Amazon EKS Kubernetes versions

PDF (eks-ug.pdf#kubernetes-versions)
Kindle (https://www.amazon.com/dp/B07GQDSDPZ)
RSS (doc-history.rss)

The Kubernetes project is rapidly evolving with new features, design updates, and bug fixes. The community releases new Kubernetes minor versions, such as 1.19, as generally available approximately every three months, and each minor version is supported for approximately twelve months after it is first released.

Available Amazon EKS Kubernetes versions

The following Kubernetes versions are currently available for new clusters in Amazon EKS:

- 1.19.6
- 1.18.9
- 1.17.12
- 1.16.15
- 1.15.12

Unless your application requires a specific version of Kubernetes, we recommend that you choose the latest available Kubernetes version supported by Amazon EKS for your clusters. As new Kubernetes versions become available in Amazon EKS, we recommend that you proactively update your clusters to use the latest available version. For more information, see Updating a cluster (./update-cluster.html) . For more information, see Amazon EKS Kubernetes release calendar (#kubernetes-release-calendar) and Amazon EKS version support and FAQ (#version-deprecation) .

Kubernetes 1.19

Kubernetes 1.19 is now available in Amazon EKS. For more information about Kubernetes 1.19, see the official release announcement (https://kubernetes.io/blog/2020/08/26/kubernetes-release-1.19-accentuate-the-paw-sitive/) .

Important

- Starting with 1.19, Amazon EKS no longer adds the `kubernetes.io/cluster/<cluster-name>` tag to subnets passed in during cluster creation. This subnet tag is only required if you want to influence where the Kubernetes service controller or AWS Load Balancer Controller places Elastic Load Balancers. For more information about the requirements of subnets passed to Amazon EKS during cluster creation, see updates to Cluster VPC considerations (./network_reqs.html) .
  - Subnet tags are not modified on existing clusters updated to 1.19.
  - The AWS Load Balancer Controller version `v2.1.1` and earlier required the `<cluster-name>` subnet tag. In version `v2.1.2` and later, you can specify the tag to refine subnet discovery, but it's not required. For more information about the AWS Load Balancer Controller, see AWS Load Balancer Controller (./aws-load-balancer-controller.html) . For more information about subnet tagging when using a load balancer, see Application load balancing on Amazon EKS (./alb-ingress.html) and Network load balancing on Amazon EKS (./load-balancing.html) .

- You're no longer required to provide a security context for non-root containers that need to access the web identity token file for use with IAM roles for service accounts. For more information, see IAM roles for service accounts (./iam-roles-for-service-accounts.html) andproposal for file permission handling in projected service account volume (https://github.com/kubernetes/enhancements/pull/1598) on GitHub.

- The pod identity webhook has been updated to address the missing startup probes (https://github.com/aws/amazon-eks-pod-identity-webhook/issues/84) GitHub issue. The webhook also now supports an annotation to control token expiration. For more information, see the GitHub pull request (https://github.com/aws/amazon-eks-pod-identity-webhook/pull/97) .

- CoreDNS version 1.8.0 is the recommended version for Amazon EKS 1.19 clusters. This version is installed by default in new Amazon EKS 1.19 clusters. For more information, see Installing or upgrading CoreDNS (./coredns.html) .

- Amazon EKS optimized Amazon Linux 2 AMIs include the Linux kernel version 5.4 for Kubernetes version 1.19. For more information, see Amazon EKS optimized Amazon Linux AMI (./eks-linux-ami-versions.html#eks-al2-ami-versions) .

- The `CertificateSigningRequest API` has been promoted to stable `certificates.k8s.io/v1` with the following changes:

  - `spec.signerName` is now required. You can't create requests for `kubernetes.io/legacy-unknown` with the `certificates.k8s.io/v1` API.

  - You can continue to create CSRs with the `kubernetes.io/legacy-unknown` signer name with the `certificates.k8s.io/v1beta1` API.

  - You can continue to request that a CSR to is signed for a non-node server cert, webhooks, for example, with the `certificates.k8s.io/v1beta1` API. These CSRs aren't auto-approved.

  - To approve certificates, a privileged user requires `kubectl` 1.18.8 or later.

