the earliest opportunity. Log on to the ACK console to upgrade your ACK clusters.

For more information about the Kubernetes vulnerability *CVE-2019-11253*, see CVE-2019-11253.

Affected versions

- Kubernetes v1.0.x~1.12.x
- Kubernetes v1.13.0 to 1.13.11 (fixed in 1.13.12)
- Kubernetes v1.14.0 to 1.14.7 (fixed in 1.14.8)
- Kubernetes v1.15.0 to 1.15.4 (fixed in 1.15.5)
- Kubernetes v1.16.0 to 1.16.1 (fixed in 1.16.2)

Fixes

Log on to the the ACK console to upgrade your ACK clusters to 1.14.8. For more information about how to upgrade an ACK cluster and the considerations to which you must pay attention, see Upgrade the Kubernetes version of an ACK cluster.

If you cannot immediately upgrade your ACK clusters, perform the following operations to reduce the risks caused by this vulnerability and perform an upgrade at a later time.

- You can follow the principle of least privilege (POLP) and grant Resource Access Management (RAM) users the minimum permissions on the ACK cluster that they need to access. Do not grant the RAM users the permissions to create or modify ACK clusters. For more information, see Authorization overview.
- You can also use your Alibaba Cloud account to revoke KubeConfig credentials from users that may be exposed to the risk of disclosing their KubeConfig credentials.

18.20. Vulnerability fixed: CVE-2019-16276 in Golang

Alibaba Cloud has fixed vulnerability *CVE-2019-16276* of Golang for Container Service for Kubernetes (ACK). This topic describes the impact and how to fix the vulnerability.

Background

Vulnerability *CVE-2019-16276* is discovered by Golang. Kubernetes users can write a request header in a specific format to bypass the filter conditions in the authentication proxy and send authenticated requests to the backend API server on behalf of other users or groups. Golang has fixed this vulnerability. We recommend that you upgrade your Golang version.

For more information about vulnerability *CVE-2019-16276*, see CVE-2019-16276.

Affected versions

Clusters that use an authenticating proxy for authentication and the authenticating proxy server is written in Go.

Fixes

Upgrade Golang. For more information, see Install Go. You can download Golang 1.12.10 or 1.13.1 to recompile and deploy the authenticating proxy server. After Go is installed, you can run the go version command to check its version.

18.21. Vulnerability fixed: CVE-2019-1002101 in kubectl cp

The kubectl cp command allows users to copy files between containers and user machines. Attackers can implant a malicious TAR package that has a header with a symbolic link to images or running containers. When the kubectl cp command decompresses the TAR package, it can both modify and follow the files in the symbolic link. This vulnerability is fixed in kubectl 1.11.9, 1.12.7, 1.13.5, and 1.14.0. For more information, see Install and set up kubectl. You can use kubectl of the preceding versions to avoid this vulnerability.

18.22. Vulnerability fixed: CVE-2020-8555 in kube-controller-manager

Alibaba Cloud has fixed vulnerability *CVE-2020-8555* in kube-controller-manager for Container Service for Kubernetes (ACK). Vulnerability CVE-2020-8555 is a Server Side Request Forgery (SSRF) vulnerability of kube-controller-manager. Authorized users can forge requests of server-side applications to obtain arbitrary information from unprotected endpoints in the host network of master nodes. This topic describes the impacts, solution, and prevention measures for this vulnerability.

The Common Vulnerability Scoring System (CVSS) score of this vulnerability is 3.0. For more information, see CVSS:3.0/AV:N/AC:H/PR:L/UI:N/S:C/C:H/I:N/A:N. The risk level is medium.

Impact scope

Prerequisites of attacks:

- A local port of kube-apiserver that allows unauthorized access is open.
- Unprotected services are open to the host network of master nodes.
- Malicious users have the permissions to create pods or write StorageClass objects in a Kubernetes cluster.

Affected versions:

- kube-controller-manager v1.16.0~v1.16.8

- kube-controller-manager<v1.15.11

The affected volume types are GlusterFS, Quobyte, StorageFS, and ScaleIO.

Fixes

An authorized user may exploit this vulnerability to create a pod that is mounted with a vulnerable volume (GlusterFS, Quobyte, StorageFS, or ScaleIO) or write a StorageClass object in a Kubernetes cluster. The user can send GET or POST requests to Services that are open to the host network of master nodes. This way, the user can probe and attack the host network without authorization. For example, an attacker may use the unprotected port 8080 of kube-apiserver to obtain Kubernetes Secrets.

