Ruby master - Bug #14434

IO#reopen fails after EPIPE

02/02/2018 10:32 AM - nobu (Nobuyoshi Nakada)

Status:	Open		
Priority:	Normal		
Assignee:			
Target version:			
ruby -v:		**Backport:**	2.3: UNKNOWN, 2.4: UNKNOWN, 2.5: UNKNOWN

Description

Consider the following code, which emulates yes | head -1.

```
IO.popen("head -1", "w") do |f|
  f.sync = false
  stdout = STDOUT.dup
  STDOUT.reopen(f)
  loop{puts "y"} rescue break $!
ensure
  STDOUT.reopen(stdout)
end
```

This fails with Errno::EPIPE in IO#reopen.

```
Traceback (most recent call last):
    4: from -:1:in `<main>'
    3: from -:1:in `popen'
    2: from -:7:in `block in <main>'
    1: from -:7:in `ensure in block in <main>'
-:7:in `reopen': Broken pipe (Errno::EPIPE)
```

BTW, this "broken pipe" IO can't even close.

```
r, w = IO.pipe
w.sync = false # make buffered
w.print("foo")
r.close
# w.reopen(STDOUT) rescue p $! #=> #<Errno::EPIPE: Broken pipe>
w.close rescue p $! #=> #<Errno::EPIPE: Broken pipe>
```

This is caused by flushing buffered data.
I think there is no way to recover "broken pipe" FD, so nothing can be done for the remained data and should be discarded gently.

```
diff --git i/io.c w/io.c
index 0a4e66f5ed..5eca6762f2 100644
--- i/io.c
+++ w/io.c
@@ -7150,8 +7150,7 @@ io_reopen(VALUE io, VALUE nfile)
     }
      }
      if (fptr->mode & FMODE_WRITABLE) {
-         if (io_fflush(fptr) < 0)
-             rb_sys_fail(0);
+         fptr_finalize_flush(fptr, TRUE, FALSE);
      }
      else {
     io_tell(fptr);
@@ -7177,7 +7176,8 @@ io_reopen(VALUE io, VALUE nfile)
      if (fd != fd2) {
     if (IS_PREP_STDIO(fptr) || fd <= 2 || !fptr->stdio_file) {
         /* need to keep FILE objects of stdin, stdout and stderr */
-        if (rb_cloexec_dup2(fd2, fd) < 0)
```

```
+       fd = (fd < 0) ? rb_cloexec_dup(fd2) : rb_cloexec_dup2(fd2, fd);
+       if (fd < 0)
        rb_sys_fail_path(orig->pathv);
             rb_update_max_fd(fd);
    }
```