OpenShift Dedicated 4

Applications

Configuring OpenShift Dedicated for your applications
OpenShift Dedicated 4 Applications

Configuring OpenShift Dedicated for your applications
Abstract

This document provides information about configuring OpenShift Dedicated for your application deployments. This includes setting up custom wildcard domains.
# Table of Contents

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

1.1. CUSTOM DOMAINS FOR APPLICATIONS 3
   1.1.1. Configuring custom domains for applications 3
   1.1.2. Renewing a certificate for custom domains 5
1. CUSTOM DOMAINS FOR APPLICATIONS

You can configure a custom domain for your applications. Custom domains are specific wildcard domains that can be used with OpenShift Dedicated applications.

1.1. Configuring custom domains for applications

The top-level domains (TLDs) are owned by the customer that is operating the OpenShift Dedicated cluster. The Custom Domains Operator sets up a new ingress controller with a custom certificate as a second day operation. The public DNS record for this ingress controller 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 domain or wildcard domain, such as *.apps.<company_name>.io
- A custom certificate or 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**

   ```sh
   $ 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:
   ```
The custom domain.

The type of load balancer for your custom domain. This type can be the default **classic** or **NLB** if you use a network load balancer.

The secret created in the previous step.

Optional: Filters the set of routes serviced by the CustomDomain ingress. If no value is provided, the default is no filtering.

Optional: Filters the set of namespaces serviced by the CustomDomain ingress. If no value is provided, the default is no filtering.

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**

<table>
<thead>
<tr>
<th>NAME</th>
<th>ENDPOINT</th>
<th>DOMAIN</th>
<th>STATUS</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;company_name&gt;</td>
<td>xxrywp.&lt;company_name&gt;.cluster-01.opln.s1.openshiftapps.com</td>
<td>.apps.&lt;company_name&gt;.io     Ready</td>
<td></td>
</tr>
</tbody>
</table>

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

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

Troubleshooting

- Error creating TLS secret
- Troubleshooting: CustomDomain in NotReady state

1.1.2. Renewing a certificate for custom domains

You can renew certificates with the Custom Domains Operator (CDO) by using the `oc` CLI tool.

Prerequisites

- You have the latest version `oc` CLI tool installed.

Procedure

1. Create new secret

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

2. Patch CustomDomain CR

```shell
$ oc patch customdomain <company_name> --type='merge' -p '{"spec":{"certificate":{"name":"<secret-new>"}}}'}
```

3. Delete old secret

```shell
$ oc delete secret <secret-old> -n <my_project>
```

Troubleshooting

- Error creating TLS secret