

---

Subject: [PATCH 0/4][cryo] Test pipes  
Posted by [Sukadev Bhattiprolu](#) on Tue, 24 Jun 2008 03:31:33 GMT  
[View Forum Message](#) <> [Reply to Message](#)

---

PATCH[1/4]: Support-multiple-pipe-test-cases.patch  
PATCH[2/4]: Testcase-3-continous-read-write-to-pipe.patch  
PATCH[3/4]: Test4-Non-consecutive-pipe-fds.patch  
PATCH[4/4]: Test-5-Read-write-using-dup-of-pipe-fds.patch

---

Containers mailing list  
Containers@lists.linux-foundation.org  
<https://lists.linux-foundation.org/mailman/listinfo/containers>

---

---

Subject: [PATCH 1/4]: Support multiple pipe test cases  
Posted by [Sukadev Bhattiprolu](#) on Tue, 24 Jun 2008 03:32:39 GMT  
[View Forum Message](#) <> [Reply to Message](#)

---

>From a99deb9bcdd611c52589fa733dd90057f1f134bf Mon Sep 17 00:00:00 2001  
From: Sukadev Bhattiprolu <sukadev@linux.vnet.ibm.com>  
Date: Sun, 22 Jun 2008 21:05:48 -0700  
Subject: [PATCH] Support multiple pipe test cases

Modify pipe.c to support multiple test cases and to select a test case using the -t option.

Implement two tests:

- empty pipe
- single write to pipe followed by continuous read

---

tests/pipe.c | 93 +++-----  
1 files changed, 75 insertions(+), 18 deletions(-)

diff --git a/tests/pipe.c b/tests/pipe.c  
index 0812cb3..76be6cc 100644

```
--- a/tests/pipe.c
+++ b/tests/pipe.c
@@ -1,52 +1,109 @@
#include <stdio.h>
#include <sys/types.h>
+#include <sys/fcntl.h>
#include <unistd.h>
#include <stdlib.h>
#include <string.h>
#include <errno.h>
-#include <sys/fcntl.h>
```

```
-int main(int argc, char *argv[])
```

```

+char *test_descriptions[] = {
+ NULL,
+ "Test with an empty pipe",
+ "Test with single-write to and continous read from a pipe",
+ "Test continous reads/writes from pipe",
+ "Test non-consecutive pipe-fds",
+ "Test with read-fd > write-fd",
+ "Test with read-fd/write-fd swapped",
+ "Test with all-fds in use",
+ "Test with all-fds in use for pipes",
+};
+
+static int last_num = 2;
+usage(char *argv[])
+{
+ int i;
+ printf("Usage: %s -t <test-number>\n", argv[0]);
+ printf("\t where <test-number is in range 1..%d\n", last_num);
+ for (i = 0; i < last_num; i++)
+ printf("\t test-%d: %s\n", i, test_descriptions[i]);
+ exit(1);
+}
+
+int test1()
+{
+ int i = 0;
- int rc;
+ int fds[2];
+
+ if (pipe(fds) < 0) {
+ perror("pipe()");
+ exit(1);
+ }
+
+ printf("Running as %d\n", getpid());
+ while (i<100) {
+ sleep(1);
+ printf("i is %d (pid %d)\n", i, getpid());
+ i++;
+ }
+}
+
+int test2()
+{
+ int i = 0;
+ int c;
- int empty;
+ int rc;

```

```

+ int fds[2];
  char *buf = "abcdefghijklmnopqrstuvwxy";

- /*
-  * -e: test with an empty pipe
-  */
- empty = 0;
- if (argc > 1 && strcmp(argv[1], "-e") == 0)
-   empty = 1;
-
  if (pipe(fds) < 0) {
    perror("pipe()");
    exit(1);
  }

- if (!empty)
-   write(fds[1], buf, strlen(buf));
+ write(fds[1], buf, strlen(buf));

  if (fcntl(fds[0], F_SETFL, O_NONBLOCK) < 0) {
    perror("fcntl()");
    exit(1);
  }
+
  printf("Running as %d\n", getpid());
  while (i<100) {
    sleep(1);
    if (i%5 == 0) {
      c = errno = 0;
      rc = read(fds[0], &c, 1);
-   if (rc != 1) {
-     perror("read() failed");
-   }
-   printf("i is %d (pid %d), c is %c\n", i, getpid(), c);
-
+   if (rc != 1)
+     perror("read() pipe failed\n");
+   printf("i is %d (pid %d), next byte is %d\n", i,
+         getpid(), c);
  }
  i++;
}

+int
+main(int argc, char *argv[])
+{
+ int c;

