Contents

[References: - 1](#_Toc162198364)

# How Information is Passed from Kernel to User Space

# The Linux kernel version 2.6.10 (and later versions) introduced the uevent notification mechanism for kernel and user-space communication. These user events (uevents) generated from the kernel are used by user space daemons to either create/ remove device files, to run programs, or load/remove a driver in user land. Inside the kernel, these uevents are linked to the kernel data structure called kobject. This data structure’s life cycle (linked with a device) is what is notified as uevents to user space.

## Netlink

The Linux kernel uses netlink to send kernel uevents to the user space. Netlink is a socket-like mechanism used in Linux to pass information between kernel and user processes. Netlink, similar to a generic BSD socket infrastructure, supports primitive APIs like socket(), bind(), sendmsg() and recvmsg().

## Uevent actions

A uevent message communicates the device’s or subsystems’ states inside the kernel to user space.

The kobject\_action data structure indicates the kernel object’s state, and is defined in include/linux/kobject.h as enum kobject\_action:

enum kobject\_action {

KOBJ\_ADD,

KOBJ\_REMOVE,

KOBJ\_CHANGE,

KOBJ\_MOVE,

KOBJ\_ONLINE,

KOBJ\_OFFLINE,

KOBJ\_MAX

};

Each uevent message sent to user space is tagged with one of these kobject actions. For example, actions like KOBJ\_ADD or KOBJ\_REMOVE are used to notify the addition or deletion of a kernel object—which happen when a device is added or deleted inside the kernel, using device\_add or device\_del. KOBJ\_CHANGE is the action most used by drivers to notify changes in the device, like the configuration or state. Other actions like KOBJ\_ONLINE/KOBJ\_OFFLINE are generally used to indicate when a CPU’s state changed; KOBJ\_MOVE is used only by network interfaces when renaming a kobject.

 kernel uevent framework: lib/kobject\_uevent.c

The uevent framework exports two important APIs:

1. kobject\_uevent\_env()

It allows a driver to send extra information to user space. This extra information is termed ‘environmental data’ by the framework. Internal to the framework

1. kobject\_uevent

The kobject\_uevent API is just a call of the kobject\_uevent\_env API with ‘environmental data’ set to NULL.

The basic message passed to user space from the kernel uevent framework is as follows:

ACTION= DEVPATH= SUBSYSTEM= SEQNUM=

The DEVPATH string holds the path of the kobject and SUBSYSTEM indicates the subsystem the uevent originated from. Any extra data other than the above four, is termed as environment data.

Information in this environment data is specific to the subsystem or the kernel object. The environment data, which is basically an array of strings, holds state changes or other information useful to user-space code.

The environment data is packed in a kobj\_uevent\_env data structure, defined in /include/linux/kobject.h:

struct kobj\_uevent\_env

{

char \*envp[UEVENT\_NUM\_ENVP];

int envp\_idx;

char buf[UEVENT\_BUFFER\_SIZE];

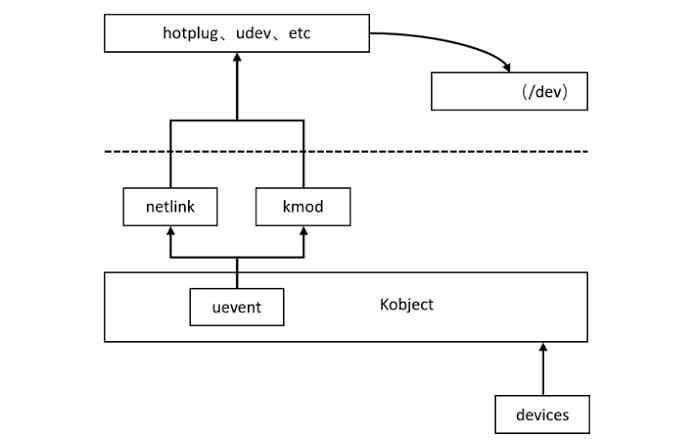
int buflen;

};

The uevent framework provides the add\_uevent\_var API to add more environment data to a uevent message.

Mechanism for flow of uevents from kernel to user space: -

1. Kernel driver sends uevent message
2. netlink driver broadcasts the uevent messages
3. User Space client processes uevent messages



# How it is used by a driver

When an Android device is connected as a USB device, the Android framework notifies the user with a USB icon in the notification bar. To do this, the Android USB framework needs certain information from the kernel when the Android device is configured as a USB device. This information is passed on through uevents by the kernel USB driver. Let us see how USB-specific state information is passed on to the Android USB framework.

The following code is from the android\_work function of the drivers/usb/gadget/android.c file.

# References: -

1. <https://linux-kernel-labs.github.io/refs/pull/183/merge/labs/device_model.html>
2. <https://issuu.com/hibadweib/docs/open_source_for_you_-_october_2012/s/13663276>
3. <https://linux-kernel-labs.github.io/refs/heads/master/labs/device_model.html>