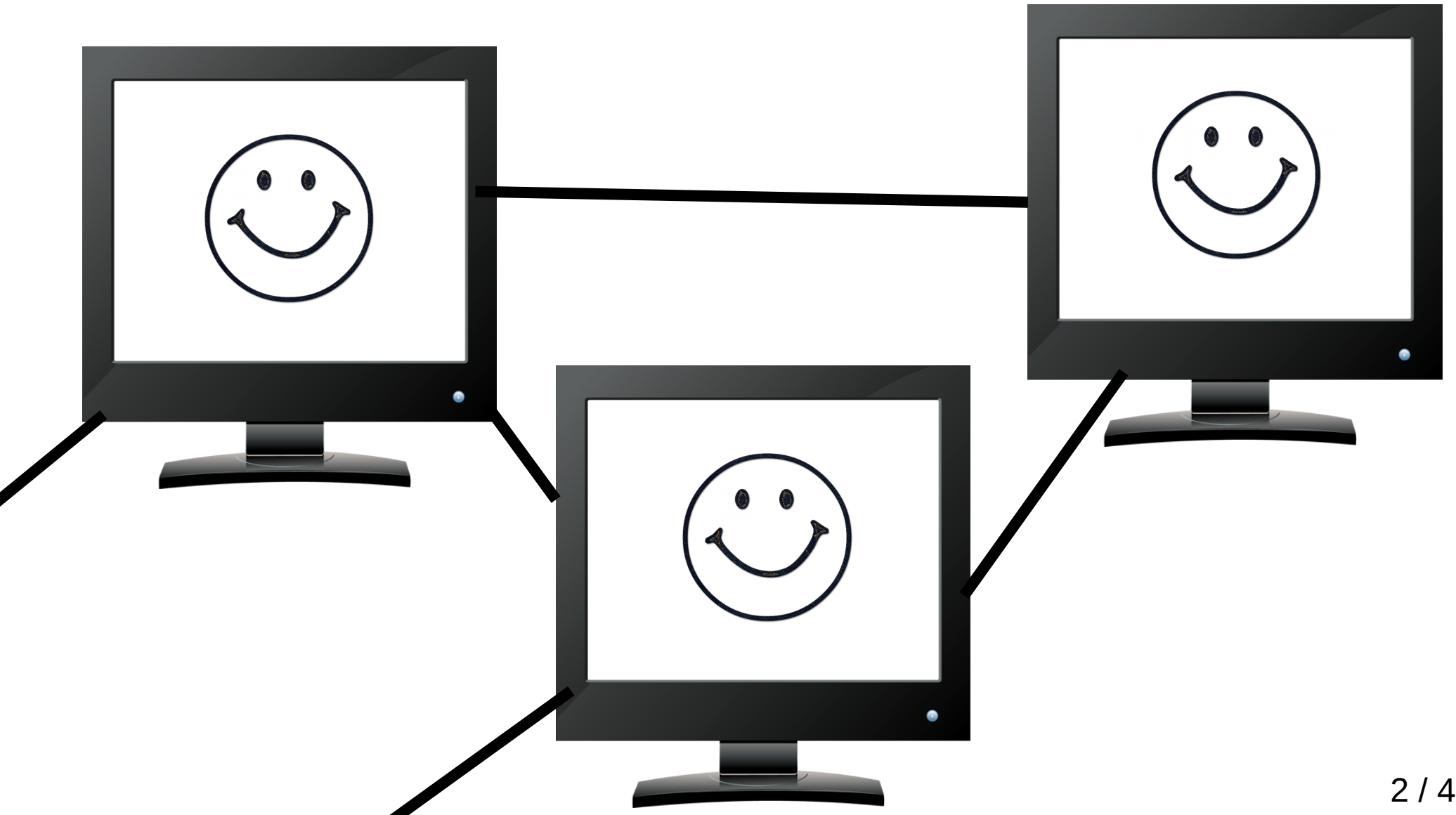


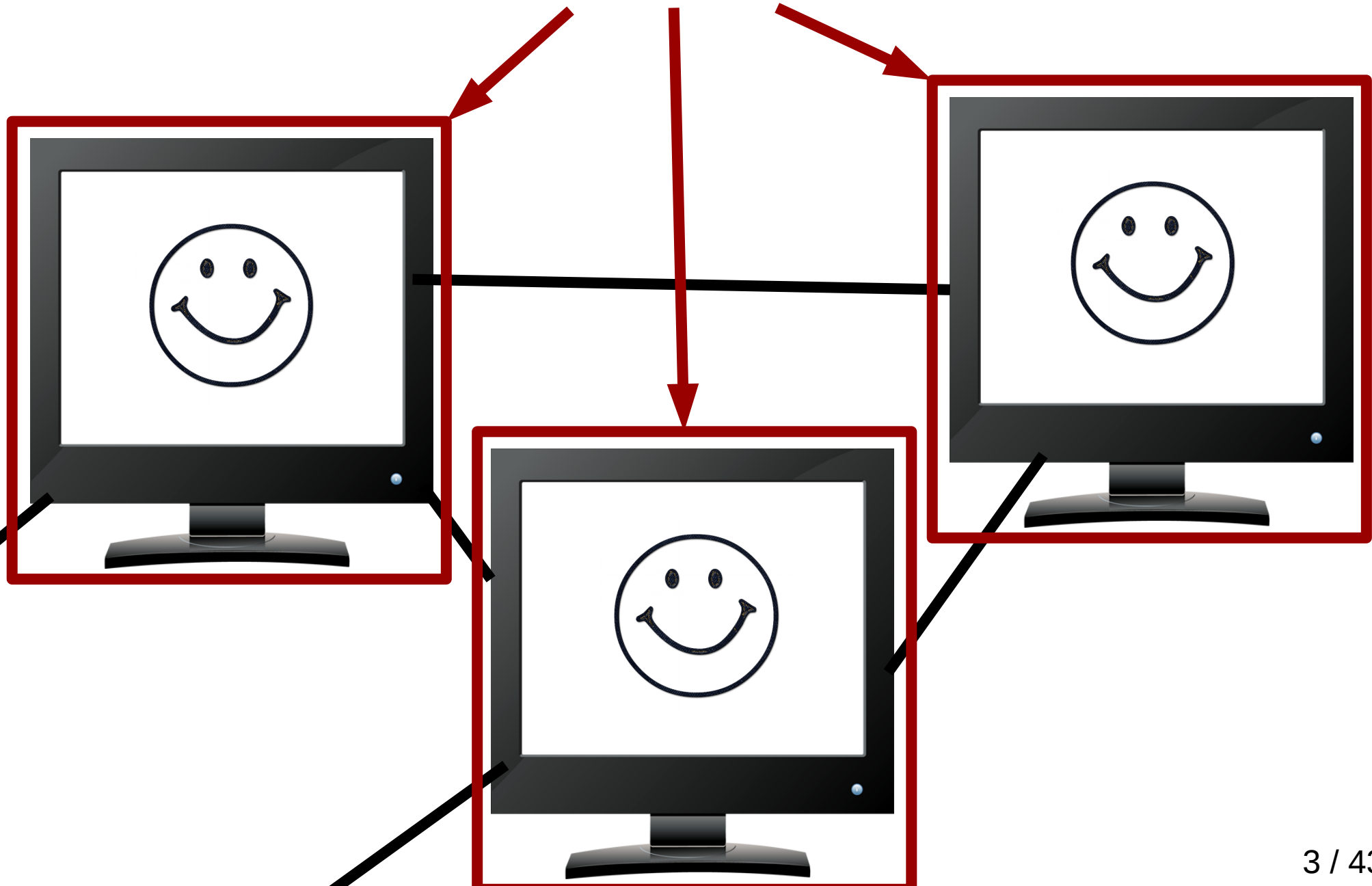
НПС



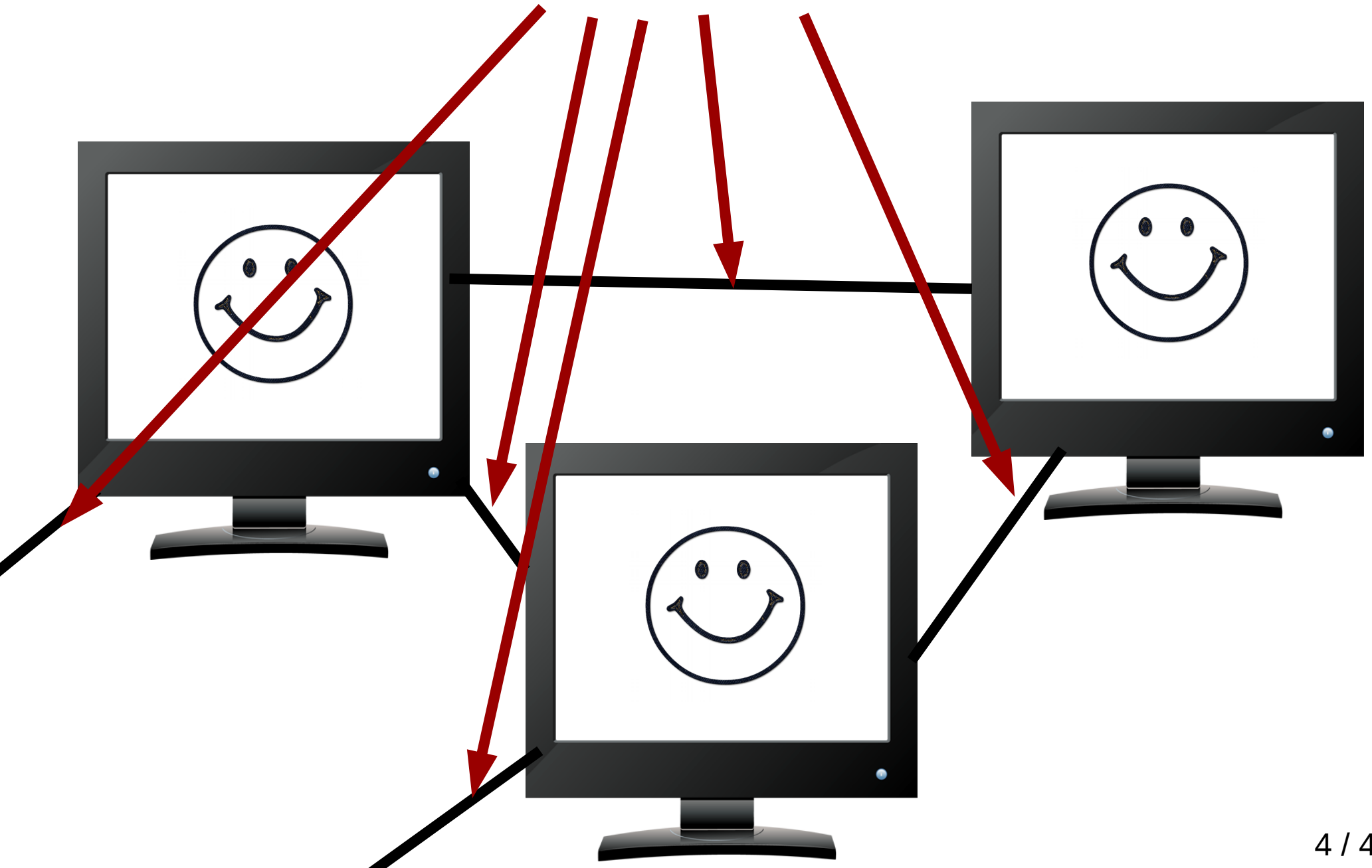
Remember?



