PURESTORAGE

Deploying Pure Service Orchestrator in an IBM Cloud Private

Simon Dodsley, Director of New Stack Technologies

Version 1.0, 26 November 2018

Contents

Notices

Pure Storage, Inc.
650 Castro Street
Mountain View, CA 94041

Scope

This white paper describes the implementation of the Pure Service Orchestrator in IBM® Cloud™ Private, an on-premises application platform based on Kubernetes. It is designed for developers, system architects and storage administrators who are evaluating or deploying Pure Service Orchestrator in an IBM Cloud Private deployment.

It is assumed that the reader understands how to deploy IBM Cloud Private, understands Kubernetes, Helm and other container platform components, as well as has access to Pure Storage platforms that can be used to provide persistent storage to IBM Cloud Private.

This white paper will not cover the deployment or configuration of IBM Cloud Private or the configuration of Kubernetes client nodes to communicate with the main application platform.

IBM Cloud Private

IBM Cloud Private is an application platform for developing and managing on-premises, containerized applications. It is an integrated environment for managing containers that includes the container orchestrator Kubernetes, a private image registry, a management console and monitoring frameworks.

IBM Cloud Private delivers a customer-managed container solution for enterprises. It is also available in a community edition, IBM Cloud Private-CE, which provides a limited offering that is available at no charge and is ideal for test environments.

Pure Service Orchestrator

Pure Service Orchestrator integrates seamlessly with container orchestration frameworks like Docker, Kubernetes, and Mesosphere, and functions as the control plane virtualization layer for your containerized environment. It allows developers to move from consuming storage-as-a-device to consuming storage-as-a-service.

Pure Service Orchestrator was designed to provide a similar experience to your developers that they expect they can only get from the public cloud. Pure Service Orchestrator can provide a seamless container-as-a-service environment that is:

- **Simple, Automated and Integrated:** Automatically provisions on-demand storage via policy, and seamlessly integrates with your environment, thereby enabling DevOps and Developer friendly ways to consume storage
- **Elastic:** Allows you to start small and scale your storage environment with ease and flexibility, mixing and matching varied configurations as your environment grows
- **Multi-protocol:** Supports both file and block protocols
- **Enterprise-grade:** Delivers the same Tier1 resilience, reliability and protection that your mission-critical applications depend upon, for stateful applications in your Kubernetes clusters

- **Shared:** Makes shared storage a viable and preferred architectural choice for the next generation, containerized data centers by delivering a vastly superior experience relative to direct-attached storage alternatives.

Installing Pure Service Orchestrator in IBM Cloud Private

It is assumed that IBM Cloud Private has been deployed and that you have configured a client node to communicate with the main deployment using `kubectl` and that the IBM Cloud Private CLI, `cloudctl`, has been installed on the client node. You must also have a Helm installed on your client node to be able to use the Helm chart used by Pure Service Orchestrator in its deployment.

Prerequisites

There are two actions that need to be performed on every IBM Cloud Private worker node in your cluster before performing the installation of Pure Service Orchestrator from the client node:

- ensure the latest multipath software package is installed and enabled
- ensure the `/etc/multipath.conf` file exists and contains the Pure Storage stanza as described in the Pure Storage Linux Best Practices.

PSO Installation

All commands for this installation are performed from a client node that has been configured for access to your IBM Cloud Private deployment.

Note that Pure Service Orchestrator manages the installation of all required files across all worker nodes in your IBM Cloud Private deployment using a DaemonSet to perform the cross-node installation. The DaemonSet runs a pod on each node in the cluster, which copies the required files in the right path on the host for the kubelet to access. It will keep the config updated and ensure that files are installed safely.

Installation of the Pure Service Orchestrator for IBM Cloud Private requires that you have a Helm client installed on client node.

- Add the `pure` repo to Helm

```
# helm --tls repo add pure http://purestorage.github.io/helm-charts
# helm --tls repo update
# helm --tls search pure-k8s-plugin
```

- Ensure you have authenticated your client node to target the `kube-system` namespace as it is the namespace where this guide will install Pure Service Orchestrator. This can be done by using `cloudctl` as shown in the example below:

```
# cloudctl login https://xx.xx.xx.xx:8443 --skip-ssl-validation
API endpoint: https://xx.xx.xx.xx:8443

Username> admin

Password>
```

```
Authenticating...
OK

Select an account:
1. mycluster Account (id-mycluster-account)
Enter a number> 1
Targeted account mycluster Account (id-mycluster-account)

Select a namespace:
1. cert-manager
2. Default
3. istio-system
4. kube-public
5. kube-system
6. Platform
7. Services
Enter a number> 5
Targeted namespace kube-system

Configuring kubectl ...
Property "clusters.mycluster" unset.
Property "users.mycluster-user" unset.
Property "contexts.mycluster-context" unset.
Cluster "mycluster" set.
User "mycluster-user" set.
Context "mycluster-context" created.
Switched to context "mycluster-context".
OK

Configuring helm: /root/.helm
OK
```

