

The screenshot shows the AWS CloudSearch results for the query 'elastic kubernetes Service'. The results are categorized under 'Services' and 'Features'.

Services

- Elastic Kubernetes Service (Thumbnail: orange square with white icon, Rating: 5 stars, Description: 'The most trusted way to start, run, and scale Kubernetes')
- Elastic Transcoder (Thumbnail: orange square with white icon, Rating: 5 stars, Description: 'Easy-to-Use Scalable Media Transcoding')
- Elastic Beanstalk (Thumbnail: orange square with white icon, Rating: 5 stars, Description: 'Run and Manage Web Apps')
- Elastic Container Service (Thumbnail: orange square with white icon, Rating: 5 stars, Description: 'Highly secure, reliable, and scalable way to run containers')

Features

- Clusters (Thumbnail: orange square with white icon, Description: 'Elastic Kubernetes Service feature')
- Elastic IPs (Thumbnail: orange square with white icon, Description: 'EC2 feature')

On the right side of the search results, there is a sidebar titled 'CloudWatch Metrics' with a 'Default layout' section containing a 'Create application' button and a 'Recent applications' list. The sidebar also includes sections for 'Metrics' and 'Logs'.

Click on Add Cluster

The screenshot shows the EKS Clusters management page. The left sidebar contains navigation links for 'Clusters', 'Amazon EKS Anywhere', 'Enterprise Subscriptions', 'Related services' (Amazon ECR, AWS Batch), and 'Console settings', 'Documentation', 'Submit feedback'.

The main content area displays the 'Clusters (0)' list. It includes a 'Filter clusters' input field, sorting options ('Cluster name ▲', 'Status ▼', 'Kubernetes version ▼', 'Support period ▼'), and filtering options ('Upgrade policy ▼', 'Created ▼'). A message states 'No clusters' and 'You do not have any clusters.' Below this is a 'Create cluster' button.

At the bottom of the page, there is a footer bar with links for 'Display a menu', 'Feedback', '© 2024, Amazon Web Services, Inc. or its affiliates.', 'Privacy', 'Terms', and 'Cookie preferences'.

Click on Create Cluster

The screenshot shows the 'Configure cluster' step of the EKS cluster creation wizard. On the left, a sidebar lists steps from 1 to 6. Step 1 is 'Configure cluster', which is currently active. The main area is titled 'Configure cluster' and contains two sections: 'Cluster configuration' and 'Kubernetes version settings'. In the 'Cluster configuration' section, the 'Name' field is set to 'demo_cluster1'. Below it, a note says: 'The cluster name should begin with letter or digit and can have any of the following characters: the set of Unicode letters, digits, hyphens and underscores. Maximum length of 100.' The 'Cluster IAM role' dropdown is set to 'eks-cluster'. A button 'Create a role in IAM console' is visible. In the 'Kubernetes version settings' section, the 'Kubernetes version' dropdown is set to '1.28'. Below it, an 'Upgrade policy' dropdown has 'Extended' selected. The bottom of the page includes standard AWS navigation links like 'Display a menu', 'Feedback', and copyright information.

The screenshot shows the IAM role details for 'eks-cluster'. The left sidebar shows the IAM navigation path. The main area is titled 'eks-cluster' and displays the 'Summary' of the role. It shows the creation date (July 26, 2024), last activity (2 months ago), ARN (arn:aws:iam::637423339839:role/eks-cluster), and maximum session duration (1 hour). Below the summary is the 'Permissions' tab, which lists one policy: 'AmazonEKSClusterPolicy' (AWS managed). Other tabs include 'Trust relationships', 'Tags', 'Last Accessed', and 'Revoke sessions'. The bottom of the page includes standard AWS navigation links.

Ex of Roles.

The screenshot shows the AWS EKS Cluster Configuration page. At the top, there's a navigation bar with links to various services like Apple, iCloud, Google, Facebook, Twitter, LinkedIn, The Weather Channel, NDTV, PT registration, and CodeGPT. On the right side of the navigation bar, it shows "N. Virginia" and "ShalAdmin".

The main content area has several sections:

- Kubernetes version**: Set to 1.28.
- Upgrade policy**: Shows two options:
 - Extended**: This option supports the Kubernetes version for 26 months after the release date. The extended support period has an additional hourly cost that begins after the standard support period ends. When extended support ends, your cluster will be auto-upgraded to the next version.
 - Standard**: This option supports the Kubernetes version for 14 months after the release date. There is no additional cost. When standard support ends, your cluster will be auto-upgraded to the next version.
- Cluster access**:
 - Bootstrap cluster administrator access**: Set to **Allow cluster administrator access**.
 - Cluster authentication mode**: Set to **EKS API and ConfigMap**.

At the bottom of the page, there are links for "Display a menu", "Feedback", "© 2024, Amazon Web Services, Inc. or its affiliates.", "Privacy", "Terms", and "Cookie preferences".

Click on Next

Screenshot of the AWS EKS Create EKS cluster wizard Step 2: Specify networking.

The Networking section shows:

- VPC**: vpc-082939a3ed23f7468 | Default
- Subnets**: subnet-01a8b4c483f84bbc5 (us-east-1b 172.31.80.0/20) and subnet-07c36d7bb676cc782 (us-east-1e 172.31.48.0/20)
- Security groups**: sg-0471e51c0a9421150 | default (default VPC security group)

Choose cluster IP address family: IPv4 is selected.

Display a menu **Feedback** **© 2024, Amazon Web Services, Inc. or its affiliates.** **Privacy** **Terms** **Cookie preferences**

Screenshot of the AWS EKS Create EKS cluster wizard Step 3: Review and create.

The Security groups section shows:

- Select security groups**: sg-0471e51c0a9421150 | default (default VPC security group)
- Clear selected security groups** button is visible.

Choose cluster IP address family: IPv4 is selected.

Configure Kubernetes service IP address block: Specified range is "Specify the range from which cluster services will receive IP addresses."

Cluster endpoint access: Public and private options are available.

Cancel **Previous** **Next** **Display a menu** **Feedback** **© 2024, Amazon Web Services, Inc. or its affiliates.** **Privacy** **Terms** **Cookie preferences**

Click on Next.

