

Spectrum Scale and Container Support

**Spectrum Scale Cloud Team V 1.09.1
Presented By John Lewars**

Disclaimer

- The information in this document is IBM CONFIDENTIAL.
- This information is provided on an "AS IS" basis without warranty of any kind, express or implied, including, but not limited to, the implied warranties of merchantability and fitness for a particular purpose. Some jurisdictions do not allow disclaimers of express or implied warranties in certain transactions; therefore, this statement may not apply to you.
- This information is provided for information purposes only as a high level overview of possible future products. PRODUCT SPECIFICATIONS, ANNOUNCE DATES, AND OTHER INFORMATION CONTAINED HEREIN ARE SUBJECT TO CHANGE AND WITHDRAWAL WITHOUT NOTICE.
- USE OF THIS DOCUMENT IS LIMITED TO SELECT IBM PERSONNEL AND TO BUSINESS PARTNERS WHO HAVE A CURRENT SIGNED NONDISCLOSURE AGREEMENT ON FILE WITH IBM. THIS INFORMATION CAN ALSO BE SHARED WITH CUSTOMERS WHO HAVE A CURRENT SIGNED NONDISCLOSURE AGREEMENT ON FILE WITH IBM, BUT THIS DOCUMENT SHOULD NOT BE GIVEN TO A CUSTOMER EITHER IN HARDCOPY OR ELECTRONIC FORMAT.
- IBM's statements regarding its plans, directions, and intent are subject to change or withdrawal without notice at IBM's sole discretion. Information regarding potential future products is intended to outline our general product direction and it should not be relied on in making a purchasing decision. The information mentioned regarding potential future products is not a commitment, promise, or legal obligation to deliver any material, code, or functionality. The development, release, and timing of any future features or functionality described for our products remains at our sole discretion.
- IBM reserves the right to change product specifications and offerings at any time without notice. This publication could include technical inaccuracies or typographical errors. References herein to IBM products and services do not imply that IBM intends to make them available in all countries.

Agenda

- Containers – brief list of some of the benefits
- Spectrum Scale and containers what is possible?
- Ways to access Spectrum Scale from containers (flow chart)
- Ubiquity Details
- Examples of configurations in which Spectrum Scale may be used with ICP

Containers



Benefits:

- Workload Isolation (less CPU/memory overhead and faster deployment than VMs)
- Resource limitations (cgroups can constrain containers' memory and CPU usage)
- Layered image architecture such that each change to the package is tracked
- Build once/ Run anywhere:
Provides a way to package an application and all its dependencies so that the application can be moved freely between environments – packages are light weight, stand-alone, including code, runtime environment, libraries, etc

Spectrum Scale and Containers

What is Possible?

- Can Spectrum Scale file systems be used by containerized workloads?
 - Yes – and this presentation will focus on possible approaches.
- Can Spectrum Scale itself be containerized?
 - This is not currently supported but IBM Research has been investigating
 - Note that Spectrum Scale kernel extension requirements, as the code is currently implemented, would require a privileged container to run
- Can Spectrum Scale be installed from a container image?
 - IBM doesn't provide a way to do this (so it's not supported) but IBM Research has been doing prototype work in this area
- What about Container Storage Interface (CSI)?
 - The team working on IBM Storage Enabler for Containers (SEC) tentatively plans to support CSI in 2019 (the Scale Research and Product teams are looking at CSI)

Containers and Spectrum Scale

Use Case 1/4 – Command Line Mounts

- In this presentation we focus on Docker use cases which is the most common container technology, but IBM knows some customer user other containers (e.g. Singularity) and we are interested in if there's desire to do more with these other containers
- Spectrum Scale can be bind mounted manually to allow native Spectrum Scale access, and NFS exports of Spectrum Scale file systems may also be manually bind mounted

(e.g. `docker run -v`) `run -v /source_path:/destination_bind_mount_path [. . .]`

- This approach is not Scalable on a shared multi-user system as it requires the users to manage the assignment of workloads to resources
- Requires that users have root access, or sudo access to the Docker command, which is typically not acceptable on a big system (e.g. all the users must trust each other)

Containers and Spectrum Scale

Use Case 2/4 – Running under LSF

(Or another Workload Manager)

- Some workload management platforms, or High Performance job schedulers, have support for running containers
- For example, IBM Spectrum Compute LSF) can automatically established the required bind mounts to allow containers to access Spectrum Scale for directories specified in the job command file (including the job's current working, spool, temporary, checkpoint, and \$LSF_TOP directories.)
- This approach is more scalable and 'secure' in a multi-user shared system compared to approach (1)
- On a system which is focused on container workloads, container orchestration systems, such as Kubernetes and Docker Swarm may be a better fit for managing container workloads, compared to traditional job management systems like LSF

Containers and Spectrum Scale

Use Case 3/4 – What Currently is Available to Use Spectrum Scale with Kubernetes?

