UEVENT事件格式

2017年6月2日
20:51

由内核发出 **event** 事件.

1. kobject_uevent() 产生 uevent 事件(lib/kobject_uevent.c 中), 产生的 uevent 先由 netlink_broadcast_filtered() 发出, 最后调用 uevent_helper[] 所指定的程序来处理.
2. uevent_helper[] 里默认指定 "/sbin/hotplug", 但可以通过 /sys/kernel/uevent_helper (kernel/ksysfs.c) 或 /proc/kernel/uevent_helper (kernel/sysctl.c) 来修改成指定的程序.
3. 在 OpenWrt 并不使用 user_helper[] 指定程序来处理 uevent (/sbin/hotplug 不存在), 而是使用 PF_NETLINK 来获取来自内核的 uevent.

用户空间监听 **uevent**

openwrt 中, procd 作为 init 进程会处理许多事情, 其中就包括 hotplug.

procd/plug/hotplug.c 中, 创建一个 PF_NETLINK 套接字来监听内核 netlink_broadcast_filtered() 发出的 uevent.

收到 uevent 之后, 再根据 /etc/hotplug.json 里的描述来处理.

通常情况下, /etc/hotplug.json 会调用 /sbin/hotplug-call 来处理, 它根据 uevent 的 $SUBSYSTEM 变量来分别调用 /etc/hotplug.d/ 下不同目录中的脚本.

比如, 插入U盘或SD卡时, 会产生的事件消息如下:

```
procd: rule_handle_command(355): Command: makedev
procd: rule_handle_command(357):  /dev/sda1
procd: rule_handle_command(357):  0644
procd: rule_handle_command(358):
procd: rule_handle_command(360): Message:
procd: rule_handle_command(362):  ACTION=add
procd: rule_handle_command(362):
DEVPATH=/devices/101c0000.ehci/usb1/1-1/1-1.3/1-1.3:1.0/host16/target16:0:0/1
6:0:0:0/block/sda/sda1
procd: rule_handle_command(362):  SUBSYSTEM=block
procd: rule_handle_command(362):  MAJOR=8
procd: rule_handle_command(362):  MINOR=1
procd: rule_handle_command(362):  DEVNAME=sda1
procd: rule_handle_command(362):  DEVTYPE=partition
procd: rule_handle_command(362):  SEQNUM=865
procd: rule_handle_command(363):
procd: rule_handle_command(355): Command: exec
procd: rule_handle_command(357):  /sbin/hotplug-call
procd: rule_handle_command(357):  block
procd: rule_handle_command(358):
procd: rule_handle_command(360): Message:
procd: rule_handle_command(362):  ACTION=add
procd: rule_handle_command(362):
DEVPATH=/devices/101c0000.ehci/usb1/1-1/1-1.3/1-1.3:1.0/host16/target16:0:0/1
6:0:0:0/block/sda/sda1
procd: rule_handle_command(362):  SUBSYSTEM=block
procd: rule_handle_command(362):  MAJOR=8
procd: rule_handle_command(362):  MINOR=1
procd: rule_handle_command(362):  DEVNAME=sda1
procd: rule_handle_command(362):  DEVTYPE=partition
procd: rule_handle_command(362):  SEQNUM=865
procd: rule_handle_command(363):
```

第一个 makedev 会创建 /dev/sda1 节点. 第二个 exec 命令, 其附带的消息中指定了 ACTION, DEVPATH, SUBSYSTEM, DEVNAME, DEVTYPE 等变量.

于是 hotplug-call 会尝试执行 /etc/hotplug.d/block/ 目录下的所有可执行脚本.

所以我们可以在这里放置我们的自动挂载/卸载处理脚本.

按键 button 的检测

openwrt 中, 按键的检测也是通过 hotplug 来实现的.

它首先写了一个内核模块: gpio_button_hotplug, 用于监听按键, 有中断和 poll 两种方式. 然后在发出事件的同时, 将记录并计算得出的两次按键时间差也作为 uevent 变量发出来.