By default, the unprotected port 8080 is closed for an ACK cluster. All Resource Access Management (RAM) users must be granted role-based access control (RBAC) permissions to perform the preceding operations. By default, all RAM users except the user who creates the cluster are unauthorized to create pods or write StorageClass objects. To prevent data leaks from unprotected Services in the host network of master nodes, implement the measures described in pr.k8s.io/89794. A new version of kube-controller-manager is also provided to fix this vulnerability.

Prevention and mitigation

> **Note** This vulnerability is fixed in the latest ACK version V1.16.9-aliyun.1. We recommend that you upgrade your clusters to V1.16.9-aliyun.1.

If cluster upgrades are not allowed by your business, we recommend that you implement the following prevention measures:

- Do not open the unprotected port 8080 on kube-apiserver. By default, this port is closed in ACK clusters.
- Check whether request authentication is enabled for Services that are open to the host network of master nodes. Find and disable the unprotected Services that may cause data leaks.
- Do not authorize untrusted users to create pods or write StorageClass objects.

18.23. Vulnerability fix: CVE-2020-8558

To enable nodes in a Kubernetes cluster to access services that listen on 127.0.0.1, the Linux kernel parameter `net.ipv4.conf.all.route_localnet` is set to 1 for kube-proxy in both iptables and ipvs modes. This causes a security vulnerability. An attacker may log on to a container connected to the network of a vulnerable host or an adjacent host of the vulnerable host in the same local area network (LAN). Then, the attacker attempts to access TCP and UDP services that are deployed on the vulnerable host and listen on 127.0.0.1. If a TCP or UDP service does not require authentication, the attacker may access the service. This causes data breaches.

Affected versions

This vulnerability affects kube-proxy of the following versions:

- kube-proxy v1.18.0~v1.18.3
- kube-proxy v1.17.0~v1.17.6
- kube-proxy V1.16.10 and earlier

By default, in a Kubernetes cluster, users must be authenticated to access services that listen on 127.0.0.1, and kube-apiserver disables unprotected ports. The kubelet service opens read-only port 10255 that is bound to 0.0.0.0 but does not require user authentication. Attackers can access the kubelet service from containers that use the host network or privileged containers through port 10255 even if the vulnerability is fixed. Therefore, this vulnerability has little impact on the ACK cluster.

Impacts

Assume that an attacker has the permission to configure a host network or log on to a container that has CAP_NET_RAW enabled. The attacker may exploit this vulnerability to obtain the socket addresses of services that listen on 127.0.0.1. If services on a node listen on 127.0.0.1 and the services do not require user authentication, an attacker may exploit this vulnerability to access the services. For more information, see Issue.

Common Vulnerability Scoring System (CVSS) rating:

- If kube-apiserver opens the unprotected port, an attacker may exploit this vulnerability to access the kube-apiserver information. In this case, this vulnerability has high severity and is scored 8.8. The default unprotected port for kube-apiserver is 8080.
- If kube-apiserver disables the unprotected port, this vulnerability has medium severity and is scored 5.4.

Attackers may log on to the following objects to launch attacks:

- Hosts that are connected to the same vSwitch as the vulnerable host
- Containers that run on the vulnerable host

Preventative measures

We recommend that you take the following preventative measures:

- Do not open the unprotected port of kube-apiserver. The default unprotected port is 8080. By default, this port is disabled.
- Run the following command to configure an iptables rule for each node in the cluster. This rule blocks traffic that is sent from other nodes to 127.0.0.1.

```
 iptables -I INPUT --dst 127.0.0.0/8 ! --src 127.0.0.0/8 -m conntrack ! --ctstate RELATED
,ESTABLISHED,DNAT -j DROP
```

- Control user permissions to log on to cluster nodes. For example, you can invalidate kubeconfig files that may have been leaked to malicious users.
- Do not enable CAP_NET_RAW for containers. If this feature is enabled, run the following command to disable it:

```
securityContext:
      capabilities:
        drop: ["NET_RAW"]
```

- Use PodSecurityPolicy to control the deployment of privileged containers and containers that use the network of the host. You can also disable CAP_NET_RAW for containers by configuring the requiredDropCapabilities parameter in the policy.