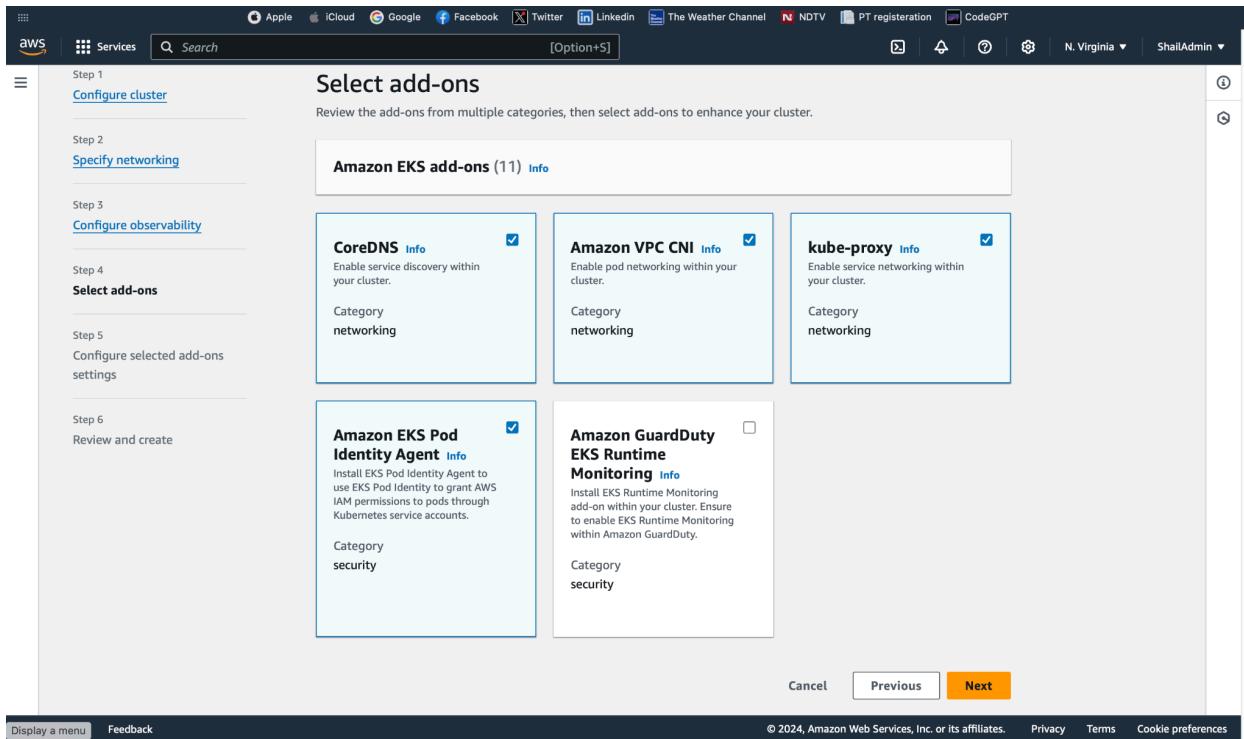
Configure Observability. Leave it as Default

The screenshot shows the AWS EKS Create EKS Cluster wizard at Step 3: Configure observability. The left sidebar lists steps 1 through 6. Step 3 is selected, titled "Configure observability". The main content area has a section titled "About observability" and a "Metrics" section. Under "Metrics", there are two options: "Prometheus" and "CloudWatch". The "CloudWatch" option is highlighted with a blue border and contains a callout box with the following text:

ⓘ You can enable CloudWatch Observability in your clusters through the CloudWatch Observability add-on.
After your cluster is created, navigate to the add-ons tab and install CloudWatch Observability add-on to enable CloudWatch Application Signals and Container Insights and start ingesting telemetry into CloudWatch.

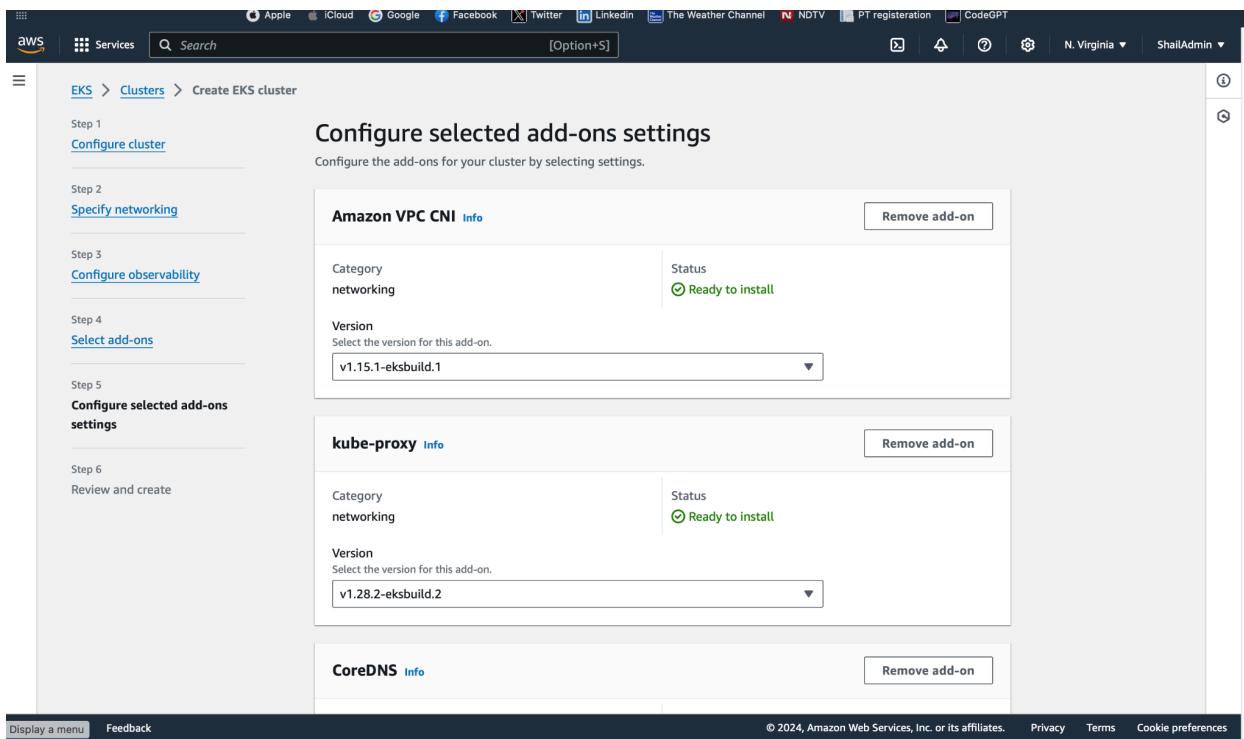
At the bottom of the page, there are links for "Display a menu", "Feedback", "© 2024, Amazon Web Services, Inc. or its affiliates.", "Privacy", "Terms", and "Cookie preferences".

