Relatório do Projeto 1
MC504 - Grupo 12

Matheus Domingues Casanova Nogueira (185135)
André Rodrigues Alves da Silva (231392)
Daniel de Sousa Cipriano (233228)
Guilherme Gomes Gonçalves (170927)

***Resumo*—Neste relatório de projeto, apresentamos nossa primeira experiência com o sistema operacional Minix 3, um S.O. gratuito e open-source que é utilizado no mundo inteiro como uma ferramenta para o aprendizado de sistemas operacionais (mais detalhes sobre o sistema serão dados na próxima seção).**

Abordamos nosso procedimento para a instalação e configuração do sistema, e explicamos nossas dificuldades e soluções que encontramos para resolvê-las. Além disso, mostramos como realizamos algumas modificações simples no Kernel do Minix: alteramos algumas mensagens padrão do Kernel do S.O. e modificamos levemente o funcionamento do comando `exec`. Explicamos como conseguimos recompilar o sistema para aplicar essas modificações e os desafios que todas essas etapas trouxeram para o trabalho.

Por fim, concluímos o relatório com uma breve reflexão sobre nosso aprendizado no projeto, relembrando os pontos principais do trabalho e descrevendo novos conhecimentos e mudanças de paradigmas que tivemos com a realização da atividade. Um pequeno parágrafo descrevendo como dividimos as tarefas também está incluso nessa seção.

I. Introdução

MINIX é um sistema operacional gratuito e open-source tipo Unix que segue uma arquitetura de microkernel. O sistema foi desenvolvido por Andrew S. Tanenbaum em 1987, na Vrije Universiteit em Amsterdam, com o intuito de exemplificar e ensinar princípios de design de sistemas operacionais para estudantes e pessoas interessadas em aprender como eles funcionam. A versão 1.0 do Minix tinha um total de 12 mil linhas de código escritas na linguagem de programação C, e contava com um kernel, um gerenciador de memória e um sistema de arquivos [1]. A relativa simplicidade da implementação fez com que o Minix se tornasse uma excelente ferramenta para o aprendizado de diversos estudantes.

Durante vários anos, Tanenbaum lançou diversas atualizações de compatibilidade [2] com novas arquiteturas de hardware para o Minix, e em 2005 foi anunciada a versão 3.0 do sistema operacional (Minix 3), que além de continuar com a simplicidade e a didática das versões anteriores, tinha como objetivo tornar o Minix um "sistema operacional viável em computadores e sistemas embarcados com recursos limitados para executarem aplicações que precisem de alta estabilidade" [3].

A versão 3 do sistema operacional é, atualmente, a mais recente lançada, e conta com suporte para arquitetura IA-32 e ARM. É possível usar o sistema operacional num formato Live CD (sem instalações), numa instalação normal em um disco rígido e também em um ambiente de máquina virtual [4]. É também possível instalar várias aplicações comuns de sistemas Unix e compilar e escrever em diversas linguagens de programação nessa versão do sistema [4]. O Minix 3 também possui suporte para memória virtual e é capaz de reiniciar e reparar drivers problemáticos sem que outros processos sejam afetados, tornando ele um sistema operacional "auto-reparável" [5].

II. Instalação e configuração

Como explicado anteriormente, existem várias formas para instalação e utilização do Minix. Para esse projeto, executamos o sistema operacional em um hypervisor de máquina virtual, o VirtualBox, atualmente desenvolvido pela Oracle. Utilizamos a última versão lançada do VirtualBox, e a versão do Minix instalada foi a versão 3.4.0rc6, adquirida gratuitamente no site do sistema operacional. Seguimos as etapas que foram descritas no documento disponibilizado para o Projeto 1, e também realizamos várias snapshots do estado da nossa máquina virtual ao longo do projeto.

A aplicação do VirtualBox apresentou alguns desafios quando instalada no sistema operacional Macos Ventura 13.5.1, em um processador ARM64 Apple Silicon. Após baixarmos o arquivo .iso do Minix e configurarmos a máquina com os recursos de hardware, conseguimos realizar a instalação sem nenhum problema, mas na etapa de remover o arquivo .iso da máquina virtual (seção 3.1 do documento, etapa 11), o sistema não reconheceu que a ISO havia sido retirada e pediu para realizarmos a instalação do início novamente. Tentamos fazer mais uma vez o procedimento e mesmo assim não conseguimos prosseguir com a instalação no Macos Ventura 13.5.1. Felizmente, tínhamos acesso a outros sistemas operacionais para instalarmos o VirtualBox (Linux Mint e Windows 10), e repetimos a etapa da mesma maneira e conseguimos fazer a instalação funcionar. Acreditamos que possa existir um bug com a versão específica que baixamos do VirtualBox, ou até com a versão do Macos Ventura 13.5.1, já que VirtualBox descontinuou o suporte à processadores ARM64 Apple Silicon. O restante da configuração foi realizado sem nenhum problema, exceto na seção 3.1 do documento, etapa 18, em que era pedido para executar um `printf` para escrever no arquivo `/etc/rc.conf`. O comando descrito no documento não funcionou algumas vezes, mas ao utilizar a aplicação `nano` conseguimos editar e salvar o arquivo normalmente.