Уже знаем

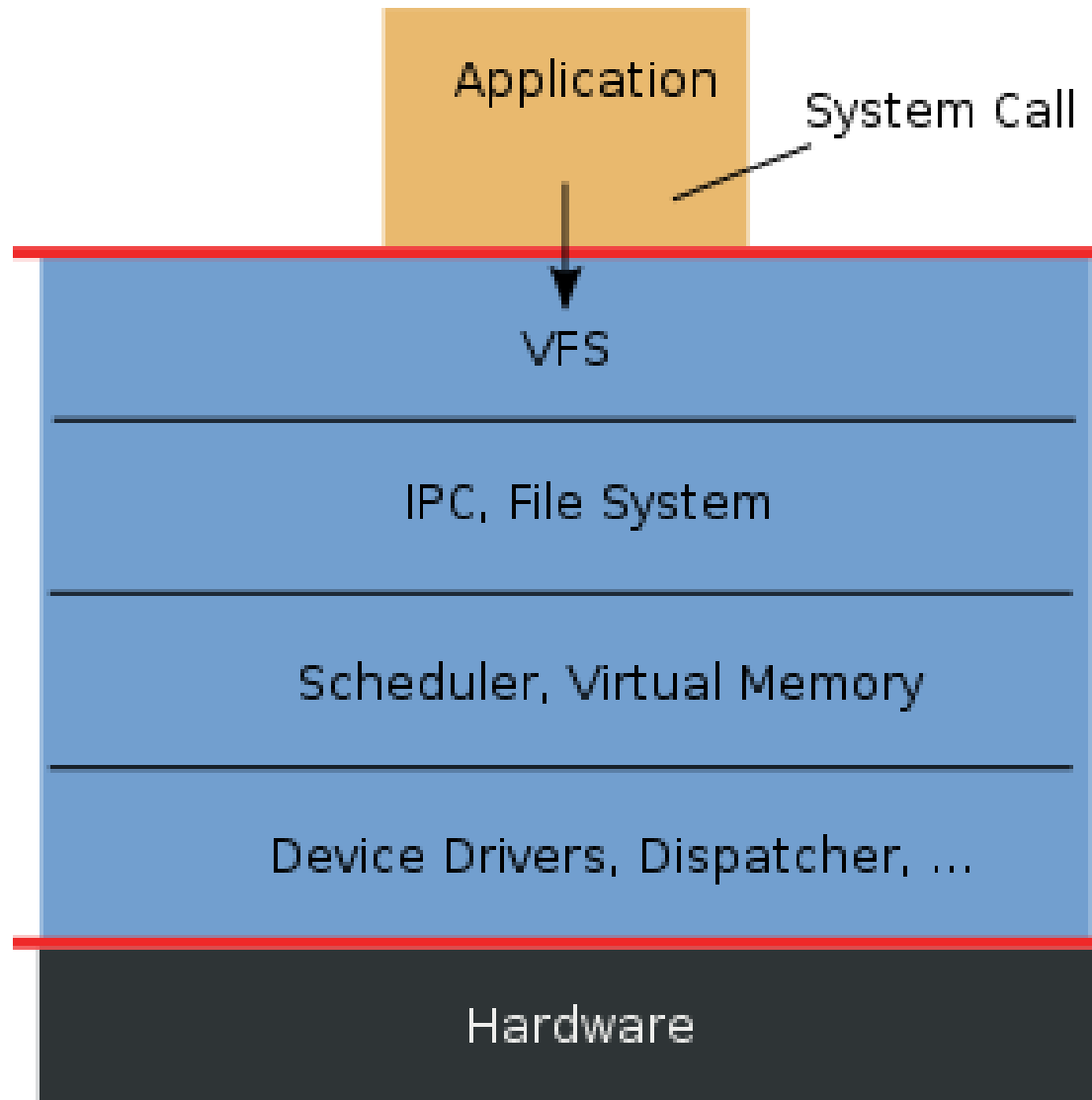


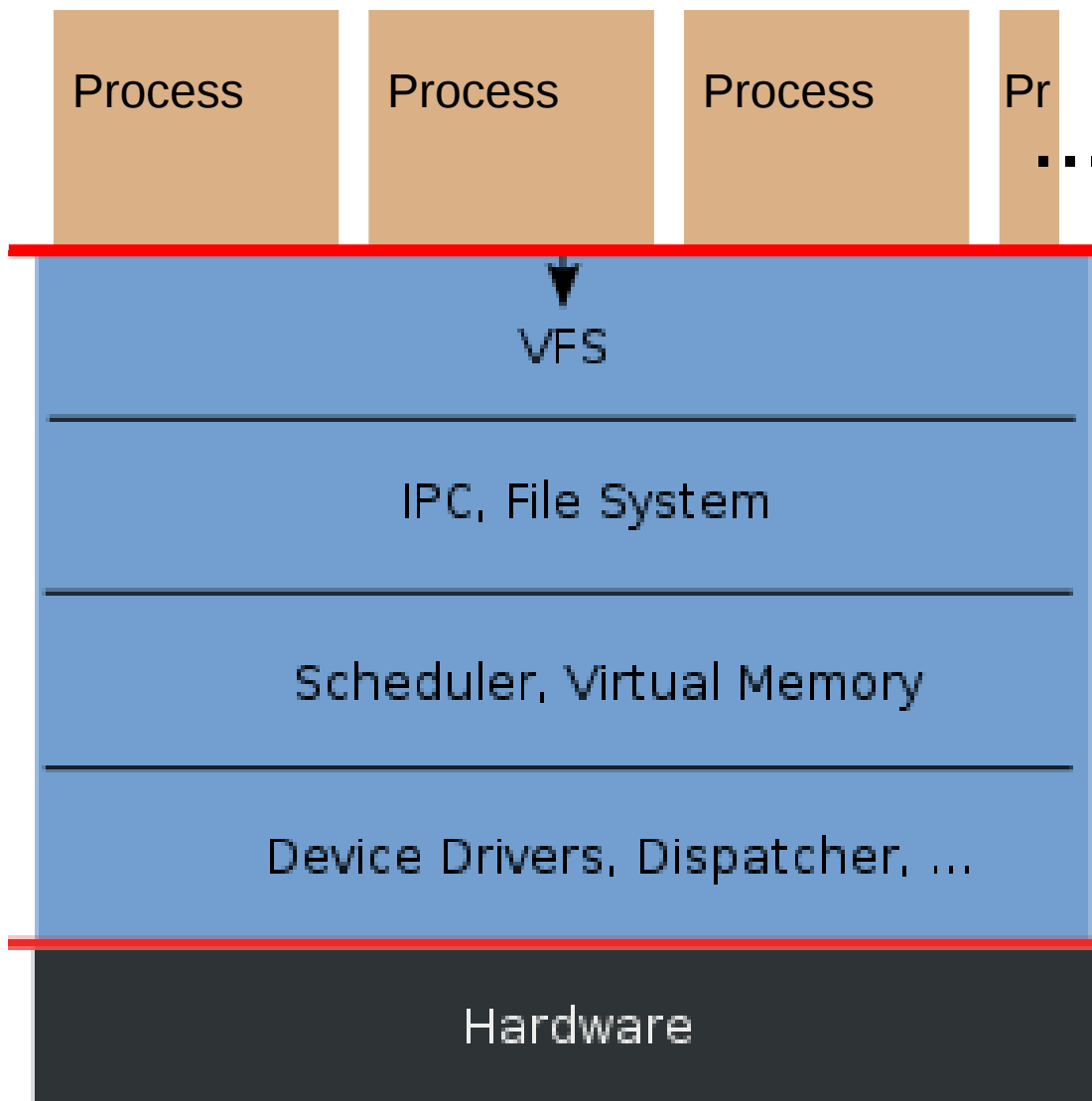
Уже знаем

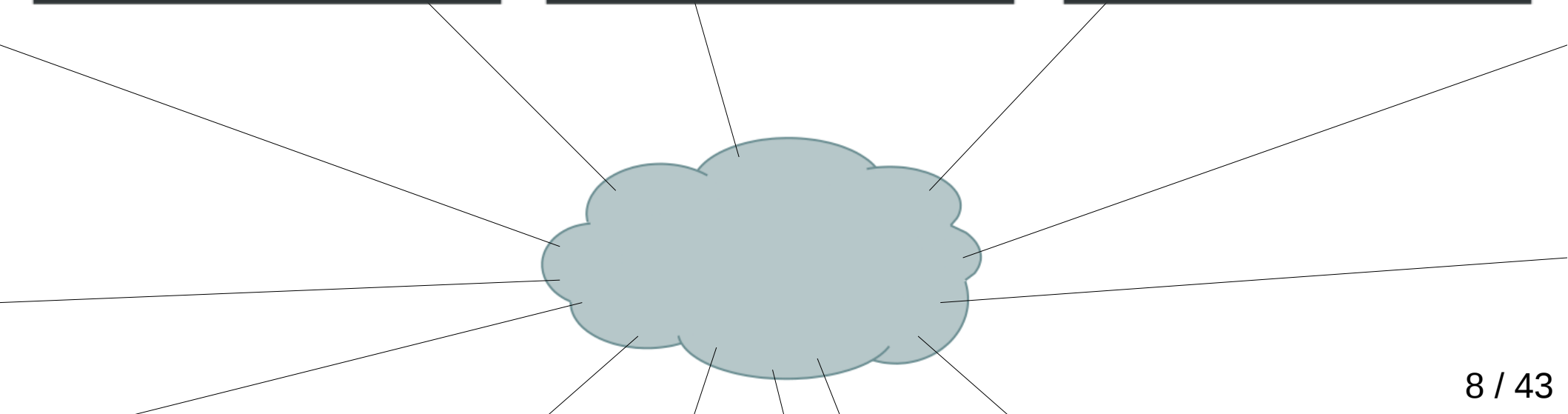
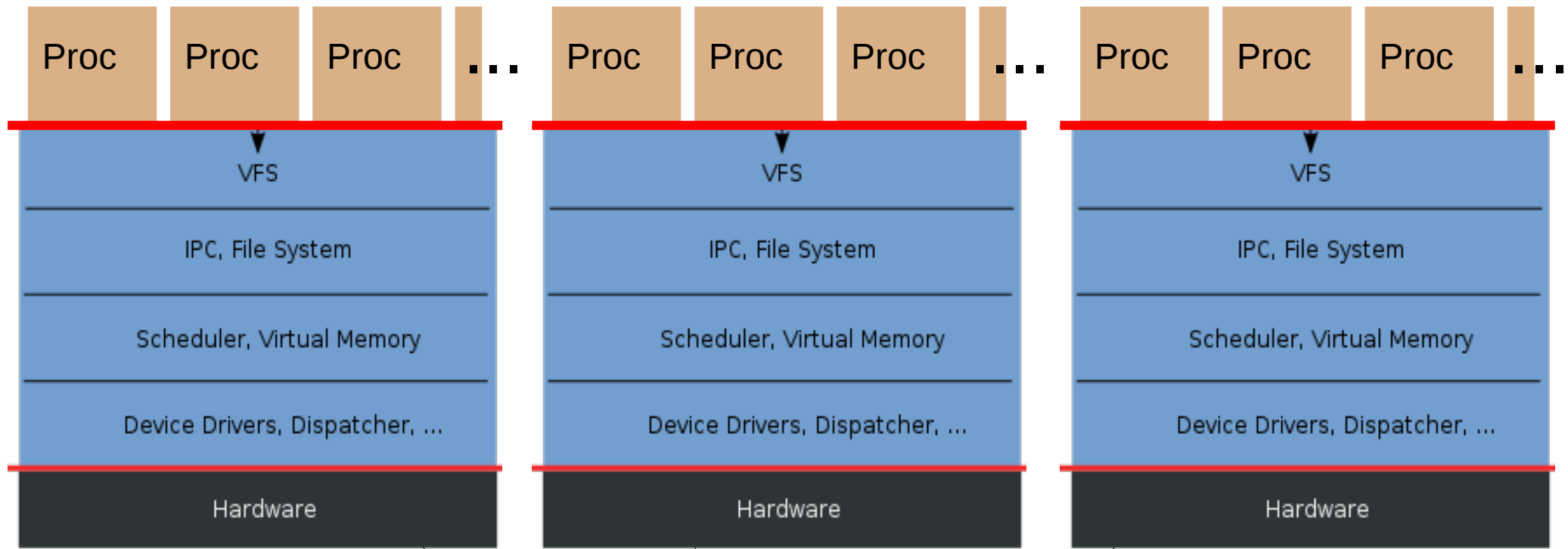


?

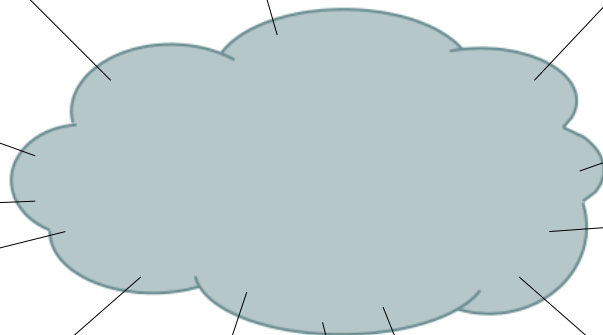
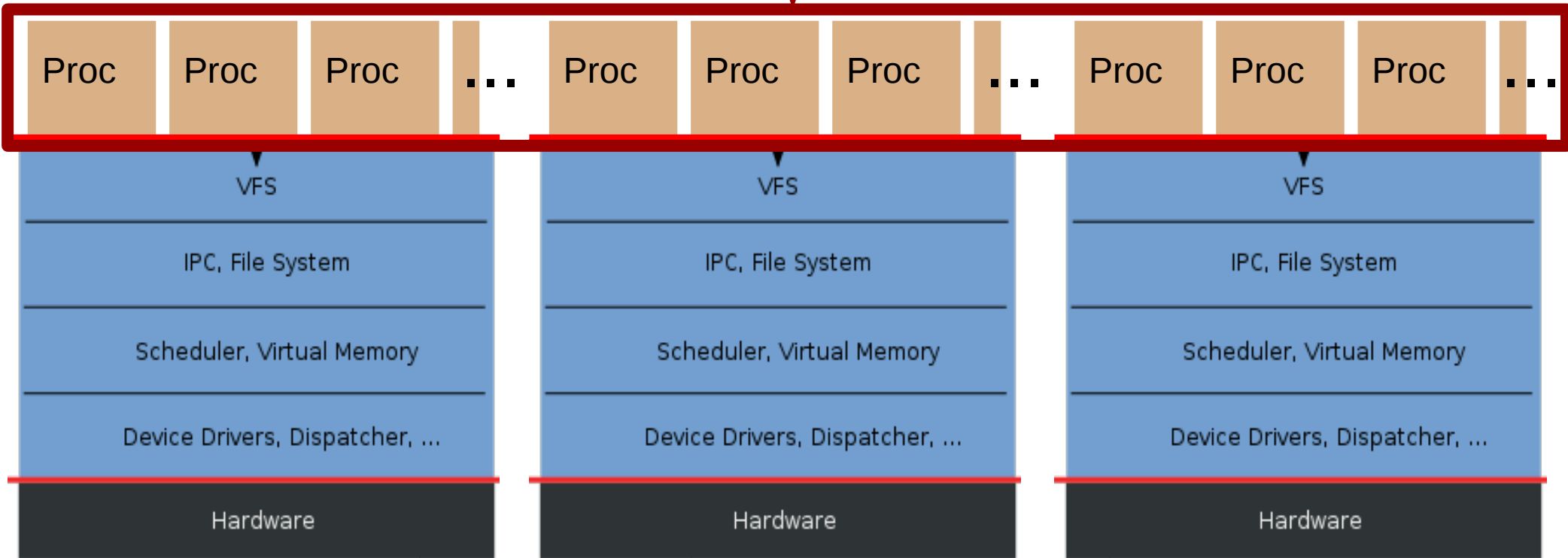