```

```
+ int tc_num;
+
+ while((c = getopt(argc, argv, "t:")) != EOF) {
+ switch(c) {
+ case 't':
+ tc_num = atoi(optarg);
+ break;
+ default:
+ printf("Unknown option %c\n", c);
+ usage(argv);
+ }
+ }
+
+ switch(tc_num) {
+ case 1: test1(); break;
+ case 2: test2(); break;
+ default:
+ printf("Unsupported test case %d\n", tc_num);
+ usage(argv);
+ }
+}
--
1.5.2.5
```

---

Containers mailing list

Containers@lists.linux-foundation.org

<https://lists.linux-foundation.org/mailman/listinfo/containers>

---

---

Subject: [PATCH 2/4]: Test3: continous read/write to pipe

Posted by [Sukadev Bhattiprolu](#) on Tue, 24 Jun 2008 03:33:25 GMT

[View Forum Message](#) <> [Reply to Message](#)

---

>From ade1b719f7d9968e0f934daf736ca1746cb6747d Mon Sep 17 00:00:00 2001

From: Sukadev Bhattiprolu <sukadev@linux.vnet.ibm.com>

Date: Sun, 22 Jun 2008 22:26:18 -0700

Subject: [PATCH] Testcase 3: continous read/write to pipe

---

```
tests/pipe.c | 82 ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
1 files changed, 81 insertions(+), 1 deletions(-)
```

```
diff --git a/tests/pipe.c b/tests/pipe.c
```

```
index 76be6cc..c81ac2a 100644
```

```
--- a/tests/pipe.c
```

```
+++ b/tests/pipe.c
```

```
@ @ -6,6 +6,8 @ @
```

```

#include <string.h>
#include <errno.h>

#define min(a, b) ((a) < (b) ? (a) : (b))
+static char *temp_file;
char *test_descriptions[] = {
    NULL,
    "Test with an empty pipe",
@@ -18,7 @@ char *test_descriptions[] = {
    "Test with all-fds in use for pipes",
};

-static int last_num = 2;
+static int last_num = 3;
usage(char *argv[])
{
    int i;
@@ -82,12 +84,89 @@ int test2()
}
}

+int read_write_pipe()
+{
+ int i = 0;
+ int rc;
+ int fds[2];
+ int fd1, fd2;
+ int c;
+ char *wbufp;
+ char wbuf[256];
+ char rbuf[256];
+ char *rbufp;
+
+ rbufp = &rbuf[0];
+ wbufp = &wbuf[0];
+
+ strcpy(wbufp, "abcdefghijklmnopqrstuvwxyABCDEFGHIJKLMNOPQRSTUVWXYZ");
+ memset(rbufp, '\0', sizeof(rbuf));
+
+ if (pipe(fds) < 0) {
+ perror("pipe()");
+ exit(1);
+ }
+ printf("fds[0] %d, fds[1] %d\n", fds[0], fds[1]);
+
+ if (fcntl(fds[0], F_SETFL, O_NONBLOCK) < 0) {
+ perror("fcntl()");
+ exit(1);

```

```

+ }
+
+ printf("Running as %d\n", getpid());
+ for (i = 0; i < 50; i++) {
+ sleep(1);
+ if (i%2 == 0) {
+ c = errno = 0;
+ rc = read(fds[0], rbufp, 3);
+
+ if (rc < 0)
+ perror("read() failed");
+ else {
+ printf("i is %d (pid %d), rbufp %p read %s\n",
+ i, getpid(), rbufp, rbufp);
+ rbufp += rc;
+ }
+
+ if (*wbufp == '\0')
+ continue;
+
+ errno = 0;
+ rc = write(fds[1], wbufp, min(3, strlen(wbufp)));
+ if (rc < 0) {
+ perror("write() to pipe");
+ } else {
+ if (rc != 3) {
+ printf("Wrote %d of 3 bytes, "
+ "error %d\n", rc, errno);
+ }
+ wbufp += rc;
+ }
+ }
+ }
+
+ if (strncmp(wbuf, rbuf, strlen(wbufp))) {
+ printf("Wrote: %s\n", wbuf);
+ printf("Read : %s\n", rbuf);
+ printf("Test FAILED\n");
+ } else {
+ printf("Test passed\n");
+ }
+}
+
+static void test3()
+{
+ read_write_pipe();
+}
+