  For more details on the certificate v1 API, see Certificate Signing Requests (https://kubernetes.io/docs/reference/access-authn-authz/certificate-signing-requests/) in the Kubernetes documentation.

The following Amazon EKS Kubernetes resources are critical for the Kubernetes control plane to work. We recommend that you don't delete or edit them.

Permission	Kind	Namespace	Reason
`eks:certificate-controller`	`Rolebinding`	`kube-system`	Impacts signer and approver functionality in the control plane.
`eks:certificate-controller`	`Role`	`kube-system`	Impacts signer and approver functionality in the control plane.
`eks:certificate-controller`	`ClusterRolebinding`	All	Impacts kubelet's ability to request server certificates which affects certain cluster functionality like `kubectl exec` and `kubectl logs`.

The following Kubernetes features are now supported in Kubernetes 1.19 Amazon EKS clusters:

- The `ExtendedResourceToleration` admission controller is enabled. This admission controller automatically adds tolerations for taints to pods requesting extended resources, such as GPUs, so you don't have to manually add the tolerations. For more information, see ExtendedResourceToleration (https://kubernetes.io/docs/reference/access-authn-authz/admission-controllers/#extendedresourcetoleration) in the Kubernetes documentation.

- Elastic Load Balancers (CLB and NLB) provisioned by the in-tree Kubernetes service controller support filtering the nodes included as instance targets. This can help prevent reaching target group limits in large clusters. For more information, see the related GitHub issue (https://github.com/kubernetes/kubernetes/pull/90943) and the `service.beta.kubernetes.io/aws-load-balancer-target-node-labels` annotation under Other ELB annotations (https://kubernetes.io/docs/concepts/services-networking/service/#other-elb-annotations) in the Kubernetes documentation.

- Pod Topology Spread has reached stable status. You can use topology spread constraints to control how pods are spread across your cluster among failure-domains such as regions, zones, nodes, and other user-defined topology domains. This can help to achieve high availability, as well as efficient resource utilization. For more information, see Pod Topology Spread Constraints (https://kubernetes.io/docs/concepts/workloads/pods/pod-topology-spread-constraints/) in the Kubernetes documentation.

- The Ingress API has reached general availability. For more information, see Ingress (https://kubernetes.io/docs/concepts/services-networking/ingress/) in the Kubernetes documentation.

- EndpointSlices are enabled by default. EndpointSlices are a new API that provides a more scalable and extensible alternative to the Endpoints API for tracking IP addresses, ports, readiness, and topology information for Pods backing a Service. For more information, see Scaling Kubernetes Networking With EndpointSlices (https://kubernetes.io/blog/2020/09/02/scaling-kubernetes-networking-with-endpointslices/) in the Kubernetes blog.

- Secret and ConfigMap volumes can now be marked as immutable, which significantly reduces load on the API server if there are many Secret and ConfigMap volumes in the cluster. For more information, see ConfigMap (https://kubernetes.io/docs/concepts/configuration/configmap/) and Secret (https://kubernetes.io/docs/concepts/configuration/secret/) in the Kubernetes documentation.

For the complete Kubernetes 1.19 changelog, see https://github.com/kubernetes/kubernetes/blob/master/CHANGELOG/CHANGELOG-1.19.md (https://github.com/kubernetes/kubernetes/blob/master/CHANGELOG/CHANGELOG-1.19.md) .

Kubernetes 1.18

Kubernetes 1.18 is now available in Amazon EKS. For more information about Kubernetes 1.18, see the official release announcement (https://kubernetes.io/blog/2020/03/25/kubernetes-1-18-release-announcement/) .

The following Kubernetes features are now supported in Kubernetes 1.18 Amazon EKS clusters:

- Topology Manager has reached beta status. This feature allows the CPU and Device Manager to coordinate resource allocation decisions, optimizing for low latency with machine learning and analytics workloads. For more information, see Control Topology Management Policies on a node (https://kubernetes.io/docs/tasks/administer-cluster/topology-manager/) in the Kubernetes documentation.

- Server-side Apply is updated with a new beta version. This feature tracks and manages changes to fields of all new Kubernetes objects, allowing you to know what changed your resources and when. For more information, see What is Server-side Apply? (https://kubernetes.io/blog/2020/04/01/kubernetes-1.18-feature-server-side-apply-beta-2/#what-is-server-side-apply) in the Kubernetes documentation.

