Introdução ao Minix
Relatório do Projeto 1

Diogo T. Sant'Anna, *RA 169966,* Luciano G. Zago, *RA 182835,*
e Maria A. Paulino, *RA 183465*

Resumo—Este projeto teve por objetivo promover a familiarização com o ambiente do sistema operacional Minix 3, o qual foi emulado pelo QEMU. Ao longo ddeste relatório é descrito o processo para instalação do Minix, os problemas encontrados, soluções para esses e os meios adotados para resolução dos objetivos propostos no enunciado do projeto em questão - compilação do kernel, mudança da syscall exec e do banner do kernel e identificação da syscall 33 (que concluímos ser relacionada ao comando truncate(2)). Além disso, o projeto utilizou um versionamento GIT do GitLab do Instituto de Computação, configurado com um sistema de Continuous Integration para compilação do SO.

1 Introdução

O Objetivo deste projeto é instalar o sistema operacional MINIX através de uma máquina virtual, e fazer algumas alterações em seu código fonte, a fim de conhecer a estrutura na prática de um sistema operacional e entender o seu funcionamento.

O Minix foi desenvolvido por Andrew Tenenbaum para compensar a proibição da AT&T contra o estudo de sistemas operacionais baseados em código UNIX e promover uma ferramenta de ensino para seus alunos. Originalmente, foi projetado para ser compatível com a versão 7 do UNIX e em seguida passou a ser desenvolvido baseado no padrão POSIX. O Minix foi escrito a partir do zero e mesmo sendo compatível com UNIX, não contém AT&T possibilitando sua distribuição livre.

O Minix 1 e 2 são entendidos como ferramenta de ensino. A versão 3 tem por objetivo se tornar usável um computadores com recursos limitados, sistemas embarcados e aplicações que demandam grande confiabilidade.

O MINIX 3 é um sistema operacional gratuito, e open-source, projetado para ser confiável, flexível e seguro. Ele é descrito no livro "Operating Systems: Design and Implementation 3nd Edition" de Andrew S. Tanenbaum e Albert S. Woodhull.

2 Desenvolvimento

Como foi utilizado o QEMU[1] para virtualizar o MINIX(3.4), os passos a seguir serão específicos para tal ferramenta.

2.1 Instalação: desafios e soluções

A seguir, será apontado o passo a passo executado pelo grupo e os obstáculos vistos em cada um desses. Em suma, seguiu-se o guia de instalação do MINIX[1].

Utilizamos o comando abaixo para criar a imagem da máquina virtual.

```
$ qemu-img create minix.img 8G
```

Em seguida, inicializamos a máquina virtual com a ISO de instalação do MINIX[2], versão de maio de 2017, no CD-ROM.

```
$ qemu-system-x86_64 -localtime
-net user -net nic -m 256 -cdrom
minix.iso -hda minix.img -boot d
```

Após o boot da ISO, seguiu-se o script de instalação sem problemas. Em seguida,

1. https://www.qemu.org/
2. http://download.minix3.org/iso/snapshot/

finalizou-se a sessão com o comando:
`# shutdown -h now.`

Inicializou-se novamente a máquina virtual, desta vez com o comando:
```
$ qemu-system-x86_64 -rtc base=utc
-net user -net nic -m 256 -hda
minix.img
```

Ao tentar prosseguir, integrantes do grupo observaram que não era possível utilizar a tecla '/' devido a escolha pelo teclado ABNT2 durante a instalação. Para resolver tal problema, alteramos o padrão para us-swap.
```
# loadkeys /usr/lib/keymaps/
us-swap.map
# cp /usr/lib/keymaps/us-swap.map
/etc/keymap
```

O último comando foi utilizado para salvar a configuração feita.

Depois de ter devidamente instalado o MINIX, optamos por clonar a imagem criada para ter um backup para o caso de algum teste comprometa a utilizada.