Click on Next.



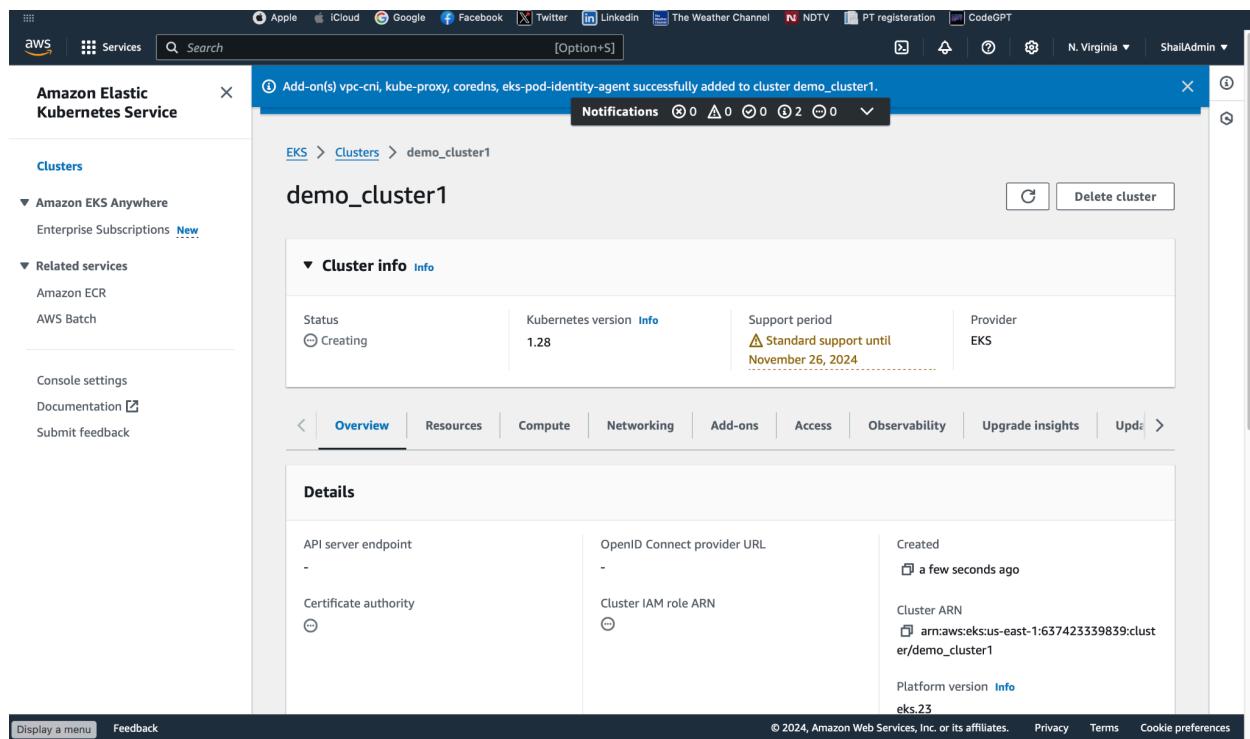
The screenshot shows the 'Select add-ons' step of the AWS EKS cluster creation wizard. On the left, a sidebar lists steps from 1 to 6. Step 1 is 'Configure cluster', Step 2 is 'Specify networking', Step 3 is 'Configure observability', Step 4 is 'Select add-ons' (which is currently selected), Step 5 is 'Configure selected add-ons settings', and Step 6 is 'Review and create'. The main area is titled 'Select add-ons' and contains a heading 'Amazon EKS add-ons (11)'. It lists five add-ons under the 'networking' category: CoreDNS, Amazon VPC CNI, kube-proxy, Amazon EKS Pod Identity Agent, and Amazon GuardDuty EKS Runtime Monitoring. The 'Amazon EKS Pod Identity Agent' is checked. At the bottom right are 'Cancel', 'Previous', and 'Next' buttons, with 'Next' being highlighted.

Click on Next



The screenshot shows the 'Configure selected add-ons settings' step of the AWS EKS cluster creation wizard. The sidebar shows steps 1 through 6. The main area is titled 'Configure selected add-ons settings' and shows three add-ons: Amazon VPC CNI, kube-proxy, and CoreDNS. Each add-on has a 'Remove add-on' button. The 'Amazon VPC CNI' section shows it is 'Ready to install'. The 'kube-proxy' section also shows it is 'Ready to install'. The 'CoreDNS' section is partially visible. At the bottom right are 'Cancel', 'Previous', and 'Next' buttons, with 'Next' being highlighted.

Click on Next. Review and Create.



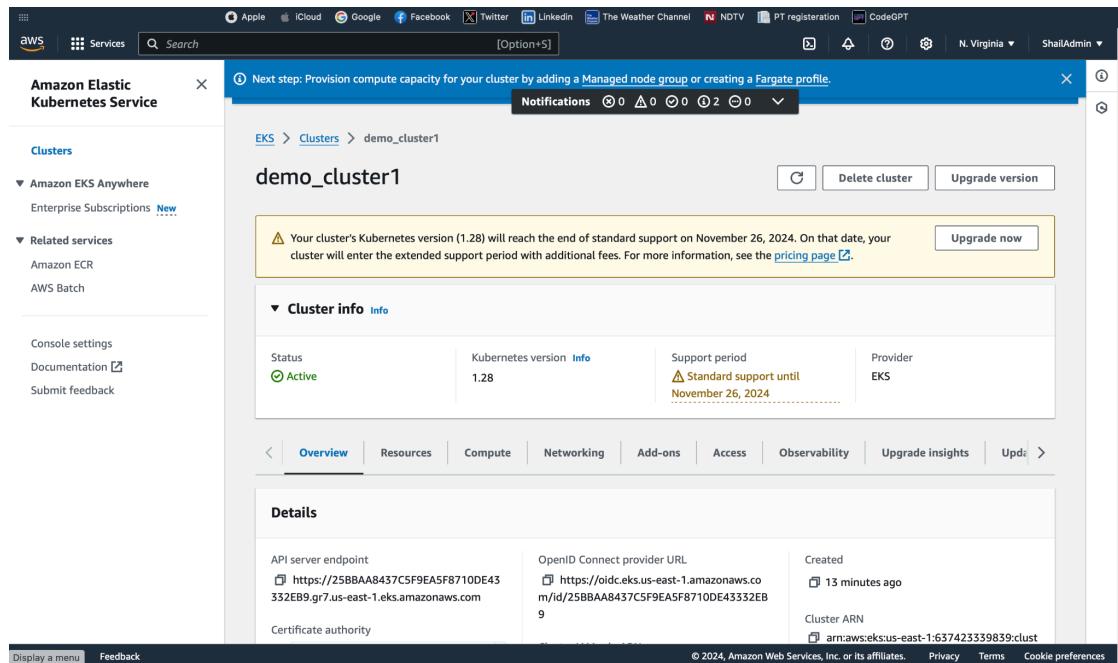
The screenshot shows the 'Clusters' section of the Amazon EKS service. A cluster named 'demo_cluster1' is being created, indicated by the 'Creating' status in the 'Cluster info' section. The 'Overview' tab is selected. A notification at the top states: 'Add-on(s) vpc-cni, kube-proxy, coredns, eks-pod-identity-agent successfully added to cluster demo_cluster1.'