- A new `pathType` field and a new `IngressClass` resource has been added to the Ingress specification. These features make it simpler to customize Ingress configuration, and are supported by the AWS Load Balancer Controller (./alb-ingress.html) (formerly called the ALB Ingress Controller). For more information, see Improvements to the Ingress API in Kubernetes 1.18 in the Kubernetes documentation. (https://kubernetes.io/blog/2020/04/02/improvements-to-the-ingress-api-in-kubernetes-1.18/)

- Configurable horizontal pod autoscaling behavior. For more information, see Support for configurable scaling behavior (https://kubernetes.io/docs/tasks/run-application/horizontal-pod-autoscale/#support-for-configurable-scaling-behavior) in the Kubernetes documentation.

- In 1.18 clusters, you no longer need to include the `AWS_DEFAULT_REGION=<region-code>` environment variable to pods when using IAM roles for service accounts in China regions, whether you use the mutating web hook or configure the environment variables manually. You still need to include the variable for all pods in earlier versions.

For the complete Kubernetes 1.18 changelog, see https://github.com/kubernetes/kubernetes/blob/master/CHANGELOG/CHANGELOG-1.18.md (https://github.com/kubernetes/kubernetes/blob/master/CHANGELOG/CHANGELOG-1.18.md) .

Kubernetes 1.17

Kubernetes 1.17 is now available in Amazon EKS. For more information about Kubernetes 1.17, see the official release announcement (https://kubernetes.io/blog/2019/12/09/kubernetes-1-17-release-announcement/) .

⚠ Important

- EKS has not enabled the `CSIMigrationAWS` feature flag. This will be enabled in a future release, along with detailed migration instructions. For more info on CSI migration, see the Kubernetes blog (https://kubernetes.io/blog/2019/12/09/kubernetes-1-17-feature-csi-migration-beta/) .

- Updating a cluster from 1.16 to 1.17 will fail if any of your AWS Fargate pods have a `kubelet` minor version earlier than 1.16. Before updating your cluster from 1.16 to 1.17, you need to recycle your Fargate pods so that their `kubelet` is 1.16 before attempting to update the cluster to 1.17. To recycle a Kubernetes deployment on a 1.15 or later cluster, use the following command.

  ```
  kubectl rollout restart deployment <deployment-name>
  ```

The following Kubernetes features are now supported in Kubernetes 1.17 Amazon EKS clusters:

- Cloud Provider Labels (https://kubernetes.io/docs/reference/kubernetes-api/labels-annotations-taints) have reached general availability. If you are using the beta labels in your pod specs for features such as node affinity, or in any custom controllers, then we recommend that you start migrating them to the new GA labels. For information about the new labels, see the following Kubernetes documentation:
  - node.kubernetes.io/instance-type (https://kubernetes.io/docs/reference/labels-annotations-taints/#nodekubernetesioinstance-type)
  - topology.kubernetes.io/region (https://kubernetes.io/docs/reference/labels-annotations-taints/#topologykubernetesioregion)

  - topology.kubernetes.io/zone (https://kubernetes.io/docs/reference/labels-annotations-taints/#topologykubernetesiozone)

- The ResourceQuotaScopeSelectors (https://kubernetes.io/docs/concepts/policy/resource-quotas/#quota-scopes) feature has graduated to generally available. This feature allows you to limit the number of resources a quota supports to only those that pertain to the scope.

- The TaintNodesByCondition (https://kubernetes.io/docs/concepts/scheduling-eviction/taint-and-toleration/#taint-nodes-by-condition) feature has graduated to generally available. This feature allows you to taint nodes that have conditions such as high disk or memory pressure.

- The CSI Topology (https://kubernetes-csi.github.io/docs/topology.html) feature has graduated to generally available, and is fully supported by the EBS CSI driver (https://github.com/kubernetes-sigs/aws-ebs-csi-driver#features-1) . You can use topology to restrict the Availability Zone where a volume is provisioned.

- Finalizer protection (https://kubernetes.io/docs/tasks/access-application-cluster/create-external-load-balancer/#garbage-collecting-load-balancers) for services of type `LoadBalancer` has graduated to generally available. This feature ensures that a service resource is not fully deleted until the correlating load balancer is also deleted.

