Red Hat Developer Hub 0.2

Azure Container Registry plugin for Backstage

The Azure Container Registry (ACR) plugin for Backstage
Red Hat Developer Hub 0.2 Azure Container Registry plugin for Backstage

The Azure Container Registry (ACR) plugin for Backstage
Abstract

The ACR plugin displays information about your container images available in the Azure Container Registry.
Table of Contents

CHAPTER 1. AZURE CONTAINER REGISTRY PLUGIN FOR BACKSTAGE ........................................ 3
  1.1. FOR ADMINISTRATORS .................................................................................. 3
    1.1.1. Installing and configuring the ACR plugin .............................................. 3
  1.2. FOR USERS ................................................................................................... 4
    1.2.1. Using the ACR plugin in Backstage ....................................................... 4
CHAPTER 1. AZURE CONTAINER REGISTRY PLUGIN FOR BACKSTAGE

The Azure Container Registry (ACR) plugin displays information about your container images available in the Azure Container Registry.

1. FOR ADMINISTRATORS

1.1. Installing and configuring the ACR plugin

The Red Hat Plug-ins for Backstage (RHPIB) packages are hosted in a separate NPM registry, which is maintained by Red Hat. To use these packages, you must adjust your NPM configuration to pull the @redhat scoped packages:

```bash
# update your .npmrc or .yarnrc file
yarn config set "@redhat:registry" https://npm.registry.redhat.com
# then pull a package
yarn add @redhat/backstage-plugin-quay
```

For more information, see npm docs.

Creating a .npmrc file ensures that all the packages are scoped under @redhat and are fetched from Red Hat’s NPM registry, while the rest dependencies remain sourced from other registry.

Procedure

1. Run the following command to install the ACR plugin:

```
yarn workspace app add @redhat/backstage-plugin-acr
```

2. Set the proxy to the desired ACR server in the app-config.yaml file as follows:

```
# app-config.yaml
proxy:
  '/acr/api':
    target: 'https://mycontainerregistry.azurecr.io/acr/v1/
changeOrigin: true
headers:
  # If you use Bearer Token for authorization, please replace the 'Basic' with 'Bearer' in the following line.
  Authorization: 'Basic ${ACR_AUTH_TOKEN}'
  # Change to "false" in case of using self hosted artifactory instance with a self-signed certificate
  secure: true
```

3. Set the authorization using one of the following options:
   
   - Basic authorization:
     
     - Navigate to the ACR portal and go to the Access Keys tab.

     - Retrieve the username and password of the Admin user and use the [Basic Auth Header Generator tool](https://www.debugbear.com/basic-auth-header-generator) or run
echo printf '<username>:<password>' | base64 in a terminal to convert the credentials to a basic token.

- Set the generated token as `ACR_AUTH_TOKEN` in environment variables.

- **OAuth2:**
  - Generate bearer access token using the process described in Authenticate with an Azure Container Registry.
  - You can generate a bearer token using your basic authorization token, for example:

```
curl --location 'https://<yourregistry>.azurecr.io/oauth2/token?scope=repository%3A%3A*&service=<yourregistry>.azurecr.io' \--header 'Authorization: Basic <basic_token>'
```

- Set the generated token as `ACR_AUTH_TOKEN` in environment variables. Also ensure that you replace the `Basic` with `Bearer` in the `app-config.yaml` file.

4. Enable an additional tab on the entity view page using the `packages/app/src/components/catalog/EntityPage.tsx` file as follows:

```typescript
/* highlight-add-start */
import { AcrPage, isAcrAvailable } from '@redhat/backstage-plugin-acr';

/* highlight-add-end */

const serviceEntityPage = (
  <EntityPageLayout>
    // ...
    { /* highlight-add-start */
      <EntityLayout.Route path="/acr" title="ACR">
        <Grid container spacing={3} alignItems="stretch">
          <EntitySwitch>
            <EntitySwitch.Case if={e => Boolean(isAcrAvailable(e))}>
              <Grid item sm={12}>
                <AcrPage />
              </Grid>
            </EntitySwitch.Case>
          </EntitySwitch>
        </Grid>
      </EntityLayout.Route>
    { /* highlight-add-end */
  </EntityPageLayout>

);```

5. Annotate your entity using the following annotations:

```json
metadata:
  annotations:
    'azure-container-registry/repository-name': '<REPOSITORY-NAME>',
```

### 1.2. FOR USERS

#### 1.2.1. Using the ACR plugin in Backstage
ACR is a front-end plugin that enables you to view information about the container images from your Azure Container Registry in Backstage.

**Prerequisites**

- Your Backstage application is installed and running.
- You have installed the ACR plugin. For installation instructions, see Section 1.1.1, “Installing and configuring the ACR plugin”.

**Procedure**

1. Open your Backstage application and select a component from the Catalog page.
2. Go to the ACR tab.

![Image of the ACR tab in the Backstage UI](image)

The ACR tab in the Backstage UI contains a list of container images and related information, such as TAG, CREATED, LAST MODIFIED, and MANIFEST.