[Shared Publicly]

Proposal: Device Assignment

By dashpole@ updated 07/20/2018

Background

The Pod API Object contains the metadata, specification, and status of the smallest unit of computing that can be managed by kubernetes. In order to support Network Devices through the device plugin, it has been proposed (in Network Devices Support Changes Proposal) to add Pod UID to the Allocate call, and then have the CNI ask the Network Device Plugin for it to do network plumbing. Note that this is a very tentative proposal, and the proposal below does not rely on adding the PodUID to the Allocate call. Device Monitoring requires external agents to be able to determine the set of devices in-use by containers, and attach pod and container metadata for these devices.

Motivation

- We want to provide monitoring agents on the node the mapping between devices consumed, and container metadata so they can determine the set of devices in-use by containers, and attach pod and container metadata for these devices.
- We want to make it possible to assign devices to containers at the cluster level (e.g. with a custom scheduler). This would be useful in cases where pods have strong requirements about the set of devices they are assigned (e.g. specific numa topology requirements). While the node can, in best-effort fashion, provide the best set of devices available, the best available may not meet the workload's needs. In such cases, it would be useful to be able to use a custom scheduler to provide "guaranteed" properties about the set of devices given. However, the default case is still for the kubelet to assign devices in a best-effort manner.
- We want to enable CNI plugins to determine the set of devices assigned to a pod during network setup so it does not need to coordinate with network device plugins

Proposal: DeviceIDs in the Container Spec

```
type Container struct {
        ...
        // ComputeDevices contains the devices assigned to this container
        // This field is alpha-level and is only honored by servers that enable the
DeviceAssignment feature.
        // +optional
        ComputeDevices []ComputeDevice
        ...
}
// ComputeDevice describes the devices assigned to this container for a given
ResourceName
Type ComputeDevice struct {
        DeviceIDs           []string
        ResourceName        string
}
```

- Add `AssignedDevices` (or a more expansive api as suggested in the Resource Class Proposal) as a field in container spec. The kubelet assigns the DeviceIDs during pod admission following the Late Initialization pattern, and subsequently calls Allocate to complete pod admission. DeviceIDs would be reflected to the APIServer after they have been selected, but before Allocate is called. The kubelet only assigns DeviceIds not already present in the pod spec.
    - The kubelet needs to persist this information to the API Server, and observe the change locally before creating the pod, or serving the pod on the `/pods` endpoint.
    - If DeviceIDs are already assigned, the kubelet keeps the current assignment. This way, on kubelet restarts, the kubelet will not reassign Devices. This also means a scheduler can set DeviceIDs during scheduling.
- Device Monitoring Agents can discover the relationship between pods given the Pod API Object, which can be obtained by querying the `/pods` endpoint, or a future watch-style endpoint.
- Network device plugins discover and monitor the health of network devices on the node, and do not necessarily return DeviceSpecs or Mounts in their AllocateResponse, although some will (a memif interface created by VPP, which is represented as files on the host system). The CNI plugin queries the `/pods` endpoint each time it needs to perform network setup for a pod to get the devices it should plumb to the pod. CNI can get the devices required for a pod because it is passed the pod name and namespace during network initialization. CNI plugins would only be able to set-up networking for ResourceNames that it recognizes.

Downsides

- DeviceID is a concept in a **Beta** plugin API. Adding this into a **GA** Pod API is not ideal, since the plugin API could change.
    - Note: This is less of a downside if it starts as an Alpha Field
- Both cluster-level device scheduling and network device integration into the device plugin's designs are not complete. This change may not actually be an important piece of those designs.
- It may be confusing for users who attempt to manually set DeviceID. Specifying a device without specifying a node could result in scheduling to a node where the device does not exist, or is already being used. Specifying a Device that does not exist, is unhealthy, or is already in-use would result in the pod being repeatedly created, and rejected by the kubelet.

Alternative Proposal: DeviceIDs in the container status

- Add DeviceIDs map[v1.ResourceName][]types.UID as a field in container status. Populate this field based on the DeviceManager.
    - Kubelet restarts rely on consuming the device manager checkpoint in order to populate this.

- Device Monitoring agents can use the `/pods` endpoint to find which devices are in use, and which container is using each device. There may be a period of time when the pod has been created, but the status has not yet propagated to the pods endpoint.
- CNI plugins cannot use the `/pods` endpoint because pod status is only populated after the pod is created, and status may not always include device assignments if we fail to recover the checkpoint.