Troca de Contextos

Amir Annahas, Leonardo Reis

Setembro de 2019

1 Introdução

Este trabalho é resultado de uma pesquisa sobre trocas de contexto no sistema operacional Linux, suas chamadas POSIX, que facilitam a manipulação de contextos e um estudo sobre um código fornecido, que utiliza a implementação dessas chamadas.

O texto será dividido na explicação de cada uma das quatro funções propostas (getcontext, setcontext, swapcontext e makecontext) na explicação do significado dos campos da estrutura ucontext_t, seguido de uma breve conclusão da pesquisa.

2 Chamadas POSIX

Este capítulo contém as explicações das quatro chamadas POSIX propostas (getcontext, setcontext, swapcontext e makecontext), utilizando as informações do manual [1].

2.1 getcontext

Na chamada:

```
int getcontext(ucontext_t *ucp);
```

É inicializada a estrutura referenciada pelo ponteiro ucp com o contexto ativo no momento da execução da chamada. Em caso de sucesso, o retorno é zero e -1 caso contrário.

2.2 setcontext

Na chamada:

```
int setcontext(const ucontext_t *ucp);
```

Ocorre a restauração do contexto apontado pelo ponteiro ucp. Este contexto deve ter sido anteriormente obtido pela função getcontext acima. Em caso de sucesso, não há retonrno e o retorno será -1 caso contrário.

2.3 swapcontext

Na chamada:

```
int swapcontext(ucontext_t *oucp, const ucontext_t *ucp);
```

É salvo o contexto atual no endereço apontado pelo ponteiro oucp. Também ocorre a restauração do contexto apontado pelo ponteiro ucp. Em caso de sucesso, a função não retorna nada e retorna -1 caso contrário.

2.4 swapcontext

Na chamada:

```
void makecontext(ucontext_t *ucp, void (*func)(), int argc, ...);
```

É alterado as informações do contexto salvo na posição apontada por ucp. Antes de chamar essa função, é necessário alocar um novo espaço na memória para a pilha.

3 ucontext_t

A estrutura mínima de ucontext_t é:

```
typedef struct ucontext_t {
   struct ucontext_t *uc_link;
    sigset_t          uc_sigmask;
    stack_t           uc_stack;
    mcontext_t        uc_mcontext;
    ...
} ucontext_t;
```

Nessa estrutura, uc_link aponta para o contexto que será retomado quando a tarefa atual terminar. uc_sigmask representa os sinais bloqueados neste contexto. uc_stack é a pilha utilizada pelo contexto. E uc_mcontext é a representação específica da máquina do contexto salvo, que inclui os registradores referentes a threads.

4 Conclusão

A troca de contexto pode ser uma tarefa muito complicada, uma vez que envolve vários detalhes diferentes e são específicos para cada arquitetura.

Neste trabalho foi possível visualizar uma abordagem bem mais simplificada dessa atividade. O sistema operacional Linux oferece várias ferramentas que facilitam muito, abstraindo as informações internas a arquitetura dos desenvolvedores.

Referências

[1] Linux Programmer's Manual. Linux programmer's manual. http://man7.org/linux/man-pages/man3/getcontext.3.html, 2019. acessado em 20/09/2019.