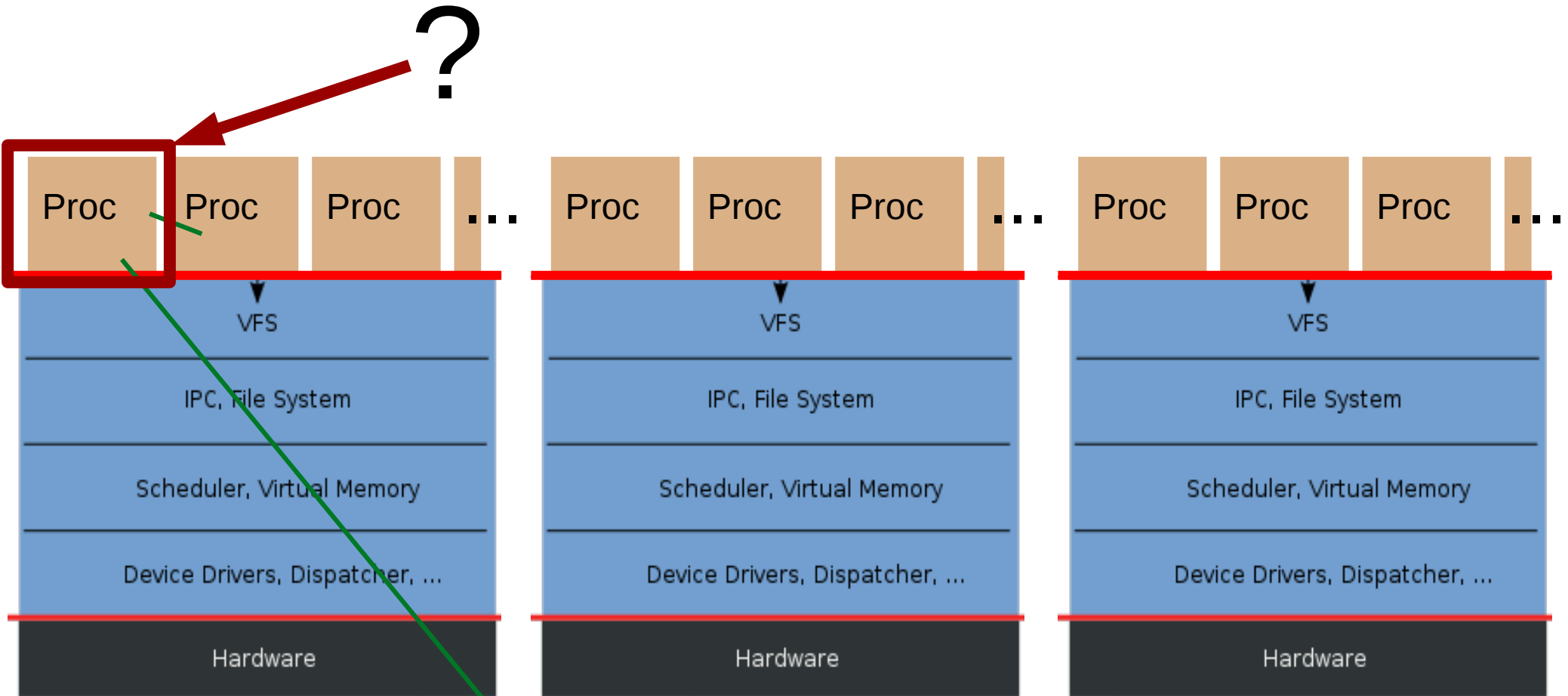


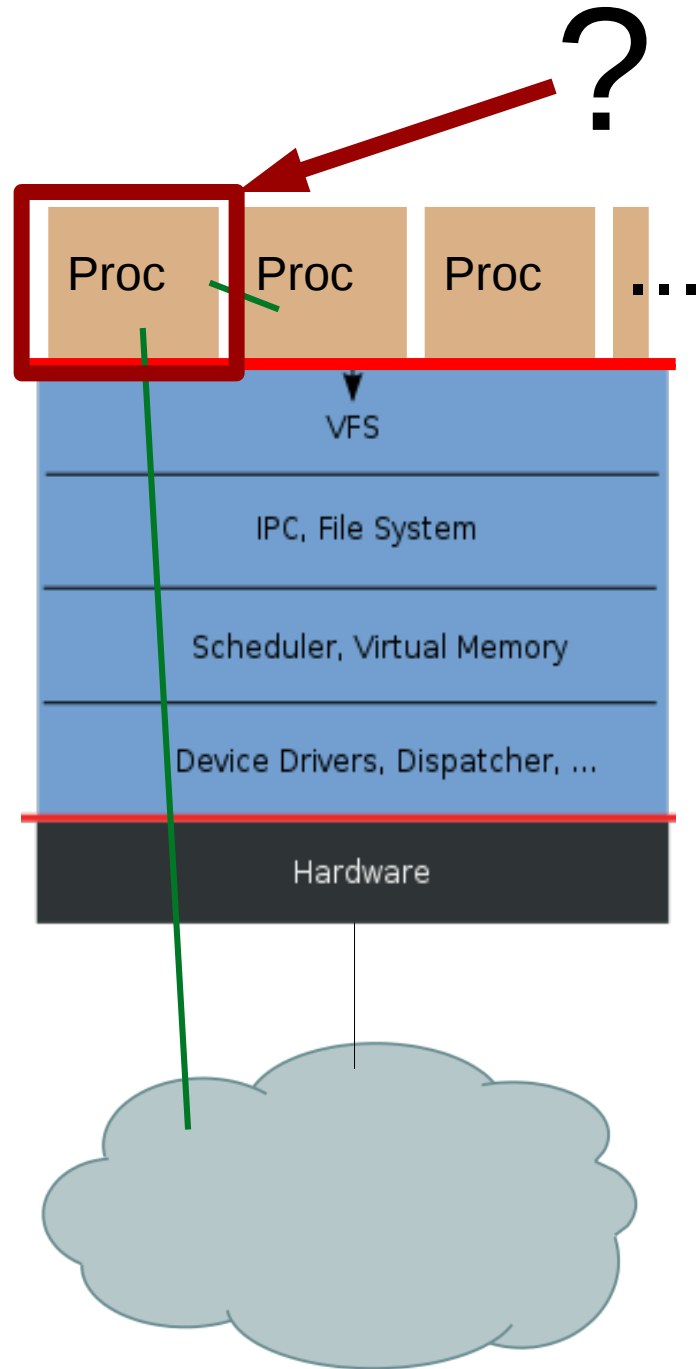


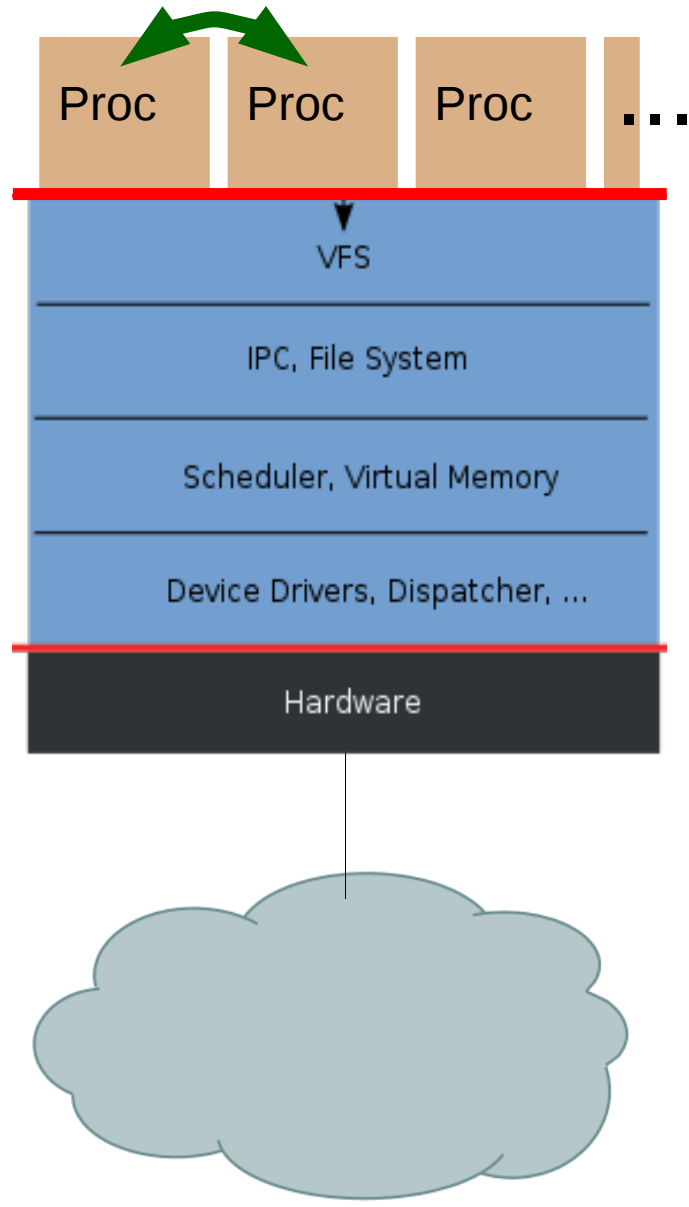


Взаимодействие

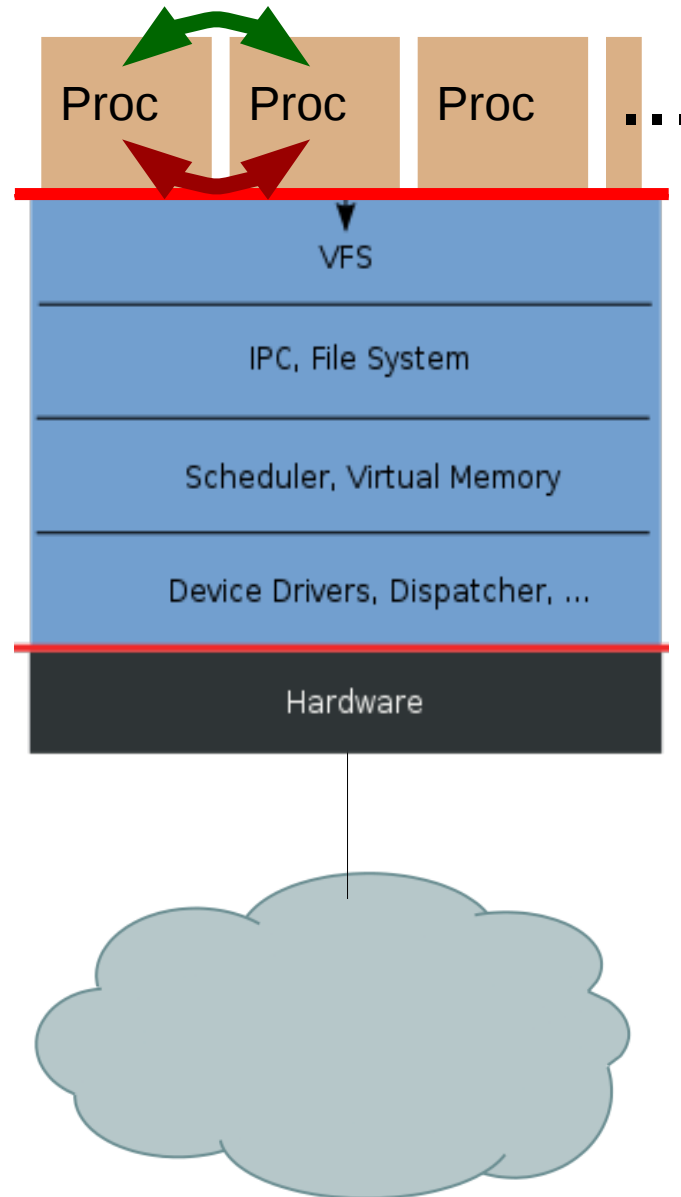




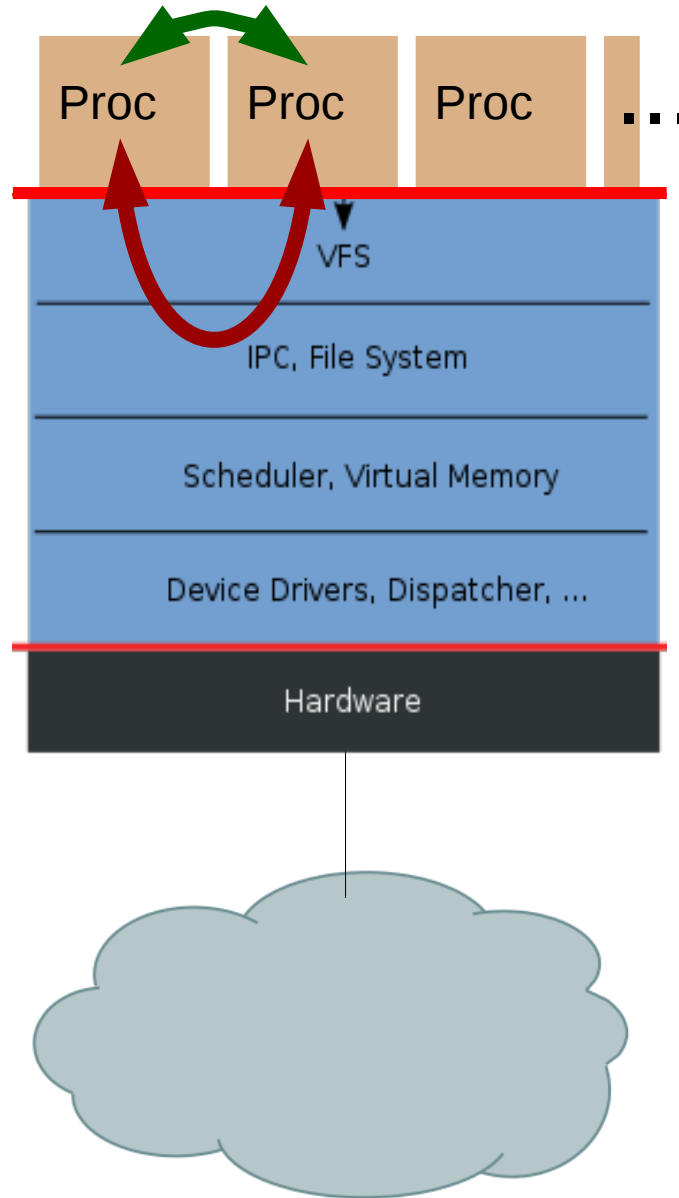




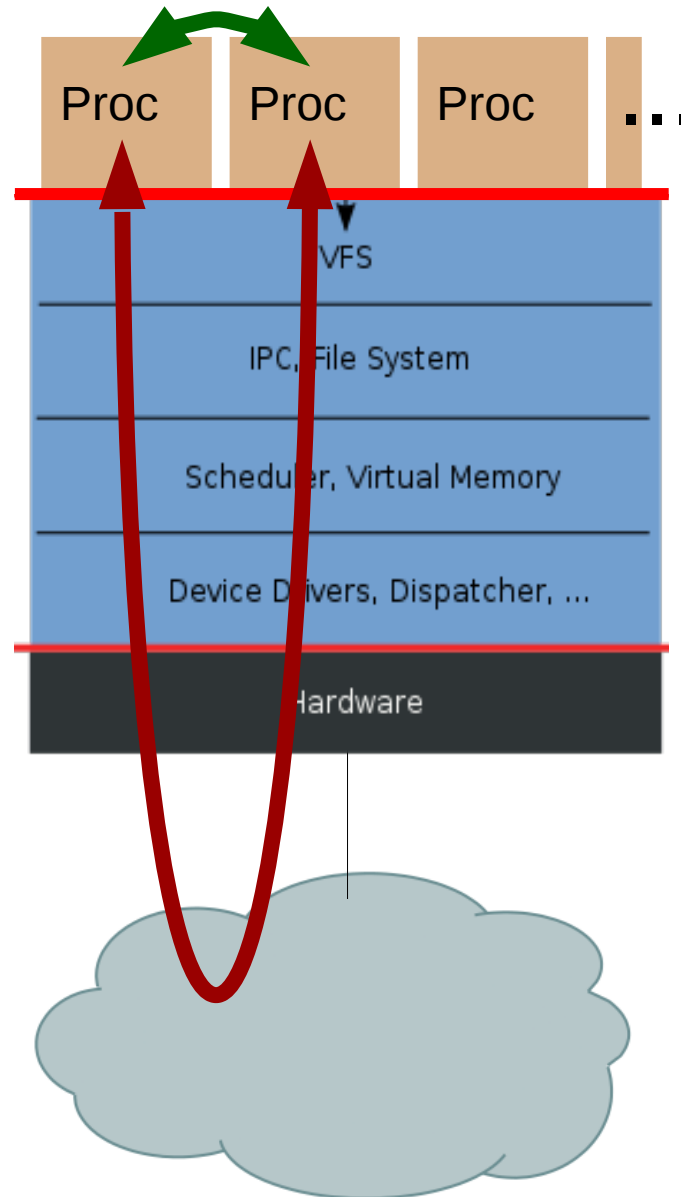
Shared memory



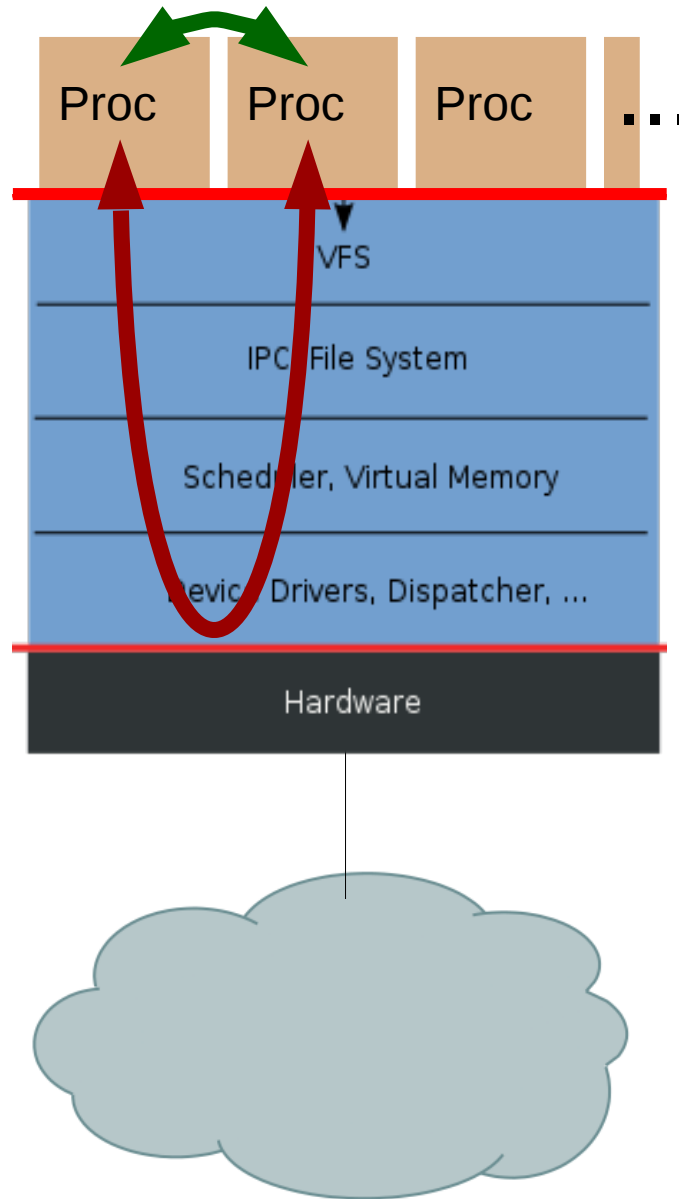
