MC514–Sistemas Operacionais: Teoria e Prática

1s2010

Processos e Threads 2

Objetivos

- Pthreads
  - Revisão create e join
  - Passagem de parâmetros
  - Valor de retorno
- Pilha de execução

Create e Join

```
int pthread_create(pthread_t  *thread,
                   pthread_attr_t *attr,
                   void * (*start_routine)(void *),
                   void *arg);

int pthread_join(pthread_t thr,
                 void **thread_return);
```

Veja o código: create_join.c

Como passar algo mais que void *?

- void* é um apontador genérico
- casting pode permitir a passagem de estruturas
- Veja os códigos: estruturas?.c

Como encerrar a execução de uma thread

- Comando `return` na função principal da thread (passada como parâmetro em `pthread_create`)
- Análogo ao comando `return` na função `main()`

Veja os códigos: return0.c, return1.c pthread_return.c

Como encerrar a execução de uma thread

- `void pthread_exit(void *retval);`
- Análogo ao comando `exit(status);`
- Alternativa para término da thread principal sem terminar o programa.

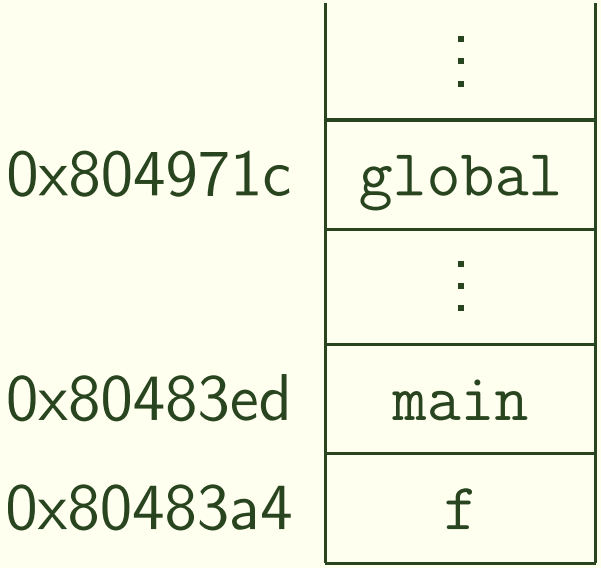
Veja os códigos: exit?.c e pthread_exit?.c

Pilha de execução:

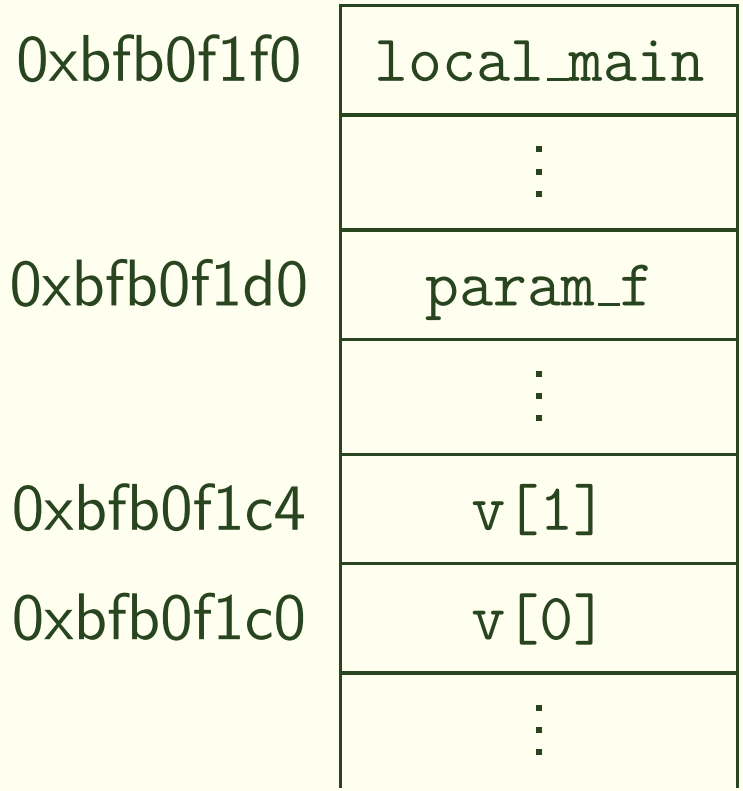
- Espaço para valor de retorno da função
- Argumentos
- Endereço de retorno
- Registradores
- Variáveis locais