Assim como indicado pelo guia de pós instalação do MINIX [2], inseriu-se uma senha para o usuário root com `$ passwd`. Em seguida, proseguiu-se com a instalação de pacotes para facilitar o uso do sistema operacional, o primeiro escolhido teve por objetivo proporcionar o uso de acesso via ssh. Para isso, seguiu-se o processo descrito no site do sistema operacional[3] e no enunciado deste projeto, assim como a seguir.
```
# pkgin update
# pkgin install openssh
# pkgin install rsync
# cp /usr/pkg/etc/rc.d/sshd
/etc/rc.d/sshd
# etc/rc.d/sshd start
# printf 'sshd=YES\n' >>
/etc/rc.conf
```

Além disso, em `etc/ssh/sshd_config`, mudou-se o estado de `PermitirRootLogin` para `yes`[4] a partir de:
```
# printf 'PermitirRootLogin yes' >>
/usr/pkg/etc/ssh/sshd_config
```

Para permitir uma conexão direta com o ambiente minix emulado na máquina através do simples comando `ssh minix`, fez-se a seguinte alteração no arquivo `.ssh/config` da máquina anfitriã:

```
Host minix
    Hostname localhost
    Port 10022
    User root
```

Sendo assim, precisamos alterar a forma de inicialização da máquina virtual para direcionar a porta 1022 da máquina anfitriã (nosso computador) para a porta 22 do minix, possibilitando a comunicação por ssh:
```
# qemu-system-x86_64 -rtc base=utc
-net user,hostfwd=tcp::10022-:22
-net nic -m 256 -hda minix.img
```

A troca de arquivos entre os ambientes (simulado e realmente presente na máquina) é feita por `rsync -rptOv origem minix:destino`, que utiliza a configuração do ssh `minix:` para enviar arquivos à máquina virtual[5].

2.2 Estrutura do GIT

Na branch Master foi criado um diretório `projeto-1` e dentro deste, outro nomeado `minix` que contém código fonte do sistema operacional modificado. Como a estrutura dos arquivos do repositório foi alterada, alterou-se o arquivo `.gitlab-ci.yml` fornecido, que configura o Continuous Integration. A alteração feita foi substituir `--host-minix-source` `` `pwd` `` por `--host-minix-source` `` `pwd`/projeto-1/minix ``.

Além disso, para o CI funcionar, foi necessário habilitar a opção `Enable shared Runners` em `Settings > CI/CD > Runner Settings`

2.3 Compilação do kernel

Antes de de fato conseguir compilar o kernel efetuado alguma modificação, tentou-se recompilá-lo sem efetuar nenhuma modificação para validar o processo.

Primeiro tentou-se utilizando o comando `make build` dentro da pasta `/usr/src/`, mas a máquina virtual simplesmente parava no meio da execução. Tentou-se também o comando `make hdboot` na

pasta `/usr/src/releasetools`, mas a execução terminava em erro indicando arquivos faltantes ou incompletos da pasta `/usr/stc/minix/commands/fsck.mfs/`

Por fim membros do grupo conseguiram executar a compilação utilizando o comando `make build` ou somente `make` na pasta `/usr/src/`, seguido pelo comando `make hdboot` dentro da pasta `/usr/src/releasetools`. [11]

O comando `make build` demorou 11 horas até ser finalizado. Já o `make hdboot`, que só compila e instala o kernel, demorou cerca de 30 minutos.

2.4 Mudança da syscall exec

Para encontrar o arquivo no qual é definida a syscall exec, começamos a procura pela pasta `/usr/src/minix/kernel` e lendo o conteúdo dos arquivos e principalmente os comentários chegamos ao arquivo `/usr/src/minix/kernel/system/do_exec.c`. Alterando a função `do_exec` conseguimos fazer o kernel imprimir o nome do arquivo executado cada vez que algo é executado, mas não conseguíamos obter o caminho completo para o diretório onde se encontrava o arquivo executável.