Após o término da configuração de setup inicial do Minix 3 na máquina virtual, houveram alguns problemas relacionados à

especificação de rede da máquina. Rodando o sistema em uma rede cabeada, configurada em modo Bridge Adaptor, não houve reconhecimento de rede pelo VirtualBox, porém, ao alterar a especificação para modo NAT, funcionou corretamente. O não reconhecimento de rede gera problemas ao tentar instalar e atualizar pacotes com o gerenciador de pacotes padrão do Minix 3 (pkgin). Esse contratempo ocorre apenas ao tentar utilizar o sistema em rede cabeada, a utilização em rede sem fio no modo Bridge Adaptor foi bem sucedida.

O sistema operacional pode ser visto na Figura 1, que foi feita após realizarmos com êxito a instalação e as configurações descritas no documento do Projeto 1.

III. Modificações no Kernel

Como explicado na introdução, o Minix é um sistema open-source. Todas as versões do Minix 3 podem ser encontradas em um repositório no Git, disponível no site oficial do sistema operacional. Com acesso ao código fonte, é possível realizar diversas modificações no sistema. O procedimento padrão para realizá-las é clonar o repositório do Git no diretório `/usr/src` (dentro do próprio Minix), realizar alterações neste diretório (que é o código-fonte), e recompilar o kernel usando o comando `make build` [7].

O comando `make build` apresentou diversos problemas nas máquinas de nossa equipe. O procedimento não deveria demorar muito, pois o sistema é leve e utiliza poucos recursos. Apesar disso, não conseguimos compilar em algumas de nossas máquinas, pois mesmo após 10 horas o procedimento não terminava. Tentamos recompilar o sistema mais vezes, e ainda assim após 6 horas a compilação ainda não havia terminado. Infelizmente, não conseguimos resolver esse problema mesmo trocando o S.O. da máquina física utilizada e reinstalando todos os arquivos. Conseguimos, porém, recompilar o Minix com o mesmo comando em outras máquinas, e em menos de 1 hora.

Houveram também tentativas de solucionar esses erros de compilação, utilizando outro hypervisor de máquina virtual, no caso o VMware, porém não obtivemos sucesso e os mesmos problemas permaneceram. Tentamos realizar a recompilação em vários sistemas operacionais diferentes, entre eles estão, Windows 10, Windows 11, Linux Mint e Kali Linux. Constatamos que independente do sistema operacional ou máquina virtual utilizada, os mesmos problemas se mantinham. Em relação aos sistemas Linux Mint e Kali Linux, eles foram utilizados rodando em modo Live USB, com isso, conseguimos testar os mesmos procedimentos, nos mesmos sistemas, em máquinas físicas diferentes, com sucesso na recompilação. Concluímos que esses erros relacionados a recompilação, estão ligados de alguma forma ao hardware específico das máquinas e não necessariamente ao tipo de sistema utilizado.

Também obtivemos alguns erros de pós compilação. Algumas vezes, mesmo após a recompilação do kernel, as mudanças realizadas não foram refletidas no sistema. É válido também citar que, consultando a página que contém o tutorial de instalação do Minix 3 no VirtualBox [6], notamos que a versão do VirtualBox utilizada para testes da versão de sistema do projeto (Minix 3.4.0.rc6), foi uma versão antiga, no caso a versão VirtualBox 5.2.4. A versão utilizada para o projeto é superior ao VirtualBox 6.1.

A seguir, explicamos detalhadamente as alterações que conseguimos realizar no Kernel do Minix 3.

A. Alteração dos Banners

Foram realizadas alterações em algumas mensagens que são mostradas na inicialização do sistema. Para começar, modificamos a mensagem que aparece durante o boot do Minix: "Copyright 2016, Vrije Universiteit...". Acrescentamos novas informações a essa mensagem, escrevendo um pequeno banner com elas.