Status	Kubernetes version	Support period	Provider
Creating	1.28	Standard support until November 26, 2024	EKS

Details section:

API server endpoint	OpenID Connect provider URL	Created
-	-	a few seconds ago
Certificate authority	Cluster IAM role ARN	Cluster ARN
-	-	arn:aws:eks:us-east-1:637423339839:cluster/demo_cluster1

Clusters are getting created.



The screenshot shows the same 'Clusters' section after the cluster has been successfully created. The 'demo_cluster1' status is now 'Active'. A yellow warning message in the 'Cluster info' section informs the user that the Kubernetes version (1.28) will reach the end of standard support on November 26, 2024, and encourages them to upgrade now. The 'Overview' tab is selected.

Status	Kubernetes version	Support period	Provider
Active	1.28	Standard support until November 26, 2024	EKS

Details section:

API server endpoint	OpenID Connect provider URL	Created
https://25BBA8437C5F9EA5F8710DE4332EB9.gr7.us-east-1.eks.amazonaws.com	https://oidc.eks.us-east-1.amazonaws.com/id/25BBA8437C5F9EA5F8710DE4332EB9	13 minutes ago
Certificate authority	Cluster ARN	arn:aws:eks:us-east-1:637423339839:cluster/demo_cluster1

Click on Add Node Group

The screenshot shows the AWS EKS service page. On the left, there's a sidebar with 'Clusters' (Amazon EKS Anywhere, Enterprise Subscriptions), 'Related services' (Amazon ECR, AWS Batch), and links for 'Console settings', 'Documentation', and 'Submit feedback'. The main area has three sections:

- Nodes (0)**: Shows a search bar and a table header for 'Node name', 'Instance type', 'Node group', 'Created', and 'Status'. Below it says 'No Nodes' and 'This cluster does not have any Nodes, or you don't have permission to view them.'
- Node groups (0)**: Shows a table header for 'Group name', 'Desired size', 'AMI release version', 'Launch template', and 'Status'. Below it says 'No node groups' and 'This cluster does not have any node groups.' It includes an 'Add node group' button.
- Fargate profiles (0)**: Shows a table header for 'Profile name', 'Namespaces', and 'Status'. Below it says 'No Fargate profiles' and 'This cluster does not have any Fargate profiles.'

At the bottom, there are navigation links for 'Display a menu', 'Feedback', and copyright information: '© 2024, Amazon Web Services, Inc. or its affiliates.', 'Privacy', 'Terms', and 'Cookie preferences'.

Click on Add node group

The screenshot shows the AWS EKS Node Groups configuration interface. The top navigation bar includes the AWS logo, a services menu, a search bar, and account information for 'N. Virginia' and 'ShailAdmin'. The main content area is titled 'Configure node group' with a sub-section 'Node group configuration'. It shows a note: 'A node group is a group of EC2 instances that supply compute capacity to your Amazon EKS cluster. You can add multiple node groups to your cluster.' Below this, there's a 'Name' field containing 'node_group_1' with a note: 'The node group name should begin with letter or digit and can have any of the following characters: the set of Unicode letters, digits, hyphens and underscores. Maximum length of 63.' A 'Node IAM role' section contains a 'Select role' dropdown, a 'Create a role in IAM console' button, and a note: 'The selected role must not be used by a self-managed node group as this could lead to a service interruption upon managed node group deletion.' A 'Launch template' section is also present. On the left sidebar, steps 1 through 4 are listed: 'Configure node group', 'Set compute and scaling configuration', 'Specify networking', and 'Review and create'. The bottom of the screen includes standard AWS footer links for privacy, terms, and cookie preferences.

Create a Role in IAM console

The screenshot shows the AWS IAM 'Create role' process. The top navigation bar includes the AWS logo, a services menu, a search bar, and account information for 'Global' and 'ShailAdmin'. The main content area is titled 'Select trusted entity' with a sub-section 'Trusted entity type'. It lists five options: 'AWS service' (selected), 'AWS account', 'Web identity', 'SAML 2.0 federation', and 'Custom trust policy'. Below this, a 'Use case' section allows selecting a service like EC2. A note states: 'Allow an AWS service like EC2, Lambda, or others to perform actions in this account.' A 'Service or use case' dropdown is set to 'EC2'. A 'Choose a use case for the specified service' section shows 'EC2' selected with the note: 'Allows EC2 instances to call AWS services on your behalf.' The bottom of the screen includes standard AWS footer links for privacy, terms, and cookie preferences.

Click on Next

Step 2: Add permissions

Permissions policy summary

Policy name	Type	Attached as
AmazonEC2ContainerRegistryReadOnly	AWS managed	Permissions policy
AmazonEKS_CNI_Policy	AWS managed	Permissions policy
AmazonEKSWorkerNodePolicy	AWS managed	Permissions policy

The screenshot shows the AWS IAM 'Create role' wizard at Step 2: Add permissions. The title is 'Name, review, and create'. On the left, a sidebar shows steps: Step 1 (Select trusted entity), Step 2 (Add permissions, currently selected), and Step 3 (Name, review, and create). The main area is titled 'Role details' and contains fields for 'Role name' (EKS_NODE) and 'Description' (Allows EC2 instances to call AWS services on your behalf). Below this is a code editor showing the JSON trust policy:

```
1 - [ {  
2 -     "Version": "2012-10-17",  
3 -     "Statement": [  
4 -         {  
5 -             "Effect": "Allow",  
6 -             "Action": [  
7 -                 "sts:AssumeRole"  
8 -             ],  
9 -             "Principal": "*"  
0 -         }  
1 -     ]  
2 - }
```

At the bottom, there's a 'Step 1: Select trusted entities' section with an 'Edit' button, and a note about the trust policy being applied to the entire account.

