

Adicionar uma nova chamada de sistema ao kernel do Linux

Fabício K. Januário Gabriel A. Silva João G. Guimarães

19 de janeiro de 2021

1 Material

Para realização dessa atividade, foram utilizados os seguintes softwares:

- Kernel: 5.8.0-25-generic
- Sistema Operacional: Ubuntu 20.10

2 Passos

2.1 Preparação de ambiente

Atualizar todas as dependências do sistema operacional e do processo de compilação do Kernel.

```
$ sudo apt update && sudo apt upgrade -y &&  
sudo apt install build-essential libncurses-dev  
bison flex libssl-dev libelf-dev dwarves
```

2.2 Baixar a versão correta do Kernel

O versionamento do Kernel do Linux segue o padrão **SemVer**, sendo assim, para evitar diversos conflitos que possam ser gerados utilizando o novo Kernel, baixe um que somente o último valor da versão mude.

Ex.: caso o comando a baixo retorne a versão 4.7.1, baixe versões 4.7.x.

Verificar a versão atual do Kernel

```
$ uname -r
```

Baixar uma versão compatível seguindo a orientação dada acima e extraí-la.

```
$ wget https://mirrors.edge.kernel.org/pub/linux/  
kernel/v5.x/linux-5.8.1.tar.gz && tar -xf linux-5.8.1.tar.gz
```

3 Criando a nova funcionalidade no Kernel

A nova funcionalidade adicionada ao Kernel consiste em adicionar uma mensagem de log ao buffer do Kernel e retornar o valor '0' quando o método for chamado. Essa funcionalidade foi implementada com o seguinte código:

```
#include <linux/kernel.h>
#include <linux/syscalls.h>

SYSCALL_DEFINE0(helloworld) {
    printk("\nHello World!\n");

    return 0;
}
```

Obs.: a mensagem de log gerada pode ser consultada pelo terminal utilizando o comando 'dmesg'.

4 Editando o Kernel

Adicionando o novo método a tabela Syscall que se encontra no diretório *arch/x86/entry/syscalls/syscall_64.tbl*.

440	64	helloworld	sys_helloworld
-----	----	------------	----------------

*Obs.: Note que o kernel na versão 5.8.1 no arquivo *syscall_64.tbl* vc deverá adicionar o texto junto com as chamadas comuns.*

Criando a assinatura do método, para isso foi necessário editar o arquivo *include/linux/syscalls.h*

```
asmlinkage long sys_helloworld(void);
```

Alterando o Makefile para compilar o diretório *helloworld/* criado no passo 3.

```
core-y += kernel/ certs/ mm/ fs/ ipc/ security/ crypto/
block/ helloworld/
```

5 Compilando

Utilizando as configurações atuais da máquina.

```
$ make localmodconfig
```

Compilando utilizando 4 núcleos do processador para um melhor desempenho.

```
$ make -j4
```

6 Instalação

```
$ sudo make modules_install install -j4
$ sudo update-grub
```

7 Teste da chamada implementada

Para verificar se o kernel foi atualizado, digite:

```
$ uname -r
```

Para fazer o teste da chamada implementada, deverá ser criado um arquivo `userspace.c` com o seguinte código:

```
#include <stdio.h>
#include <linux/kernel.h>
#include <sys/syscall.h>
#include <unistd.h>

int main()
{
    long int amma = syscall(440);
    printf("System call sys_hello returned %ld\n", amma);
    return 0;
}
```

Para executar o teste, digite os seguintes comandos:

```
$ sudo gcc userspace.c
$ sudo ./a.out
```

E para visualizar a mensagem no kernel:

```
$ sudo dmesg
```

Obs.: As imagens de saídas estão no Capítulo Anexos.

8 Questionário

8.1 A implementação da chamada de sistema e o teste funcionaram corretamente?

Sim, mas teve que ser feitas algumas alterações do tutorial passado em sala, pois a partir da versão 5.0 do Kernel, os métodos adicionais não se chamam *main* e sim *SYSCALL_DEFINE0*

8.2 Qual o nome do arquivo executável que corresponde ao Kernel?

O nome do arquivo é *bzImage*.

8.3 Após a instalação do Kernel, em qual local (diretório) foi armazenado o executável do Kernel?

O arquivo compilado do Kernel se encontra no diretório *arch/x86/boot/*

8.4 Em qual nível de privilégio a chamada de sistema implementada irá executar (usuário ou kernel/núcleo)?

Como a instrução foi compilada juntamente com o código do Kernel e utilizando suas bibliotecas internas, o código será executado em nível de Kernel.

8.5 Um roteiro típico contendo as etapas da execução de uma chamada de sistema é apresentado nas páginas 23 e 24 do livro (Capítulo 2 do livro do Mazieiro). Você entendeu todos os passos de 1 a 8?

Sim.