Veja o código: pilha.c

Exemplo de endereços



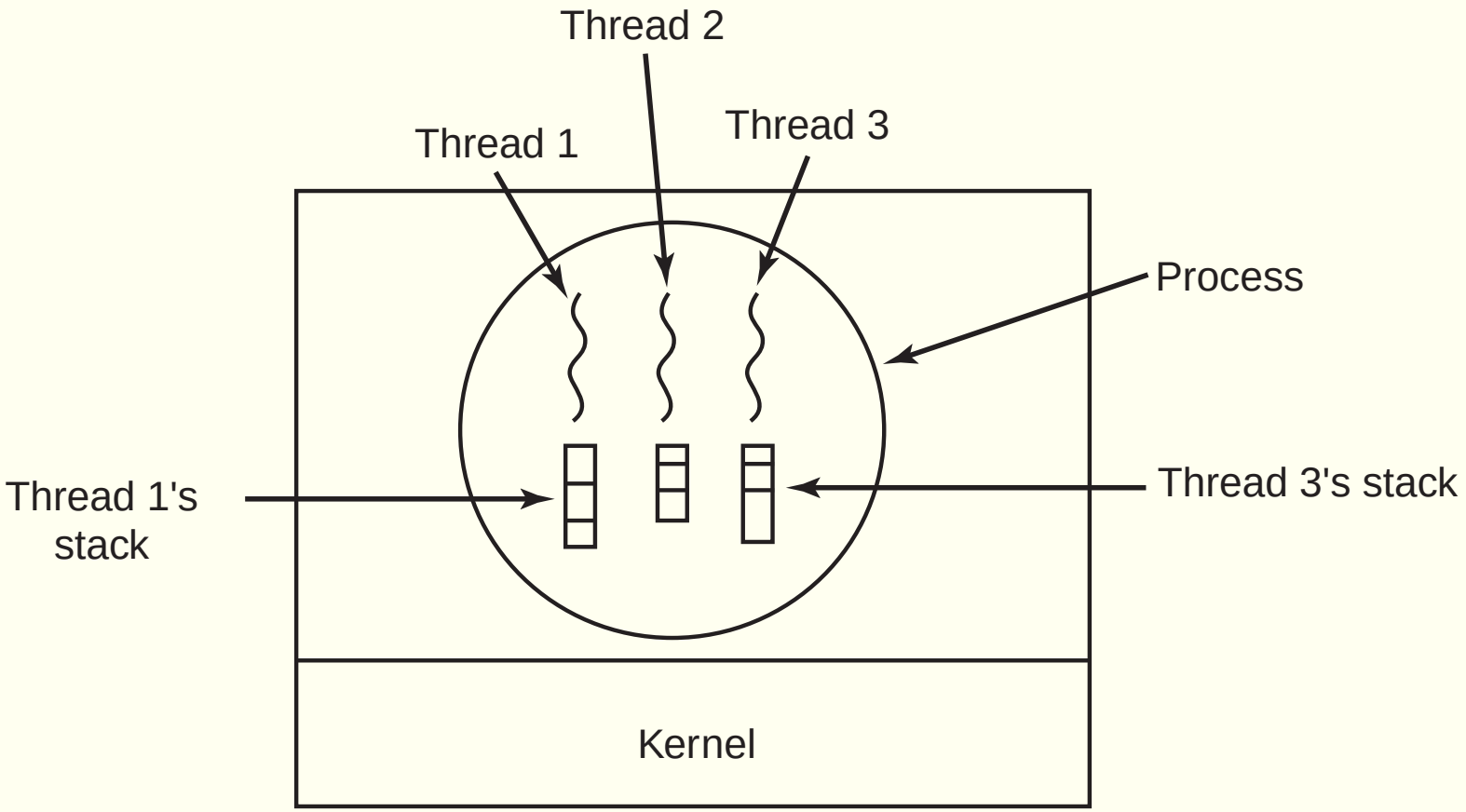
Exemplo de endereços



É muito fácil corromper a pilha

- Basta fazer acesso a posições não alocadas de um vetor
- Veja os códigos: corrompe_pilha.c e corrompe_pilha1.c

Pilhas independentes



Tanenbaum: Figura 2.8

Veja o código: pilhas.c

Uma thread pode corromper a pilha de outra thread

- Pilhas são independentes, mas não protegidas
- Veja o código: corrompe_thread.c