- Custom resources now support default values. (https://kubernetes.io/docs/tasks/extend-kubernetes/custom-resources/custom-resource-definitions/#defaulting) You specify values in an OpenAPI v3 validation schema (https://kubernetes.io/docs/tasks/extend-kubernetes/custom-resources/custom-resource-definitions/#validation) .

- The Windows containers RunAsUsername (https://kubernetes.io/docs/tasks/configure-pod-container/configure-runasusername/) feature is now in beta, allowing you to run Windows applications in a container as a different user name than the default.

For the complete Kubernetes 1.17 changelog, see https://github.com/kubernetes/kubernetes/blob/master/CHANGELOG/CHANGELOG-1.17.md (https://github.com/kubernetes/kubernetes/blob/master/CHANGELOG/CHANGELOG-1.17.md) .

Kubernetes 1.16

Kubernetes 1.16 is now available in Amazon EKS. For more information about Kubernetes 1.16, see the official release announcement (https://kubernetes.io/blog/2019/09/18/kubernetes-1-16-release-announcement/) .

⚠Important

- Kubernetes 1.16 removes a number of discontinued APIs. Changes to your applications may be required before updating your cluster to 1.16. Carefully follow the 1.16 update prerequisites (./update-cluster.html#1-16-prerequisites) before updating.

- Starting with 1.16, the Amazon EKS certificate authority will honor certificate signing requests with SAN X.509 extensions, which resolves the EKS CA should honor SAN x509 extension (https://github.com/aws/containers-roadmap/issues/750) feature request from GitHub.

The following Kubernetes features are now supported in Kubernetes 1.16 Amazon EKS clusters:

- Volume expansion in the CSI specification has moved to beta, which allows for any CSI spec volume plugin to be resizeable. For more information, see Volume Expansion (https://kubernetes-csi.github.io/docs/volume-expansion.html) in the Kubernetes CSI documentation. The latest version of the EBS CSI driver (https://github.com/kubernetes-sigs/aws-ebs-csi-driver/tree/master/examples/kubernetes/resizing) supports volume expansion when running on an Amazon EKS 1.16 cluster.

- Windows GMSA support has graduated from alpha to beta, and is now supported by Amazon EKS. For more information, see Configure GMSA for Windows Pods and containers ↗ (https://kubernetes.io/docs/tasks/configure-pod-container/configure-gmsa/) in the Kubernetes documentation.

- A new annotation: `service.beta.kubernetes.io/aws-load-balancer-eip-allocations` is available on service type `LoadBalancer` to assign an elastic IP address to Network Load Balancers. For more information, see the Support EIP Allocations with AWS NLB ↗ (https://github.com/kubernetes/kubernetes/issues/63959) GitHub issue.

- Finalizer protection for service load balancers is now in beta and enabled by default. Service load balancer finalizer protection ensures that any load balancer resources allocated for a Kubernetes Service object, such as the AWS Network Load Balancer (https://docs.aws.amazon.com/eks/latest/userguide/load-balancing.html) , will be destroyed or released when the service is deleted. For more information, see Garbage Collecting Load Balancers ↗ (https://kubernetes.io/docs/tasks/access-application-cluster/create-external-load-balancer/#garbage-collecting-load-balancers) in the Kubernetes documentation.

- The Kubernetes custom resource definitions and admission webhooks extensibility mechanisms have both reached general availability. For more information, see Custom Resources ↗ (https://kubernetes.io/docs/concepts/extend-kubernetes/api-extension/custom-resources/) and Dynamic Admission Control ↗ (https://kubernetes.io/docs/reference/access-authn-authz/extensible-admission-controllers/) in the Kubernetes documentation.

- The server-side apply feature has reached beta status and is enabled by default. For more information, see Server Side Apply ↗ (https://kubernetes.io/docs/reference/using-api/api-concepts/#server-side-apply) in the Kubernetes documentation.

- The `CustomResourceDefaulting` feature is promoted to beta and enabled by default. Defaults may be specified in structural schemas through the `apiextensions.k8s.io/v1` API. For more information, see Specifying a structural schema ↗ (https://kubernetes.io/docs/tasks/extend-kubernetes/custom-resources/custom-resource-definitions/#specifying-a-structural-schema) in the Kubernetes documentation.