Go to Node Group

The screenshot shows the AWS EKS Node Group Configuration wizard at Step 2. The left sidebar lists steps: Step 2 (Set compute and scaling configuration), Step 3 (Specify networking), and Step 4 (Review and create). The main content area is titled "Node group configuration". It includes a "Name" field with "node_group_1" entered, a note about naming rules, and an "IAM role" dropdown set to "EKS_NODE". A warning message states: "The selected role must not be used by a self-managed node group as this could lead to a service interruption upon managed node group deletion." A link to "Learn more" is provided. Below this is a "Launch template" section with a radio button for "Use launch template" and a note about configuring the node group using an EC2 launch template. The bottom section is titled "Kubernetes labels" with an "Info" link.

Click on Next

The screenshot shows the AWS EKS Node Group Configuration wizard at Step 3. The left sidebar lists steps: Step 2 (Set compute and scaling configuration), Step 3 (Specify networking), and Step 4 (Review and create). The main content area is titled "Node group compute configuration". It includes an "AMI type" dropdown set to "Amazon Linux 2 (AL2_x86_64)", a "Capacity type" dropdown set to "On-Demand", and an "Instance types" search bar showing "t2.medium" with details: vCPU: 2 vCPUs, Memory: 4 GiB, Network: Low to Moderate, Max ENI: 3, Max IPs: 18. Below this is a "Disk size" input field with "20" and a "GiB" unit. The bottom section is titled "Node group scaling configuration" and contains "Desired size" and "Minimum size" fields, both currently set to "2".

aws Services Search [Option+S] N. Virginia ShaileshAdmin

Desired size
Set the desired number of nodes that the group should launch with initially.
 nodes
 Desired node size must be greater than or equal to 0

Minimum size
Set the minimum number of nodes that the group can scale in to.
 nodes
 Minimum node size must be greater than or equal to 0

Maximum size
Set the maximum number of nodes that the group can scale out to.
 nodes
 Maximum node size must be greater than or equal to 1 and cannot be lower than the minimum size

Node group update configuration Info

Maximum unavailable
Set the maximum number or percentage of unavailable nodes to be tolerated during the node group version update.

Number Enter a number
 Percentage Specify a percentage

Value
 node
 Node count must be greater than 0.

Cancel Previous Next

Display a menu Feedback © 2024, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

Click on Next

aws Services Search [Option+S] N. Virginia ShaileshAdmin

EKS > Clusters > demo_cluster1 > Node groups > Add node group

Step 1 Configure node group

Step 2 Set compute and scaling configuration

Step 3 **Specify networking**

Step 4 Review and create

Specify networking

Node group network configuration
These properties cannot be changed after the node group is created.

Subnets Info
Specify the subnets in your VPC where your nodes will run. To create a new subnet, go to the corresponding page in the [VPC console](#).

Select subnets Configure remote access to nodes [Info](#)

subnet-01a8b4c483f84bbc5 X us-east-1b 172.31.80.0/20 subnet-0330084cd3e4f88d4 X us-east-1a 172.31.0.0/20

Clear selected subnets

Cancel Previous Next

Display a menu Feedback © 2024, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

Click on Next. Click on Create

The screenshot shows two views of the AWS EKS Node Group configuration interface.

Top View (Node Group Configuration):

- Node group configuration:** Kubernetes version 1.28, AMI release version 1.28.13-20241011, AMI type AL2_x86_64, Status Creating.
- Details:** Node group ARN: arn:aws:eks:us-east-1:63742339839:nodegroup/demo_cluster1/node_group_1/dac94fca-7378-5b22-a829-68313f119458. Created: a few seconds ago. Autoscaling group name: (empty). Node IAM role ARN: (empty). Capacity type: On-Demand. Desired size: 2 nodes. Minimum size: 2 nodes. Maximum size: (empty). Subnets: subnet-01a8b4c483f84bbc5, subnet-0330084cd3e4f88d4. Configure remote access to nodes: off.

Bottom View (Nodes List):

- Nodes (2) Info:** Filter Nodes by property or value. Two nodes listed:
 - ip-172-31-7-191.ec2.internal, t2.medium, node_group_1, Created 22 minutes ago, Ready.
 - ip-172-31-80-89.ec2.internal, t2.medium, node_group_1, Created 24 minutes ago, Ready.

Click on Cloudshell

The screenshot shows the AWS Management Console with the 'Amazon Elastic Kubernetes Service' (EKS) selected in the left sidebar. In the main content area, the 'Clusters' section is expanded, showing details for the 'eks-23' cluster. The 'Cluster IAM role ARN' is listed as `arn:aws:iam::637423339839:role/eks-cluster/demo_cluster1`. Below this, the 'Kubernetes version settings' section shows an 'Upgrade policy' of 'Extended'. At the bottom of the page, a CloudShell window is open, titled 'us-east-1'. It displays a terminal session with the following command history:

```
[cloudshell-user@ip-10-132-61-114 ~]$ aws s3 ls
2024-09-20 14:03:23 elasticbeantalk-us-east-1-637423339839
2024-10-06 07:40:34 linkedin-06102024
2024-10-10 15:55:52 shaileshgupta-me
[cloudshell-user@ip-10-132-61-114 ~]$
```

This screenshot shows the same AWS Management Console setup as the first one, but the CloudShell window now displays a different terminal session. The command history includes:

```
[cloudshell-user@ip-10-132-61-114 ~]$ aws s3 ls
2024-09-20 14:03:23 elasticbeantalk-us-east-1-637423339839
2024-10-06 07:40:34 linkedin-06102024
2024-10-10 15:55:52 shaileshgupta-me
[cloudshell-user@ip-10-132-61-114 ~]$ aws eks --region us-east-1 update-kubeconfig --name demo_cluster1
Added new context: arn:aws:eks:us-east-1:637423339839:cluster/demo_cluster1 to /home/cloudshell-user/.kube/config
[cloudshell-user@ip-10-132-61-114 ~]$
```

