Device Whitelist Controller

1. Description:

Implement a cgroup to track and enforce open and mknod restrictions on device files. A device cgroup associates a device access whitelist with each cgroup. A whitelist entry has 4 fields. 'type' is a (all), c (char), or b (block). 'all' means it applies to all types and all major and minor numbers. Major and minor are either an integer or * for all. Access is a composition of r (read), w (write), and m (mknod).

The root device cgroup starts with rwm to 'all'. A child device cgroup gets a copy of the parent. Administrators can then remove devices from the whitelist or add new entries. A child cgroup can never receive a device access which is denied by its parent. However when a device access is removed from a parent it will not also be removed from the child(ren).

2. User Interface

An entry is added using devices.allow, and removed using devices.deny. For instance

```
        echo 'c 1:3 mr' > /cgroups/1/devices.allow
```

allows cgroup 1 to read and mknod the device usually known as /dev/null. Doing

```
        echo a > /cgroups/1/devices.deny
```

will remove the default 'a *:* rwm' entry. Doing

```
        echo a > /cgroups/1/devices.allow
```

will add the 'a *:* rwm' entry to the whitelist.

3. Security

Any task can move itself between cgroups. This clearly won't suffice, but we can decide the best way to adequately restrict movement as people get some experience with this. We may just want to require CAP_SYS_ADMIN, which at least is a separate bit from CAP_MKNOD. We may want to just refuse moving to a cgroup which isn't a descendant of the current one. Or we may want to use CAP_MAC_ADMIN, since we really are trying to lock down root.

CAP_SYS_ADMIN is needed to modify the whitelist or move another task to a new cgroup. (Again we'll probably want to change that).

A cgroup may not be granted more permissions than the cgroup's parent has.