Node specific secrets

Status: In Progress

Sponsor User: *IBM*

Date of Submission: 29 Mar 2023

Submitted by: David Booz

Affiliation(s): IBM

<Please fill out the above fields, and the Overview, Design and User Experience sections below for an initial review of the proposed feature.>

Scope and Signoff: *(to be filled out by Chair)*

Overview

<Briefly describe the problem being solved, not how the problem is solved, just focus on the problem. Think about why the feature is needed, and what is the relevant context to understand the problem.>

Open Horizon has had support for secret deployment for some time now. The current capability enables a service developer to express a requirement for a secret, and enables a service deployer to bind secrets to a service at service deployment time. In this case, it's the same secret that is deployed to all of the nodes where the the service is placed. This works well for large scale application deployment where node specific secrets are not really required. However, if the service(s) being deployed is configured to execute as a privileged container, there are more strict security requirements which result in the need for specific (i.e. different) secrets to be deployed to each node where the service runs. This can be accomplished in Open Horizon by creating a deployment policy that deploys a service to one and only one node, but that approach does not scale beyond a handful of nodes. Open Horizon needs the capability to deploy node specific secrets that scales up easily.

Design

<Describe how the problem is fixed. Include all affected components. Include diagrams for clarity. This should be the longest section in the document. Use the sections below to call out specifics related to each aspect of the overall system, and refer back to this section for context. Provide links to any relevant external information.>

This design uses the configuration by convention approach to provide an optional and scalable capability. In short, the design enables an org admin to name a secret that complies with a pattern (defined in this design) identifying a specific node, so that when the secret is deployed, it is only deployed to the identified node. Nodes for which there is no correspondingly named secret will instead receive the org or user secret as they would have before this capability was added. For example, a deployer binds a secret named myGenericSecret to the service using a deployment policy. The deployer has also created a secret named "node/node1/myGenericSecret" which is specific to node1. When the policy is published, the agbot determines that the service should be placed on node1, node2, and node3. The agreement proposal sent to node1 contains the contents of node/node1/myGenericSecret. The agreement proposal sent to node 2 and 3 contains the contents of myGenericSecret. The removal of the node specific secret from the system would cause the agbot to update the agreement with node1, resulting in the delivery of myGenericSecret to node1.

This design enables deployers to opt-in to the use of this new capability without affecting existing usage behavior. It also supports the current scalability of service placement in OH.

To enable this capability the following changes are introduced. The secret binding in a pattern or deployment policy is extended with the "enableNodeLevelSecrets" flag that enables node specific secrets:

```
 "secretBinding": [
   {"serviceOrgid": "$HZN_ORG_ID",
    "serviceUrl": "$SERVICE_NAME",
    "serviceVersionRange": "0.0.1",
    "enableNodeLevelSecrets": <boolean>,
    "secrets": [
      {"cloudAIservice": "aitoken"}, <– org level secret
      {"cloudsqlservice": "user/{userid}/sqltoken"} <– user level secret
    ]
   }
 ]
```

The flag "enableNodeLevelSecrets" is a boolean typed field and is false by default. When set to true, it causes the agbot to search for node specific secrets when making agreements. The agbot will limit it's search to the secrets specified in the "secrets" array of the same serviceBinding element where the "enableNodeLevelSecrets" flag is set. Further, this design extends the secretBinding array to support more than 1 element that references the same service. This enables the deployer to enable node specific secrets for some service secrets but disable it for others.

In order for a secret to be considered a node specific secret, it's name must adhere to the following pattern:

- For org wide secrets; node/<node-id>/<secret-name>, where the deployer specifies <secret-name> in the deployment policy.
- For user specific secrets; user/<user-id>/node/<node-id>/<secret-name>, where the deployer specifies user/<user-id/<secret-name> in the deployment policy.

The supported format for <secret-name> is not changed in this design.

The permissions required to access a node specific secret are not changed by this design. That is, for org wide node specific secrets, the required permissions are the same as for an org wide secret (in terms of Add, Delete, List and Read).

User Experience

<Describe which user roles are related to the problem AND the solution, e.g. admin, deployer, node owner, etc. If you need to define a new role in your design, make that very clear. Remember this is about what a user is thinking when interacting with the system before and after this design change. This section is not about a UI, it's more abstract than that. This section should explain all the aspects of the proposed feature that will surface to users.>

As a service deployer, I want to configure a specific secret for each node on which a service is placed.

As a service deployer, I want to configure a specific secret to some of the nodes where a given service is placed.

As a service deployer, I want to roll out node specific secrets to the nodes in small batches.

Command Line Interface

<Describe any changes to the hzn CLI, including before and after command examples for clarity. Include which users will use the changed CLI. This section should flow very naturally from the User Experience section.>

hzn exchange pattern publish and **hzn exchange deployment policy add**

The syntax and flags supported by these commands is not changed, but the implementation is updated to support the changes described above. Notably, validation of the secretBinding section should produce a warning to create node specific secrets if the specified secret is not found AND "enableNodeLevelSecrets" is true.

hzn secretsmanager secrete add/list/read/remove

The syntax and flags supported by these commands is not changed, but the implementation is updated to support node specific secret names.

hzn deploycheck

The syntax and flags supported by these commands is not changed, but the implementation is updated to check for node specific secrets.

External Components

<Describe any new or changed interactions with components that are not the agent or the management hub.>

The vault auth plugin is updated to support node specific secrets.

Affected Components

<List all of the internal components (agent, MMS, Exchange, etc) which need to be updated to support the proposed feature. Include a link to the github epic for this feature (and the epic should contain the github issues for each component).>

Agent, Agbot, Exchange, CLI, vault auth plugin

Security

<Describe any related security aspects of the solution. Think about security of components interacting with each other, users interacting with the system, components interacting with external systems, permissions of users or components>

None. Node specific secrets are secured in the same way that org wide and user specific secrets are secured before this new capbility.

APIs

<Describe and new/changed/deprecated APIs, including before and after snippets for clarity. Include which components or users will use the APIs.>

Exchange API is updated to support the new deployment policy and pattern schema for the "secretBinding" section.

Agbot secure API is updated to support node specific secrets:

"/org/{org}/secrets/node/{node}" the LIST API is added

"/org/{org}/secrets/node/{node}/{secret:[\w\/\-]+}" the "GET", "LIST", "PUT", "POST", "DELETE" APIs are added

"/org/{org}/secrets/user/{user}/node/{node}" the LIST API is added

"/org/{org}/secrets/user/{user}/node/{node}/{secret:[\w\/\-]+}" the "GET", "LIST", "PUT", "POST", "DELETE" is added

Build, Install, Packaging

<Describe any changes to the way any component of the system is built (e.g. agent packages, containers, etc), installed (operators, manual install, batch install, SDO), configured, and deployed (consider the hub and edge nodes).>

None

Documentation Notes

<Describe the aspects of documentation that will be new/changed/updated. Be sure to indicate if this is new or changed doc, the impacted artifacts (e.g. technical doc, website, etc) and links to the related doc issue(s) in github.>

Documentation is needed to:

- Describe the feature in general, as per the design section and the user experience section.
- Describe the "enableNodeLevelSecrets" flag in deployment policies and patterns
- Illustrate an example using a node specific secret

Test

<Summarize new automated tests that need to be added in support of this feature, and describe any special test requirements that you can foresee.>

The e2edev tests are updated to support:

- use of node specific secrets
- a mix of node specific and non-node-specific secret deployments for the same service deployment
- Adding a node specific secret after a service is deployed
- Removing a node specific secret after a service is deployed