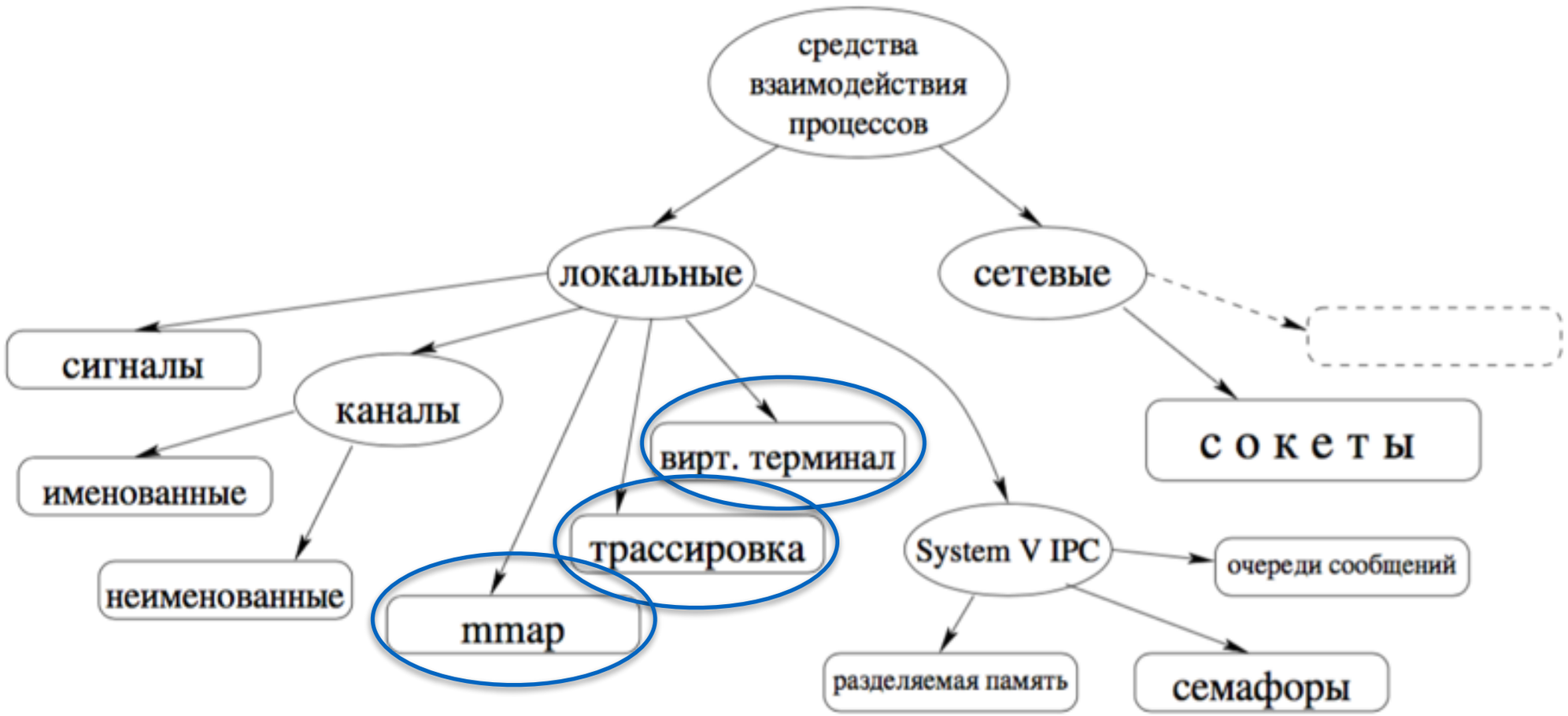
Операционные системы

Разделяемая память. Взаимодействие через псевдотерминал. Трассировка

Олег Французов

2017

Средства межпроцессного взаимодействия в Unix



Отображение файлов в виртуальное адресное пространство. Разделяемая память

Отображение файла в память

- Некоторая область в памяти ставится в соответствие содержимому файла
- При работе с этой памятью файл также меняется