IPC, FS



Network



Network (one host)



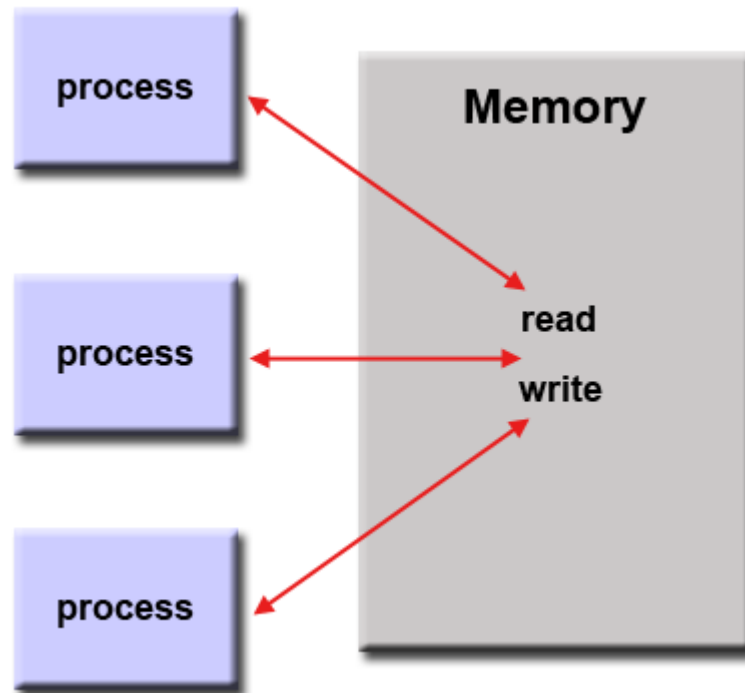
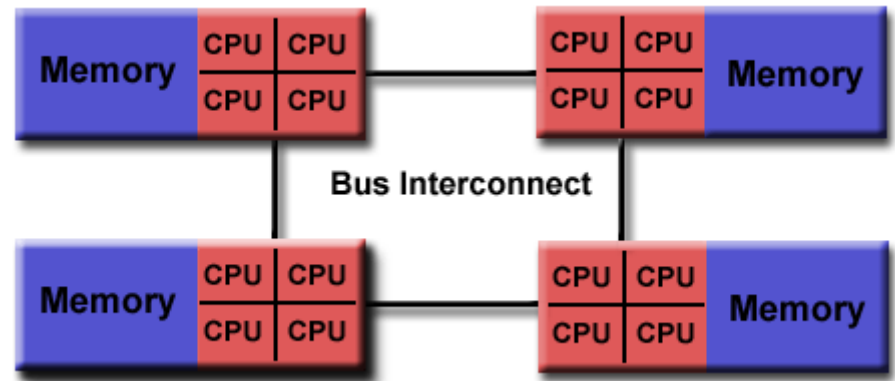
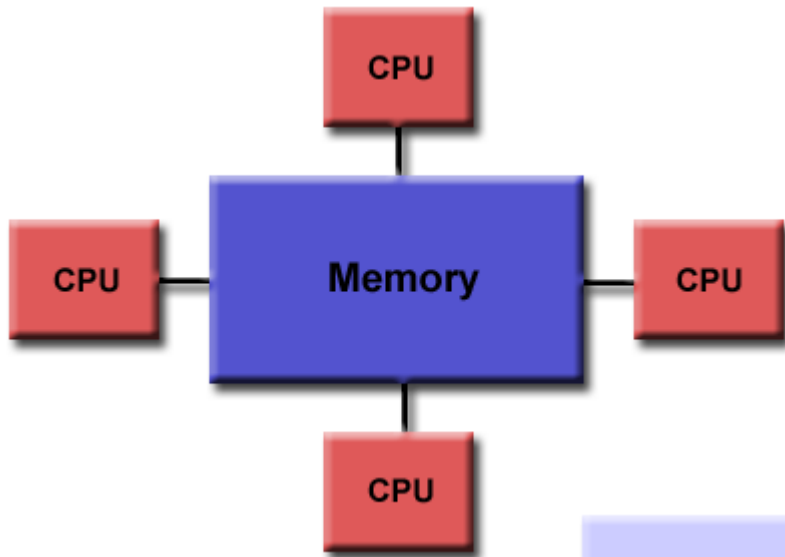
Итого

- Shared memory
- IPC, FS
- Network

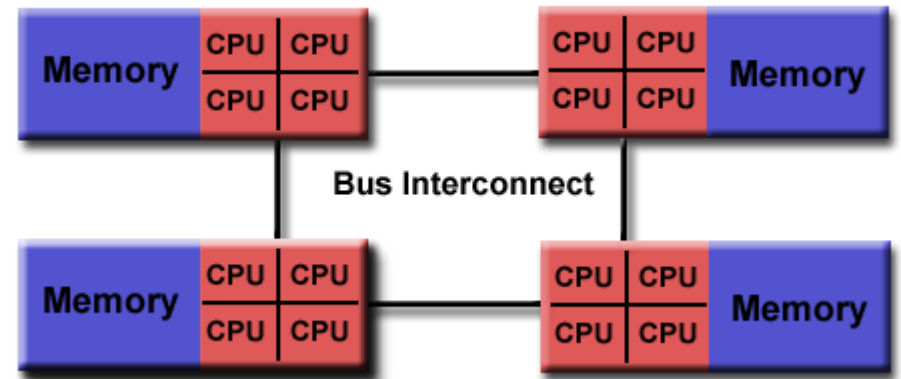
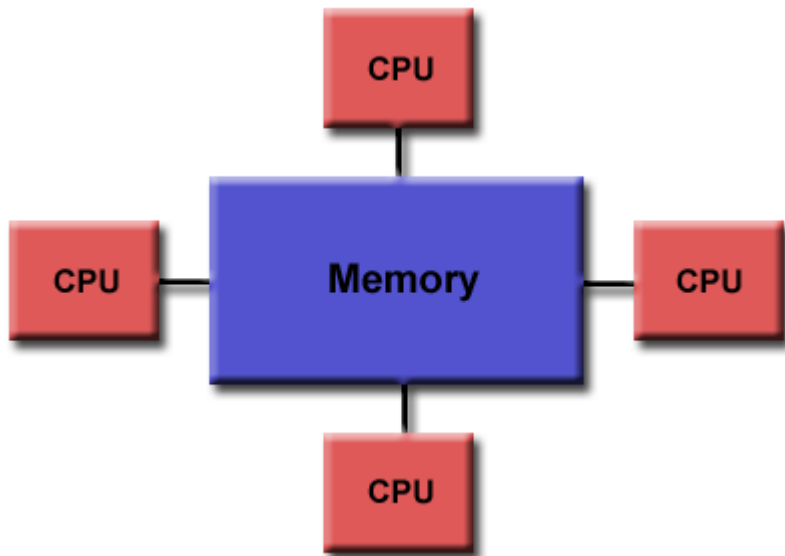
Итого