```

```

int
main(int argc, char *argv[])
{
    int c;
    int tc_num;

+ temp_file = argv[0];
+
    while((c = getopt(argc, argv, "t:")) != EOF) {
        switch(c) {
            case 't':
@@ -102,6 +181,7 @@ main(int argc, char *argv[])
        switch(tc_num) {
            case 1: test1(); break;
            case 2: test2(); break;
+ case 3: test3(); break;
            default:
                printf("Unsupported test case %d\n", tc_num);
                usage(argv);
--
1.5.2.5

```

---

Containers mailing list  
Containers@lists.linux-foundation.org  
<https://lists.linux-foundation.org/mailman/listinfo/containers>

---



---

Subject: [PATCH 3/4][cryo]: Test 4: Non-consecutive pipe fds  
Posted by [Sukadev Bhattiprolu](#) on Tue, 24 Jun 2008 03:34:12 GMT  
[View Forum Message](#) <> [Reply to Message](#)

---

>From c7276c8cb59247faa13d42a1b871c853a80d80d1 Mon Sep 17 00:00:00 2001  
From: Sukadev Bhattiprolu <sukadev@linux.vnet.ibm.com>  
Date: Sun, 22 Jun 2008 22:43:01 -0700  
Subject: [PATCH] Test4: Non-consecutive pipe fds

---  
tests/pipe.c | 76 +++-----  
1 files changed, 66 insertions(+), 10 deletions(-)

```

diff --git a/tests/pipe.c b/tests/pipe.c
index c81ac2a..5b04f46 100644
--- a/tests/pipe.c
+++ b/tests/pipe.c
@@ -5,6 +5,7 @@
#include <stdlib.h>
#include <string.h>

```

```

#include <errno.h>
#include <sys/stat.h>

#define min(a, b) ((a) < (b) ? (a) : (b))
static char *temp_file;
@@ -20,7 +21,7 @@ char *test_descriptions[] = {
    "Test with all-fds in use for pipes",
};

-static int last_num = 3;
+static int last_num = 4;
usage(char *argv[])
{
    int i;
@@ -84,13 +85,50 @@ int test2()
}
}

-int read_write_pipe()
+
+reset_pipe_fds(int *tmpfds, int *testfds, int close_unused)
+{
+ struct stat statbuf;
+ int rc;
+
+ if (fstat(testfds[0], &statbuf) == 0) {
+ printf("fd %d is in use...\n", testfds[0]);
+ exit(1);
+ }
+ if (fstat(testfds[1], &statbuf) == 0) {
+ printf("fd %d is in use...\n", testfds[1]);
+ exit(1);
+ }
+
+ rc = dup2(tmpfds[0], testfds[0]);
+ if (rc < 0) {
+ printf("dup2(%d, %d) failed, error %d\n",
+ tmpfds[0], testfds[0], rc, errno);
+ exit(1);
+ }
+
+ rc = dup2(tmpfds[1], testfds[1]);
+ if (rc < 0) {
+ printf("dup2(%d, %d) failed, error %d\n",
+ tmpfds[1], testfds[1], rc, errno);
+ exit(1);
+ }
+

```



```

+ if (close_unused) {
+ close(tmpfds[0]);
+ close(tmpfds[1]);
+ }
+}
+
+#define TEST_NON_CONSECUTIVE_FD 0x1
+
+int read_write_pipe(int *testfdsp, int close_unused)
{
    int i = 0;
    int rc;
- int fds[2];
- int fd1, fd2;
+ int tmpfds[2];
    int c;
+ int read_fd, write_fd;
    char *wbufp;
    char wbuf[256];
    char rbuf[256];
@@ -102,13 +140,23 @@ int read_write_pipe()
    strcpy(wbufp, "abcdefghijklmnopqrstuvwxyABCDEFGHIJKLMNOPQRSTUVWXYZ");
    memset(rbuf, '\0', sizeof(rbuf));

- if (pipe(fds) < 0) {
+ if (pipe(tmpfds) < 0) {
    perror("pipe()");
    exit(1);
}
- printf("fds[0] %d, fds[1] %d\n", fds[0], fds[1]);

- if (fcntl(fds[0], F_SETFL, O_NONBLOCK) < 0) {
+ if (testfdsp) {
+ reset_pipe_fds(tmpfds, testfdsp, close_unused);
+ read_fd = testfdsp[0];
+ write_fd = testfdsp[1];
+ } else {
+ read_fd = tmpfds[0];
+ write_fd = tmpfds[1];
+ }
+
+ printf("read_fd %d, write_fd %d\n", read_fd, write_fd);
+
+ if (fcntl(read_fd, F_SETFL, O_NONBLOCK) < 0) {
    perror("fcntl()");
    exit(1);
}
@@ -118,7 +166,7 @@ int read_write_pipe()