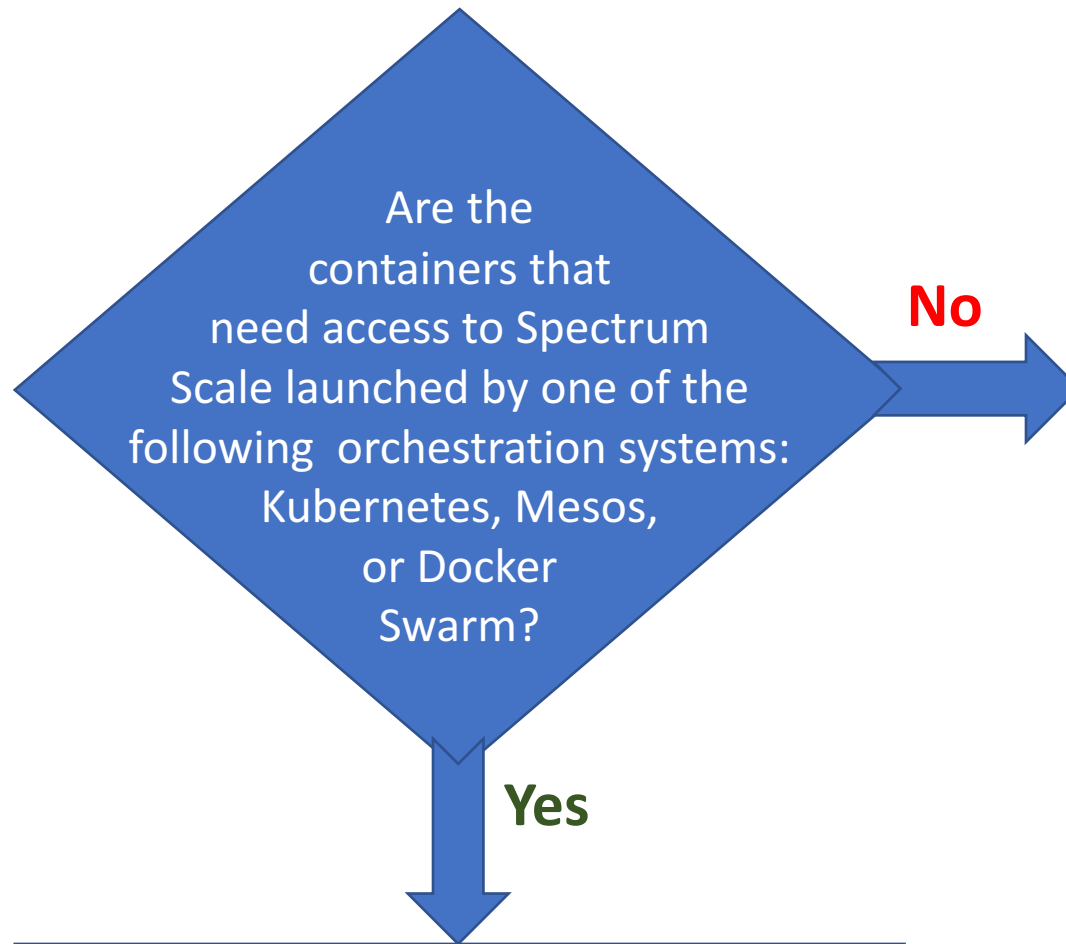
- When using the Kubernetes orchestration, Spectrum Scale may be made available via native access using the [hostPath driver](#)
- Spectrum Scale NFS exported file systems may be made available using the Kubernetes NFS driver as per [NFS Kubernetes access examples](#).
- This use case is valid for cases in which the required directories, with the correct access permissions, are setup in advance of the job running.
- This approach therefore does not support Dynamic Provisioning, which is a method by which users of Kubernetes may dynamically request the creation of new Kubernetes volumes to self provision storage for their containers

Containers and Spectrum Scale

Use Case 4/4 – Extending Support for Container Orchestration Systems

- IBM Developed an Open Source project, the [Ubiquity plugin](#), to enable using IBM Storage with Kubernetes, Mesos, or Docker Swarm orchestration systems
- The project focused first on Spectrum Scale , allowing for native Spectrum Scale access, or NFS exported file system access, and later incorporated block support
- The Spectrum Scale's team work on the [Ubiquity plugin](#) was temporarily suspended in 2017, and versions of Ubiquity after V0.4, have problems working with Spectrum Scale
- Currently (as of Sept. 2018), IBM intends to have a new fork of the Ubiquity code available in October 2018, which will address the Spectrum Scale regressions. There will be no formal support for this code.
- In December 2018, IBM intends to release the code base delivered via the [Ubiquity GitHub](#) in a more formal product offering (IBM Storage Enabler for Containers) that will support Spectrum Scale access from containers

Accessing a Scale file system from a container- What do I need? Step 1



No

Yes

If you're using Kubernetes, Mesos, or Docker Swarm orchestration software, you may access Scale via a [Ubiquity service](#) plug-in, or (only on Kubernetes) [the hostPath driver](#).

Proceed to step 2 for details.

If there is no orchestration system in place, you should not need a storage driver/plug-in for Spectrum Scale access. Some possible cases:

1) If you're launching containers directly via a "docker run" command line statements, then you may use the docker '-v' flag (or, as of Docker 17.06, the '--mount' flag) to bind mount a Spectrum Scale file system into your container, e.g. the Scale file system '/scale/scratch' mounted on the host could be made available at the mount point '/scratch' in a newly launched container as follows:

```
docker run -v /scale/scratch:/scratch [ . . . ]
```

