

# alpha

<ed/tech>

LINUX

Aula 01

<Módulo 07/>

## Linux

No ano de 1991, enquanto ainda era um estudante de Ciências da Computação na Universidade de Helsinki, Finlândia, Linus Torvalds tomou a decisão de embarcar em um projeto de desenvolvimento de um sistema operacional mais robusto do que o MINIX, motivado principalmente por interesses pessoais.

Com o intuito de compartilhar sua ideia e envolver outros colaboradores, ele postou uma mensagem em um grupo na Usenet. A resposta a esse apelo inicial resultou no lançamento da versão 0.02 do Kernel por Linus Torvalds ainda no mesmo ano. Determinado e dedicado ao seu projeto, Torvalds continuou aprimorando o sistema operacional ao longo dos anos, culminando no lançamento da versão 1.0 em 1994. Esse marco representou um passo significativo na evolução do que viria a ser conhecido como o sistema operacional **Linux**.

É importante destacar que o **Linux não constitui um sistema operacional integral**; em vez disso, é um **kernel**. Podemos entender o kernel como o núcleo central de um sistema operacional, responsável por gerenciar recursos e interações de hardware. Embora o Linux compartilhe semelhanças com o Unix, é crucial notar que eles têm origens distintas e foram desenvolvidos de maneiras diferentes. O Linux, assim como o núcleo de uma semente, é o cerne vital que, quando combinado com outros componentes, forma um sistema operacional completo. O Linux precisou ser combinado com o **projeto GNU** para formar um sistema operacional completo. Antes de descobirmos o que é projeto GNU, vamos ver um pouco mais sobre Kernel.

## Kernel

O **kernel** (ou núcleo, em português) é a parte central e fundamental de um sistema operacional. Ele atua como uma camada intermediária entre o hardware e o software, facilitando a comunicação e a gestão dos recursos do sistema. O kernel controla todas as principais funções do hardware, seja este um smartphone, um laptop, um servidor ou qualquer outro tipo de computador.

Citando suas principais características:

- **Gestão de Recursos:** O kernel gerencia os recursos do sistema, como processadores, memória, dispositivos de entrada/saída e armazenamento. Ele aloca recursos conforme necessário e garante que diferentes partes do software tenham acesso a esses recursos de maneira ordenada.
- **Comunicação Hardware-Software:** O kernel atua como uma interface entre o hardware e os programas em execução no sistema. Ele fornece abstrações de hardware, permitindo que os programas interajam com o hardware sem precisar entender detalhes específicos da arquitetura.
- **Execução de Tarefas do Sistema:** O kernel é responsável pela execução de tarefas essenciais do sistema, como a inicialização do sistema, o gerenciamento de processos e a manipulação de interrupções. Ele também controla o agendamento de processos, determinando quais processos têm acesso à CPU em um determinado momento.
- **Segurança e Proteção:** O kernel implementa políticas de segurança e proteção, garantindo que diferentes processos e partes do sistema operacional não interfiram indevidamente entre si. Ele controla o acesso a recursos críticos para evitar falhas e violações de segurança.
- **Drivers de Dispositivos:** O kernel inclui drivers de dispositivos, que são módulos de software responsáveis por permitir a comunicação entre o sistema operacional e dispositivos de hardware específicos, como placas de vídeo, impressoras, discos rígidos, entre outros.

## Introdução

No vasto universo da computação, um sistema operacional destaca-se como um ícone da liberdade, estabilidade e flexibilidade: o Linux. Desde sua concepção, este sistema operacional open source conquistou o coração de entusiastas e profissionais de TI ao redor do mundo. Seu surgimento não foi apenas um marco na história da computação, mas sim uma revolução que desafiou as normas estabelecidas. Nesta jornada, exploraremos as raízes do Linux, desvendando os eventos e mentes brilhantes por trás de sua criação, para entendermos como esse sistema operacional singular se tornou um pilar essencial no panorama tecnológico global.

## A origem do Linux

Em 1991, Linus Torvalds criou o Linux. A origem do nome desse sistema vem da combinação LINUS + UNIX. Para entender como o Linux foi criado, é necessário conhecer um pouco sobre outros sistemas.