- **Shared memory** ←
- IPC, FS
- Network

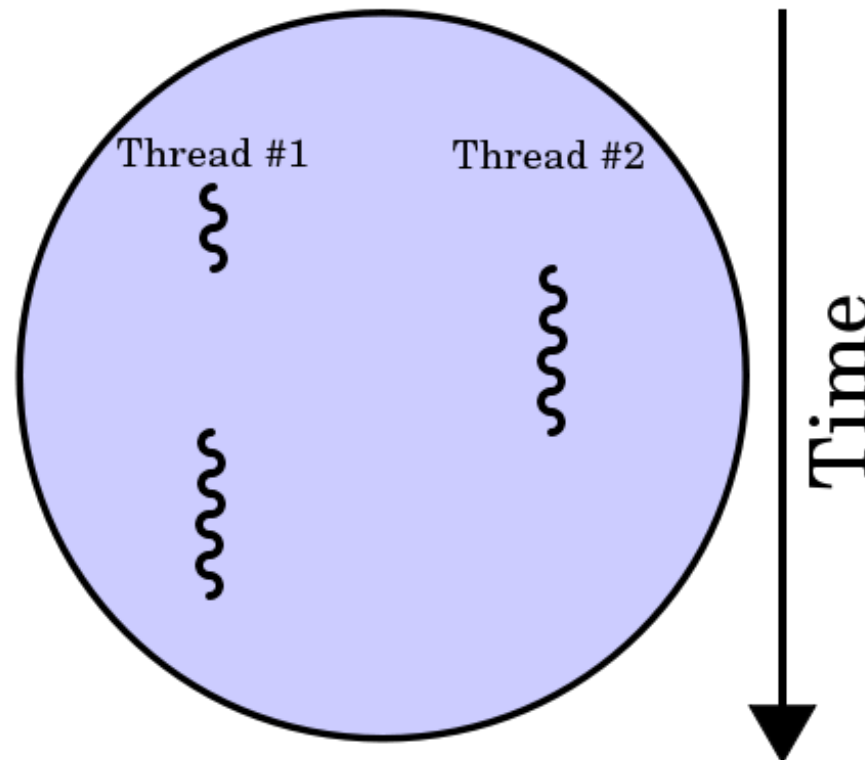
Shared memory



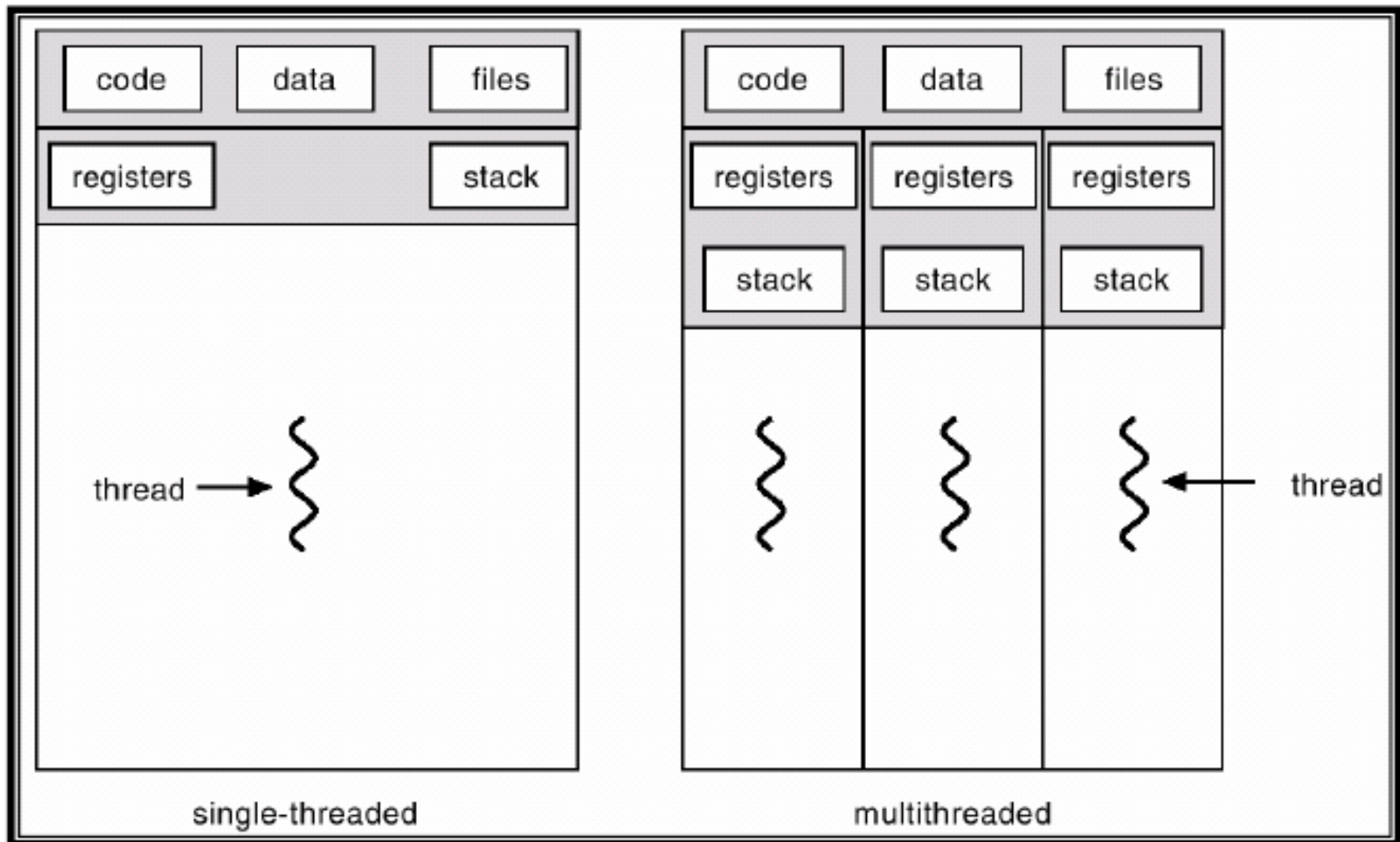
МНОГОПОТОЧНОСТЬ



Process



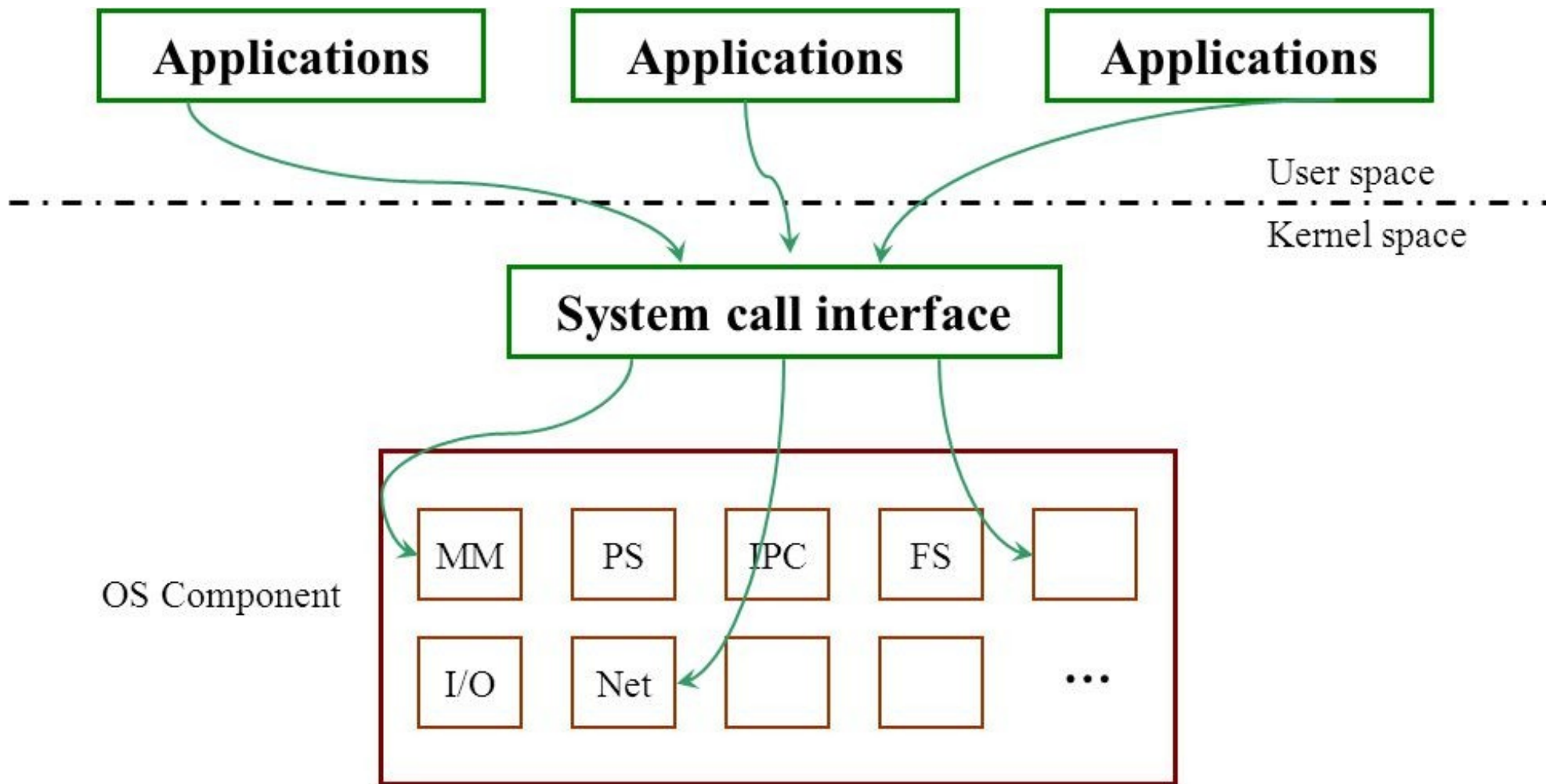
МНОГОПОТОЧНОСТЬ

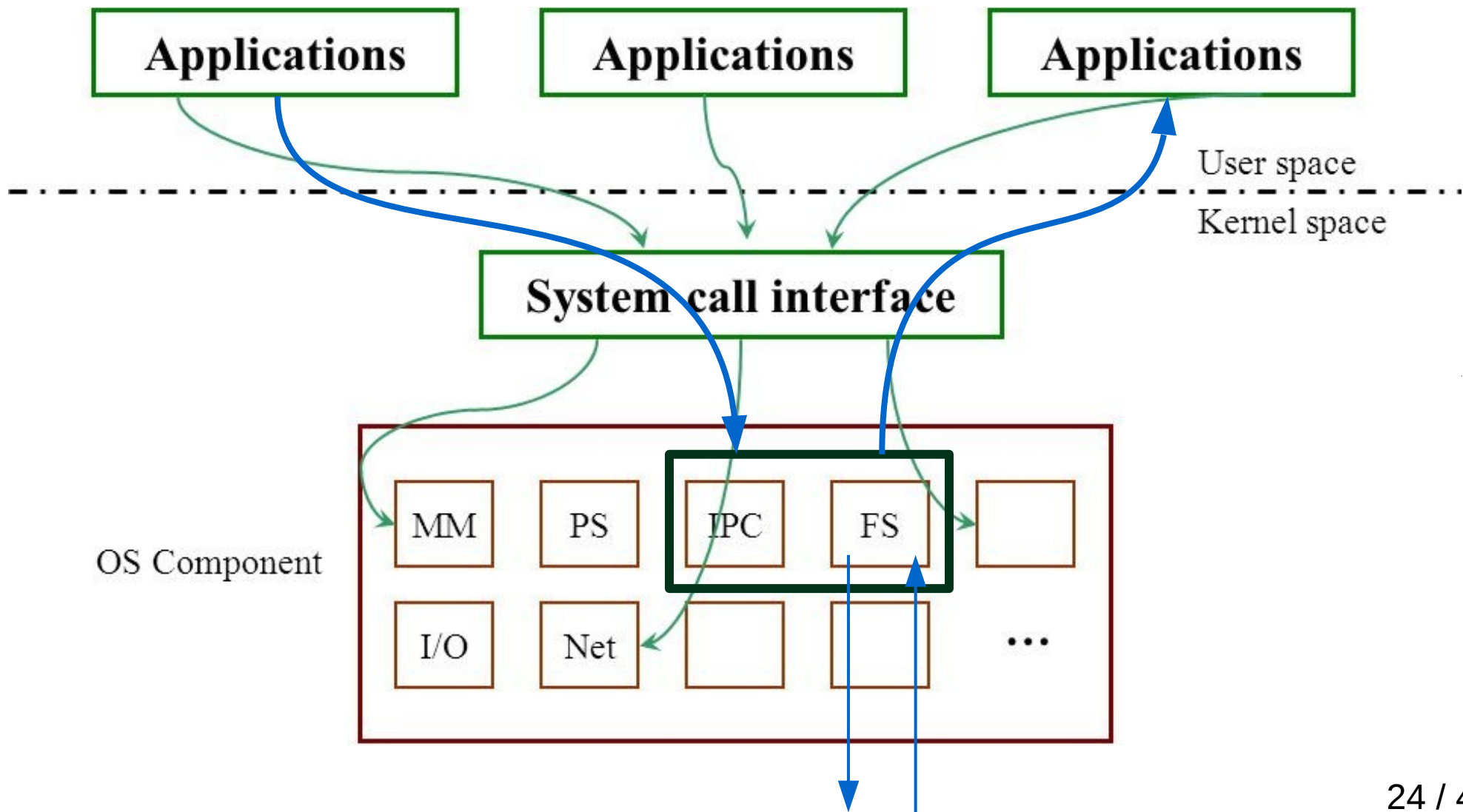


Итого

- Shared memory
- **IPC, FS**
- Network



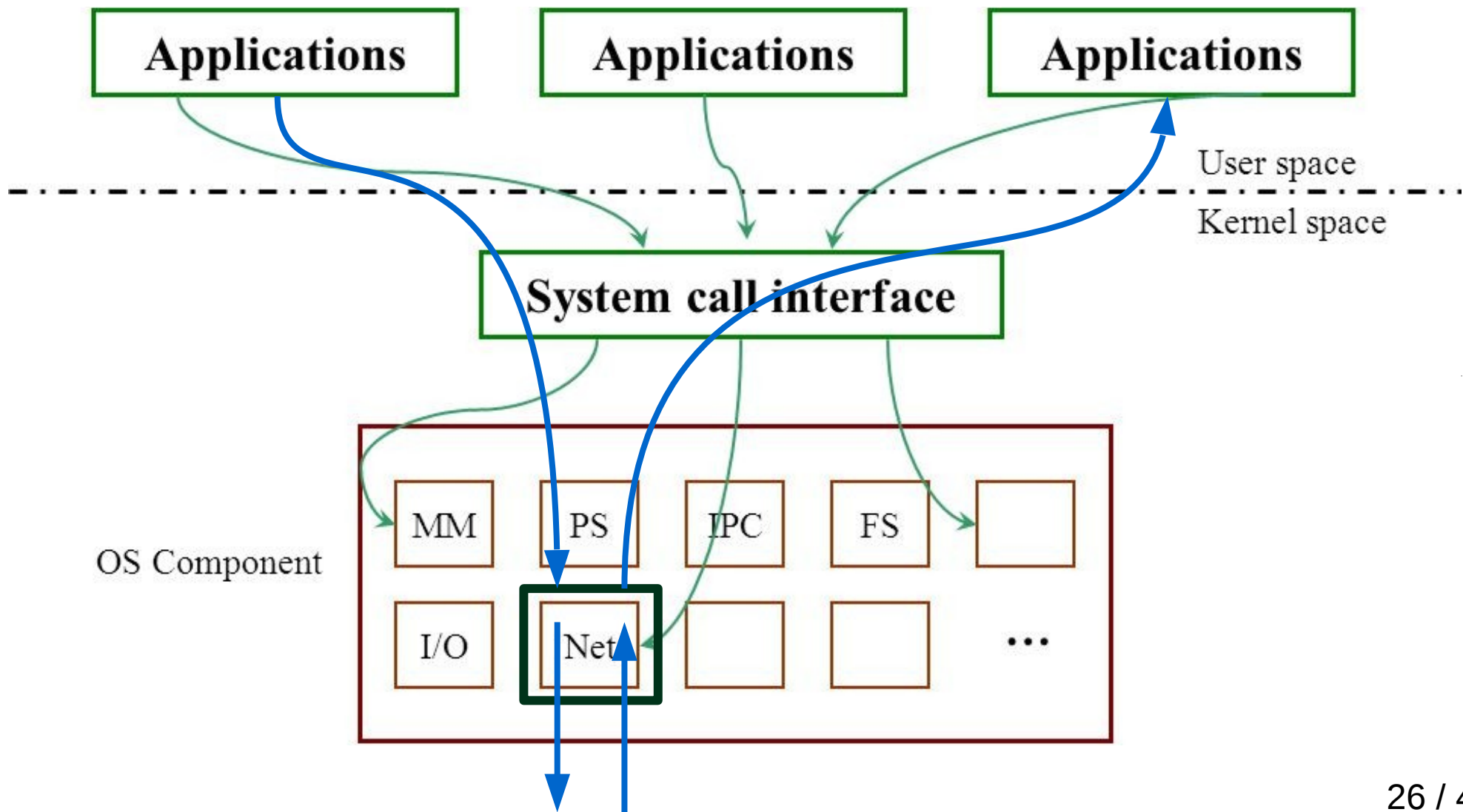




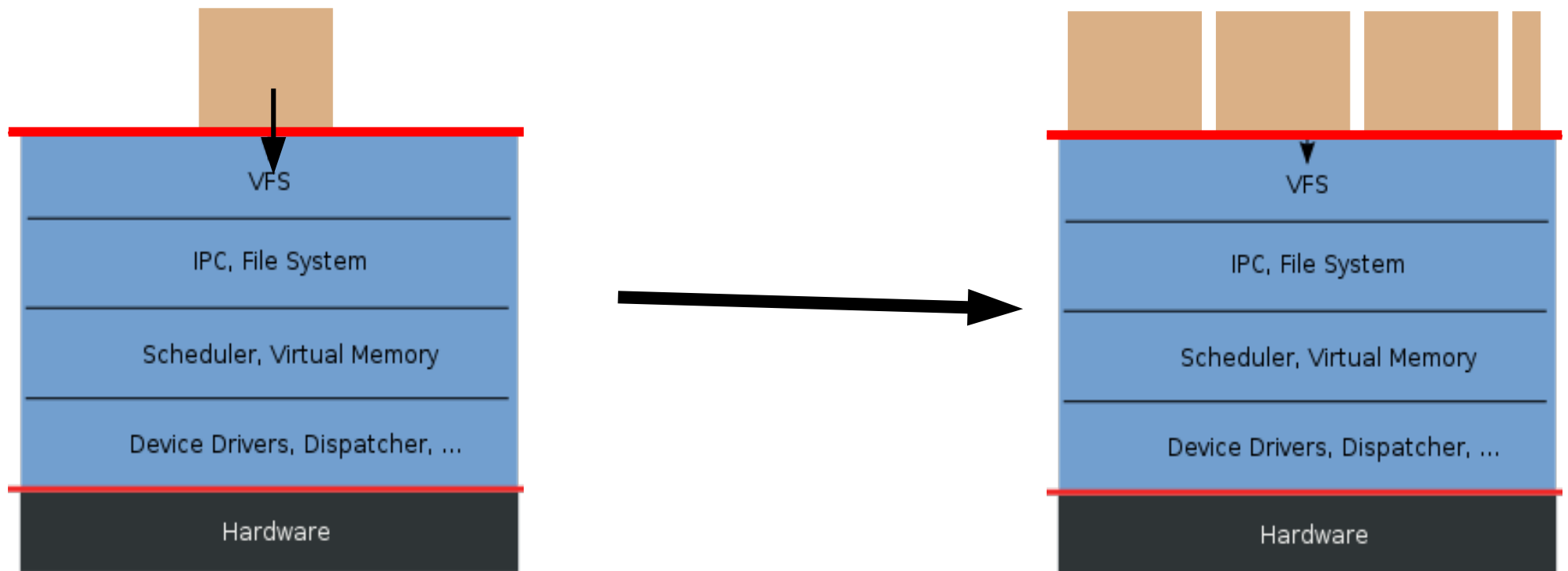