- To enable the Service Account used by Pure Service Orchestrator to use privileged containers, it is necessary: to create a new ClusterRoleBinding. Create a file called `pso-crb.yaml` to contain the following:

```
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRoleBinding
metadata:
  name: pure-pso-user
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: ClusterRole
  name: privileged
subjects:
- kind: ServiceAccount
  name: pure
  namespace: kube-system
```

Invoke this file using the following command:

```
# kubectl apply -f pso-crb.yaml
```

- Update the configuration file

To enable the Pure Service Orchestrator for Kubernetes to communicate with your Pure Storage backend storage platforms, it is required to update the configuration file to reflect the access information for the backend storage providers. The file is called `values.yaml` and needs to contain the management IP address of the backend devices, together with a valid, privileged, API token for each device, together with an NFS Data VIP address for each FlashBlade™.

Take a copy of the `values.yaml` provided by the Helm Chart and update the `arrays` parameters in the configuration file with your site-specific information as shown in the following example:

```
arrays:
  FlashArrays:
    - MgmtEndPoint: "1.2.3.4"
      APIToken: "a526a4c6-18b0-a8c9-1afa-3499293574bb"
      Labels:
        rack: "22"
        env: "prod"
    - MgmtEndPoint: "1.2.3.5"
      APIToken: "b526a4c6-18b0-a8c9-1afa-3499293574bb"
  FlashBlades:
    - MgmtEndPoint: "1.2.3.6"
      APIToken: "T-c4925090-c9bf-4033-8537-d24ee5669135"
      NfsEndPoint: "1.2.3.7"
      Labels:
        rack: "7b"
        env: "dev"
    - MgmtEndPoint: "1.2.3.8"
      APIToken: "T-d4925090-c9bf-4033-8537-d24ee5669135"
      NfsEndPoint: "1.2.3.9"
      Labels:
        rack: "6a"
```

Ensure that the values you enter are correct for your own Pure Storage devices. Note: the "Labels" entries above are optional but can be useful to target specific storage devices (see below).

If you wish to use Fibre Channel as your communication protocol for FlashArrays, you must also update the following parameter in the configuration file:

```
flasharray.sanType: FC
```

- Labels

  You will see in the above example that entries have one or more labels assigned to them. Labels can be used to filter the list of backends. Labels are arbitrary (key, value) pairs that can be added to any backend as seen in the example above. More than one backend can have the same (key, value) pair. When creating a new volume, label (key = value) pairs can be specified to filter the list of backends to a given set and enforce that the new volume be created in one of the storage devices in that set. The plugin also provides the following well-known labels that can be used:

  - `purestorage.com/backend`: Holds the value `file` for FlashBlade™ and `block` for FlashArray™.
  - `purestorage.com/hostname`: Holds the host name of the backend.
  - `purestorage.com/id`: Holds the ID of the backend.
  - `purestorage.com/family`: Holds either `FlashArray` or `FlashBlade`

- Install the plugin

It is advisable to perform a 'dry run' installation to ensure that your `values.yaml` file is correctly formatted:

```
# helm install --tls --name pure-storage-driver pure/pure-k8s-plugin -f
<your_own_dir>/<your_own_values>.yaml --dry-run --debug
```

Run the actual install.

```
# helm install --tls --name pure-storage-driver pure/pure-k8s-plugin -f
<your_own_dir>/<your_own_values>.yaml
```

The values set in your own YAML file will overwrite the ones in the default, provided, values.yaml file. However, the `--set` option can also take precedence over any value in either YAML file, for example:

```
# helm install --tls --name pure-storage-driver pure/pure-k8s-plugin -f <your_own_dir>/<your own
values>.yaml --set flasharray.sanType=FC,namespace.pure=k8s_xxx
```

However, it is recommended to use the `values.yaml` file rather than the –-set option for ease of use, especially should modifications be required to your configuration in the future.

Checking Correct Installation

After running the installer there are a few things we can check to ensure that the plugin has been installed correctly. Run these commands on the client node, ensuring that you are still targeting the `kube-system` namespace.