这样在用户空间收到这个 uevent 事件时就知道该次按键按下了多长时间.

hotplug.json 中有描述, 如果 uevent 中含有 BUTTON 字符串, 而且 SUBSYSTEM 为 "button", 则执行 /etc/rc.button/ 下的 %BUTTON% 脚本来处理.

```
        [ "if",
                [ "and",
                        [ "has", "BUTTON" ],
                        [ "eq", "SUBSYSTEM", "button" ],
                ],
                [ "exec", "/etc/rc.button/%BUTTON%" ]
        ],
```

使用 export DBGLVL=10; procd -h /etc/hotplug.json 截获一些打印信息看看:

```
{{"HOME":"\/","PATH":"\/sbin:\/bin:\/usr\/sbin:\/usr
\/bin","SUBSYSTEM":"button","ACTION":"pressed","BUTTON":"reset","SEEN":"862",
"SEQNUM":"593"}}
procd: rule_handle_command(355): Command: exec
procd: rule_handle_command(357):  /etc/rc.button/reset
procd: rule_handle_command(358):
procd: rule_handle_command(360): Message:
procd: rule_handle_command(362):  HOME=/
procd: rule_handle_command(362):  PATH=/sbin:/bin:/usr/sbin:/usr/bin
procd: rule_handle_command(362):  SUBSYSTEM=button
procd: rule_handle_command(362):  ACTION=pressed
procd: rule_handle_command(362):  BUTTON=reset
procd: rule_handle_command(362):  SEEN=862
procd: rule_handle_command(362):  SEQNUM=593
procd: rule_handle_command(363):

procd: rule_handle_command(355): Command: exec
procd: rule_handle_command(357):  /sbin/hotplug-call
procd: rule_handle_command(357):  button
procd: rule_handle_command(358):
procd: rule_handle_command(360): Message:
procd: rule_handle_command(362):  HOME=/
procd: rule_handle_command(362):  PATH=/sbin:/bin:/usr/sbin:/usr/bin
procd: rule_handle_command(362):  SUBSYSTEM=button
procd: rule_handle_command(362):  ACTION=pressed
procd: rule_handle_command(362):  BUTTON=reset
procd: rule_handle_command(362):  SEEN=862
procd: rule_handle_command(362):  SEQNUM=593
procd: rule_handle_command(363):

{{"HOME":"\/","PATH":"\/sbin:\/bin:\/usr\/sbin:\/usr
\/bin","SUBSYSTEM":"button","ACTION":"released","BUTTON":"reset","SEEN":"3","
SEQNUM":"594"}}
procd: rule_handle_command(355): Command: exec
procd: rule_handle_command(357):  /etc/rc.button/reset
procd: rule_handle_command(358):
procd: rule_handle_command(360): Message:
procd: rule_handle_command(362):  HOME=/
procd: rule_handle_command(362):  PATH=/sbin:/bin:/usr/sbin:/usr/bin
procd: rule_handle_command(362):  SUBSYSTEM=button
procd: rule_handle_command(362):  ACTION=released
procd: rule_handle_command(362):  BUTTON=reset
procd: rule_handle_command(362):  SEEN=3
procd: rule_handle_command(362):  SEQNUM=594
procd: rule_handle_command(363):

procd: rule_handle_command(355): Command: exec
```

```
procd: rule_handle_command(357):  /sbin/hotplug-call
procd: rule_handle_command(357):  button
procd: rule_handle_command(358):
procd: rule_handle_command(360): Message:
procd: rule_handle_command(362):  HOME=/
procd: rule_handle_command(362):  PATH=/sbin:/bin:/usr/sbin:/usr/bin
procd: rule_handle_command(362):  SUBSYSTEM=button
procd: rule_handle_command(362):  ACTION=released
procd: rule_handle_command(362):  BUTTON=reset
procd: rule_handle_command(362):  SEEN=3
procd: rule_handle_command(362):  SEQNUM=594
procd: rule_handle_command(363):
```

从 <http://www.cnblogs.com/sammei/p/4119659.html> 插入