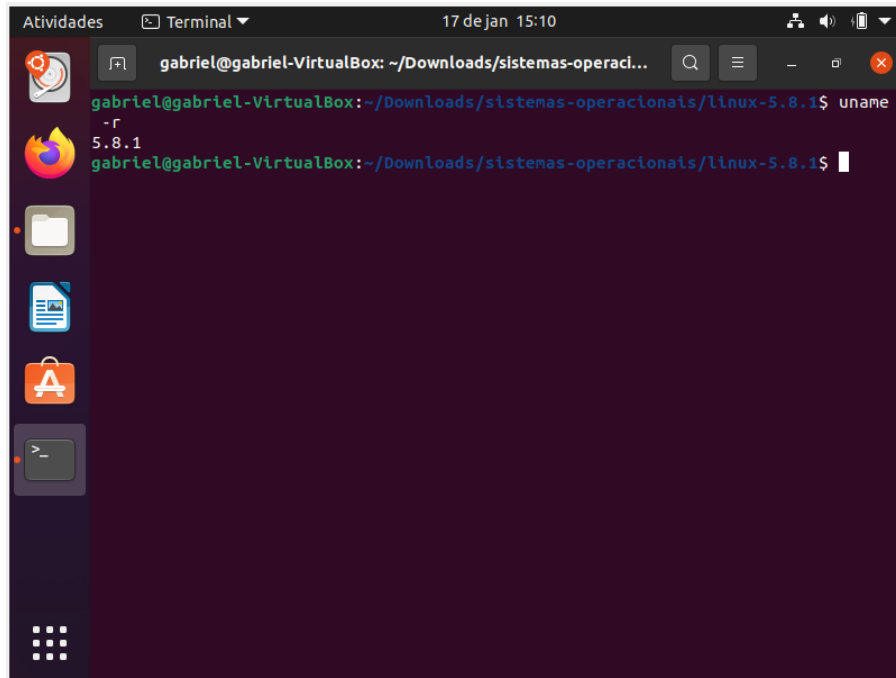
8.6 O Kernel precisará de ser recompilado toda vez que uma nova funcionalidade for adicionada ao kernel? Explique. Provavelmente, você terá que pesquisar na internet. Não precisa se preocupar em dar a resposta correta, apenas pense sobre o assunto e procure responder a questão.

Em versões mais novas, todas as alterações no Kernel necessitam uma recompilação, porém em versões antigas como as *3.x.x*, é possível alterar a instrução a ser executar no endereço de memória.

Obs.: considerando que módulos/drivers não são adição de novas funcionalidades.

9 Anexos

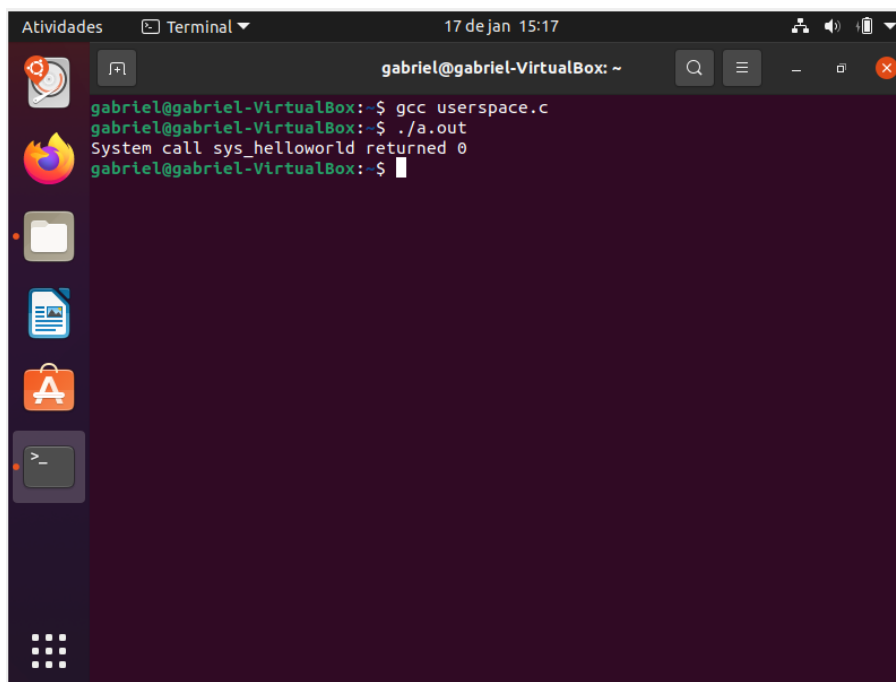
9.1 Nova versão do kernel



A terminal window titled "gabriel@gabriel-VirtualBox: ~/Downloads/sistemas-operaci..." is shown. The window has a dark background with a sidebar on the left containing icons for various applications. The terminal output shows the command "uname" being executed, resulting in the output "5.8.1".

```
gabriel@gabriel-VirtualBox: ~/Downloads/sistemas-operaci...$ uname  
-r  
5.8.1  
gabriel@gabriel-VirtualBox: ~/Downloads/sistemas-operaci...$
```