18.24. Vulnerability fixed: CVE-2020-13401

To implement dynamic address assignment in IPv6, Kubernetes supports both Dynamic Host Configuration Protocol (DHCP) and Router Advertisement. This causes the vulnerability CVE-2020-13401. Router Advertisement allows routers to periodically send messages to nodes. The messages provide information about the network status such as routing table entries. The client uses Neighbor Discovery Protocol (NDP) to configure the network based on the information. This topic describes the impacts of the vulnerability CVE-2020-13401.

> **Notice** IPv6 is disabled for ACK clusters. Therefore, this vulnerability does not affect your clusters and no further action is required.

Affected scenarios

This vulnerability affects a node if IPv6 is enabled and the Container Network Interface (CNI) plug-in version is earlier than v0.8.6.

Impacts

A malicious attacker may exploit this vulnerability to tamper with the IPv6 routing tables of hosts or containers. This enables man-in-the-middle attacks. If the DNS server returns both A (IPv4) and AAAA (IPv6) records, HTTP libraries may use the IPv6 record for connections even if no IPv6 traffic exists in the cluster. If the connection fails, the IPv4 record is used.

The following kubelet versions contain the kubernetes-cni service. Therefore, these versions are affected by the vulnerability.

- kubelet v1.18.0~v1.18.3
- kubelet v1.17.0~v1.17.6
- kubelet<v1.16.11

> **Notice** IPv6 is disabled for ACK clusters. Therefore, this vulnerability does not affect your clusters and no further action is required.

18.25. Vulnerability fixed: CVE-2020-8559 for kube-apiserver

Kubernetes revealed a security vulnerability (CVE-2020-8559) in kube-apiserver. Attackers can intercept certain upgrade requests sent to the kube-apiserver on a node. The attackers then send a redirect response to clients that use the same credentials carried in the intercepted requests. As a result, subsequent requests from these clients are redirected to another node. This leads to privilege escalation from a compromised node. This topic describes the impacts of CVE-2020-8559, the affected kube-apiserver versions, and the suggested solutions for prevention and mitigation.

Affected versions

CVE-2020-8559 is discovered in the following kube-apiserver versions:

- kube-apiserver 1.18.0 to 1.18.5 (fixed in kube-apiserver v1.18.6)
- kube-apiserver 1.18.0 to 1.18.5 (fixed in kube-apiserver v1.18.6)kube-apiserver 1.17.0 to 1.17.8 (fixed in kube-apiserver v1.17.9)
- kube-apiserver 1.16.0 to 1.16.12 (fixed in kube-apiserver v1.16.13)

CVE-2020-8559 impacts services in the following scenarios:

- A Kubernetes cluster is used by multiple tenants and nodes that belong to different tenants are isolated.
- Multiple Kubernetes clusters share the same certificate authority and authentication credentials.

Impacts

- The apiserver proxy built into the kube-apiserver can redirect upgrade requests back to clients. Attackers can intercept certain upgrade requests sent to the kube-apiserver on a node. The attackers then send a redirect response to clients that use the same credentials carried in the intercepted requests. As a result, subsequent requests from these clients are redirected to another node. This leads to privilege escalation from a compromised node. The Common Vulnerability Scoring System (CVSS) score of this vulnerability is 6.4. This indicates that CVE-2020-8559 is a medium-severity vulnerability.
- If multiple clusters share the same certificate authority certificates and authentication credentials, attackers can exploit this vulnerability to attack other clusters. In this case, this vulnerability is a high-severity vulnerability.

Prevention and mitigation

To defend against cross-cluster attacks, clusters of Alibaba Cloud Container Service for Kubernetes (ACK) use separate certificate authorities. Credentials are completely isolated among clusters.

To defend against cross-node attacks inside a cluster, we recommend that you perform the following steps:

- Enable audit log for kuber-apiserver. If a response code between 300 and 399 is returned to any of the following requests, it may be evidence of an attack.
  - pods/exec
  - pods/attach
  - pods/portforward
  - Any proxy resources, such as pods/proxy and services/proxy
- Revoke the kubeconfig credentials that may be disclosed, and remove unnecessary Role-based access control (RBAC) permissions from roles that are bound to the following resources: pods/exec, pods/attach, pods/portforward, and all proxy resources.

18.26. Vulnerability fixed: CVE-2020-8557 for kubelet

The eviction manager of kubelet does not track the ephemeral storage consumed by the */etc/hosts* file that is mounted to pods on a node. In this case, a malicious pod that is mounted with the */etc/hosts* file may exhaust the storage resource of the node by writing data into this file. As a result, the result stops responding to requests. This topic describes the impacts of CVE-2020-8557, the affected kubelet versions, and the suggested solutions for prevention and mitigation.