CloudShell

Actions ▾ +

```
2024-10-06 07:40:34 linkdine-06102024
2024-10-10 15:55:52 shaleshgupta-me
[cldshell-user@ip-10-132-61-114 ~]$ aws eks --region us-east-1 update-kubeconfig --name demo_cluster1
Added new context arn:aws:eks:us-east-1:63742339839:cluster/demo_cluster1 to /home/cloudshell-user/.kube/config
[cldshell-user@ip-10-132-61-114 ~]$ kubectl get nodes
NAME           STATUS   ROLES      AGE   VERSION
ip-172-31-7-191.ec2.internal   Ready    <none>   34m   v1.28.13-eks-a737599
ip-172-31-80-89.ec2.internal   Ready    <none>   36m   v1.28.13-eks-a737599
[cldshell-user@ip-10-132-61-114 ~]$ kubectl get pods
No resources found in default namespace.
[cldshell-user@ip-10-132-61-114 ~]$ kubectl getns
error: unknown command "getns" for "kubectl"

Did you mean this?
  get
[cldshell-user@ip-10-132-61-114 ~]$ kubectl get ns
NAME           STATUS   AGE
default        Active   70m
kube-node-lease Active  70m
kube-public    Active  70m
kube-system   Active  70m
[cldshell-user@ip-10-132-61-114 ~]$ kubectl create ns dev
namespace/dev created
[cldshell-user@ip-10-132-61-114 ~]$ kubectl get ns
NAME           STATUS   AGE
default        Active   70m
dev            Active   8s
kube-node-lease Active  70m
kube-public    Active  70m
kube-system   Active  70m
[cldshell-user@ip-10-132-61-114 ~]$ kubectl create ns dev
namespace/dev created
[cldshell-user@ip-10-132-61-114 ~]$ kubectl get ns
NAME           STATUS   AGE
default        Active   70m
dev            Active   8s
kube-node-lease Active  70m
kube-public    Active  70m
kube-system   Active  70m
[cldshell-user@ip-10-132-61-114 ~]$ kubectl run nginx-pod --image=nginx
error: unknown flag: --image=nginx
See 'kubectl run --help' for usage.
[cldshell-user@ip-10-132-61-114 ~]$ kubectl run nginx-pod --image=nginx
pod/nginx-pod created
[cldshell-user@ip-10-132-61-114 ~]$ ■
```

AWS CloudShell

Display a menu Feedback © 2024, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

```
aws cloudshell user@ip-10-132-61-114 ~]$ kubectl get nodes
NAME           STATUS   ROLES      AGE   VERSION
ip-172-31-7-191.ec2.internal   Ready    <none>   34m   v1.28.13-eks-a737599
ip-172-31-80-89.ec2.internal   Ready    <none>   36m   v1.28.13-eks-a737599
[cldshell-user@ip-10-132-61-114 ~]$ kubectl get pods
No resources found in default namespace.
[cldshell-user@ip-10-132-61-114 ~]$ kubectl getns
error: unknown command "getns" for "kubectl"

Did you mean this?
  get
[cldshell-user@ip-10-132-61-114 ~]$ kubectl get ns
NAME           STATUS   AGE
default        Active   70m
kube-node-lease Active  70m
kube-public    Active  70m
kube-system   Active  70m
[cldshell-user@ip-10-132-61-114 ~]$ kubectl create ns dev
namespace/dev created
[cldshell-user@ip-10-132-61-114 ~]$ kubectl get ns
NAME           STATUS   AGE
default        Active   70m
dev            Active   8s
kube-node-lease Active  70m
kube-public    Active  70m
kube-system   Active  70m
[cldshell-user@ip-10-132-61-114 ~]$ kubectl run nginx-pod --image=nginx
error: unknown flag: --image=nginx
See 'kubectl run --help' for usage.
[cldshell-user@ip-10-132-61-114 ~]$ kubectl run nginx-pod --image=nginx
pod/nginx-pod created
[cldshell-user@ip-10-132-61-114 ~]$ ■
```

AWS CloudShell

Display a menu Feedback © 2024, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

CloudShell

Actions ▾ +

```
us-east-1
```

```
No resources found in default namespace.
[cldshell-user@ip-10-132-61-114 ~]$ kubectl getns
error: unknown command "getns" for "kubectl"

Did you mean this?
  get
[cldshell-user@ip-10-132-61-114 ~]$ kubectl get ns
NAME           STATUS   AGE
default        Active   70m
kube-node-lease Active  70m
kube-public    Active  70m
kube-system   Active  70m
[cldshell-user@ip-10-132-61-114 ~]$ kubectl create ns dev
namespace/dev created
[cldshell-user@ip-10-132-61-114 ~]$ kubectl get ns
NAME           STATUS   AGE
default        Active   70m
dev            Active   8s
kube-node-lease Active  70m
kube-public    Active  70m
kube-system   Active  70m
[cldshell-user@ip-10-132-61-114 ~]$ kubectl run nginx-pod --image=nginx
error: unknown flag: --image=nginx
See 'kubectl run --help' for usage.
[cldshell-user@ip-10-132-61-114 ~]$ kubectl run nginx-pod --image=nginx
pod/nginx-pod created
[cldshell-user@ip-10-132-61-114 ~]$ kubectl get pods
NAME    READY  STATUS  RESTARTS  AGE
nginx-pod 1/1   Running  0          30s
[cldshell-user@ip-10-132-61-114 ~]$ kubectl run nginx-pod2 --image=nginx -n dev
pod/nginx-pod2 created
[cldshell-user@ip-10-132-61-114 ~]$ ■
```

AWS CloudShell

Display a menu Feedback © 2024, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

```
user@host: ~
```

```
aws cloudshell user@ip-10-132-61-114: ~]$ kubectl get ns
namespace/dev created
[cloudshell-user@ip-10-132-61-114 ~]$ kubectl get ns
NAME      STATUS   AGE
default   Active   70m
dev       Active   8s
kube-node-lease Active  70m
kube-public Active  70m
kube-system Active  70m
[cloudshell-user@ip-10-132-61-114 ~]$ kubectl run nginx-pod --image=nginx
error: unknown flag: --image=nginx
See 'kubectl run --help' for usage.
[cloudshell-user@ip-10-132-61-114 ~]$ kubectl run nginx-pod --image=nginx
pod/nginx-pod created
[cloudshell-user@ip-10-132-61-114 ~]$ kubectl get pods
NAME      READY   STATUS  RESTARTS   AGE
nginx-pod 1/1    Running 0          30s
[cloudshell-user@ip-10-132-61-114 ~]$ kubectl run nginx-pod2 --image=nginx -n dev
pod/nginx-pod2 created
[cloudshell-user@ip-10-132-61-114 ~]$ kubectl pods -n dev
error: unknown command "pods" for "kubectl"

Did you mean this?
  logs
[cloudshell-user@ip-10-132-61-114 ~]$ kubectl get pods -n dev
NAME      READY   STATUS  RESTARTS   AGE
nginx-pod2 1/1    Running 0          47s
[cloudshell-user@ip-10-132-61-114 ~]$
```

