sytemcall fdatasync ()

The fdatasync() system call on openSUSE (and other Linux systems) is used to flush all modified data of a file descriptor to disk, ensuring that the file's data is physically stored on the disk device. fdatasync() focuses on flushing the file's data and the minimal metadata required to retrieve that data (e.g., file length), which can improve performance.

Syntax:

```
int fdatasync(int fd);
```

Parameters:

fd: File descriptor of the file to be synchronized.

Use case:

Useful when you want to ensure data integrity (e.g., in databases or file transactions) without the overhead of syncing all metadata.

User Space: glibc Wrapper

In openSUSE, which uses glibc as the C standard library, the user-level function fdatasync() is provided as a wrapper:

glibc source (e.g., /usr/include/unistd.h):

```
extern int fdatasync(int fd);
```

glibc implementation (glibc/sysdeps/unix/sysv/linux/fdatasync.c):

```
int fdatasync(int fd) {
   return INLINE_SYSCALL(fdatasync, 1, fd);
}
```

This wraps the syscall number and arguments to make a trap into kernel space.

4. Kernel Entry Point

Kernel syscall handler (in fs/sync.c):

```
SYSCALL_DEFINE1(fdatasync, int, fd)
{
    struct fd f = fdget(fd);
    int ret = -EBADF;

    if (f.file) {
        ret = vfs_fsync(f.file, 1);
        fdput(f);
    }

    return ret;
}
```

fdget() obtains the struct file from the file descriptor.

vfs_fsync() is called with datasync = 1, meaning it will flush data but may skip some metadata.