Red Hat OpenShift Service on AWS 4

Application development

Configuring Red Hat OpenShift Service on AWS for your applications
Configuring Red Hat OpenShift Service on AWS for your applications
Abstract

This document provides information about configuring Red Hat OpenShift Service on AWS (ROSA) for your application deployments. This includes setting up custom wildcard domains.
# Table of Contents

CHAPTER 1. DEPLOYMENTS ................................................................. 3

1.1. CONFIGURING CUSTOM DOMAINS FOR APPLICATIONS 3

1.1.1. Configuring custom domains for applications 3
1.1. Configuring custom domains for applications

Custom domains are specific wildcard domains that can be used with Red Hat OpenShift Service on AWS applications. The top-level domains (TLDs) are owned by the customer that is operating the Red Hat OpenShift Service on AWS cluster. The Custom Domains Operator sets up a new ingresscontroller with a custom certificate as a second day operation. The public DNS record for this ingresscontroller can then be used by an external DNS to create a wildcard CNAME record for use with a custom domain.

**NOTE**

Custom API domains are not supported because Red Hat controls the API domain. However, customers can change their application domains. For private custom domains with a private IngressController, set `.spec.scope` to `Internal` in the `CustomDomain` CR.

**Prerequisites**

- A user account with `dedicated-admin` privileges
- A unique wildcard domain, such as `*.apps.<company_name>.io`
- A wildcard custom certificate, such as `CN=*.apps.<company_name>.io`
- Access to a cluster with the latest version of the `oc` CLI installed

**IMPORTANT**

Do not use the reserved names `default` or `apps*`, such as `apps` or `apps2`, in the `metadata/name:` section of the `CustomDomain` CR.

**Procedure**

1. Create a new TLS secret from a private key and a public certificate, where `fullchain.pem` and `privkey.pem` are your public or private wildcard certificates.

   **Example**

   ```bash
   $ oc create secret tls <name>-tls --cert=fullchain.pem --key=privkey.pem -n <my_project>
   ```

2. Create a new `CustomDomain` custom resource (CR):

   **Example `<company_name>-custom-domain.yaml`**

   ```yaml
   apiVersion: managed.openshift.io/v1alpha1
   kind: CustomDomain
   metadata:
     name: <company_name>
   spec:
     domain: apps.<company_name>.io
   ```
scope: External
certificate:
  name: <name>-tls
  namespace: <my_project>

1. The custom domain.
2. The secret created in the previous step.

3. Apply the CR:

   **Example**

   ```
   $ oc apply -f <company_name>-custom-domain.yaml
   ```

4. Get the status of your newly created CR:

   ```
   $ oc get customdomains
   ```

   **Example output**

   ```
   NAME     ENDPOINT                          DOMAIN                        STATUS
   <company_name>     xxrywp.<company_name>.cluster-01.opln.s1.openshiftapps.com
   *.apps.<company_name>.io     Ready
   ```

5. Using the endpoint value, add a new wildcard CNAME recordset to your managed DNS provider, such as Route53, Azure DNS, or Google DNS.

   **Example**

   ```
   *.apps.<company_name>.io -> xxrywp.<company_name>.cluster-01.opln.s1.openshiftapps.com
   ```

6. Create a new application and expose it:

   **Example**

   ```
   $ oc new-app --docker-image=docker.io/openshift/hello-openshift -n my-project
   $ oc create route edge --service=hello-openshift hello-openshift-tls --hostname hello-openshift-tls-my-project.apps.acme.io -n my-project
   $ oc get route -n my-project
   $ curl https://hello-openshift-tls-my-project.apps.<company_name>.io
   Hello OpenShift!
   ```