Итого

- Shared memory
- IPC, FS
- **Network**

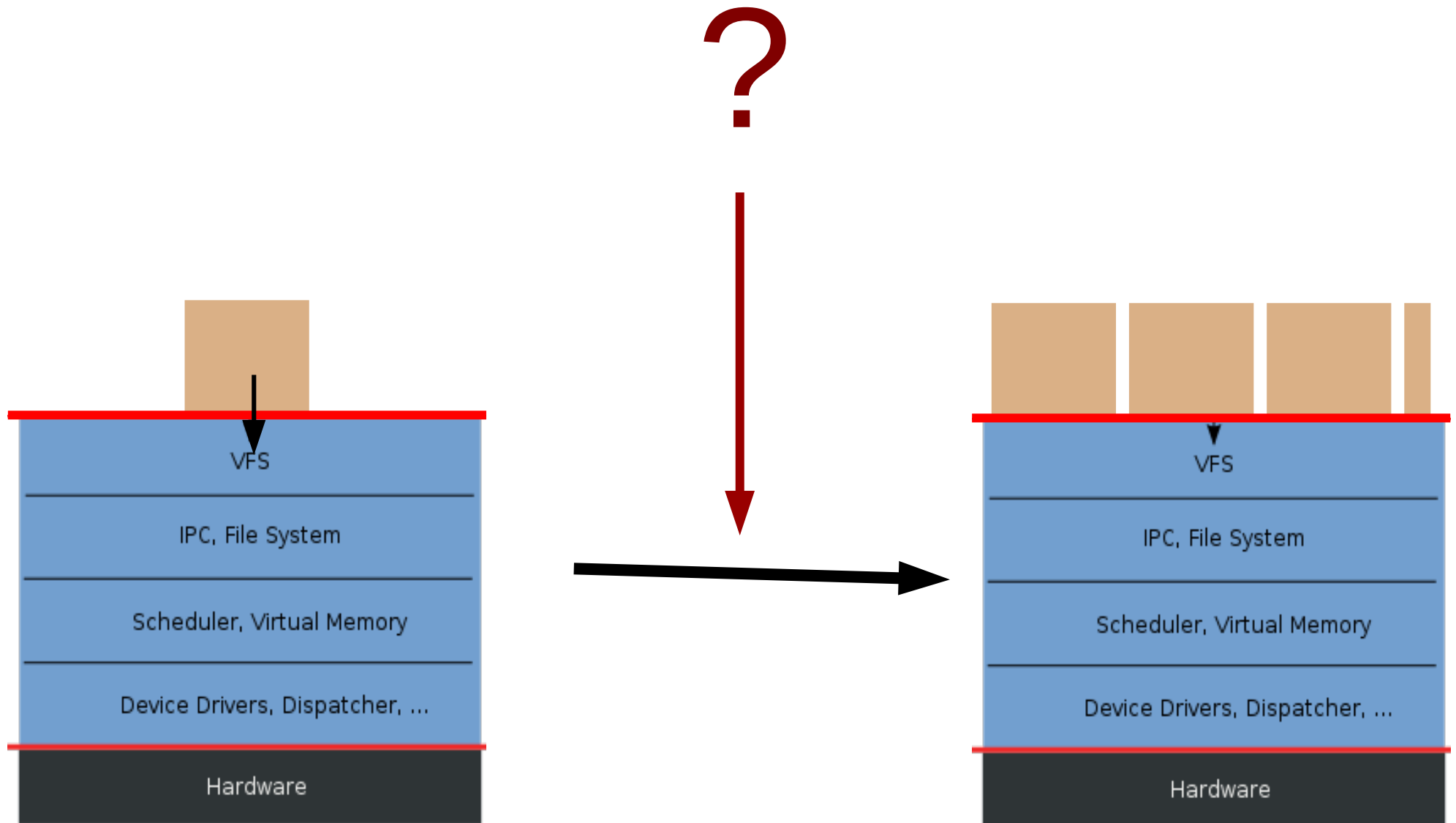




Processes, threads

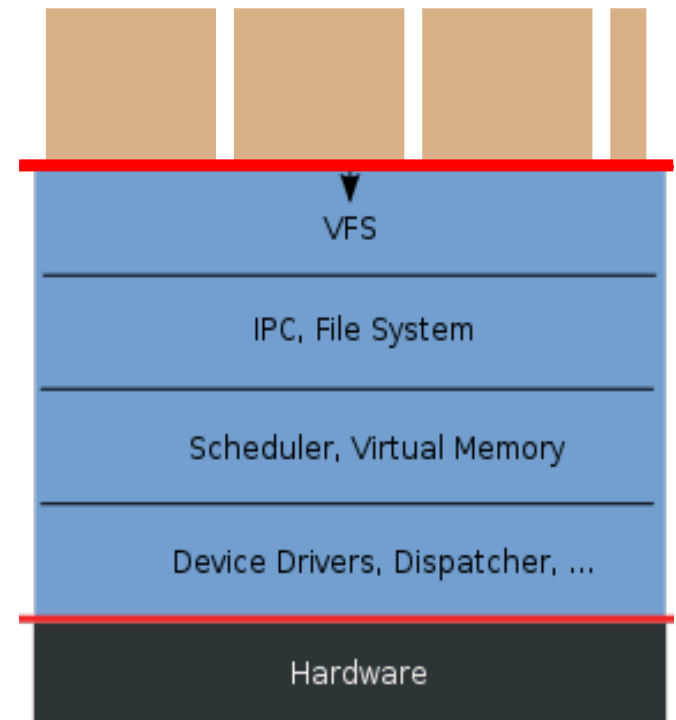
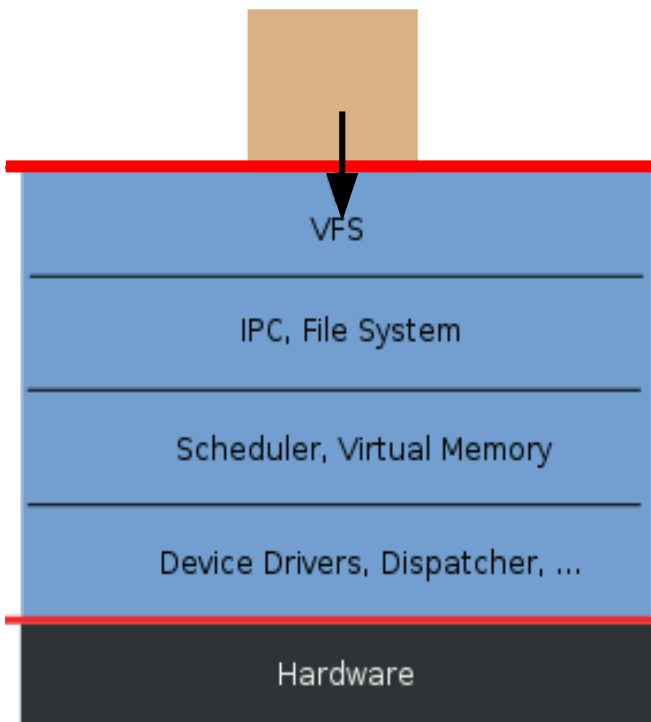
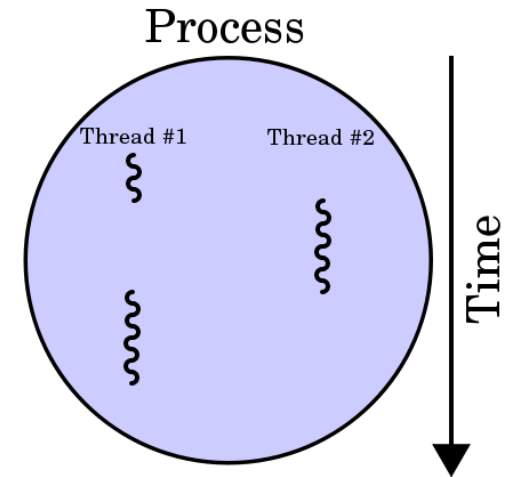


Processes, threads



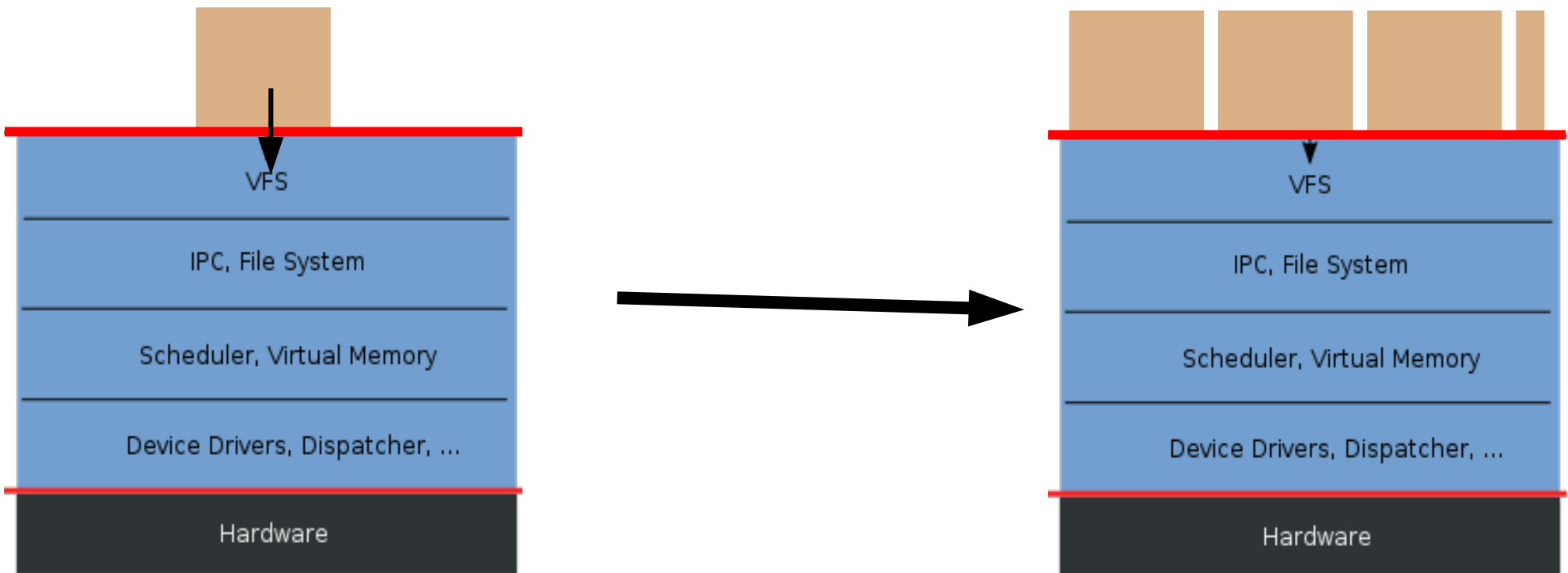
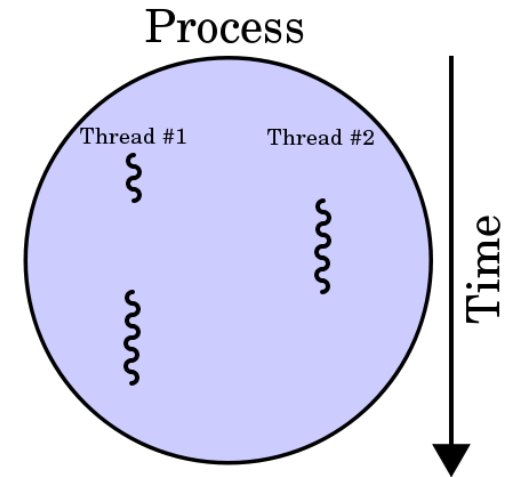
Processes, threads

fork()
pthread_create()



