Getting Started with Amazon EKS

This getting started guide helps you to create all of the required resources to use Amazon EKS during the preview.

Amazon EKS Preview Prerequisites

Before you can create an Amazon EKS cluster, you must create an IAM role that Kubernetes can assume to create AWS resources. For example, when a load balancer is created, Kubernetes assumes the role to create an Elastic Load Balancing load balancer in your account. This only needs to be done one time and can be used for multiple EKS clusters.

You must also create a VPC with certain tagging and security group requirements. Although the VPC and security groups can be used for multiple EKS clusters, we recommend that you use a separate VPC for each EKS cluster to provide better network isolation.

This section also helps you to install a custom **kubectl** binary that is configured to work with Amazon EKS.

Optionally, you can download a custom version of the AWS CLI to use with Amazon EKS.

Create your Amazon EKS Service Role

1. Open the AWS CloudFormation console at https://us-west-2.console.aws.amazon.com/cloudformation.
2. From the navigation bar, select the **US West (Oregon)** region.

   Note

   At this time, Amazon EKS is only available in the **US West (Oregon)** region.

3. Choose **Create stack**.
4. For **Choose a template**, select **Specify an Amazon S3 template URL**.
5. Paste the following URL into the text area and choose **Next**:

   ```
   https://amazon-eks.s3-us-west-2.amazonaws.com/2018-04-04/amazon-eks-service-role.yaml
   ```

6. On the **Specify Details** page, fill out the parameters accordingly, and then choose **Next**.
   - **Stack name**: Choose a stack name for your AWS CloudFormation stack. For example, you can call it **eks-service-role**.

7. (Optional) On the **Options** page, tag your stack resources. Choose **Next**.
8. On the **Review** page, review your information and acknowledge that the stack might create IAM resources. Choose **Create**.
9. When your stack is created, select it in the console and choose **Outputs**.
10. Record the **RoleArn** for the role that was created. You need this when you create your EKS cluster.

Create your Amazon EKS Cluster VPC

1. Open the AWS CloudFormation console at https://us-west-2.console.aws.amazon.com/cloudformation.
2. From the navigation bar, select the **US West (Oregon)** region.

   Note

   At this time, Amazon EKS is only available in the **US West (Oregon)** region.

3. Choose **Create stack**.
4. For **Choose a template**, select **Specify an Amazon S3 template URL**.
5. Paste the following URL into the text area and choose **Next**:

   ```
   https://amazon-eks.s3-us-west-2.amazonaws.com/2018-04-04/amazon-eks-vpc-sample.yaml
   ```

6. On the **Specify Details** page, fill out the parameters accordingly, and then choose **Next**.
   - **Stack name**: Choose a stack name for your AWS CloudFormation stack. For example, you can call it **eks-vpc**.
   - **Cluster name**: Choose a name to use for your Amazon EKS cluster. **You must use the same name when you create the cluster later.**
   - **VPC CIDR block**: Choose a CIDR range for your VPC. You may leave the default value.
   - **Subnet 1 block**: Choose a CIDR range for subnet 1. You may leave the default value.
   - **Subnet 2 block**: Choose a CIDR range for subnet 2. You may leave the default value.
7. (Optional) On the **Options** page, tag your stack resources. Choose **Next**.
8. On the **Review** page, choose **Create**.
9. When your stack is created, select it in the console and choose **Outputs**.
10. Record the **SecurityGroups** value for the security group that was created. You need this when you create your EKS cluster; this security group is applied to the cross-account elastic network interfaces that are created in your subnets that allow the Amazon EKS control plane to communicate with your worker nodes.
11. Record the **SubnetIds** for the subnets that were created. You need this when you create your EKS cluster; these are the subnets that your worker nodes are launched into.

Download and Install the Custom kubectl Binary

Amazon EKS clusters require custom-built **kubectl** and **kubelet** binaries that include the Heptio Authenticator to allow IAM authentication for your Kubernetes cluster. This custom version can replace your existing **kubectl** binary, and you do not need to maintain multiple versions.

Note

This requirement will be obsolete in Kubernetes version 1.10.