Impact scope

CVE-2020-8557 is discovered in the following kubelet versions:

- kubelet v1.18.0~v1.18.5

- kubelet v1.17.0~v1.17.9
- kubelet<v1.16.13

Impacts

The eviction manager of kubelet does not track the ephemeral storage consumed by the */etc/hosts* file that is mounted to pods on a node. In this case, a malicious pod that is mounted with the */etc/hosts* file may exhaust the storage resource of the node by writing data into this file. As a result, the node stops responding to requests. The Common Vulnerability Scoring System (CVSS) score of this vulnerability is 5.5. This indicates that CVE-2020-8557 is a medium-severity vulnerability.

A pod with the following configurations can write data into the */etc/hosts* file:

- A pod with the CAP_DAC_OVERRIDE Linux capability (authorized by default).
- A pod that is launched by a root user (with UID 0) or a pod where the allowPrivilegeEscalation field in the security context settings is set to true (the default value is true).

Prevention and mitigation

We recommend that you perform the following steps:

- Remove the CAP_DAC_OVERRIDE Linux capability from pods by using pod security policies or other admission control mechanisms. For more information, see Use pod security policies.
- Forbid root users to launch pods. You can perform this task by using pod security policies or other admission control mechanisms, or by setting allowPrivilegeEscalation to false. For more information, see Use pod security policies.
- Monitor the size of the */etc/hosts* file. For example, you can enable tamper protection in the Security Center console. For more information, see Enable the web tamper proofing feature.
- You can run the following command on a node to find pods with an abnormally-sized *etc-hosts* file:

```
find /var/lib/kubelet/pods/*/etc-hosts -size +1M
```

18.27. Vulnerability updates: CVE-2020-14386

A Linux kernel vulnerability was recently discovered. The CVE ID of the vulnerability is CVE-2020-14386. This vulnerability results from a bug in the packet socket facility in the Linux kernel. Attackers can exploit the vulnerability to perform an out-of-bounds write of up to 10 bytes, as stated by the vulnerability discoverer. The vulnerability may lead to unauthorized privilege escalation and container escapes, exhaust the memory of cluster nodes, and affect the applications that run on the nodes.

For more information about this vulnerability, see CVE-2020-14386.

Affected operating systems and kernel versions

The vulnerability affects multiple Linux distributions that have kernel versions later than 4.6. The affected Linux distributions include:

- Ubuntu Bionic (18.04) and later
- Debian 9
- Debian 10
- CentOS 8 and RHEL 8

The following list describes the impact of the vulnerability on an Alibaba Cloud Container Service for Kubernetes (ACK) cluster:

- If your cluster nodes run the Alibaba Cloud Linux 2 operating system that has a kernel version of 4.19.91-19.1.al7, your cluster is affected by the vulnerability.
- If your cluster nodes run the CentOS operating system that has a kernel version of 3.10.0-1062, which is earlier than the affected kernel version, your cluster is not affected by the vulnerability.

Impacts

CVE-2020-14386 is a memory corruption vulnerability in the af_packet kernel module. The CAP_NET_RAW capability is required to exploit the vulnerability. Non-root users in Linux do not have this capability. However, in a Linux OS that has a kernel version later than 4.6, a non-root user can create a user namespace that has the CAP_NET_RAW capability. By default, Kubernetes and Docker containers have the CAP_NET_RAW capability. Therefore, attackers may exploit the CVE-2020-14386 vulnerability on nodes of ACK clusters if the Linux kernel version of nodes in the cluster is later than 4.6. Attackers can exploit the vulnerability to perform an out-of-bounds write of up to 10 bytes. This may lead to unauthorized privilege escalation and container escapes. For the release notes of Alibaba Cloud Linux 2 images, see Vulnerability announcement | Linux kernel vulnerability (CVE-2020-14386). For more information about how to fix the Alibaba Cloud Linux 2.1903 vulnerability, see Security Advisories.

