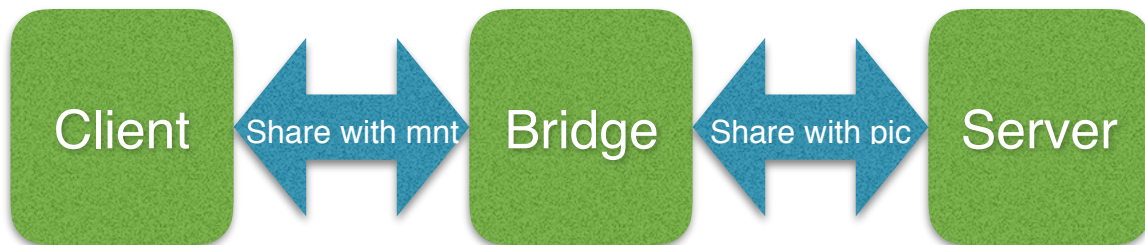


HW2_102062129許毓軒

1. Flow Chart



2. Implement

(1) Client with mnt (container1)

- 1) Initial inotify_event。
- 2) 使用inotify監控IN_DELETE(文件或目錄被刪除)，並將使用者的message寫進檔案"message"。
- 3) 使用while loop監控直到IN_DELETE發生，跳出迴圈進入下個步驟。
- 4) 如message為"exit"則程式結束。
- 5) 使用inotify監控IN_CLOSE_WRITE(以write方式打開文件並關閉)。
- 6) 使用while loop監控直到IN_CLOSE_WRITE發生，讀取檔案"message"得到回傳message，並跳出迴圈回到步驟2。

(2) Bridge with mnt and ipc (host)

- 1) set namespace with mnt Client, and set namespace with ipc server。
- 2) initial msg and inotify_event。
- 3) 使用inotify監控IN_CLOSE_WRITE。
- 4) 使用while loop監控直到IN_CLOSE_WRITE發生，讀取檔案"message"得到message，並跳出迴圈進入下個步驟2。
- 5) 使用msgsnd將message送到server。
- 6) 如message為"exit"則程式結束。
- 7) 使用msgrcv讀取server回傳的message。
- 8) 使用inotify監控IN_DELETE(文件或目錄被刪除)，並將server回傳的message寫進檔案"message"。
- 9) 使用while loop監控直到IN_DELETE發生，跳出迴圈回到步驟3。

(3) Sever with ipc (container2)

- 1) Initial msg。
- 2) 使用msgrcv讀取bridge傳來的message。
- 3) 如message為"exit"則程式結束。
- 4) 使用msgsnd將message傳回bridge。Implement

3. Result Snapshot

client

```
[susean@s102062129-0:~/HW2/10520CS542100/container1$ sudo runc run --pid-file /tmp/container1.pid container1  
[/ # ./client  
[123  
Send : 123  
Recv : 123  
[test  
Send : test  
Recv : test  
[exit  
Send : exit
```

bridge

```
[susean@s102062129-0:~/HW2/10520CS542100$ sudo ./bridge /proc/8445/ns/mnt /proc/8515/ns/ipc  
Bridge recv from mnt : 123  
Bridge send to ipc : 123  
Bridge recv from ipc : 123  
Bridge send to mnt : 123  
Bridge recv from mnt : test  
Bridge send to ipc : test  
Bridge recv from ipc : test  
Bridge send to mnt : test  
Bridge recv from mnt : exit  
Bridge send to ipc : exit
```