For the complete Kubernetes 1.16 changelog, see https://github.com/kubernetes/kubernetes/blob/master/CHANGELOG/CHANGELOG-1.16.md ↗ (https://github.com/kubernetes/kubernetes/blob/master/CHANGELOG/CHANGELOG-1.16.md) .

Kubernetes 1.15

Kubernetes 1.15 is now available in Amazon EKS. For more information about Kubernetes 1.15, see the official release announcement ↗ (https://kubernetes.io/blog/2019/06/19/kubernetes-1-15-release-announcement/) .

⚠Important

Starting with 1.15, Amazon EKS no longer tags the VPC containing your cluster.

- Subnets within the VPC of your cluster are still tagged.

- VPC tags are not modified on existing cluster updates to 1.15.

- For more information about VPC tagging, see VPC tagging requirement (./network_reqs.html#vpc-tagging) .

⚠Important

Amazon EKS has set the re-invocation policy for the Pod Identity Webhook to `IfNeeded`. This allows the webhook to be re-invoked if objects are changed by other mutating admission webhooks like the App Mesh sidecar injector. For more information about the App Mesh sidecar injector, see Install the sidecar injector (https://docs.aws.amazon.com/eks/latest/userguide/mesh-k8s-integration.html#install-injector) .

The following features are now supported in Kubernetes 1.15 Amazon EKS clusters:

- EKS now supports configuring transport layer security (TLS) termination, access logs, and source ranges for network load balancers. For more information, see Network Load Balancer support on AWS (https://kubernetes.io/docs/concepts/services-networking/service/#aws-nlb-support) on GitHub.
- Improved flexibility of Custom Resource Definitions (CRD), including the ability to convert between versions on the fly. For more information, see Extend the Kubernetes API with CustomResourceDefinitions (https://kubernetes.io/docs/tasks/extend-kubernetes/custom-resources/custom-resource-definitions/) on GitHub.
- NodeLocal DNSCache is in beta for Kubernetes version 1.15 clusters. This feature can help improve cluster DNS performance by running a DNS caching agent on cluster nodes as a DaemonSet. For more information, see Using NodeLocal DNSCache in Kubernetes clusters (https://kubernetes.io/docs/tasks/administer-cluster/nodelocaldns/) on GitHub.

  Note

  When running CoreDNS on Amazon EC2, we recommend not using `force_tcp` in the configuration and ensuring that `options use-vc` is not set in `/etc/resolv.conf`.

For the complete Kubernetes 1.15 changelog, see https://github.com/kubernetes/kubernetes/blob/master/CHANGELOG/CHANGELOG-1.15.md (https://github.com/kubernetes/kubernetes/blob/master/CHANGELOG/CHANGELOG-1.15.md) .

Amazon EKS Kubernetes release calendar

Note

Dates with only a month and a year are approximate and are updated with an exact date when it is known.

Kubernetes version	Upstream release	Amazon EKS release	Amazon EKS end of support
1.15	June 19, 2019	March 10, 2020	May 3, 2021
1.16	September 8, 2019	April 30, 2020	July, 2021
1.17	December 9, 2019	July 10, 2020	September, 2021
1.18	March 23, 2020	October 13, 2020	November, 2021
1.19	August 26, 2020	February 16, 2021	April, 2022
1.20	December 8, 2020	April, 2021	June, 2022

Amazon EKS version support and FAQ

In line with the Kubernetes community support for Kubernetes versions, Amazon EKS is committed to supporting at least four production-ready versions of Kubernetes at any given time. We will announce the end of support date of a given Kubernetes minor version at least 60 days before the end of support date. Because of the Amazon EKS qualification and release process for new Kubernetes versions, the end of support date of a Kubernetes version on Amazon EKS will be on or after the date that the Kubernetes project stops supporting the version upstream.

Frequently asked questions

Q: How long is a Kubernetes version supported by Amazon EKS?

A: A Kubernetes version is fully supported for 14 months after first being available on Amazon EKS. This is true even if upstream Kubernetes is no longer supporting a version available on Amazon EKS. We backport security patches that are applicable to the Kubernetes versions supported on Amazon EKS.

Q: Am I notified when support is ending for a Kubernetes version on Amazon EKS?