AWS CloudShell

Display a menuFeedback

© 2024, Amazon Web Services, Inc. or its affiliates. [Privacy](#) [Terms](#) [Cookie preferences](#)

```
CloudShell
```

Actions ▾

us-east-1 +

```
LL user@host: ~$
```

```
[cloudshell-user@ip-10-132-61-114 ~]$ kubectl get ns
NAME      STATUS   AGE
default   Active   70m
dev       Active   8s
kube-node-lease Active  70m
kube-public Active  70m
kube-system Active  70m
[cloudshell-user@ip-10-132-61-114 ~]$ kubectl run nginx-pod --image=nginx
error: unknown flag: --image=nginx
See 'kubectl run --help' for usage.
[cloudshell-user@ip-10-132-61-114 ~]$ kubectl run nginx-pod --image=nginx
pod/nginx-pod created
[cloudshell-user@ip-10-132-61-114 ~]$ kubectl get pods
NAME      READY   STATUS  RESTARTS   AGE
nginx-pod 1/1    Running 0          30s
[cloudshell-user@ip-10-132-61-114 ~]$ kubectl run nginx-pod2 --image=nginx -n dev
pod/nginx-pod2 created
[cloudshell-user@ip-10-132-61-114 ~]$ kubectl pods -n dev
error: unknown command "pods" for "kubectl"

Did you mean this?
  logs
[cloudshell-user@ip-10-132-61-114 ~]$ kubectl get pods -n dev
NAME      READY   STATUS  RESTARTS   AGE
nginx-pod2 1/1    Running 0          47s
[cloudshell-user@ip-10-132-61-114 ~]$ kubectl run nginx-pod3 --image=nginx --labels="app=my-nginx-app"
pod/nginx-pod3 created
[cloudshell-user@ip-10-132-61-114 ~]$ kubectl get pods
NAME      READY   STATUS  RESTARTS   AGE
nginx-pod  1/1    Running 0          4m2s
nginx-pod3 1/1    Running 0          7s
[cloudshell-user@ip-10-132-61-114 ~]$
```

AWS CloudShell

Display a menuFeedback

© 2024, Amazon Web Services, Inc. or its affiliates. [Privacy](#) [Terms](#) [Cookie preferences](#)

```
CloudShell
```

Actions ▾

us-east-1 +

```
See 'kubectl run --help' for usage.
[cloudshell-user@ip-10-132-61-114 ~]$ kubectl run nginx-pod --image=nginx
pod/nginx-pod created
[cloudshell-user@ip-10-132-61-114 ~]$ kubectl get pods
NAME      READY   STATUS  RESTARTS   AGE
nginx-pod  1/1    Running 0          30s
[cloudshell-user@ip-10-132-61-114 ~]$ kubectl run nginx-pod2 --image=nginx -n dev
pod/nginx-pod2 created
[cloudshell-user@ip-10-132-61-114 ~]$ kubectl pods -n dev
error: unknown command "pods" for "kubectl"

Did you mean this?
  logs
[cloudshell-user@ip-10-132-61-114 ~]$ kubectl get pods -n dev
NAME      READY   STATUS  RESTARTS   AGE
nginx-pod2 1/1    Running 0          47s
[cloudshell-user@ip-10-132-61-114 ~]$ kubectl run nginx-pod3 --image=nginx --labels="app=my-nginx-app"
pod/nginx-pod3 created
[cloudshell-user@ip-10-132-61-114 ~]$ kubectl get pods
NAME      READY   STATUS  RESTARTS   AGE
nginx-pod  1/1    Running 0          4m2s
nginx-pod3 1/1    Running 0          7s
[cloudshell-user@ip-10-132-61-114 ~]$ kubectl get svc
NAME           TYPE        CLUSTER-IP      EXTERNAL-IP      PORT(S)        AGE
kubernetes     ClusterIP   <none>          <none>          443/TCP       76m
[cloudshell-user@ip-10-132-61-114 ~]$ kubectl expose pod nginx-pod3 --type=NodePort --port=80 --name=my-first-service
service/my-first-service exposed
[cloudshell-user@ip-10-132-61-114 ~]$ kubectl get svc
NAME           TYPE        CLUSTER-IP      EXTERNAL-IP      PORT(S)        AGE
kubernetes     ClusterIP   10.100.0.1     <none>          443/TCP       79m
my-first-service NodePort   10.100.235.38  <none>          80:32613/TCP  11s
[cloudshell-user@ip-10-132-61-114 ~]$
```

AWS CloudShell

Display a menuFeedback

© 2024, Amazon Web Services, Inc. or its affiliates. [Privacy](#) [Terms](#) [Cookie preferences](#)

A screenshot of a CloudShell terminal window titled "CloudShell". The terminal session is named "us-east-1". The user has run several commands related to Kubernetes, including:

```
ls-Cloudshell-user@ip-10-132-61-114 ~]$ kubectl get svc
NAME      TYPE      CLUSTER-IP   EXTERNAL-IP   PORT(S)   AGE
kubernetes   ClusterIP  10.100.0.1    <none>        443/TCP   76m
[cloudshell-user@ip-10-132-61-114 ~]$ kubectl expose pod nginx-pod3 --type=NodePort --port=80 --name=my-first-service
service/my-first-service exposed
[cloudshell-user@ip-10-132-61-114 ~]$ kubectl get svc
NAME      TYPE      CLUSTER-IP   EXTERNAL-IP   PORT(S)   AGE
kubernetes   ClusterIP  10.100.0.1    <none>        443/TCP   79m
my-first-service   NodePort  10.100.235.38  <none>        80:32613/TCP  11s
[cloudshell-user@ip-10-132-61-114 ~]$ kubectl gets pods -wide
error: unknown command "gets" for "kubectl"
Did you mean this?
  set
  get
[cloudshell-user@ip-10-132-61-114 ~]$ kubectl gets pods -wide
error: unknown command "gets" for "kubectl"
Did you mean this?
  set
  get
[cloudshell-user@ip-10-132-61-114 ~]$ kubectl get po -wide
error: unknown shorthand flag: 'i' in '-ide'
See 'kubectl get --help' for usage.
[cloudshell-user@ip-10-132-61-114 ~]$ kubectl get pods -o wide
error: unable to match a printer suitable for the output format "-wide", allowed formats are: custom-columns,custom-columns-file,go-template,go-template-file,json,jsonpath,jsonpath-as-json,jsonpath-file,name,tempplate,tempplatefile,wide,yaml
[cloudshell-user@ip-10-132-61-114 ~]$ kubectl get pods -o wide
NAME    READY  STATUS   RESTARTS  AGE     IP           NODE   NOMINATED-NODE  READINESS  GATES
nginx-pod  1/1   Running  0          8m38s  172.31.10.33  ip-172-31-7-191.ec2.internal  <none>       <none>
nginx-pod3 1/1   Running  0          4m43s  172.31.86.190  ip-172-31-80-89.ec2.internal  <none>       <none>
```

The terminal also shows a message about a missing command "gets" and a suggestion to use "set" or "get". It then shows a message about an unknown shorthand flag "i" and provides usage information for the "get" command. Finally, it shows a warning about the "wide" output format and lists the available formats.

