File system Monitoring with fanotify

File system Error Reporting

Fanotify supports the FAN_FS_ERROR event type for file system-wide error reporting. It is meant to be used by file system health monitoring daemons, which listen for these events and take actions (notify sysadmin, start recovery) when a file system problem is detected.

By design, a FAN_FS_ERROR notification exposes sufficient information for a monitoring tool to know a problem in the file system has happened. It doesn't necessarily provide a user space application with semantics to verify an IO operation was successfully executed. That is out of scope for this feature. Instead, it is only meant as a framework for early file system problem detection and reporting recovery tools.

When a file system operation fails, it is common for dozens of kernel errors to cascade after the initial failure, hiding the original failure log, which is usually the most useful debug data to troubleshoot the problem. For this reason, FAN_FS_ERROR tries to report only the first error that occurred for a file system since the last notification, and it simply counts additional errors. This ensures that the most important pieces of information are never lost.

FAN_FS_ERROR requires the fanotify group to be setup with the FAN_REPORT_FID flag.

At the time of this writing, the only file system that emits FAN_FS_ERROR notifications is Ext4.

A FAN_FS_ERROR Notification has the following format:

```
::

   [ Notification Metadata (Mandatory) ]
   [ Generic Error Record  (Mandatory) ]
   [ FID record            (Mandatory) ]
```

The order of records is not guaranteed, and new records might be added in the future. Therefore, applications must not rely on the order and must be prepared to skip over unknown records. Please refer to `samples/fanotify/fs-monitor.c` for an example parser.

Generic error record

The generic error record provides enough information for a file system agnostic tool to learn about a problem in the file system, without providing any additional details about the problem. This record is identified by `struct fanotify_event_info_header.info_type` being set to FAN_EVENT_INFO_TYPE_ERROR.

```
struct fanotify_event_info_error {
     struct fanotify_event_info_header hdr;
    __s32 error;
    __u32 error_count;
};
```

The *error* field identifies the type of error using errno values. *error_count* tracks the number of errors that occurred and were suppressed to preserve the original error information, since the last notification.

FID record

The FID record can be used to uniquely identify the inode that triggered the error through the combination of fsid and file handle. A file system specific application can use that information to attempt a recovery procedure. Errors that are not related to an inode are reported with an empty file handle of type FILEID_INVALID.