- Check StorageClasses

```
# kubectl get sc
NAME             TYPE
pure             pure-provisioner
pure-block       pure-provisioner
pure-file        pure-provisioner
```

- Check FlexVolume driver running on each worker node

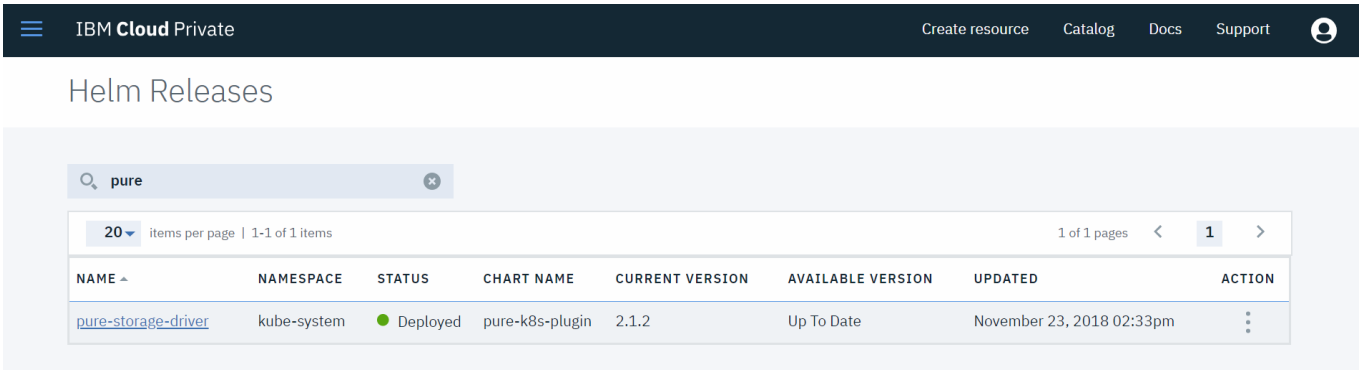
```
# kubectl get pods
NAME                                READY     STATUS    RESTARTS   AGE
pure-flex-4mkfw                     1/1       Running   0          2h
pure-flex-527zn                     1/1       Running   0          2h
pure-flex-7n9cf                     1/1       Running   0          2h
pure-flex-jxxbn                     1/1       Running   0          2h
pure-flex-w9x74                     1/1       Running   0          2h
pure-provisioner-2785090122-c69xx   1/1       Running   0          2h
```

- Check Dynamic Provisioner

```
# kubectl get deployments
NAME               DESIRED   CURRENT   UP-TO-DATE   AVAILABLE   AGE
pure-provisioner   1         1         1            1           2h
```

It is also possible to check the deployment of Pure Service Orchestrator within the IBM Cloud Private management GUI.

Navigate, using the Menu, to **Workloads : Helm Releases** and then enter 'pure' in the Search bar. You should then see something like this:



Updating your Storage Configuration

No environment is static and use of your IBM Cloud Private cluster will inevitably grow as more applications and users use the resource. As part of this growth there could be a demand for additional backend storage, be that adding more FlashArrays for greater capacity, or adding FlashBlades into a block-only environment to facilitate shared volumes across containers and applications, or just adding new labels to existing storage to allow for more granular control of storage placement.

Changing or adding Pure Storage backend devices in an existing deployment of the Pure Service Orchestrator is seamless and simple.

Update your values YAML file with your new FlashArrays or FlashBlades, or add new labels to existing devices, then update the Kubernetes configuration maps through Helm using the following command:

```
# helm upgrade --tls pure-storage-driver pure/pure-k8s-plugin -f
<your_own_dir>/<your_own_values>.yaml
```

If you used the --set option when initially deploying the plugin you must use the same option again, unless these have been incorporated into your latest YAML file.

It is important to ensure that you continue to use a client node that has been authenticated and targeted to the `kube-system` namespace when performing Pure Service Orchestrator command-line updates.

To test the proper functionality of your Pure Service Orchestrator deployment, we recommend that you follow the instructions available in the Kubernetes, Persistent Volumes and the Pure Service Orchestrator guide, starting from the "Simple Proof Point of PVs from Pure Storage" section.

Conclusion

IBM Cloud Private is a full stack of IBM applications and middleware, from virtualization engines all the way up to the services catalog.

By leveraging Pure Service Orchestrator within this environment, you can utilize the class-leading functionality this provides, backed by the enterprise-class capabilities and uptime of Pure Storage platforms.

About the Author

As Director of New Stack, Simon is responsible for developing the technical direction of Pure Storage pertaining to Open Source technologies including OpenStack, Containers, and associated orchestration and automation toolsets. His responsibilities also include developing best practices, reference architectures and configuration guides.

With over 30 years of storage experience across all aspects of the discipline, from administration to architectural design, Simon has worked with all major storage vendors' technologies and organisations, large and small, across Europe and the USA as both customer and service provider.

Blog: http://blog.purestorage.com/author/simon

PURESTORAGE

Pure Storage, Inc.
Twitter: @purestorage
www.purestorage.com

650 Castro Street, Suite #260
Mountain View, CA 94041

T: 650-290-6088
F: 650-625-9667

Sales: sales@purestorage.com
Support: support@purestorage.com
Media: pr@purestorage.com
General: info@purestorage.com