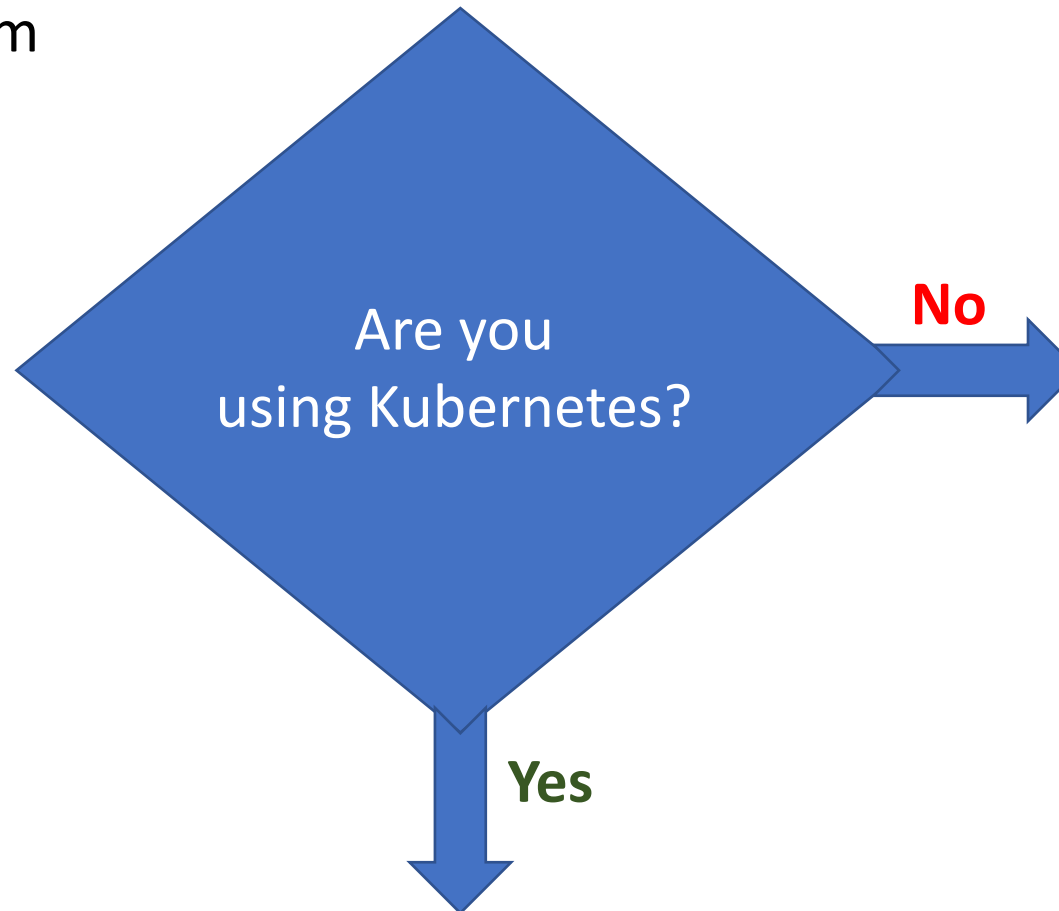
or

```
docker run --mount type=bind,source=/scale/scratch,\ntarget=/scratch [ . . . ]
```

2) If you are running a workload management platform, or High Performance job scheduler that has support for running containers (for example IBM Spectrum Compute LSF), it may already create any bind mounts needed by jobs that launch containers. For example, IBM LSF can run workloads in containers started via 'docker run' command with the '-v' flag set to automatically mount any Spectrum Scale directory required, including the job's current working, spool, temporary, checkpoint, and \$LSF_TOP directories.

Accessing a Scale file system from a container. . . What do I need? Step 2

I'm using Kubernetes, Mesos, or Docker Swarm



If you're using Kubernetes then, depending on your use case, you may choose to use [the hostPath driver](#) or the [Ubiquity plugin for Kubernetes](#).
Proceed to step 3 for details.

1) IBM has tested the [Ubiquity docker volume plugin](#) to provide Spectrum Scale access to containers launched from both Docker Swarm and Mesos.

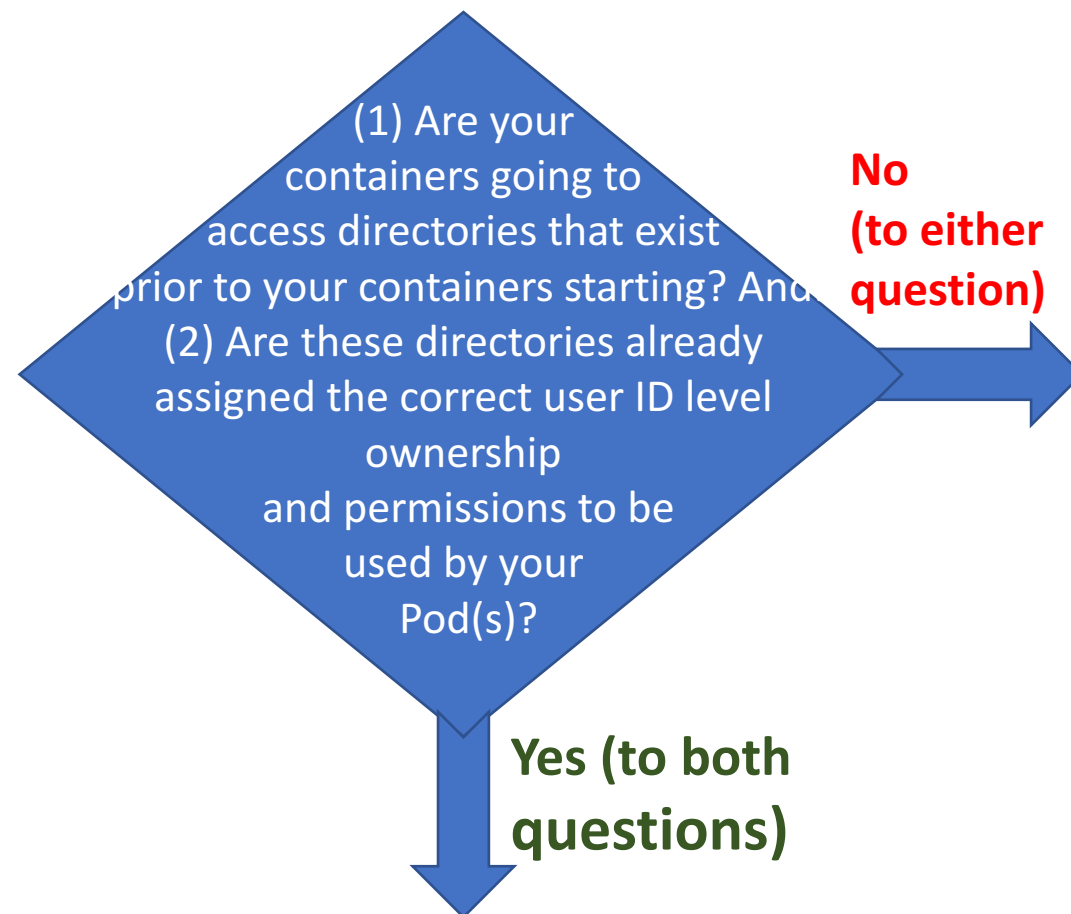
2) The hostPath driver is available for Kubernetes but, with docker swarm, we can bind mount the volume using a "docker service create" command when starting the service being launched, as per this example:

```
docker service create --replicas 1 --mount type=bind,source=/gpfs/gpfs_device,destination=/gpfs/gpfs_device --name helloworld1 alpine ping docker.com
```

