oci-dev-binder-hook

dynamic device access management based on container annotations

Roberto Majadas <rmajadas@redhat.com>
Albert Esteve <aesteve@redhat.com>

The Challenge

- Containers currently offer static device addition using --device
- But if you're working with diverse hardware boards (e.g., automotive or IoT devices), the exact set of peripheral devices is often unknown beforehand.
    - For instance, the QM container could require access to devices that could be represented in different way depending on the board, i.e: /dev/dri/cardX /dev/input/*

What can help us to solve the challenge?

- The **Open Container Initiative** (OCI) standard defines a runtime specification for containers.
  - This standard includes **hooks**, which are scripts or programs that can be triggered at different points in the container lifecycle.
  - We can use a **pre-start** hook to run a program **after** the container namespace is created but **before** the user's application starts.
- **Systemd/udev**: The standard Linux subsystem for dynamic device detection and management. It allows us to query for devices based on their properties and tags.
- Multi-Seat on Linux from Freedesktop:
  - Defines seats as a set of devices integrated with systemd/udev

oci-dev-binder-hook

- An oci-hook that inspects the udev database looking for the devices that you want to include.
- The hook can be configured using container annotations
  - podman run --annotation io.dev-binder.udev.seat=seat0 -it my-application:latest
- We can let the dynamic device management to udev and just retrieve the tagged devices.

Demo

```
oci-dev-binder-hook on  main [$?] is 📦 v0.1.0 via 🦀 v1.89.0
❯ podman run --annotation=io.dev-binder.udev.seat=seat0 --rm -it fedora find /dev/dri/card*
/dev/dri/card1
```

Next steps (?)

- Move the project to github.com/containers
- Improve the annotations' semantics
  - Filter by subsystem tag (?)

Thanks!
Questions?