To download and install the custom kubectl binary

1. Download the **kubectl** binary from Amazon S3:
   - **Linux**: https://amazon-eks.s3-us-west-2.amazonaws.com/2018-04-04/bin/linux/amd64/kubectl
   - **MacOS**: https://amazon-eks.s3-us-west-2.amazonaws.com/2018-04-04/bin/darwin/amd64/kubectl
   - **Windows**: https://amazon-eks.s3-us-west-2.amazonaws.com/2018-04-04/bin/windows/amd64/kubectl.exe

   Use the command below to download the binary, substituting the correct URL for your platform. The example below is for macOS clients.

   ```
   curl -o kubectl https://amazon-eks.s3-us-west-2.amazonaws.com/2018-04-04/bin/darwin/amd64/kubectl
   ```

2. Apply execute permissions to the binary.

   ```
   chmod +x ./kubectl
   ```

3. Copy the binary to a folder in your `$PATH`. If you have already installed **kubectl** (from Homebrew or Apt), then we recommend creating a `$HOME/bin/kubectl` and ensuring that `$HOME/bin` comes first in your `$PATH`.

   ```
   cp ./kubectl $HOME/bin/kubectl && export PATH=$HOME/bin:$PATH
   ```

4. (Optional) Add the `$HOME/bin` path to your shell initialization file so that it is configured when you open a shell.
   - For Bash shells on macOS:

     ```
     echo 'export PATH=$HOME/bin:$PATH' >> ~/.bash_profile
     ```

   - For Bash shells on Linux:

     ```
     echo 'export PATH=$HOME/bin:$PATH' >> ~/.bashrc
     ```

Download and Install the Custom AWS CLI

Because Amazon EKS is not yet generally available, it is not included in the AWS CLI. However, you can download and install a custom version of the AWS CLI model to use Amazon EKS.

You must first download and install the standard AWS CLI and then add the Amazon EKS extension model. If you do not already have the standard AWS CLI installed, see Installing the AWS Command Line Interface in the *AWS Command Line Interface User Guide*.

Note

Your system's Python version must be Python 3, or Python 2.7.9 or greater; otherwise, you will receive `hostname doesn't match` errors with AWS CLI calls to Amazon EKS. For more information, see What are "hostname doesn't match" errors? in the Python Requests FAQ.

To download and install the custom AWS CLI model for Amazon EKS

1. Download the custom model. from Amazon S3: https://amazon-eks.s3-us-west-2.amazonaws.com/2018-04-04/eks-2017-11-01.normal.json

   ```
   curl -O https://amazon-eks.s3-us-west-2.amazonaws.com/2018-04-04/eks-2017-11-01.normal.json
   ```

2. To install the custom model for Amazon EKS, run the following command:

   ```
   aws configure add-model --service-model file://eks-2017-11-01.normal.json --service-name eks
   ```

Step 1: Create Your Amazon EKS Cluster

Now you can create your Amazon EKS cluster.

Important

When the Amazon EKS cluster is created, the IAM user who creates the cluster is added to the Kubernetes RBAC authorization table as the administrator. Initially, only that IAM user can make calls to the master API using **kubectl**. Also, the custom version of **kubectl** uses the AWS SDK for Go to authenticate against your Amazon EKS cluster. If you use the console to create the cluster, you must ensure that the same IAM user credentials are in the AWS SDK credential chain when you are running **kubectl** commands on your cluster.

If you install and configure the AWS CLI, you can configure the IAM credentials for your user. These also work for **kubectl**. If the AWS CLI is configured properly for your user, then **kubectl** can find those credentials as well. For more information, see Configuring the AWS CLI in the *AWS Command Line Interface User Guide*.

To create your cluster with the console

1. Open the Amazon EKS console at https://console.aws.amazon.com/eks/home?region=us-west-2.

   Important

   You must use IAM user credentials for this step, **not root credentials**. If you create your Amazon EKS cluster using root credentials, you cannot authenticate to the cluster. For more information, see How Users Sign In to Your Account in the *IAM User Guide*.