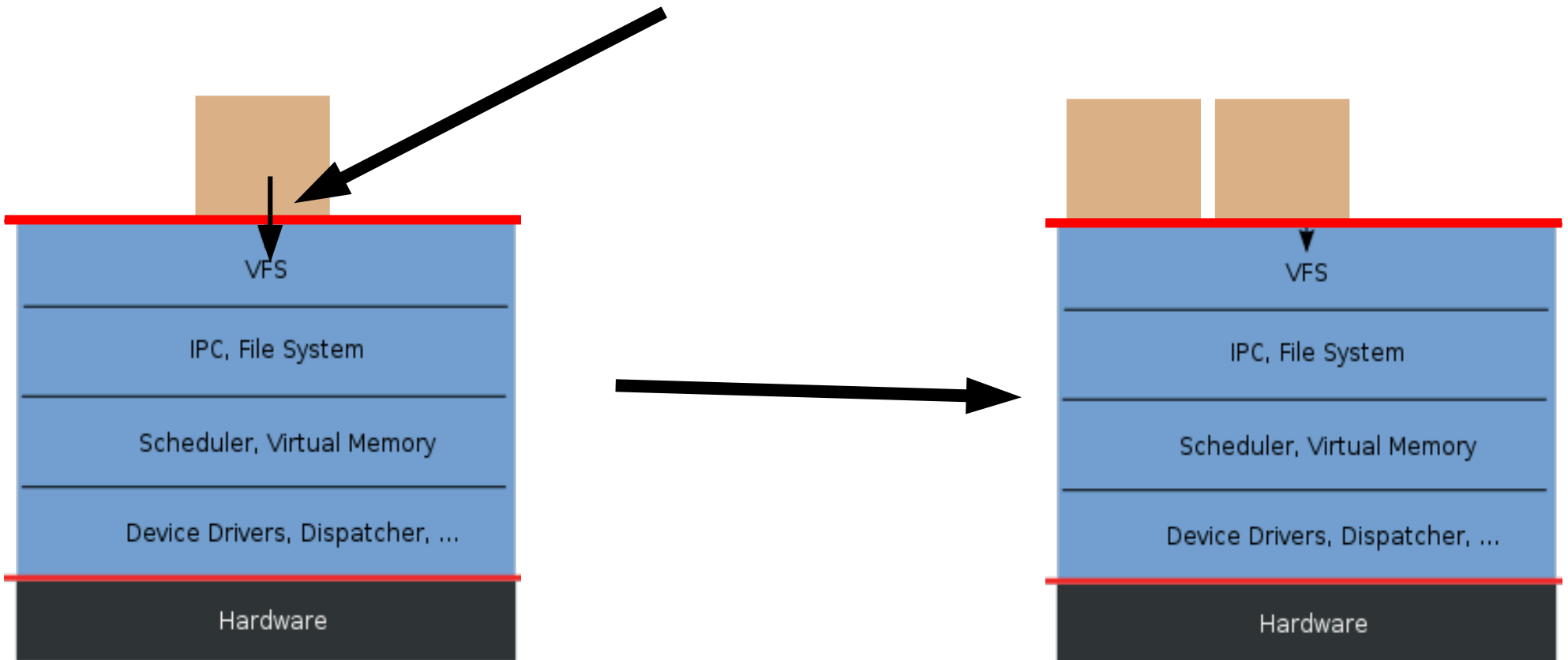
Processes, threads

clone() ← fork()
pthread_create()



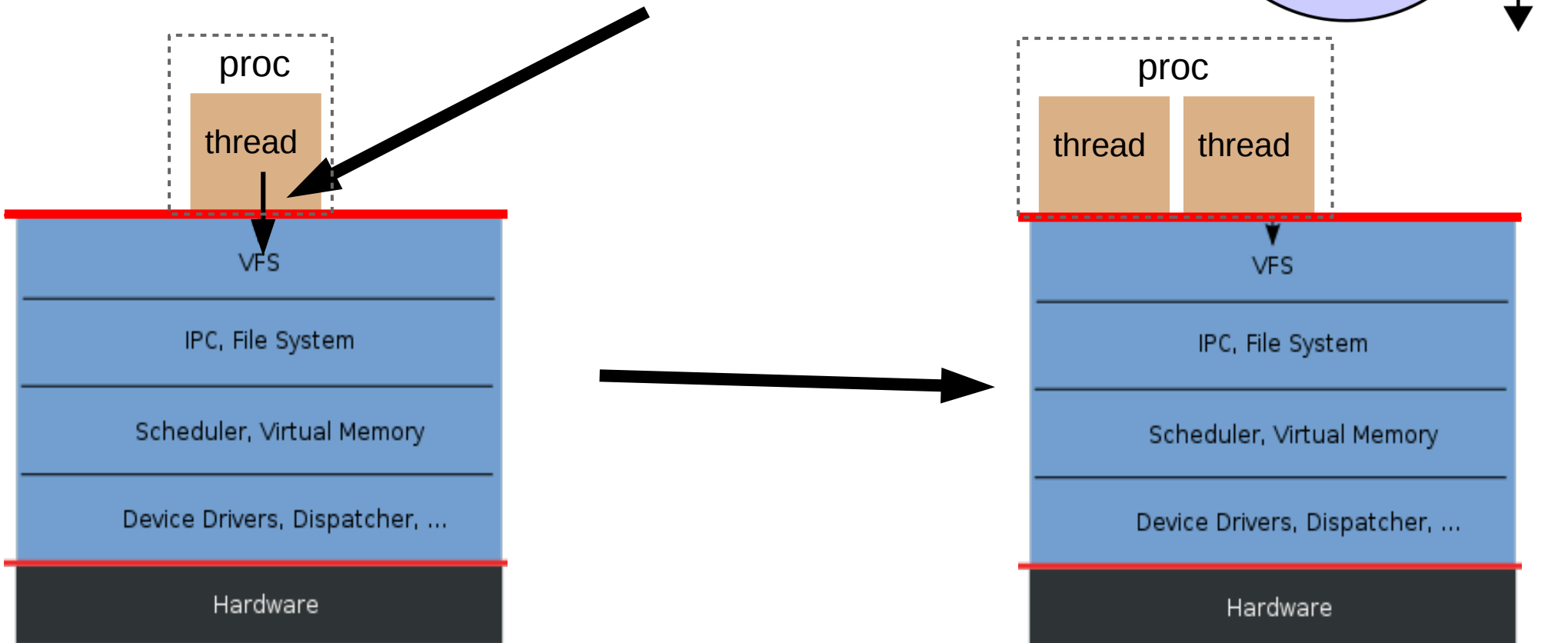
Processes, threads

syscall: clone(<как клонировать>)

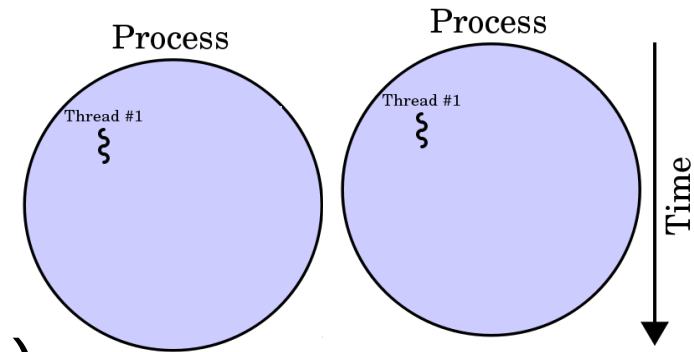


Processes, threads

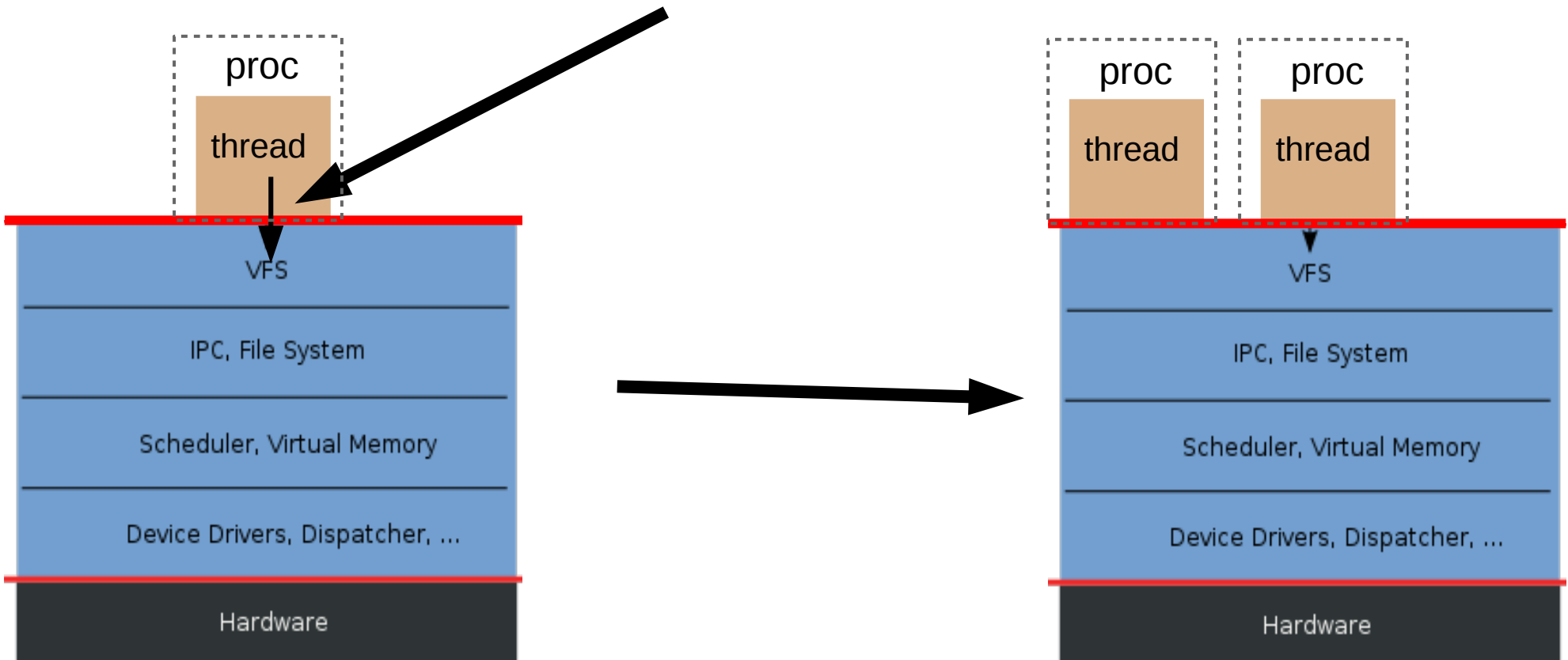
syscall: clone(всё совместное)



Processes, threads

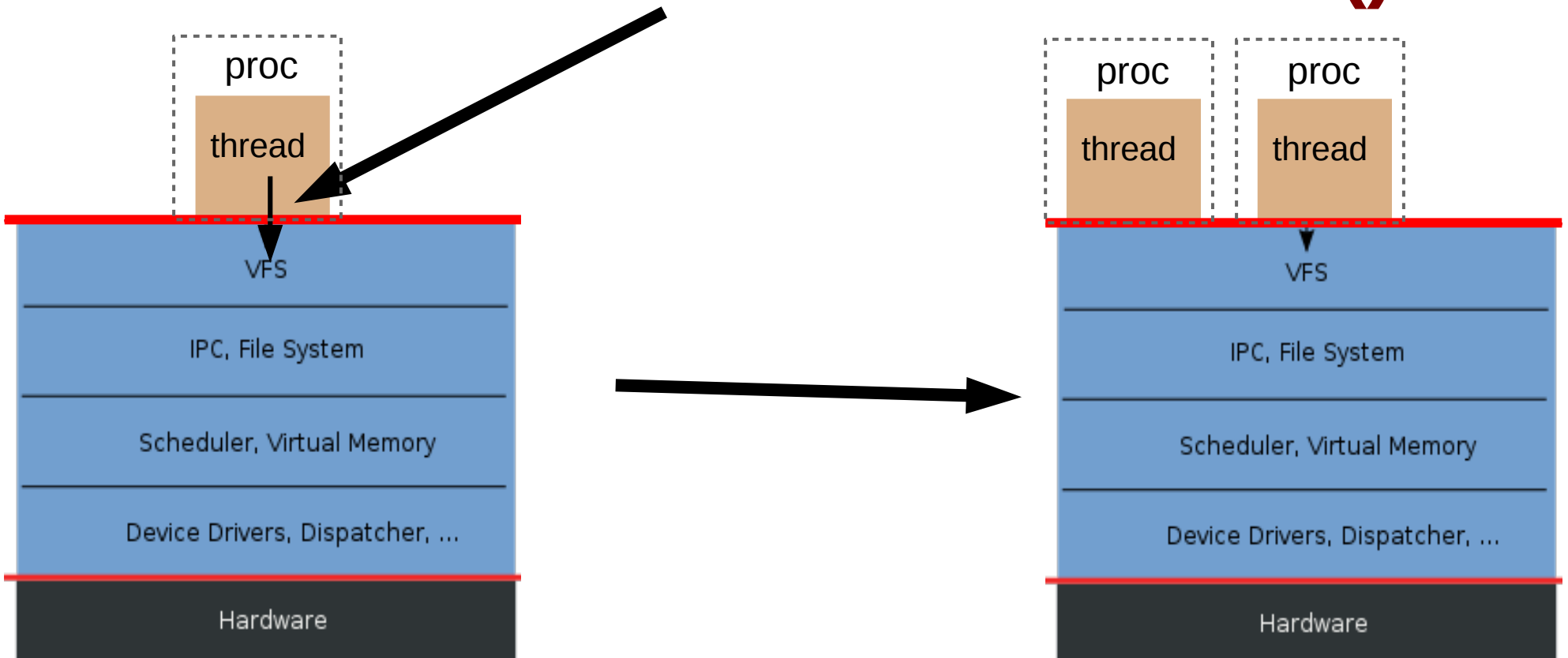


syscall: clone(всё раздельное)