Conseguimos encontrar o arquivo que inicia a execução de um processo, e que contem o caminho completo de diretórios do arquivo executável, acidentalmente: enquanto pesquisávamos pela web sobre como localizar os números de uma syscall específica, encontramos um site[12] que ensinava como criar uma syscall e dizia para inserir o arquivo contendo o código da syscall na pasta `/usr/src/minix/servers/vfs`.

Depois disso entramos na pasta citada, vimos um arquivo relacionado a syscall exec `/usr/src/minix/servers/vfs/exec.c`, no qual pudemos alterar a função `get_read_vp` inserindo uma linha com o comando `printf` para imprimir o nome completo do caminho de diretórios do arquivo executável (através da variável chamada `pathname`, um ponteiro para char / uma string).

2.5 Mudança do banner do kernel

Utilizando a função de pesquisa no repositório com a palavra `banner`, foi possível encontrar o comentário:
`/* Display the MINIX startup banner.*/`
em `minix/kernel/main.c` na função `announce()`. Logo, para alterar o banner, foi preciso somente escrever no comando `printf` dentro do `announce()`.

2.6 Syscalls

Através de [6] e [7], foi possível entender a implementação das syscalls no MINIX e conhecer os arquivos que eram utilizados para isso.

Assim, o arquivo `include/minix/callnr.h` contém as constantes com os nomes das syscalls e seus números. O arquivo `servers/pm/table.c` contém os nomes das funções relacionadas às syscalls.

Portanto, a syscall 33 é relacionada à constante `VFS_TRUNCATECALL` que se refere ao comando `truncate(2)`.

3 CONCLUSÃO

Conseguimos alcançar os objetivos propostos pelo trabalho: navegando e conhecendo o Minix foi possível recompilar o kernel depois de alterá-lo conforme as exigências propostas. Conseguimos com sucesso localizar a syscall correspondente ao número 33 relacionando-a com o comando truncate(2), modificar o texto do banner do MINIX e fazer o SO imprimir o caminho do arquivo executável ao executar cada programa.

O primeiro contato com o MINIX nos foi muito útil pois surgiram dificuldades em várias partes das configurações básicas do sistema. O trabalho demandou uma parcela do tempo navegando pelos diretórios e arquivos do kernel e do sistema operacional em geral, para nos familiarizarmos com a estrutura do SO, principalmente em busca dos arquivos específicos que nos ajudariam a resolver os problemas propostos pelo enunciado do projeto.

Contribuiu também ao fornecer o primeiro contato com o GitLab do Instituto de Computação[13] e em especial com o Continuous Integration[14].

REFERÊNCIAS

[1] http://wiki.minix3.org/doku.php?id=usersguide:runningonqemu
[2] http://wiki.minix3.org/doku.php?id=usersguide:postinstallation
[3] http://wiki.minix3.org/doku.php?id=releases:3.2.0:usersguide:installingbinarypackages
[4] https://gist.github.com/Drowze/2f7cbce35ade1fa94b2511f4138a32c2
[5] http://cobweb.cs.uga.edu/~maria/classes/4730-Fall-2016/project3minix-intro.html
[6] https://github.com/rhiguita/lab-minix/blob/master/pt_br/Atividade2-Criando_Uma_Chamada_de_Sistema-Portuguese.txt
[7] Jayaraman, Karthick. *How to Add a New System Call for Minix 3*. Department of Electrical Engineering & Computer Science, Syracuse Univerity, Syracuse, New Your. Disponível em: http://www.cis.syr.edu/~wedu/seed/Labs/Documentation/Minix3/How_to_add_system_call.pdf
[8] http://wiki.minix3.org/doku.php?id=developersguide:usinggit
[9] http://www.minix3.org/doc/A-312.html
[10] http://wiki.minix3.org/doku.php?id=developersguide:trackingcurrent
[11] http://wiki.minix3.org/doku.php?id=developersguide:rebuildingsystem
[12] http://www-di.inf.puc-rio.br/~endler//courses/inf1019/Minix/Instalacao_Syscall.pdf
[13] https://gitlab.ic.unicamp.br/
[14] https://about.gitlab.com/features/gitlab-ci-cd/