Para encontrarmos o código que realizava a impressão dessa mensagem, utilizamos o comando `grep -R \Copyright 2016, Vrije Universiteit" ./`. Encontramos então a localização do arquivo, que estava em `/usr/src/minix/kernel/main.c`. Conseguimos identificar o comando `printf` que realizava a impressão dessa mensagem, e logo após ele adicionamos um novo `printf` com o novo banner que queríamos imprimir. As alterações exatas do código podem ser vistas no repositório do GitLab do grupo 12. Após recompilar o kernel, a mensagem foi impressa normalmente, como pode ser visto na Figura 2.

Além da alteração descrita acima, também modificamos a mensagem que o Minix imprime após realizarmos o login da nossa conta: "For post-installation usage tips such....". Removemos essa mensagem e adicionamos um pequeno banner que escrevemos com algumas informações sobre a UNICAMP e a matéria MC504.

Para encontrarmos o arquivo que continha essa mensagem, utilizamos também o comando descrito anteriormente, `grep -R \For post-installation" ./`. Encontramos que o texto estava em um arquivo na localização `/usr/src/etc/motd`. Após isso, apenas tivemos que modificar o arquivo, removendo o texto que estava lá e adicionando o nosso novo texto.

É importante dizer que, segundo a documentação do Minix, o comando `make build` não atualiza todos os arquivos do diretório `/usr/src/etc/`. O arquivo `motd`, que contém a mensagem que alteramos, é um dos arquivos que não são atualizados por esse comando. Dessa maneira, para que a alteração tivesse efeito, tivemos que alterar o arquivo diretamente no diretório do próprio sistema operacional, o `/etc/motd`. Quando reiniciamos o sistema usando `reboot`, a mensagem foi alterada com sucesso. O novo arquivo `motd` pode ser encontrado no diretório no GitLab do grupo 12, mas ressaltamos que é necessário que ele seja alterado diretamente no sistema operacional para que ele funcione corretamente. A imagem da modificação pode ser vista na Figura 3.

Por fim, durante o processo de recompilação do Kernel, tivemos alguns problemas que impediram as modificações de serem concretizadas. O principais foram, tempo de compilação muito alto ou congelamento do processo, além disso, as threads do comando `make build` paravam em alguns momentos, cancelando o processo de recompilação. A imagem desse erro pode ser vista na Figura 4

B. Modificando o código `exec.c`

O objetivo aqui era fazer com que os comandos executados pelo usuário no terminal, ou até mesmo programas chamados

de forma indireta por outros programas, tivessem seu path exibido no terminal. Para isso, procuramos pelo arquivo `exec.c` responsável pela execução dos programas do terminal.

O arquivo escolhido para realizar as modificações foi o `exec.c` localizado em `src/bin/sh/`, já que a documentação indicava que esse era o responsável pela execução dos programas do shell [5]. Nesse arquivo foram adicionadas duas linhas de código, ambas consistindo em `printf` *statements* na função `shellexec`. Umas dessas linhas é colocada na primeira condição, que é o caso em que o caminho do arquivo executável é especificado na linha de comando. Se o caminho for especificado, será exibido devido a linha adicionada. A segunda linha é adicionada no bloco executado quando o caminho não é especificado. Nesse caso, um *script* que irá procurar pelo comando em diferentes diretórios listados no path. Caso esse comando seja encontrado, seu caminho será atribuído à variável *cmdname*, que em seguida será exibida pelo segundo `printf` adicionado. É possível observar essas modificações na Figura 5. Após recompilar o kernel, os caminhos dos comandos executados estavam sendo exibidos conforme o esperado. A título de demonstração no relatório, *screenshots* da execução dos comandos `ls` e `poweroff` foram tirados e podem ser visualizados nas Figuras 6 e 7, respectivamente.

IV. Conclusão

Concluindo, nesse projeto obtivemos os primeiros contatos com o sistema operacional Minix 3. Realizamos configurações iniciais de instalação e setup do sistema em um hypervisor de máquina virtual (VirtualBox) e lidamos com alguns desafios durante esse processo. O processo de recompilação do kernel gerou problemas que relacionamos a algum tipo de incompatibilidade com o hardware de algumas máquinas utilizadas nessa etapa. Também tivemos a primeira oportunidade de conhecer a árvore de diretórios padrão do Minix 3, obtendo acesso aos arquivos de configuração do S.O, nos quais realizamos alterações e visualizamos o reflexo destas mudanças dentro do sistema e, com isso, podendo obter maior familiaridade com o funcionamento e com a estrutura geral do sistema operacional.

Com esse trabalho foi possível ver na prática conceitos que tínhamos apenas visto teoricamente nas aulas ministradas, como o conceito de processos, kernel, e até mesmo organização básica de sistemas operacionais. Tudo isso foi importante para solidificarmos nossos conhecimentos teóricos e práticos em relação ao tema.