2. Choose **Create cluster**.

   Note

   If your IAM user does not have administrative privileges, you must explicitly add permissions for that user to call the Amazon EKS API operations. For more information, see Creating Amazon EKS IAM Policies

3. On the **Create master cluster** page, fill in the following fields and then choose **Create**:
   - **Master cluster name**: A unique name for your cluster. This must be the cluster name you used in Create your Amazon EKS Cluster VPC.
   - **Kubernetes version**: The version of Kubernetes to use for your cluster. By default, the latest available version is selected.
   - **Role ARN**: The **RoleARN** value from the AWS CloudFormation output that you generated with Create your Amazon EKS Service Role.
   - **Cluster Subnets**: The **SubnetIds** values (comma-separated) from the AWS CloudFormation output that you generated with Create your Amazon EKS Cluster VPC.
   - **Security Groups**: The **SecurityGroups** value from the AWS CloudFormation output that you generated with Create your Amazon EKS Cluster VPC.
4. On the **Clusters** page, choose the name of your newly created cluster to view the cluster information.
5. The **Status** field shows **CREATING** until the cluster provisioning process completes. When your cluster provisioning is complete (usually less than 10 minutes), and note the **Master endpoint** value. This is the endpoint for your Kubernetes master that you use in your **kubectl** configuration.
6. Retrieve the `certificateAuthority.data` for your cluster. Currently, this value is not displayed in the console, and you must use the following AWS CLI command to retrieve the value.

   ```
   aws eks describe-cluster --region us-west-2 --cluster-name preview  --
   query cluster.certificateAuthority.data
   ```

To create your cluster with the AWS CLI

1. Create your cluster with the following command. Substitute your cluster name, the Amazon Resource Name (ARN) of your Amazon EKS service role that you created in

Create your Amazon EKS Service Role, and the subnet and security group IDs for the VPC you created in Create your Amazon EKS Cluster VPC.

Important

You must use IAM user credentials for this step, **not root credentials**. If you create your Amazon EKS cluster using root credentials, you cannot authenticate to the cluster. For more information, see How Users Sign In to Your Account in the *IAM User Guide*.

```
aws eks create-cluster --region us-west-2 --cluster-name preview --
role-arn arn:aws:iam::111122223333:role/eks-service-role-
AWSServiceRoleForAmazonEKS-EXAMPLEBKZRQR --subnets subnet-d474a49f
subnet-e794259e --security-groups sg-e829f296
```

Note

If your IAM user does not have administrative privileges, you must explicitly add permissions for that user to call the Amazon EKS API operations. For more information, see Creating Amazon EKS IAM Policies

Output:

```
{
    "cluster": {
        "status": "NEW",
        "subnets": [
            "subnet-d474a49f",
            "subnet-e794259e"
        ],
        "clusterName": "preview",
        "roleArn": "arn:aws:iam::111122223333:role/eks-service-role-
AWSServiceRoleForAmazonEKS-EXAMPLEBKZRQR",
        "desiredMasterVersion": "1.9",
        "certificateAuthority": {},
        "securityGroups": [
            "sg-e829f296"
        ],
        "createdAt": 1522779824913000
    }
}
```

2. Cluster provisioning usually takes less than 10 minutes. You can query the status of your cluster with the following command; when your cluster status is `ACTIVE`, you can proceed.

```
aws eks describe-cluster --region us-west-2 --cluster-name preview --
query cluster.status
```

3. When your cluster provisioning is complete, retrieve the `masterEndpoint` and `certificateAuthority.data` values with the following commands. These must be added to your **kubectl** configuration so that you can communicate with your cluster.

1. Retrieve the `masterEndpoint`.

   ```
   aws eks describe-cluster --region us-west-2 --cluster-name
   preview  --query cluster.masterEndpoint
   ```

2. Retrieve the `certificateAuthority.data`.

   ```
   aws eks describe-cluster --region us-west-2 --cluster-name
   preview  --query cluster.certificateAuthority.data
   ```

Step 2: Configure `kubectl` for Amazon EKS

In this section, you create a `kubeconfig` file for your cluster. The code block below shows the `kubeconfig` elements to add to your configuration. If you have an existing configuration and you are comfortable working with `kubeconfig` files, you can merge these elements into your existing setup. Be sure to replace the *<endpoint-url>* value with the full endpoint URL (for example, *https://EXAMPLE_MASTER_ENDPOINT.yl4.us-west-2.eks.amazonaws.com*) that was created for your cluster, replace the *<base64-encoded-ca-cert>* with the `certificateAuthority.data` value you retrieved earlier, and replace the *<cluster-name>* with your cluster name.

```
apiVersion: v1
clusters:
- cluster:
    server: <endpoint-url>
    certificate-authority-data: <base64-encoded-ca-cert>
  name: kubernetes
contexts:
- context:
    cluster: kubernetes
    user: aws
  name: aws
current-context: aws
kind: Config
preferences: {}
users:
- name: aws
  user:
    auth-provider:
      config:
        cluster-id: <cluster-name>
      name: aws
```

If you do not have an existing configuration, or to add the Amazon EKS cluster without modifying your existing configuration files, you can use the following procedure to add the Amazon EKS cluster to your configuration.