Using the command from this example, if the service fails, it will get restarted on different node with the same bind mount in effect, provided the same path exists (/gpfs/gpfs_device) on the new service node.

Accessing a Scale file system from a container. . . What do I need? Step 3

I'm using Kubernetes Orchestration



If you're using Kubernetes and all of the directories your Pods want to access:

- (1) exist already and
- (2) have the ownership and permissions set to allow your Pods to access them,

then the [the hostPath driver](#) should meet your use case.

If you answered 'No' to either question, the [Ubiquity plugin for Kubernetes](#) may be the appropriate choice for your workload but read the following details to verify:

(1) The ability for a Pod to request the creation of new a Kubernetes volume as it launches refers to a concept known as Dynamic Provisioning in Kubernetes. The Spectrum Scale Ubiquity project plans to continue support two types volumes, based on the two types of Spectrum Scale filesets:

- (i) Spectrum Scale independent filesets (1000 supported per file system),
- (ii) Spectrum Scale dependent filesets (10000 supported per file system).

If you require higher volume scaling (from a single file system) your use case may not be appropriate for Spectrum Scale.

Note that Dynamic provisioning in the context of Spectrum Scale refers to the dynamic creation of Spectrum Scale directories and filesets, and does not imply the automation of Spectrum Scale installation, cluster and file system creation/configuration, etc.

You should also consider some characteristics of snapshots if you plan to use them with Spectrum Scale filesets. Note that each independent fileset supports up to 256 snapshots, and every dependent fileset belongs to one independent fileset (all dependent filesets share their parent's snapshot limit).

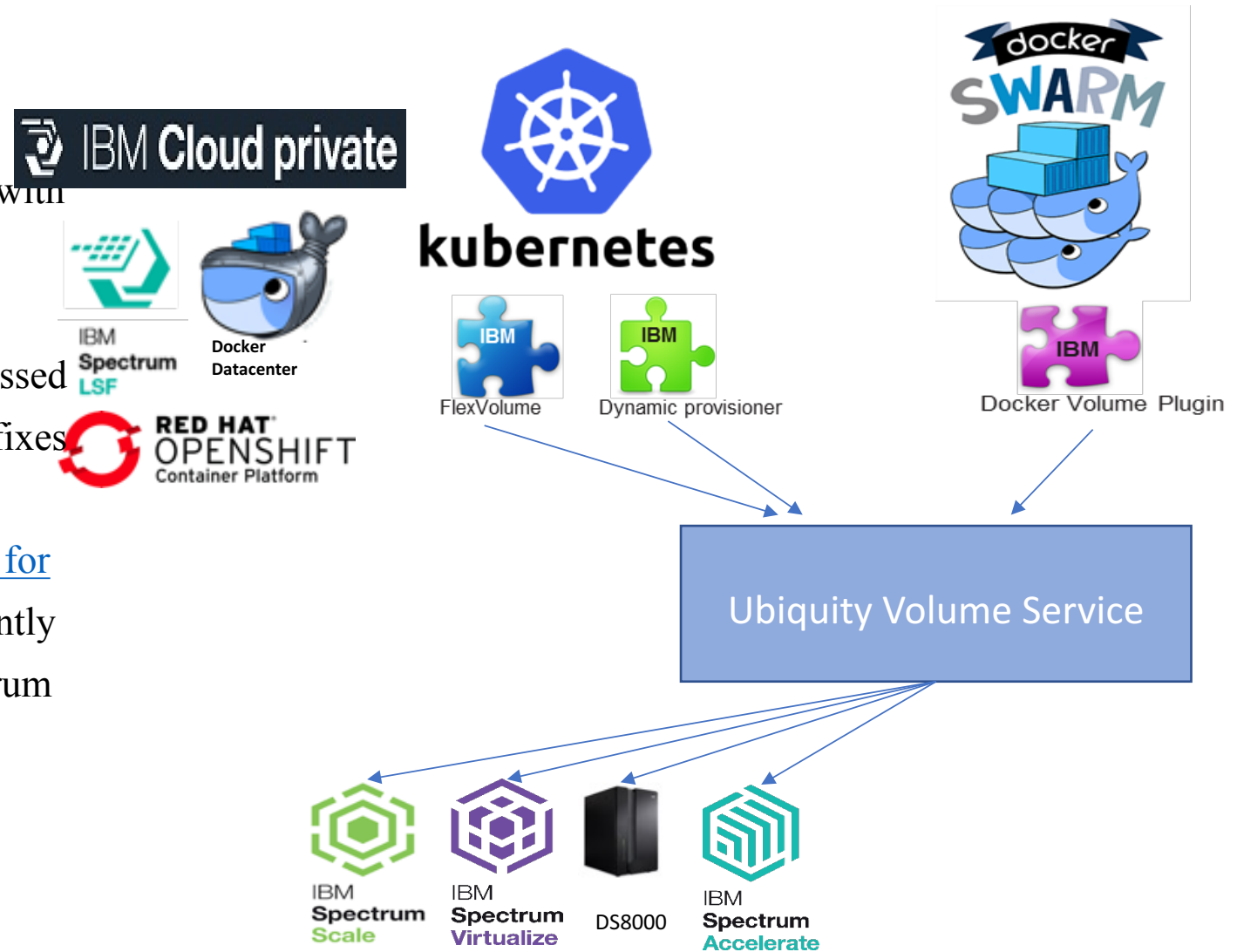
There may be a performance degradation observed with using large number of filesets and/or snapshots. Note that independent filesets have their own dedicated inode space which enables more efficient per-fileset functions, such as fileset snapshots.

(2) In a statically provisioned use case, all directories are created up front, and the Kubernetes hostPath driver should meet this use case if these storage volumes are created with the ownership and permissions that match the user ID of the pod that tries to use them (the hostPath driver allows for group ID permission changes at run time but not user ID permission changes).

Therefore the [Ubiquity plugin for Kubernetes](#) is needed for workloads that don't have all their volumes created in advance with the correct directory ownership and permissions assigned.

Ubiquity Details

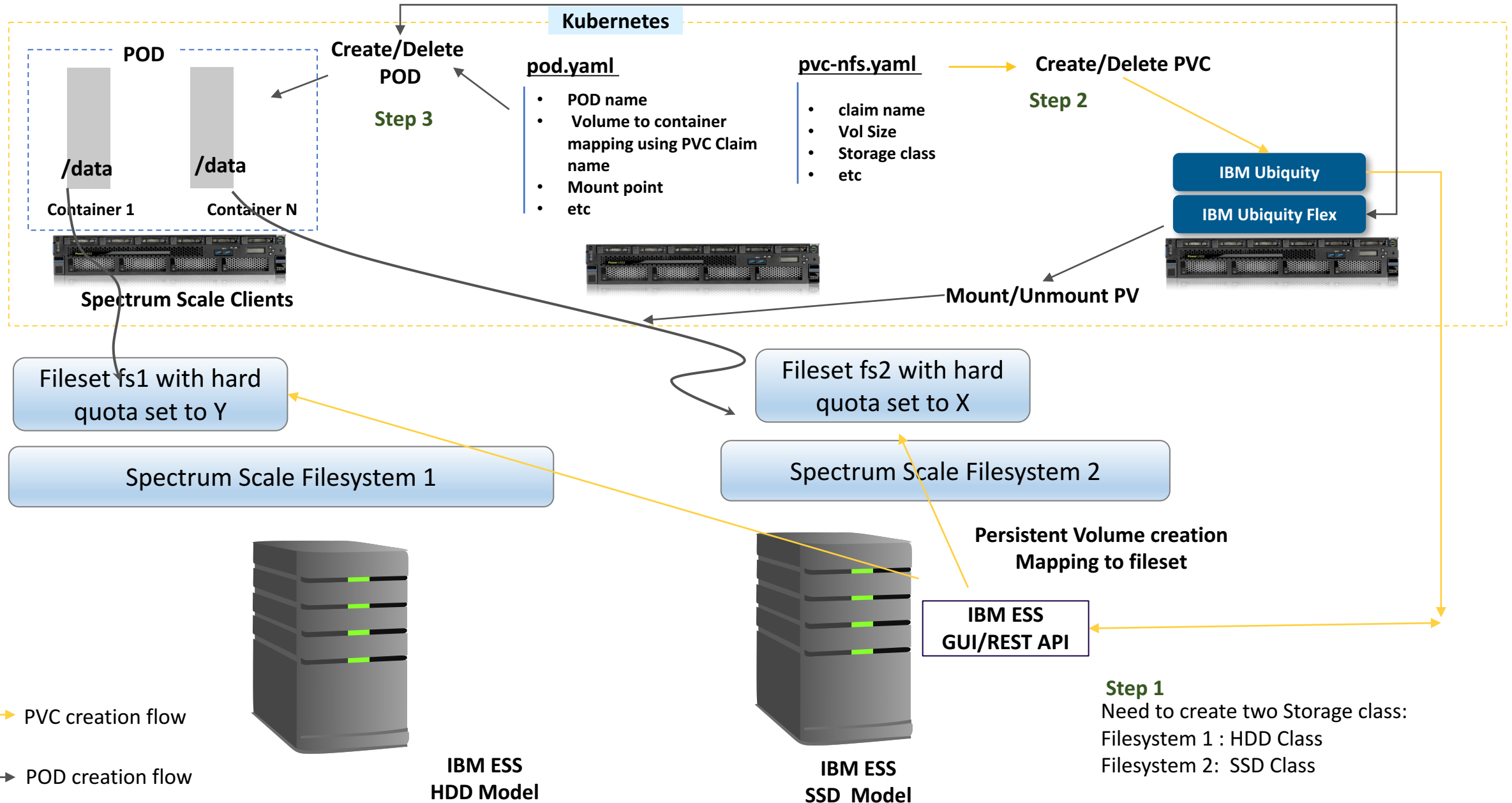
- Unified solution across IBM storage
- Supports different containers' orchestrators
- Supports Dynamic Provisioning
- Highly available & secure (use Scale's Rest API V2 with secure connections)
- Initial GA planned to support native Scale (not NFS)
- [Open Source Project](#) originally (latest code has regressed Spectrum Scale support but IBM intends to provide fixes for Spectrum Scale in Github via October 2018)
- Code base is to delivered via [IBM's Storage Enabler for Containers \(SEC\)](#) (Spectrum Scale support not currently delivered via SEC but IBM *intends* to support Spectrum Scale in December 2018, (subject to plan review))
- The code delivered in SEC is IBM strategic plan for supporting Spectrum Scale access from containers running on IBM Cloud Private (ICP)



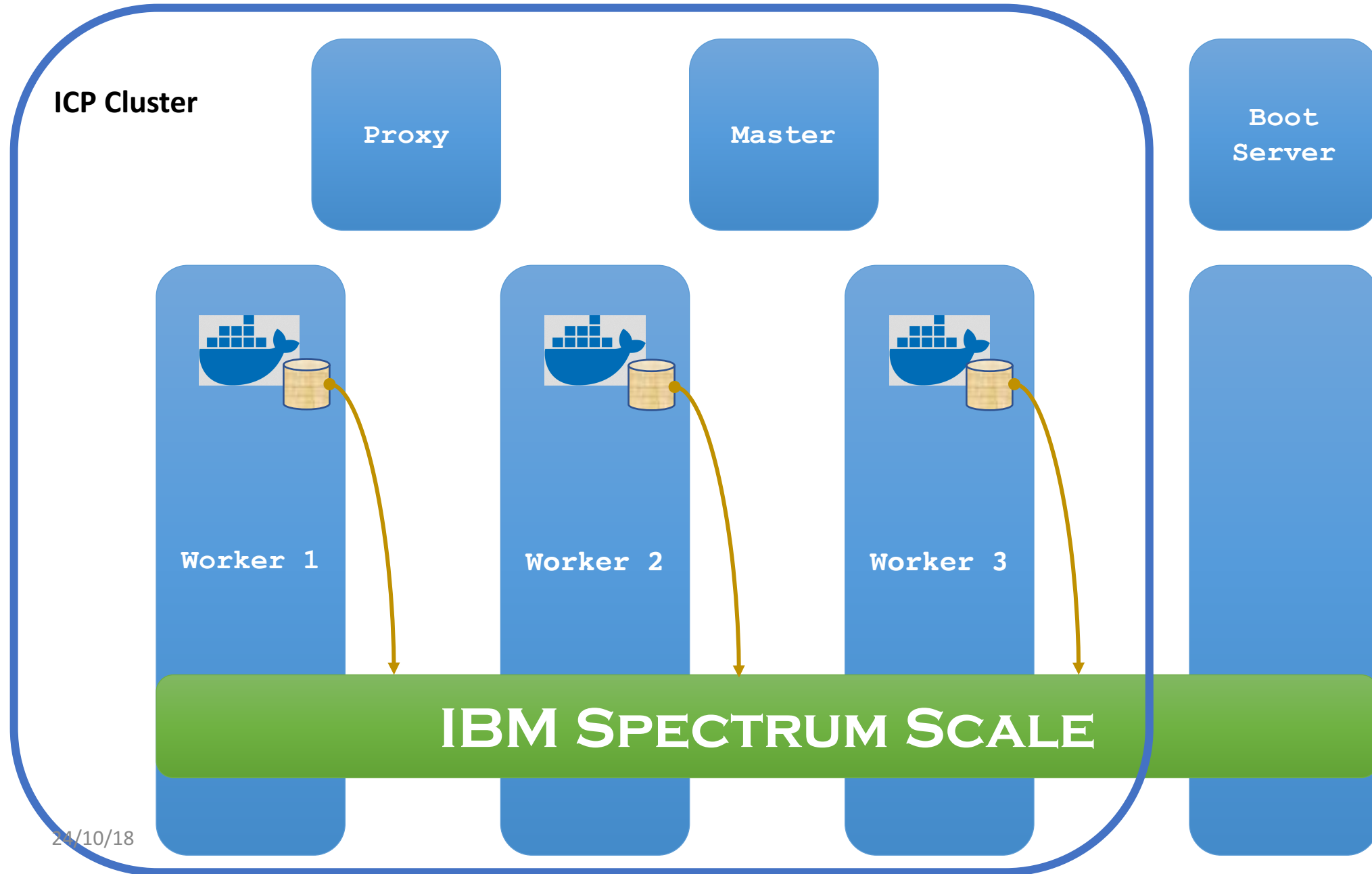
Notes on a Use Case Scenario with Containers and IBM Spectrum Scale's Upcoming SEC Support

- Question: after IBM delivers SEC support for Spectrum Scale dynamically provisioning persistent volumes for Kubernetes, how do I leverage class support to provide storage with different performance characteristics to containers?
- Answer: Let's focus on an example configuration: one flash ESS system and one all HDD ESS system
- Now we'll show a workflow that describes how to enable the following use case:
Let's say a customer desires:
 - a) X amount of capacity using using "all flash" storage class (coming from an all flash ESS) and wants to create a persistent volume for given container in a POD and:
 - b) Y amount of capacity using using "HDD" storage class (coming from an HDD ESS) to create a persistent volume for a given container in the same Kubernetes POD.

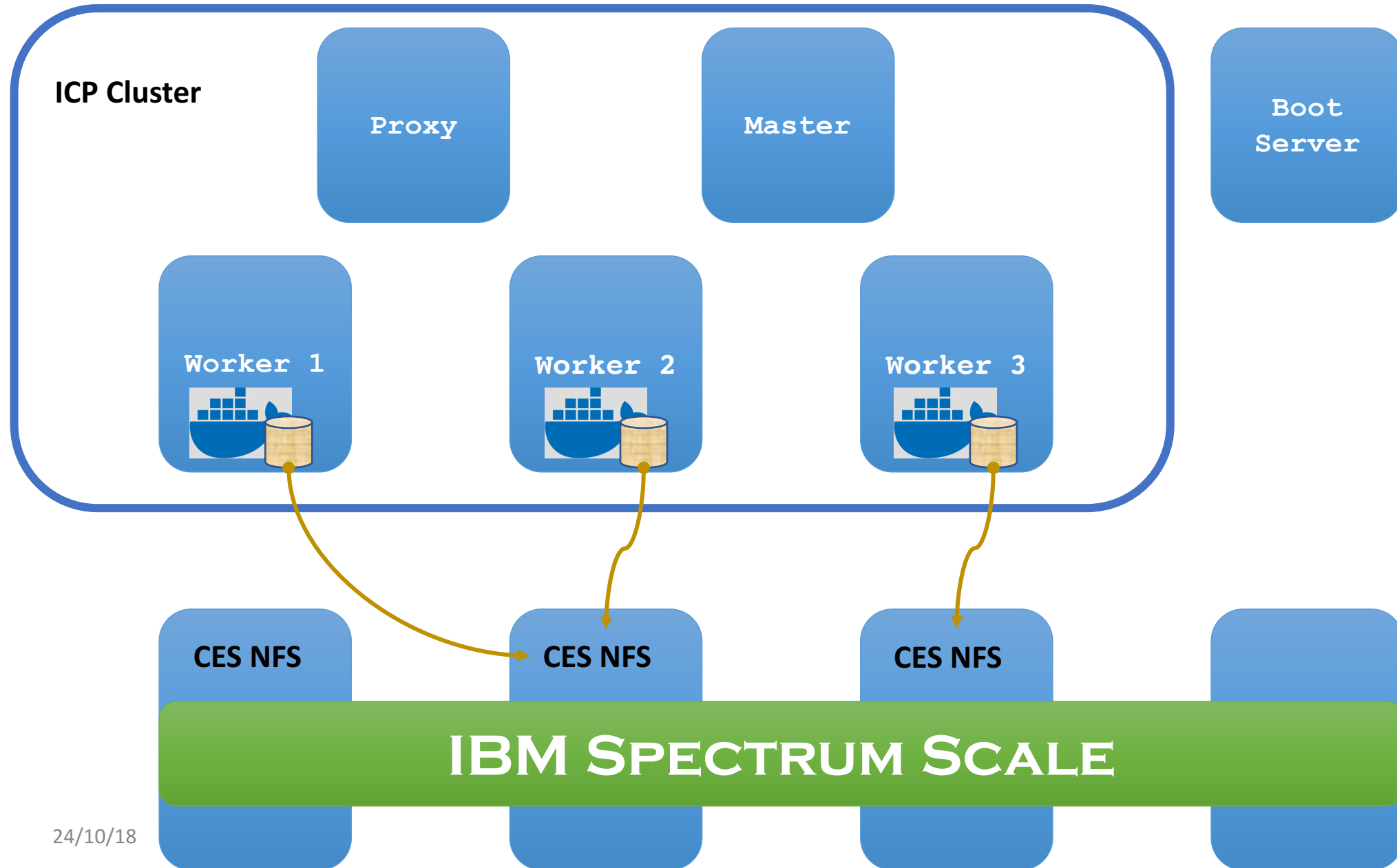
Kubernetes Storage Class Use Case for Spectrum Scale High Level Flow



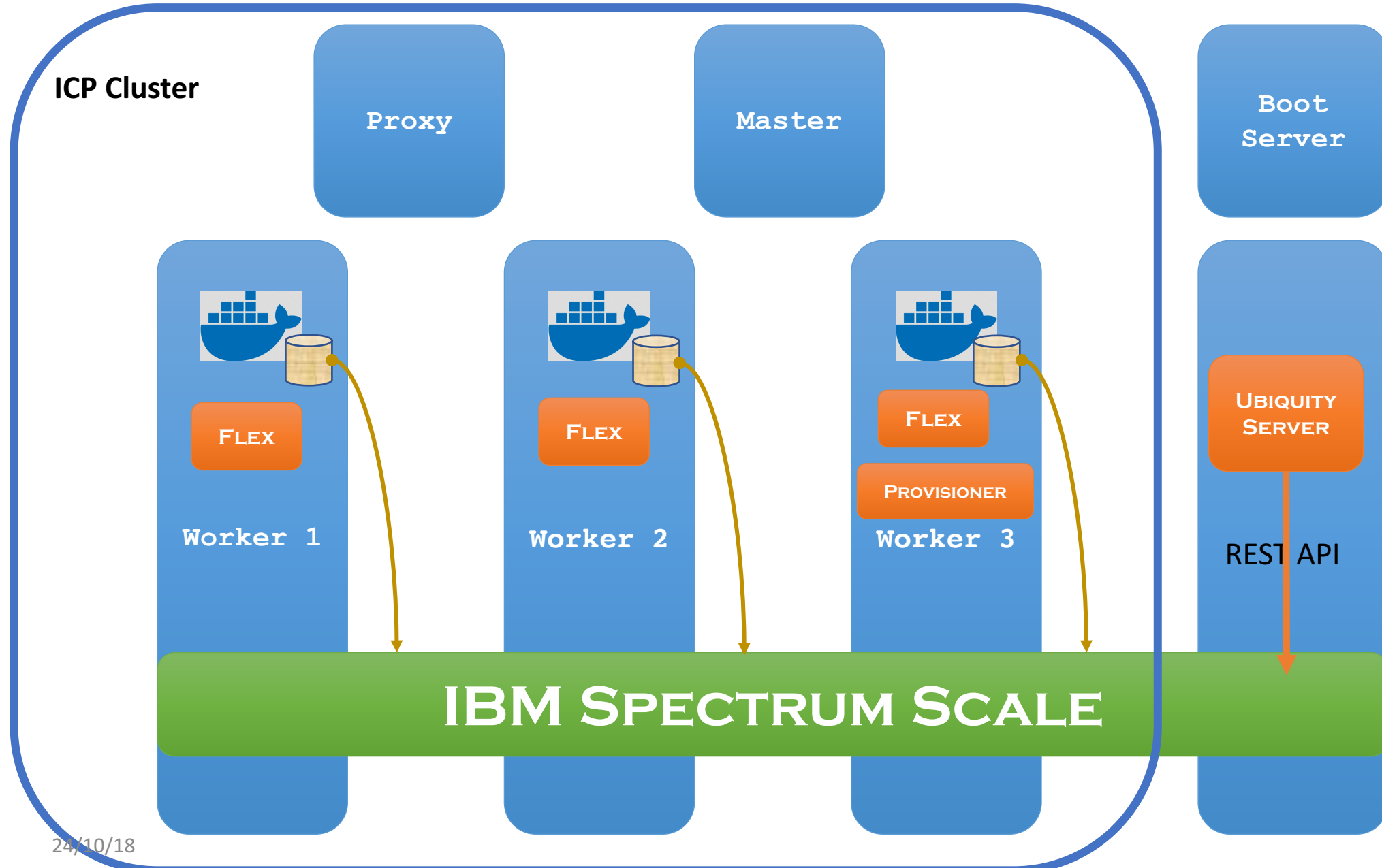
Spectrum Scale with ICP via Hostpath driver

