使用printk记录消息

printk()是Linux内核中最广为人知的函数之一。它是我们打印消息的标准工具，通常也是追踪和调试 的最基本方法。如果你熟悉printf(3)，你就能够知道**printk()**是基于它的，尽管它在功能上有一些不 同之处:

- **printk()** 消息可以指定日志级别。
- 格式字符串虽然与C99基本兼容，但并不遵循完全相同的规范。它有一些扩展和一些限制（没 有 `%n` 或浮点转换指定符）。参见:ref: 如何正确地获得printk格式指定符 <printk-specifiers> 。

所有的**printk()**消息都会被打印到内核日志缓冲区，这是一个通过/dev/kmsg输出到用户空间的环形缓冲区。读取它的通常方法是使用 `dmesg` 。

printk()的用法通常是这样的:

```
printk(KERN_INFO "Message: %s\n", arg);
```

其中 `KERN_INFO` 是日志级别（注意，它与格式字符串连在一起，日志级别不是一个单独的参数）。 可用的日志级别是:

名称	字符串	别名函数
KERN_EMERG	“0”	**pr_emerg()**
KERN_ALERT	“1”	**pr_alert()**
KERN_CRIT	“2”	**pr_crit()**
KERN_ERR	“3”	**pr_err()**
KERN_WARNING	“4”	**pr_warn()**
KERN_NOTICE	“5”	**pr_notice()**
KERN_INFO	“6”	**pr_info()**
KERN_DEBUG	“7”	**pr_debug()** and **pr_devel()** 若定义了DEBUG
KERN_DEFAULT	“”	
KERN_CONT	“c”	**pr_cont()**

日志级别指定了一条消息的重要性。内核根据日志级别和当前 console_loglevel （一个内核变量）决 定是否立即显示消息（将其打印到当前控制台）。如果消息的优先级
比 console_loglevel 高（日志级 别值较低），消息将被打印到控制台。

如果省略了日志级别，则以 `KERN_DEFAULT` 级别打印消息。

你可以用以下方法检查当前的 console_loglevel

```
$ cat /proc/sys/kernel/printk
```

```
4       4       1       7
```

结果显示了 current, default, minimum 和 boot-time-default 日志级别

要改变当前的 console_loglevel，只需在 /proc/sys/kernel/printk 中写入所需的 级别。例如，要打印所有的消息到控制台上:

```
# echo 8 > /proc/sys/kernel/printk
```

另一种方式，使用 dmesg:

```
# dmesg -n 5
```

设置 console_loglevel 打印 KERN_WARNING (4) 或更严重的消息到控制台。更多消息参见 dmesg(1) 。

作为**printk()**的替代方案，你可以使用 pr_*() 别名来记录日志。这个系列的宏在宏名中 嵌入了日志级别。例如:

```
pr_info("Info message no. %d\n", msg_num);
```

打印 KERN_INFO 消息。

除了比等效的**printk()**调用更简洁之外，它们还可以通过**pr_fmt()**宏为格式字符串使用一个通用 的定义。例如，在源文件的顶部（在任何 #include 指令之前）定义这样的内容。:

```
#define pr_fmt(fmt) "%s:%s: " fmt, KBUILD_MODNAME, __func__
```

会在该文件中的每一条 pr_*() 消息前加上发起该消息的模块和函数名称。

为了调试，还有两个有条件编译的宏: **pr_debug()**和**pr_devel()**，除非定义了 DEBUG （或者在**pr_debug()**的情况下定义了 CONFIG_DYNAMIC_DEBUG ），否则它们会被编译。

函数接口

该API在以下内核代码中:

include/linux/printk.h