To create your `kubeconfig` file

1. Create the default **kubectl** folder if it does not already exist.

```
mkdir -p ~/.kube
```

2. Open your favorite text editor and copy the above `kubeconfig` code block into it.
3. Replace the *`<endpoint-url>`* with the endpoint URL that was created for your cluster.
4. Replace the *`<base64-encoded-ca-cert>`* with the `certificateAuthority.data` that was created for your cluster.
5. Replace the *`<cluster-name>`* with your cluster name.
6. Save the file to the default **kubectl** folder, with your cluster name in the file name. For example, if your cluster name is *`preview`*, save the file to `~/.kube/config-`*`preview`*.
7. Add that file path to your `KUBECONFIG` environment variable so that **kubectl** knows where to look for your cluster configuration.

   ```
   export KUBECONFIG=$KUBECONFIG:~/.kube/config-preview
   ```

8. (Optional) Add the configuration to your shell initialization file so that it is configured when you open a shell.
   - For Bash shells on macOS:

     ```
     echo 'export KUBECONFIG=$KUBECONFIG:~/.kube/config-preview' >>
     ~/.bash_profile
     ```

   - For Bash shells on Linux:

     ```
     echo 'export KUBECONFIG=$KUBECONFIG:~/.kube/config-preview' >>
     ~/.bashrc
     ```

9. Test your configuration.

   ```
   kubectl get all
   ```

   Note

   If you receive the error `No Auth Provider found for name "aws"`, you are not using the custom **kubectl** required for Amazon EKS during the preview. For more information, see Download and Install the Custom **kubectl** Binary.

   Output:

   ```
   NAME             TYPE        CLUSTER-IP   EXTERNAL-IP   PORT(S)   AGE
   svc/kubernetes   ClusterIP   10.100.0.1   <none>        443/TCP   1m
   ```

Step 3: Launch and Configure Amazon EKS Worker Nodes

Now that your VPC and Kubernetes master are created, you can launch and configure your worker nodes.

Important

Amazon EKS worker nodes are standard Amazon EC2 instances, and you are billed for them based on normal EC2 On-Demand prices. For more information, see Amazon EC2 Pricing.

To launch your worker nodes

1. Open the AWS CloudFormation console at https://console.aws.amazon.com/cloudformation.
2. From the navigation bar, select the **US West (Oregon)** region.

   Note

   Amazon EKS is only available in the **US West (Oregon)** region at this time.

3. Choose **Create stack**.
4. For **Choose a template**, select **Specify an Amazon S3 template URL**.
5. Paste the following URL into the text area and choose **Next**.

   ```
   https://amazon-eks.s3-us-west-2.amazonaws.com/2018-04-04/amazon-eks-nodegroup.yaml
   ```

6. On the **Specify Details** page, fill out the parameters accordingly, and choose **Next**.
   - **Stack name**: Choose a stack name for your AWS CloudFormation stack. For example, you can call it ***<cluster-name>*-worker-nodes**.
   - **ClusterName**: Enter the name that you used when you created your Amazon EKS cluster.

     Important

     This name must exactly match the name you used in Step 1: Create Your Amazon EKS Cluster; otherwise, your worker nodes cannot join the cluster.

   - **ClusterControlPlaneSecurityGroup**: Choose the **SecurityGroups** value from the AWS CloudFormation output that you generated with Create your Amazon EKS Cluster VPC.
   - **NodeGroupName**: Enter a name for your node group that is included in your Auto Scaling node group name.
   - **NodeAutoScalingGroupMinSize**: Enter the minimum number of nodes that your worker node Auto Scaling group can scale in to.
   - **NodeAutoScalingGroupMaxSize**: Enter the maximum number of nodes that your worker node Auto Scaling group can scale out to.

     Note

     We ask that you limit the number of worker nodes in your cluster no more than 100 during the preview.

   - **NodeInstanceType**: Choose an instance type for your worker nodes.