O projeto foi realizado em partes, e cada parte foi feita por um ou mais integrantes. A equipe inteira colaborou com discussões gerais sobre o andamento do projeto e erros que encontrávamos pelo caminho. Como não conseguimos compilar o Minix em algumas máquinas, o Matheus ficou responsável por editar, compilar e realizar *screenshots* do S.O. após as modificações. O Daniel também conseguiu realizar as alterações do banner, mas infelizmente seu computador deixou de compilar o S.O., e após isso ele não conseguiu mais mexer no kernel. Para escrevermos o relatório, o André colaborou com o resumo, com a introdução, com a instalação e com a primeira parte da seção III. Além disso, o Matheus escreveu a parte do `exec` e o Daniel e o Guilherme complementaram o texto inteiro com novas informações de erros e com esta seção, a conclusão.

Apêndice A
Figuras

Figura 1: Minix 3.4 após instalção e configurações realizadas com sucesso.

Figura 2: Mensagem adicionada no primeiro banner exibido pelo sistema operacional durante a inicialização.

Figura 3: Mensagem adicionada no último banner exibido pelo sistema operacional durante a inicialização.

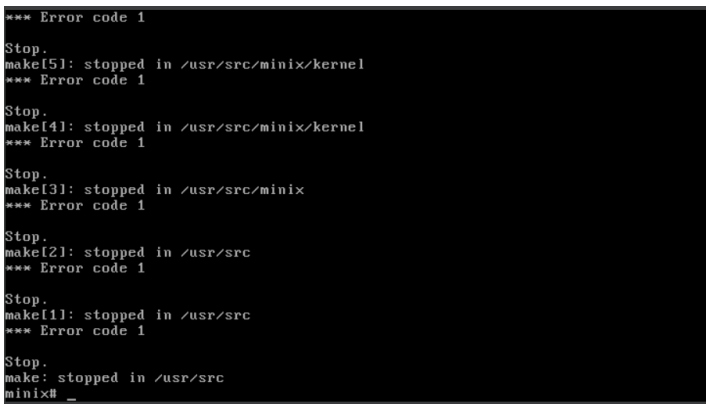

Figura 4: Threads parando durante o processo de recompilação do Kernel.

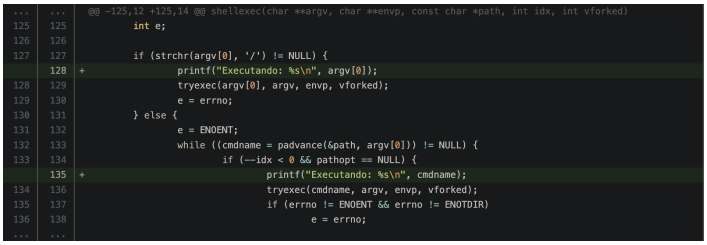

Figura 5: Modificações feitas no código para que os comandos executados no terminal tivessem seu caminho imprimido no terminal.

Figura 6: Exibição do caminho do comando `ls` durante sua execução.

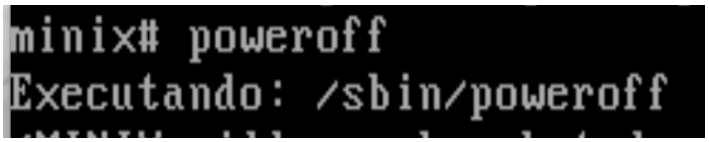

Figura 7: Exibição do caminho do comando `poweroff` durante sua execução.

REFERÊNCIAS

[1] Tanenbaum, Andrew S.; Woodhull, Albert S. (1987). Operating Systems: Design and Implementation. ISBN 9780131429383.
[2] ”Minix Releases”, url = https://web.archive.org/web/20120531025416/http://wiki.minix3.org/en/MinixReleases
[3] ”MINIX 3: a highly reliable, self-repairing operating system”, Jorrit N. Herder and Herbert Bos and Ben Gras and Philip Homburg and Andrew S. Tanenbaum, ACM SIGOPS Oper. Syst. Rev., 2006 , volume 40, pages 80-89. url = https://api.semanticscholar.org/CorpusID:30216714
[4] ”Minix Downloads”, url = https://wiki.minix3.org/doku.php?id=www:download:start
[5] Minix 3 developers guide, url = https://wiki.minix3.org/doku.php?id=developersguide:trackingcurrent
[6] “usersguide:runningonvirtualbox [Wiki]”. start [Wiki]. Consult. 2023-09-11. [Em linha]. Disponível: https://wiki.minix3.org/doku.php?id=usersguide:runningonvirtualbox