Вызов mmap()

```
void *mmap(void *start, int length,
           int protection, int flags,
           int fd, int offset);
```

```
fd     – дескриптор файла (открытого)
offset – начало отображаемого участка
length – его длина

Кратны размеру страницы:
int getpagesize();
```

Вызов mmap()

```
void *mmap(void *start, int length,
           int protection, int flags,
           int fd, int offset);
```

protection
- PROT_READ, PROT_WRITE, PROT_EXEC
- д.б. совместим с режимом fd

Вызов mmap()

```
void *mmap(void *start, int length,
           int protection, int flags,
           int fd, int offset);
```

flags
- MAP_SHARED
- MAP_PRIVATE (copy-on-write)
- MAP_ANONYMOUS (без файла)
  - переживает fork

Вызов mmap()

```
void *mmap(void *start, int length,
           int protection, int flags,
           int fd, int offset);
```

start
  - где разместить в памяти
  - обычно NULL

-> указатель или MAP_FAILED

Отображение файла

```
int fd;
char *ptr;
fd = open("file.dat", O_RDWR);
if (fd == -1)
{ /* обработка ошибки */ }

ptr = (char*) mmap(
    NULL, 4096, PROT_READ|PROT_WRITE,
    MAP_SHARED, fd, 0);
if (ptr == MAP_FAILED)
{ /* обработка ошибки */ }
```

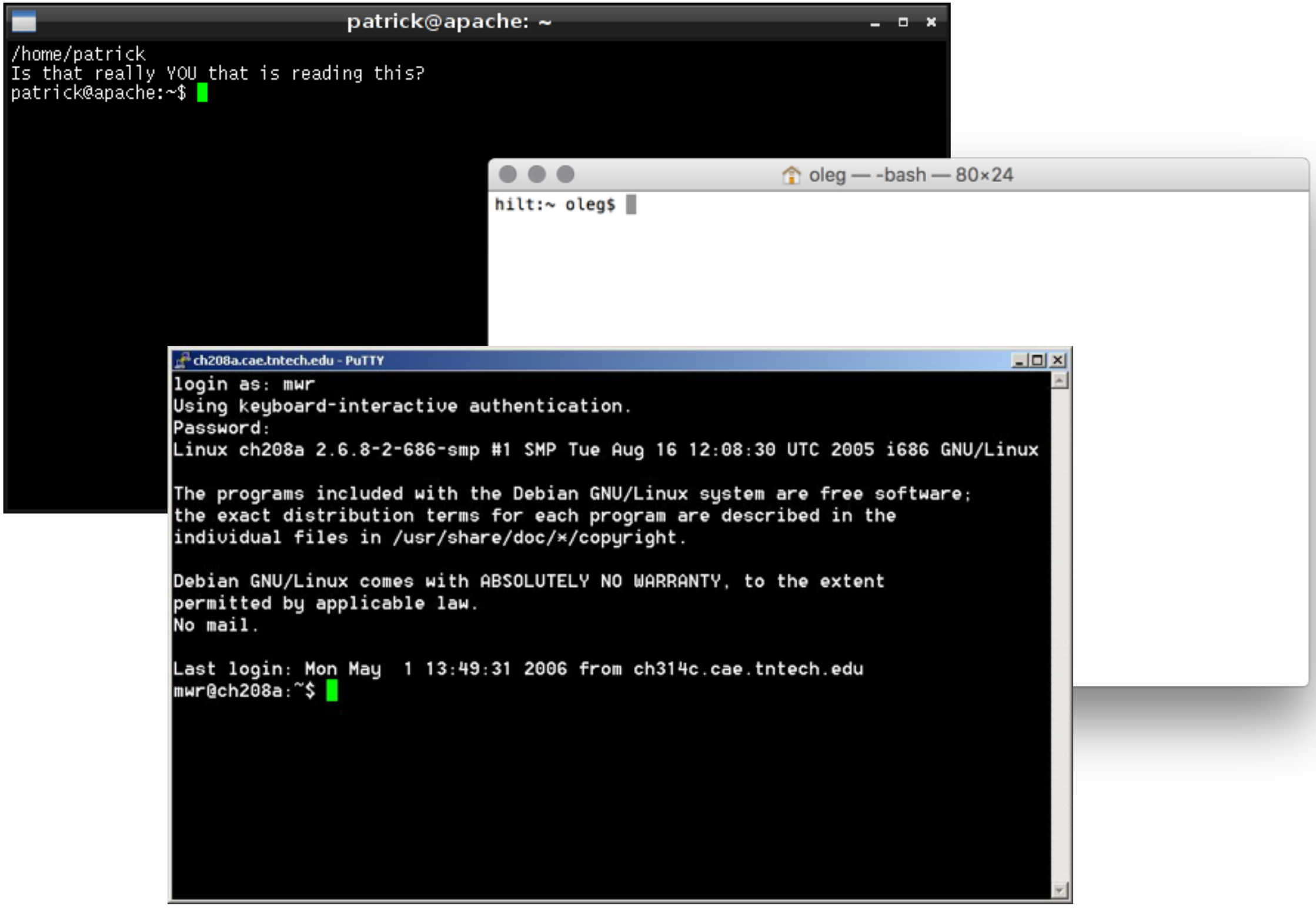
Разделяемая память

```
int *ptr;
ptr = (int*) mmap(
    NULL, 4096, PROT_READ|PROT_WRITE,
    MAP_SHARED|MAP_ANONYMOUS, 0, 0);
if (ptr == MAP_FAILED) {
    /* обработка ошибки */
}

if (fork() == 0) {
    ptr[77] = 120; /* child */
} else {
    i = ptr[77]; /* parent */
}
```