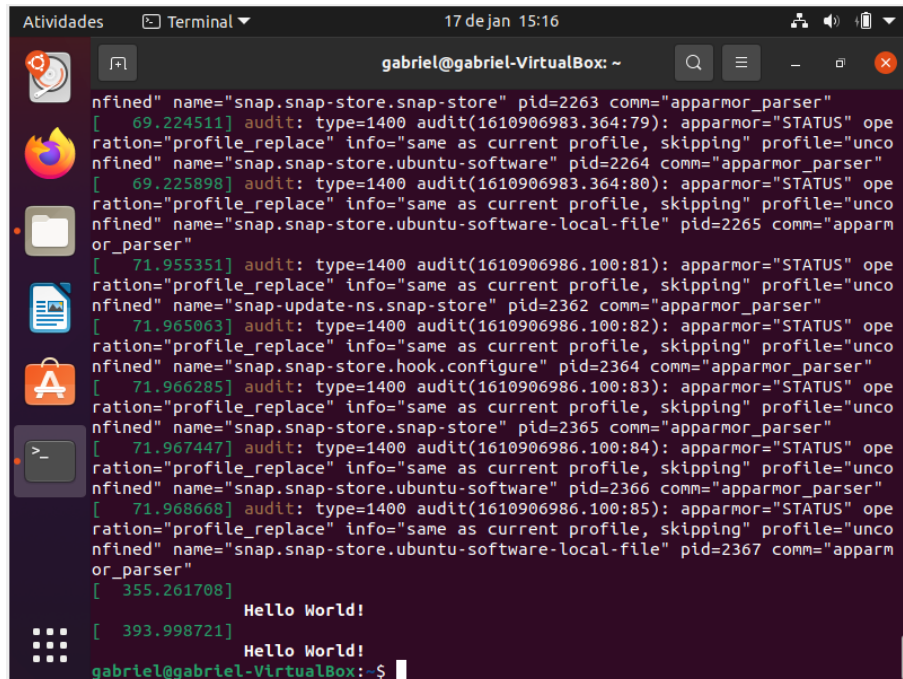
9.2 Teste para a chamada no kernel



A terminal window titled "gabriel@gabriel-VirtualBox: ~" is shown. The window has a dark background with a sidebar on the left containing icons for various applications. The terminal output shows the compilation and execution of a C program named "userspace.c". The command "gcc userspace.c" is executed, followed by the command ". /a.out". The output of the program is "System call sys_helloworld returned 0".

```
gabriel@gabriel-VirtualBox: ~$ gcc userspace.c  
gabriel@gabriel-VirtualBox: ~$ ./a.out  
System call sys_helloworld returned 0  
gabriel@gabriel-VirtualBox: ~$
```

9.3 Mensagem no kernel



A screenshot of a Linux terminal window titled "gabriel@gabriel-VirtualBox: ~". The window shows a series of kernel audit logs from the apparmor parser. The logs include timestamps, audit IDs, and details about profile replacements and status checks. After the logs, the user has typed "Hello World!" and the terminal has echoed it back. The prompt "gabriel@gabriel-VirtualBox:~\$" is visible at the bottom.

```
nfined" name="snap.snap-store.snap-store" pid=2263 comm="apparmor_parser"
[ 69.224511] audit: type=1400 audit(1610906983.364:79): apparmor="STATUS" ope
ration="profile_replace" info="same as current profile, skipping" profile="unco
nfined" name="snap.snap-store.ubuntu-software" pid=2264 comm="apparmor_parser"
[ 69.225898] audit: type=1400 audit(1610906983.364:80): apparmor="STATUS" ope
ration="profile_replace" info="same as current profile, skipping" profile="unco
nfined" name="snap.snap-store.ubuntu-software-local-file" pid=2265 comm="apparm
or_parser"
[ 71.955351] audit: type=1400 audit(1610906986.100:81): apparmor="STATUS" ope
ration="profile_replace" info="same as current profile, skipping" profile="unco
nfined" name="snap-update-ns.snap-store" pid=2362 comm="apparmor_parser"
[ 71.965063] audit: type=1400 audit(1610906986.100:82): apparmor="STATUS" ope
ration="profile_replace" info="same as current profile, skipping" profile="unco
nfined" name="snap.snap-store.hook.configure" pid=2364 comm="apparmor_parser"
[ 71.966285] audit: type=1400 audit(1610906986.100:83): apparmor="STATUS" ope
ration="profile_replace" info="same as current profile, skipping" profile="unco
nfined" name="snap.snap-store.snap-store" pid=2365 comm="apparmor_parser"
[ 71.967447] audit: type=1400 audit(1610906986.100:84): apparmor="STATUS" ope
ration="profile_replace" info="same as current profile, skipping" profile="unco
nfined" name="snap.snap-store.ubuntu-software" pid=2366 comm="apparmor_parser"
[ 71.968668] audit: type=1400 audit(1610906986.100:85): apparmor="STATUS" ope
ration="profile_replace" info="same as current profile, skipping" profile="unco
nfined" name="snap.snap-store.ubuntu-software-local-file" pid=2367 comm="apparm
or_parser"
[ 355.261708]
Hello World!
[ 393.998721]
Hello World!
gabriel@gabriel-VirtualBox:~$
```