Linux 的 timerfd 分析

timerfd 是 Linux 为用户程序提供的一个定时器接口。这个接口基于文件描述符，所以能够被用于 select/poll 的应用场景。

1. 使用方法

timerfd 提供了如下接口供用户使用

timerfd_create

int timerfd_create(int clockid, int flags);

timerfd_create 用于创建一个定时器文件。

参数 clockid 可以是 CLOCK_MONOTONIC 或者 CLOCK_REALTIME。

参数 flags 可以是 0 或者 O_CLOEXEC/O_NONBLOCK。

函数返回值是一个文件句柄 fd。

timerfd_settime

int timerfd_settime(int ufd, int flags, const struct itimerspec * utmr, struct itimerspec * otmr);

此函数用于设置新的超时时间，并开始计时。

参数 ufd 是 timerfd_create 返回的文件句柄。

参数 flags 为 1 代表设置的是绝对时间；为 0 代表相对时间。

参数 utmr 为需要设置的时间。

参数 otmr 为定时器这次设置之前的超时时间。

函数返回 0 代表设置成功。

timerfd_gettime

int timerfd_gettime(int ufd, struct itimerspec * otmr);

此函数用于获得定时器距离下次超时还剩下的时间。如果调用时定时器已经到期，并且该定时器处于循环模式（设置超时时间时 struct itimerspec::it_interval 不为 0），那么调用此函数之后定时器重新开始计时。

read

当 timerfd 为阻塞方式时，read 函数将被阻塞，直到定时器超时。

函数返回值大于 0，代表定时器超时；否则，代表没有超时（被信号唤醒，等等）。

poll/close

poll，close 与标准文件操作相同。

2. 内核实现

timerfd 的内核实现代码在 kernel/fs/timerfd.c，它的实现基于 Linux 的 hrtimer。

timerfd_create 的实现

SYSCALL_DEFINE2(timerfd_create, int, clockid, int, flags)

- 做一些定时器的初始化工作
- 调用 hrtimer_init 初始化一个 hrtimer
- 调用 anon_inode_getfd 分配一个 dentry，并得到一个文件号 fd，同时传入 timerfd 的文件操作指针 struct file_operations timerfd_fops。anno_inode_getfd 是文件系统 anon_inodefs 的一个帮助函数。anon 文件系统比较简单，整个文件系统只有一个 inode 节点，其实现代码可以在 fs/anon_inodes.c 中找到。

timerfd_settime 的实现

timerfd_settime 最终会调用 hrtimer_start 启动定时器，其超时函数被设置为 timerfd_tmrproc。

timerfd_tmrproc

timefd_tmrproc 是 timerfd 的定时器超时函数。在 timerfd 超时时，该函数会设置定时器超时标记位；增加定时器超时次数（在设置定时器循环模式时，可能会出现多次超时没有被处理的情况）；唤醒一个等待队列，从而唤醒可能存在的正被阻塞的 read、select。

timerfd_fops

```
static const struct file_operations timerfd_fops = {
    .release    = timerfd_release,
    .poll       = timerfd_poll,
    .read           = timerfd_read,
};
```

timerfd_read 函数是文件操作 read 的内核实现，读到的是定时器的超时次数。该函数在阻塞模式下会把自身挂到 timerfd 的等待队列中，等待定时器超时时被唤醒。

timerfd_poll 将 timerfd 的等待队列登记到一个 poll_table，从而在定时器超时时能唤醒 select 系统调用。

timerfd_release

timerfd_release 函数释放 timerfd_create 函数中申请的资源，删除已分配的定时器。