Processes, threads

syscall: clone(всё раздельное) = **fork()**

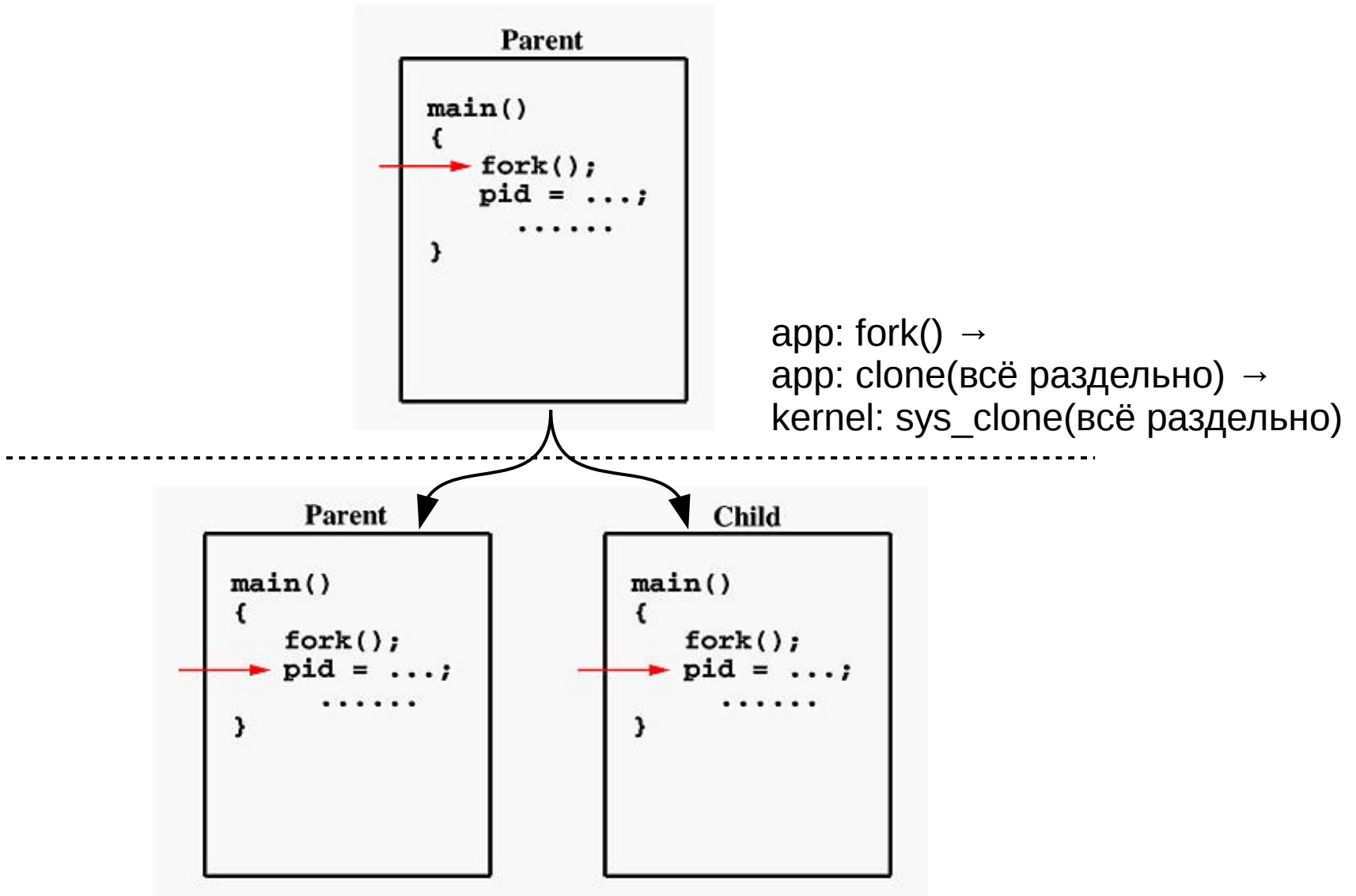


fork()

Parent

```
main()
{
  → fork();
  pid = ...;
  .....
}
```

fork()



fork()

Parent

```
main()      pid = 3456
{
  pid=fork();
  if (pid == 0)
    ChildProcess();
  else
    ParentProcess();
}

void ChildProcess()
{
  .....
}

void ParentProcess()
{
  .....
}
```

Child

```
main()      pid = 0
{
  pid=fork();
  if (pid == 0)
    ChildProcess();
  else
    ParentProcess();
}

void ChildProcess()
{
  .....
}

void ParentProcess()
{
  .....
}
```

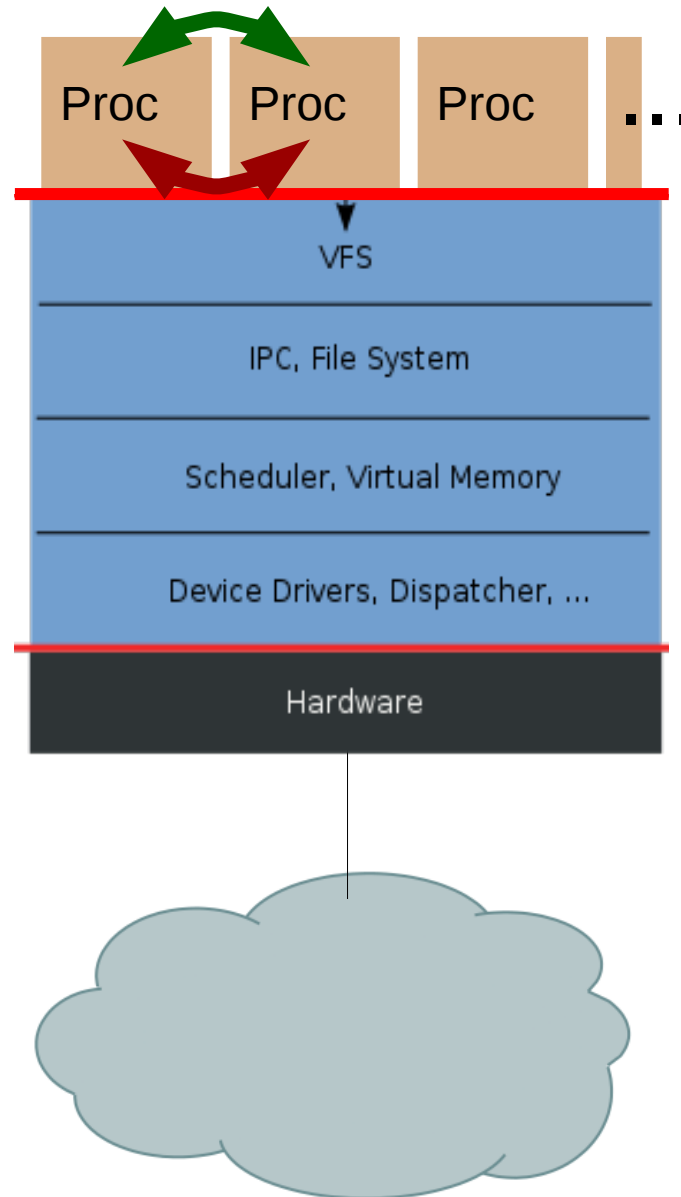
Итого

- Shared memory
- IPC, FS
- Network


Высокопроизводительное решение



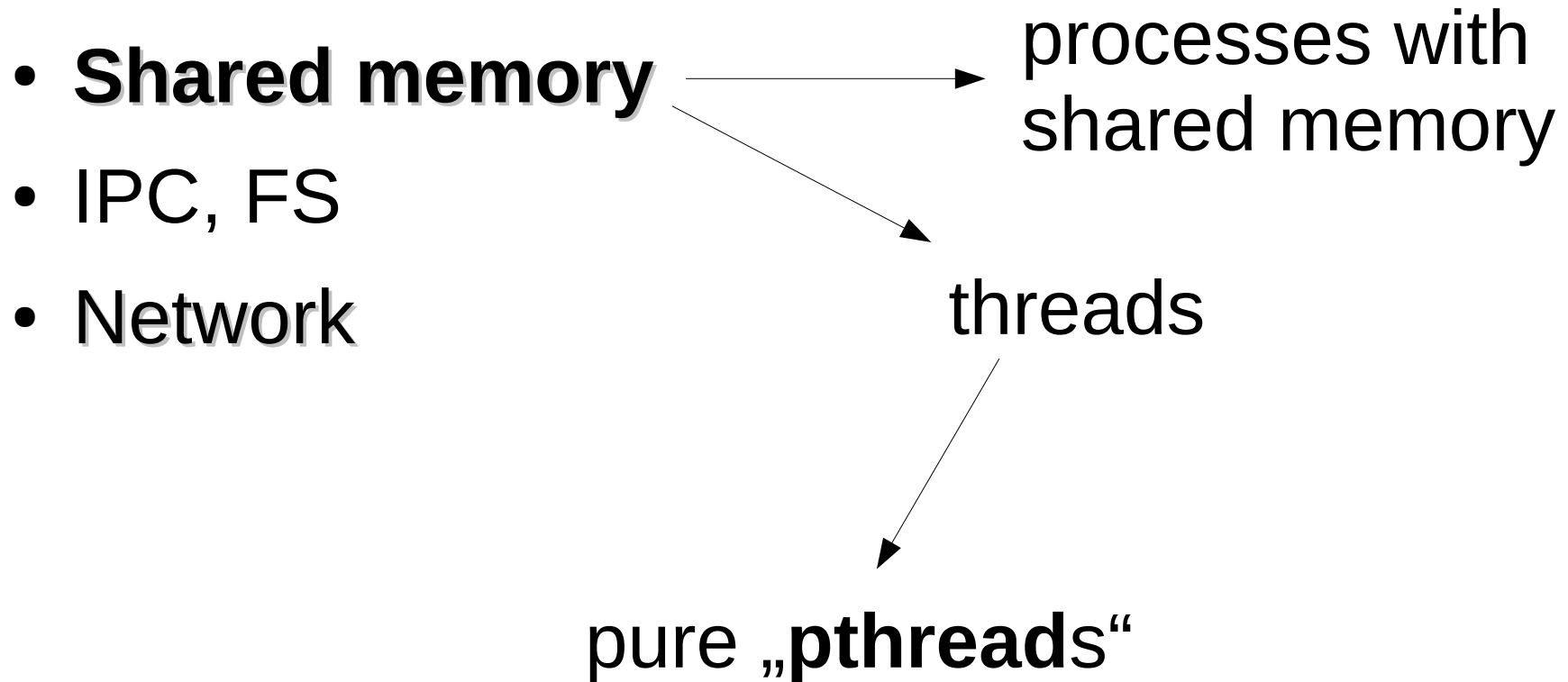
Shared memory



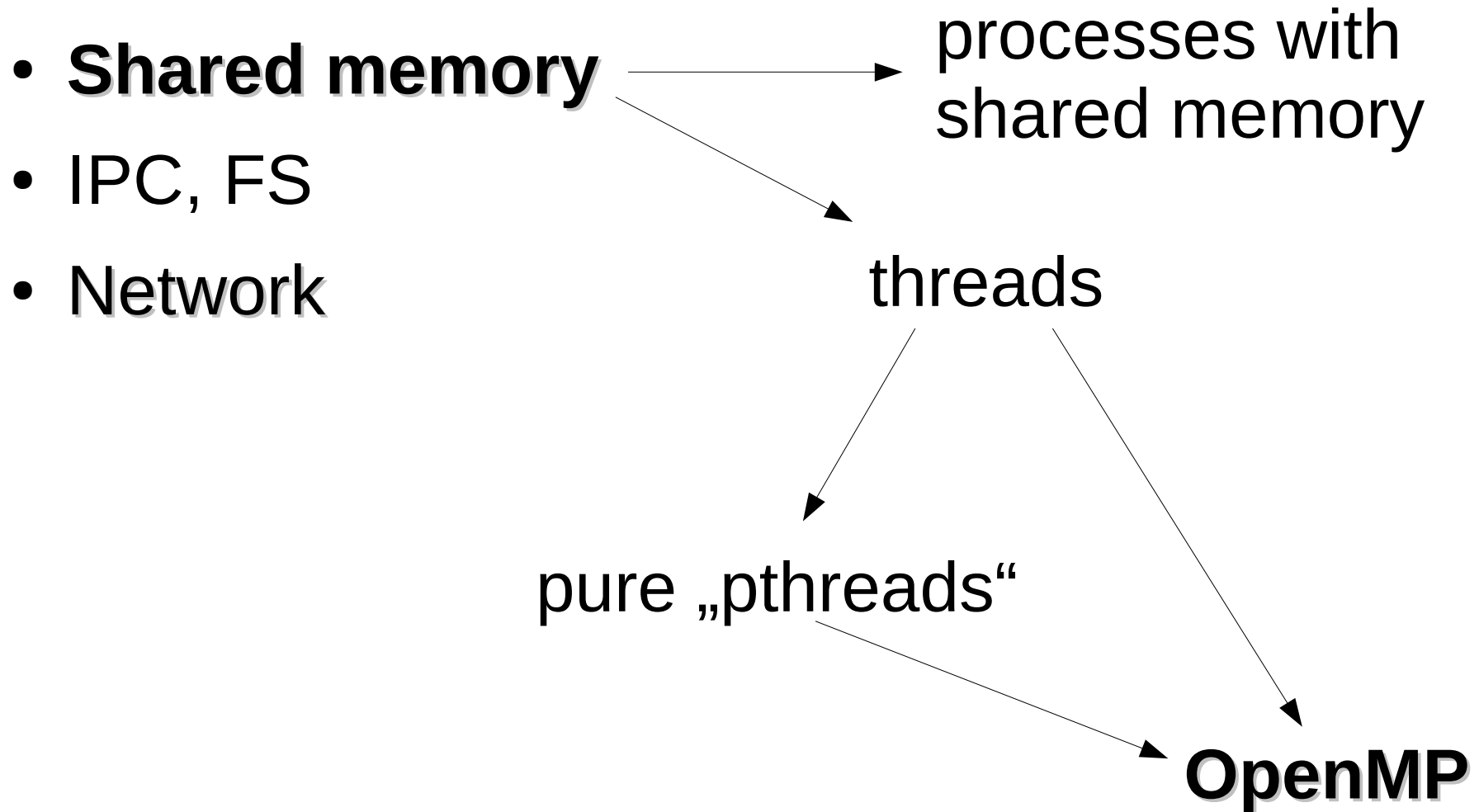
shared memory

- **Shared memory**  **processes with shared memory**
- IPC, FS
- Network

pthread



OpenMP



OpenMP

```
#include <stdio.h>
#include <omp.h>

#define N 100

int main(int argc, char *argv[])
{
    double a[N], b[N], c[N];
    int i;
    omp_set_dynamic(0);      // запретить библиотеке openmp менять число потоков во время исполнения
    omp_set_num_threads(10); // установить число потоков в 10

    // инициализируем массивы
    for (i = 0; i < N; i++)
    {
        a[i] = i * 1.0;
        b[i] = i * 2.0;
    }

    // вычисляем сумму массивов
    #pragma omp parallel for shared(a, b, c) private(i)
        for (i = 0; i < N; i++)
            c[i] = a[i] + b[i];

    printf ("%f\n", c[10]);
    return 0;
}
```