Вызов munmap()

```
/* Отменить отображение */
int munmap(void *start, int length);

/* Синхронизировать память с файлом */
int msync(void *start, int len,
          int flags);
```

Взаимодействие процессов через псевдотерминал


patrick@apache: ~
/home/patrick
Is that really YOU that is reading this?
patrick@apache:~$
oleg — -bash — 80×24
hilt:~ oleg$
ch208a.cae.tntech.edu - PuTTY
login as: mwr
Using keyboard-interactive authentication.
Password:
Linux ch208a 2.6.8-2-686-smp #1 SMP Tue Aug 16 12:08:30 UTC 2005 i686 GNU/Linux
The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.
Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
No mail.
Last login: Mon May 1 13:49:31 2006 from ch314c.cae.tntech.edu
mwr@ch208a:~$

Псевдотерминал

- Физический терминал
- Эмулятор терминала
  - Удаленный доступ: sshd
  - xterm и аналоги

Псевдотерминал

- Пользователь нажал `Ctrl-C`
- ???
- Программа получила `SIGINT`

Псевдотерминал

- Управляющий символ передается драйверу псевдотерминала
- Драйвер посылает сигнал

- Кто получает `Ctrl-C`, `Ctrl-D`?

Псевдотерминал

- Имеет два двусторонних канала связи:
  - для эмулятора терминала (master)
  - для запущенных программ (slaves)

Создание псевдотерминала

```
int getpt();
/* создать файл устройства */

int grantpt(int fd);
/* дать доступ к файлу владельцу
   текущего процесса */

int unlockpt(int fd);
/* разрешить открытие файла
   псевдотерминала */
```

Создание псевдотерминала

```
char *ptsname(int master_fd);
/* узнать имя файла устройства */
```

Далее можно порождать slave-процесс. stdin, stdout, stderr надо связать с дескриптором файла псевдотерминала.

Трассировка

Трассировка

- Применяется при отладке программ
- Отладчик контролирует выполнение отлаживаемой программы
  - остановка
  - просмотр и изменение памяти
  - пошаговый режим
  - точки останова

Трассировка

```
int ptrace(int request, int pid,
           void *addr, void *data);
```

Запуск трассируемой программы

- `fork()`
  - `PTRACE_TRACEME`
    - `exec() -> SIGCHLD`
- `wait()`
- `PTRACE_SINGLESTEP`
- `PTRACE_CONT`
- `PTRACE_GETREGS`

Присоединение к запущенному процессу

- PTRACE_ATTACH

Q & A