- **NodeImageId**: Enter the current Amazon EKS worker node AMI ID (**ami-228dee5a**).

  Note

  The Amazon EKS worker node AMI is based on Amazon Linux 2, with a custom `kubelet` and `kube-proxy` built in. You can track security or privacy events for Amazon Linux 2 at the Amazon Linux Security Center or subscribe to the associated RSS feed. Security and privacy events include an overview of the issue, what packages are affected, and how to update your instances to correct the issue.

- **KeyName**: Enter the name of an Amazon EC2 SSH key pair that you can use to connect using SSH into your worker nodes with after they launch.
- **VpcId**: Enter the ID for the VPC that you created in Create your Amazon EKS Cluster VPC.
- **Subnets**: Choose the subnets that you created in Create your Amazon EKS Cluster VPC.

7. On the **Options** page, you can choose to tag your stack resources. Choose **Next**.
8. On the **Review** page, review your information, acknowledge that the stack might create IAM resources, and then choose **Create**.
9. When your stack has finished creating, select it in the console and choose the **Outputs** tab.
10. Record the **NodeInstanceRole** for the node group that was created. You need this when you configure your Amazon EKS worker nodes.

To enable worker nodes to join your cluster

1. Download, edit, and apply the AWS authenticator configuration map.
   1. Download the configuration map.

      ```
      curl -O https://amazon-eks.s3-us-west-2.amazonaws.com/2018-04-04/aws-auth-cm.yaml
      ```

   2. Open the file with your favorite text editor, replace the *<ARN of instance role (not instance profile)>* snippet with the **NodeInstanceRole** value that you recorded in the previous procedure, and save the file.

      Important

      Do not modify any other lines in this file.

      ```
      apiVersion: v1
      kind: ConfigMap
      metadata:
        name: aws-auth
        namespace: default
      data:
      ```

```
mapRoles: |
  - rolearn: <ARN of instance role (not instance profile)>
    username: system:node:{{EC2PrivateDNSName}}
    groups:
      - system:bootstrappers
      - system:nodes
      - system:node-proxier
```

3. Apply the configuration. (This command may take a few minutes to finish)

   ```
   kubectl apply -f aws-auth-cm.yaml
   ```

   Note

   If you receive the error `No Auth Provider found for name "aws"`, you are not using the custom **kubectl** required for Amazon EKS during the preview. For more information, see Download and Install the Custom **kubectl** Binary.

2. Watch the status of your nodes and wait for them to reach the `Ready` status.

   ```
   kubectl get nodes --watch
   ```

Step 4: Launch a Guest Book Application

In this section, you create a sample guest book application to test your new cluster.

Note

For more information about setting up the guest book example, see https://github.com/kubernetes/examples/blob/master/guestbook-go/README.md in the Kubernetes documentation.

To create your guest book application

1. Create the Redis master replication controller.

   ```
   kubectl apply -f
   https://raw.githubusercontent.com/kubernetes/kubernetes/v1.9.2/examples
   /guestbook-go/redis-master-controller.json
   ```

   Note

   If you receive the error `No Auth Provider found for name "aws"`, you are not using the custom **kubectl** required for Amazon EKS during the preview. For more information, see Download and Install the Custom **kubectl** Binary.

   Output:

```
replicationcontroller "redis-master" created
```

2. Create the Redis master service.

   kubectl apply -f https://raw.githubusercontent.com/kubernetes/kubernetes/v1.9.2/examples/guestbook-go/redis-master-service.json

   Output:

   ```
   service "redis-master" created
   ```

3. Create the Redis slave replication controller.

   kubectl apply -f https://raw.githubusercontent.com/kubernetes/kubernetes/v1.9.2/examples/guestbook-go/redis-slave-controller.json

   Output:

   ```
   replicationcontroller "redis-slave" created
   ```

4. Create the Redis slave service.

   kubectl apply -f https://raw.githubusercontent.com/kubernetes/kubernetes/v1.9.2/examples/guestbook-go/redis-slave-service.json

   Output:

   ```
   service "redis-slave" created
   ```

5. Create the guestbook replication controller.

   kubectl apply -f https://raw.githubusercontent.com/kubernetes/kubernetes/v1.9.2/examples/guestbook-go/guestbook-controller.json

   Output:

   ```
   replicationcontroller "guestbook" created
   ```

6. Create the guestbook service.

   kubectl apply -f https://raw.githubusercontent.com/kubernetes/kubernetes/v1.9.2/examples/guestbook-go/guestbook-service.json

   Output:

```
service "guestbook" created
```

7. Query the services in your cluster and wait until the **External IP** column for the `guestbook` service is populated.