A: Yes. If any clusters in your account are running the version nearing the end of support, Amazon EKS sends out a notice through the AWS Personal Health Dashboard approximately 12 months after the Kubernetes version was released on Amazon EKS. The notice includes the end of support date, which is at least 60 days from the date of the notice.

Q: What happens on the end of support date?

A: On the end of support date, you are no longer able to create new Amazon EKS clusters with the unsupported version. Existing control planes are automatically updated by Amazon EKS to the oldest supported version through a gradual deployment process after the end of support date. After the automatic control plane update, you must manually update cluster add-ons and Amazon EC2 nodes. For more information, see Update an existing cluster (./update-cluster.html#update-existing-cluster) .

Q: When exactly will my control plane be automatically updated after the end of support date?

A: Amazon EKS is unable to provide specific timeframes. Automatic updates can happen at any time after the end of support date. We recommend that you take proactive action and update your control plane without relying on the Amazon EKS automatic update process. For more information, see Updating a cluster (./update-cluster.html) .

Q: Can I leave my control plane on a Kubernetes version indefinitely?

A: No. Cloud security at AWS is the highest priority. Amazon EKS does not allow control planes to stay on a version that has reached end of support.

Q: Which Kubernetes features are supported by Amazon EKS?

A: Amazon EKS supports all general availability features of the Kubernetes API, as well as beta features which are enabled by default. Alpha features are not supported.

Q: Are Amazon EKS managed node groups automatically updated along with the cluster control plane version?

A: No. A managed node group creates Amazon EC2 instances in your account. These instances aren't automatically upgraded when you or Amazon EKS update your control plane. If Amazon EKS automatically updates your control plane, the Kubernetes version on your managed node group may be more than one version earlier than your control plane. If a managed node group contains instances that are running a version of Kubernetes that is more than one version earlier than the control plane, the node group has a health issue in the **Node Groups** section of the **Compute** tab on the **Configuration** tab of your cluster in the console. If a node group has an available version update, **Update now** appears next to the node group in the console. For more information, see Updating a managed node group (./update-managed-node-group.html) . We recommend maintaining the same Kubernetes version on your control plane and nodes.

Q: Are self-managed node groups automatically updated along with the cluster control plane version?

A: No. A self-managed node group includes Amazon EC2 instances in your account. These instances aren't automatically upgraded when you or Amazon EKS update the control plane version. A self-managed node group doesn't have any indication in the console that it needs updating. You can view the `kubelet` version installed on a node by selecting the node in the **Nodes** list on the **Overview** tab of your cluster to determine which nodes need updating. You must manually update the nodes. For more information, see Self-managed node updates (./update-workers.html) .

The Kubernetes project tests compatibility between the control plane and nodes for up to two minor versions. For example, 1.17 nodes continue to operate when orchestrated by a 1.19 control plane. However, running a cluster with nodes that are persistently two minor versions behind the control plane is not recommended. For more information, see Kubernetes version and version skew support policy (https://kubernetes.io/docs/setup/version-skew-policy/) in the Kubernetes documentation. We recommend maintaining the same Kubernetes version on your control plane and nodes.

Q: Are pods running on Fargate automatically upgraded with an automatic cluster control plane version upgrade?

Yes. Fargate pods run on infrastructure in AWS owned accounts on the Amazon EKS side of the shared responsibility model (./security.html) . Amazon EKS uses the Kubernetes eviction API to attempt to gracefully drain pods running on Fargate. For more information, see The Eviction API (https://kubernetes.io/docs/tasks/administer-cluster/safely-drain-node/#eviction-api) in the Kubernetes documentation. If a pod can’t be evicted, then Amazon EKS issues a Kubernetes `delete pod` command. We strongly recommend running Fargate pods as part of a replication controller like a Kubernetes deployment so a pod is automatically rescheduled after deletion. For more information, see Deployments ((https://kubernetes.io/docs/concepts/workloads/controllers/deployment) in the Kubernetes documentation. The new version of the Fargate pod is deployed with a `kubelet` version that is the same version as your updated cluster control plane version.

Important

If you update the control plane, you must update the Fargate nodes yourself. To update Fargate nodes, delete the Fargate pod represented by the node and redeploy the pod. The new pod is deployed with a `kubelet` version that is the same version as your cluster.