Check the IP

A terminal window displaying two internal IP addresses:

```
NODE
ip-172-31-7-191.ec2.internal
ip-172-31-80-89.ec2.internal
window to reconnect and continue
```

Check Public IP

The screenshot shows the AWS EC2 Instances page. On the left, there's a navigation sidebar with links like EC2 Dashboard, EC2 Global View, Events, Instances, Images, Elastic Block Store, and Network & Security. The main area displays the instance summary for i-02edd018bb80dfa22. Key details shown include:

- Instance ID: i-02edd018bb80dfa22
- IPv6 address: -
- Hostname type: IP name: ip-172-31-80-89.ec2.internal
- Auto-assigned IP address: 44.201.184.194 [Public IP]
- IAM Role: EKS_NODE
- IMDSv2: Optional, EC2 recommends setting IMDSv2 to required | Learn more
- Private IPv4 address: 172.31.80.89
- Private IPv4 DNS name (IPv4 only): ip-172-31-80-89.ec2.internal
- Instance state: Running
- VPC ID: vpc-082939a3ed23f7468
- Subnet ID: subnet-01a8b4c483f84bbc5
- Instance ARN: arn:aws:ec2:us-east-1:637423339839:instance/i-02edd018bb80dfa22

On the right, there are sections for Private IPv4 addresses (172.31.80.89, 172.31.82.46), Public IPv4 DNS (ec2-44-201-184-194.compute-1.amazonaws.com), and Elastic IP addresses (none listed). Below the instance details, there are tabs for Details, Status and alarms, Monitoring, Security (which is selected), Networking, Storage, and Tags.

Open the https port in sg

The screenshot shows the AWS Security Groups page. The security group selected is sg-0d4f9494a30fb322, which is associated with the eks-cluster-sg-demo_cluster1-699838381 cluster. The Details section shows:

- Security group name: eks-cluster-sg-demo_cluster1-699838381
- Owner: 637423339839
- Security group ID: sg-0d4f9494a30fb322
- Description: EKS created security group applied to ENI that is attached to EKS Control Plane master nodes, as well as any managed workloads.
- VPC ID: vpc-082939a3ed23f7468
- Inbound rules count: 2 Permission entries
- Outbound rules count: 1 Permission entry

The Inbound rules section is expanded, showing two entries:

Name	Security group rule...	IP version	Type	Protocol	Port range
-	sgr-0fc29dad2206f32d2	IPv4	All TCP	TCP	0
-	sgr-0f87d24661a34fc00	-	All traffic	All	All

IP address will be like <http://44.201.184.194:32613/>

Public IP.Port No

You will get port no

```
kubernetes  CLUSTER-IP 10.100.0.1 <none> 443/TCP 76m
kubernetes  TYPE      CLUSTER-IP EXTERNAL-IP PORT(S) AGE
kubernetes  ClusterIP 10.100.0.1 <none> 443/TCP 76m
my-first-service  NodePort 10.100.235.38 <none> 80:32613/TCP 11s
[cloudshell-user@ip-10-132-61-114 ~]$ kubectl get svc
NAME      TYPE      CLUSTER-IP EXTERNAL-IP PORT(S) AGE
kubernetes  ClusterIP 10.100.0.1 <none> 443/TCP 79m
my-first-service  NodePort 10.100.235.38 <none> 80:32613/TCP 11s
[cloudshell-user@ip-10-132-61-114 ~]$ kubectl gets po -wide
error: unknown command "gets" for "kubectl"

Did you mean this?
```



Login to the Pods

```
CloudShell
us-east-1 + Actions ▾

ls
[cloudshell-user@ip-10-132-61-114 ~]$ kubectl expose pod nginx-pod3 --type=NodePort --port=80 --name=my-first-service
service/my-first-service exposed
[cloudshell-user@ip-10-132-61-114 ~]$ kubectl get svc
NAME      TYPE      CLUSTER-IP EXTERNAL-IP PORT(S) AGE
kubernetes  ClusterIP 10.100.0.1 <none> 443/TCP 79m
my-first-service  NodePort 10.100.235.38 <none> 80:32613/TCP 11s
[cloudshell-user@ip-10-132-61-114 ~]$ kubectl gets po -wide
error: unknown command "gets" for "kubectl"

Did you mean this?
  set
  get
[cloudshell-user@ip-10-132-61-114 ~]$ kubectl gets pods -wide
error: unknown command "gets" for "kubectl"

Did you mean this?
  set
  get
[cloudshell-user@ip-10-132-61-114 ~]$ kubectl get po -wide
error: unknown shorthand flag: 'i' in '-ide'
See 'kubectl get --help' for usage.
[cloudshell-user@ip-10-132-61-114 ~]$ kubectl get pods -o -wide
error: unable to match a printer suitable for the output format "-wide", allowed formats are: custom-columns,custom-columns-file,go-template,go-template-file,json,jsonpath,jsonpath-as-json,jsonpath-file,name,template,templatefile,wide,yaml
[cloudshell-user@ip-10-132-61-114 ~]$ kubectl get pods -o wide
NAME      READY STATUS RESTARTS AGE      IP           NODE      NOMINATED-NODE   READINESS GATES
nginx-pod  1/1   Running  0        8m38s  172.31.10.33 ip-172-31-7-191.ec2.internal <none>       <none>
nginx-pod3 1/1   Running  0        4m43s  172.31.86.190 ip-172-31-80-89.ec2.internal <none>       <none>
[cloudshell-user@ip-10-132-61-114 ~]$ kubectl exec -it nginx-pod3 -- bin/bash
root@nginx-pod3:~# ls
bin  boot  dev  docker-entrypoint.d  docker-entrypoint.sh  etc  home  lib  lib64  media  mnt  opt  proc  root  sbin  srv  sys  tmp  usr  var
root@nginx-pod3:~#
```