   Note

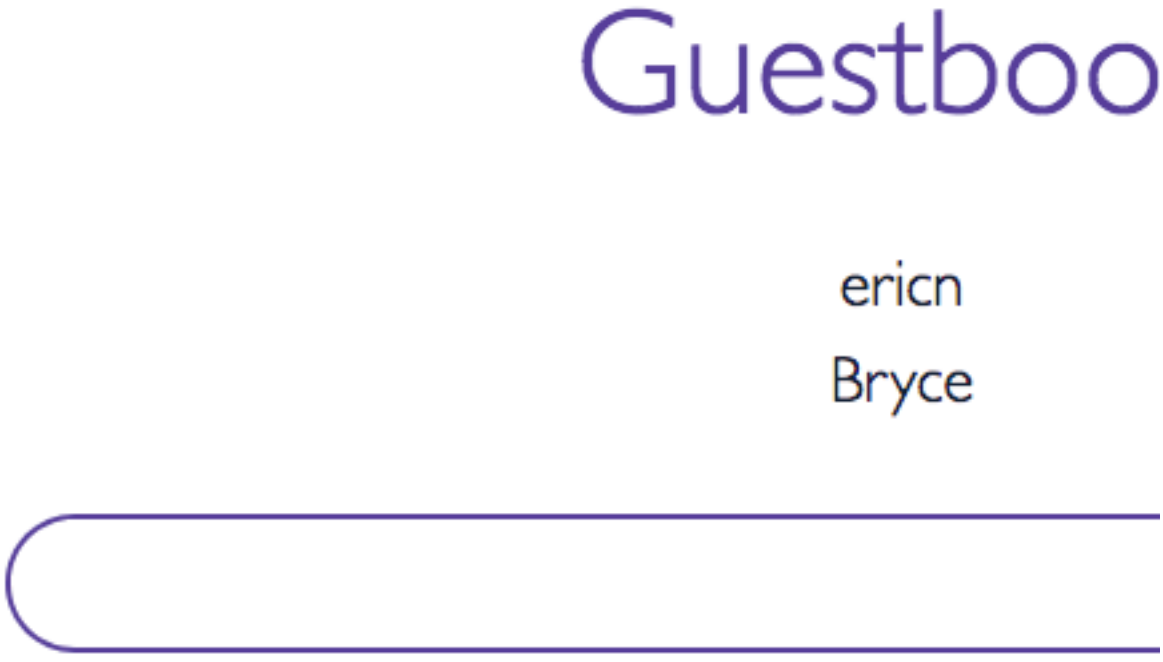
   It may take several minutes before the IP address is available.

   ```
   kubectl get services -o wide
   ```

8. After your external IP address is available, point a web browser to that address at port 3000 to view your guest book. For example, *`http://a7a95c2b9e69711e7b1a3022fdcfdf2e-1985673473.us-west-2.elb.amazonaws.com:3000`*

   Note

   It may take several minutes for DNS to propagate and for your guest book to show up.



   Important

If you are unable to connect to the external IP address with your browser, be sure that your corporate firewall is not blocking non-standards ports, like 3000. You can try switching to a guest network to verify.

Step 5: Cleaning Up Guest Book Objects

When you are finished experimenting with your guest book application, you should clean up the resources that you created for it. The following command deletes all of the services and replication controllers for the guest book application:

```
kubectl delete rc/redis-master rc/redis-slave rc/guestbook svc/redis-master
svc/redis-slave svc/guestbook
```

Note

If you receive the error `No Auth Provider found for name "aws"`, you are not using the custom **kubectl** required for Amazon EKS during the preview. Be sure that the custom **kubectl** for Amazon EKS is first in your `PATH`. For more information, see Download and Install the Custom **kubectl** Binary.

 **Javascript is disabled or is unavailable in your browser.**

To use the AWS Documentation, Javascript must be enabled. Please refer to your browser's Help pages for instructions.

Document Conventions
« Previous Next »

On this page: