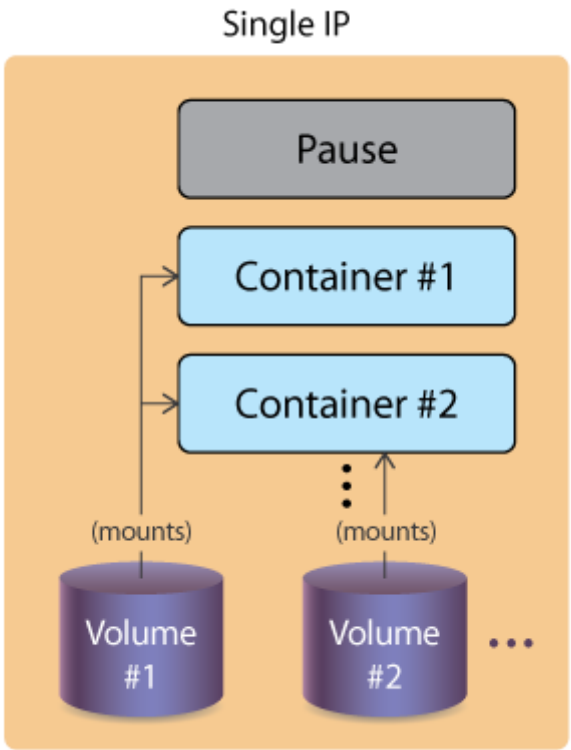
VOLUMES AND DATA

Volumes and Data

Introducing Volumes

A Pod specification can declare one or more volumes and where they are made available. Each requires a name, a type, and a mount point. The same volume can be made available to multiple containers within a Pod, which can be a method of container-to-container communication. A volume can be made available to multiple Pods, with each given an access mode to write. There is no concurrency checking, which means data corruption is probable, unless outside locking takes place.



Kubernetes Pod Volumes

A particular access mode is part of a Pod request. As a request, the user may be granted more, but not less access, though a direct match is attempted first. The cluster groups volumes with the same mode together, then sorts volumes by size, from smallest to largest. The claim is checked against each in that access mode group, until a volume of sufficient size matches. The three access modes are:

- ReadWriteOnce, which allows read-write by a single node
- ReadOnlyMany, which allows read-only by multiple nodes
- ReadWriteMany, which allows read-write by many nodes.

Thus two pods on the same node can write to a ReadWriteOnce, but a third pod on a different node would not become ready due to a FailedAttachVolume error.

When a volume is requested, the local kubelet uses the **`kubelet_pods.go`** script to map the raw

devices, determine and make the mount point for the container, then create the symbolic link on the host node filesystem to associate the storage to the container. The API server makes a request for the storage to the **`StorageClass`** plugin, but the specifics of the requests to the backend storage depend on the plugin in use.

If a request for a particular **`StorageClass`** was not made, then the only parameters used will be access mode and size. The volume could come from any of the storage types available, and there is no configuration to determine which of the available ones will be used.