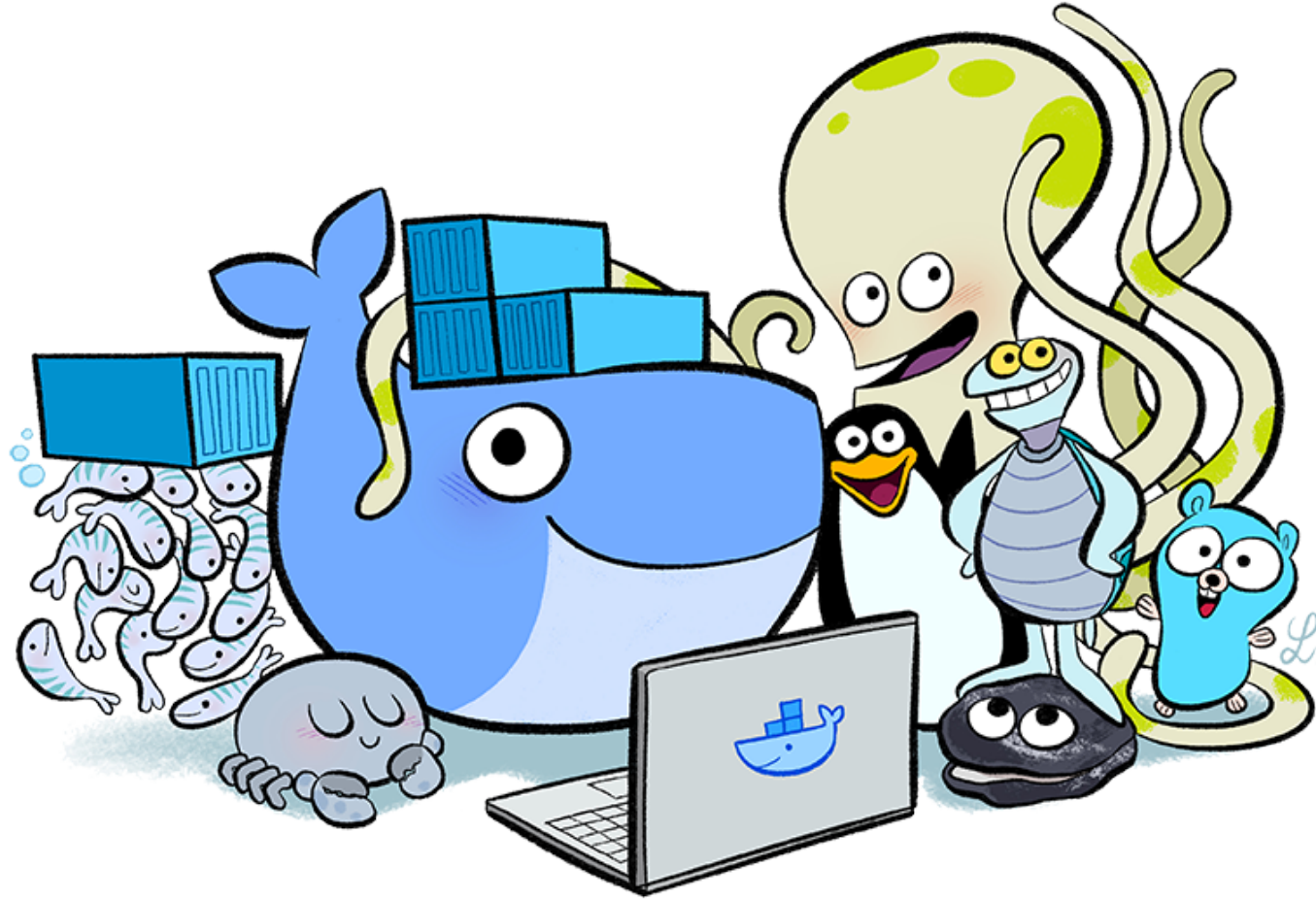


Spectrum Scale with ICP via NFS driver



Spectrum Scale with ICP w/ Ubiquity (native Scale)





Thank You !