Fixes

- Disable CAP_NET_RAW in the securityContext field in the configuration file of the containerized application.

```
spec:
  containers:
    -name: target-container
     ...
    securityContext:
      capabilities:
        drop:
          -NET_RAW
```

The CAP_NET_RAW capability is required to exploit this vulnerability. The CAP_NET_RAW capability is not required by most container services. You can configure a pod security policy (PSP) to ensure that the CAP_NET_RAW capability is disabled for a pod. The following content is a PSP template:

> **Note** You can enable the PSP feature, create a PSP, and attach the PSP to a pod in the ACK console. For more information, see Configure and enforce pod security policies.

```
apiversion: policy/v1beta1
kind: PodSecurityPolicy
metadata:
  name: no-cap-net-raw
spec:
  requiredDropCapabilities:
    -NET_RAW
     ...
```

- Install the gatekeeper add-on and the official constraint template for your cluster on the Components page. For more information, see gatekeeper and . Then, create the following constraint

to disable the CAP_NET_RAW capability of the container:

```
# Dropping CAP_NET_RAW with Gatekeeper
 # (requires the K8sPSPCapabilities template)
 apiversion: constraints.gatekeeper.sh/v1beta1
 kind:  K8sPSPCapabilities
 metadata:
   name: no-cap-net-raw
 spec:
   match:
     kinds:
       - apiGroups: [""]
       kinds: ["Pod"]
     namespaces:
       #List of namespaces to enforce this constraint on
       - default
     # If running gatekeeper >= v3.1.0-beta.5,
     # you can exclude namespaces rather than including them above.
     excludedNamespaces:
       - kube-system
   parameters:
     requiredDropCapabilities:
       - "NET_RAW"
```

- A bulletin about this vulnerability and an upgrade guide are released for Alibaba Cloud Linux 2. If your cluster nodes run Alibaba Cloud Linux 2, perform the following steps to upgrade the kernel version:
  i. Run the `yum -y install kernel-4.19.91-21.2.al7` command to upgrade the kernel to a version that has this vulnerability fixed. Alternatively, run the `yum -y update kernel` command to upgrade the kernel to the latest version.
  ii. Restart the system for the upgrade to take effect. If a node has a running service that is not deployed on other nodes, drain and restart the node during off-peak hours.
  iii. For more information about how to fix the Alibaba Cloud Linux 2.1903 vulnerability, see Security Advisories.

18.28. Vulnerability fix: CVE-2020-8564, CVE-2020-8565, and CVE-2020-8566

The Kubernetes community discloses three vulnerabilities that involve multiple components, such as kube-apiserver, kube-controller-manager, and kubectl. If the log level of a system component is equal to or higher than a specific level, sensitive data may be leaked.

Vulnerabilities CVE-2020-8564, CVE-2020-8565, and CVE-2020-8566 are rated as Medium.

Affected versions

- For open source Kubernetes clusters:

  These vulnerabilities exist in all Kubernetes versions earlier than V1.19.2. You can find solutions in the following list:

  - For information about how to fix CVE-2020-8564, see 94712.
  - For information about how to fix CVE-2020-8565, see 95316.

- For information about how to fix CVE-2020-8566, see 95245.

- For Container Service for Kubernetes (ACK) clusters:
  - These vulnerabilities do not affect standard managed or serverless clusters.
  - These vulnerabilities do not affect standard dedicated clusters if you retain the default log levels of system components.
  - For a standard dedicated cluster, if the log level of a system component is equal to or higher than the level that causes security risks, these vulnerabilities affect your cluster. For more information, see Preventative measures.

Impacts

- CVE-2020-8564

  Assume that you have stored secrets that are used to pull images from a private repository in a Docker config file. If the log level of kubelet is four or higher, an attacker may exploit the vulnerability to obtain the secrets from the logs of kubelet.

- CVE-2020-8565

  If the log level of kube-apiserver is nine or higher, an attacker may exploit the vulnerability to obtain the `bearer token` or `basic auth token`.

- CVE-2020-8566

  Assume that you have used Ceph RADOS Block Device (RBD) to store application data. If the log level of kube-controller-manager is four or higher, an attacker may exploit the vulnerability to obtain the Ceph RBD admin secret from the logs of kube-controller-manager.

Preventative measures

For a standard dedicated cluster, if the log level of a system component is equal to or higher than the level that causes security risks, we recommend that you take the following preventive measures:

- Make sure that untrusted users do not have read permissions on the logs of system components.
- If an attacker may have permissions to read logs, we recommend that you rotate the secrets and modify the log levels.
- Clients such as kubectl may also cause data leakage. If you use kubectl, make sure that the log level of kubectl is lower than the level that causes security risks or the read permissions on log data are granted to trusted users.