```

```

sleep(1);
if (i%2 == 0) {
    c = errno = 0;
- rc = read(fds[0], rbufp, 3);
+ rc = read(read_fd, rbufp, 3);

    if (rc < 0)
        perror("read() failed");
@@ -132,7 +180,7 @@ int read_write_pipe()
    continue;

    errno = 0;
- rc = write(fds[1], wbufp, min(3, strlen(wbufp)));
+ rc = write(write_fd, wbufp, min(3, strlen(wbufp)));
    if (rc < 0) {
        perror("write() to pipe");
    } else {
@@ -156,7 +204,14 @@ int read_write_pipe()

static void test3()
{
- read_write_pipe();
+ read_write_pipe(NULL, 1);
+}
+
+static void test4()
+{
+ int tmpfds[2] = { 172, 101 };
+
+ read_write_pipe(tmpfds, 1);
+}

int
@@ -182,6 +237,7 @@ main(int argc, char *argv[])
    case 1: test1(); break;
    case 2: test2(); break;
    case 3: test3(); break;
+ case 4: test4(); break;
    default:
        printf("Unsupported test case %d\n", tc_num);
        usage(argv);
--
1.5.2.5

```

Subject: [PATCH 4/4][cryo]: Test 5: Read/write using dup() of pipe fds  
Posted by [Sukadev Bhattiprolu](#) on Tue, 24 Jun 2008 03:34:54 GMT  
[View Forum Message](#) <> [Reply to Message](#)

---

>From 121c33ebe1ae201d5bb051f9c76e3eaae29329bb Mon Sep 17 00:00:00 2001  
From: Sukadev Bhattiprolu <sukadev@linux.vnet.ibm.com>  
Date: Sun, 22 Jun 2008 23:33:24 -0700  
Subject: [PATCH] Test 5: Read/write using dup() of pipe fds

---  
tests/pipe.c | 19 ++++++-----  
1 files changed, 14 insertions(+), 5 deletions(-)

diff --git a/tests/pipe.c b/tests/pipe.c  
index 5b04f46..47f5da6 100644

--- a/tests/pipe.c  
+++ b/tests/pipe.c  
@@ -16,12 +16,13 @@ char \*test\_descriptions[] = {  
 "Test continous reads/writes from pipe",  
 "Test non-consecutive pipe-fds",  
 "Test with read-fd > write-fd",  
+ "Test with dup of pipe fds around"  
 "Test with read-fd/write-fd swapped",  
 "Test with all-fds in use",  
 "Test with all-fds in use for pipes",  
};

-static int last\_num = 4;  
+static int last\_num = 5;  
usage(char \*argv[])  
{  
 int i;  
@@ -145,13 +146,13 @@ int read\_write\_pipe(int \*testfdsp, int close\_unused)  
 exit(1);  
}

+ read\_fd = tmpfds[0];  
+ write\_fd = tmpfds[1];  
 if (testfdsp) {  
 reset\_pipe\_fds(tmpfds, testfdsp, close\_unused);  
+ /\* read from dup'd fds even if main ones are open \*/  
 read\_fd = testfdsp[0];  
 write\_fd = testfdsp[1];  
- } else {  
- read\_fd = tmpfds[0];  
- write\_fd = tmpfds[1];  
 }

printf("read\_fd %d, write\_fd %d\n", read\_fd, write\_fd);

```
@@ -193,7 +194,7 @@ int read_write_pipe(int *testfdsp, int close_unused)
}
}
```

```
- if (strncmp(wbuf, rbuf, strlen(wbuf))) {
+ if (strncmp(wbuf, rbuf, strlen(wbuf))) {
    printf("Wrote: %s\n", wbuf);
    printf("Read : %s\n", rbuf);
    printf("Test FAILED\n");
@@ -214,6 +215,13 @@ static void test4()
    read_write_pipe(tmpfds, 1);
}
```

```
+static void test5()
+{
+ int tmpfds[2] = { 172, 101 };
+
+ read_write_pipe(tmpfds, 0);
+}
+
int
main(int argc, char *argv[])
{
@@ -238,6 +246,7 @@ main(int argc, char *argv[])
    case 2: test2(); break;
    case 3: test3(); break;
    case 4: test4(); break;
+ case 5: test5(); break;
    default:
        printf("Unsupported test case %d\n", tc_num);
        usage(argv);
```

--

1.5.2.5

---

Containers mailing list  
Containers@lists.linux-foundation.org  
<https://lists.linux-foundation.org/mailman/listinfo/containers>

---