### Multics

O sistema operacional **Multics (Multiplexed Information and Computing Service)** foi desenvolvido no início dos anos 1960, como um projeto que teve colaboração do Massachusetts Institute of Technology (MIT), General Electric (GE), Bell Labs e American Telephone and Telegraph (AT&T). Naquela época, Multics era de longe o sistema operacional mais robusto, permitindo que vários usuários compartilhassem recursos de um único computador, além de suportar várias atividades simultâneas, fornecendo segurança e escalabilidade.

### Unix

Após alguns anos, a empresa Bell Labs se retirou do projeto Multics. Ken Thompson, pesquisador do Multics e funcionário da Bell Labs, continuou seus estudos sobre sistemas. O objetivo era desenvolver um sistema menor, mais leve e flexível, tomando como base o Multics. Esse sistema menor ficou conhecido como **Unix**.

Em 1973, Dennis Ritchie, outro pesquisador da Bell Labs, criou a linguagem C e reescreveu o sistema Unix para a nova linguagem. A capacidade de ser facilmente portado para diferentes máquinas foi uma das razões pelas quais o Unix ganhou popularidade. É importante destacar também que o Unix é comercial.

### Minix

O **Minix** é um sistema operacional de código aberto, baseado em Unix, criado por Andrew S. Tanenbaum em 1987. Foi desenvolvido originalmente como um exemplo de sistema operacional para o livro "Operating Systems: Design and Implementation", escrito por Tanenbaum. Inicialmente, o Minix foi projetado para fins educacionais, como um sistema operacional simples e compreensível que poderia ser usado para ensinar princípios de sistemas operacionais. A sua simplicidade facilitava o entendimento dos conceitos fundamentais. O sistema teve seu código fonte liberado para estudantes, possibilitando que eles pudessem criar novos códigos ou entender a fundo o desenvolvimento do sistema.



# A origem do Linux

Em 1991, Linus Torvalds criou o Linux. A origem do nome desse sistema vem da combinação LINUS + UNIX. Para entender como o Linux foi criado, é necessário conhecer um pouco sobre outros sistemas.

## Projeto GNU

Como na década de 1980 quase todo software era proprietário, e os donos evitavam e proibiam a cooperação dos usuários, Richard Stallman deu início ao **Projeto GNU (GNU is Not Unix)**. O Projeto GNU, lançado em 1983, teve o objetivo de desenvolver um sistema operacional livre e de código aberto, semelhante ao Unix, mas sem depender de seu código-fonte. O GNU é uma tentativa de criar um sistema operacional completo e totalmente funcional que respeite as liberdades dos usuários de software. É importante ressaltar que a palavra "livre" se refere a liberdade e não ao preço do software.

Segundo o manifesto GNU, o usuário possui direito a quatro liberdades essenciais que são:

- **Executar:** O usuário tem a liberdade de utilizar o software como ele desejar, para qualquer propósito.
- **Estudar:** O usuário tem a liberdade de estudar como o software funciona e adaptá-lo para suas necessidades. Para que isso seja feito, o acesso ao código fonte é um pré-requisito.
- **Redistribuir:** O usuário tem a liberdade de redistribuir cópias do software para ajudar outros usuários.
- **Modificar:** O usuário tem a liberdade de redistribuir o projeto com modificações, para que a comunidade possa ser beneficiada com as mudanças.

Para um software ser considerado **software livre**, é necessário atender a estes requisitos. Vale ressaltar que software livre (free software) é diferente de software de código aberto (open source).

A Free Software Foundation (FSF) é uma entidade sem fins lucrativos estabelecida por Richard Stallman em 1985. A FSF desempenha um papel fundamental como patrocinadora do projeto GNU e administra as diversas licenças GNU, com o objetivo de advogar em prol do software livre. O kernel Linux opera sob a **licença GNU GPL**, concebida por Richard Stallman em 1989. A característica distintiva desta licença reside no fato de que qualquer trabalho derivado de um projeto que utiliza a licença GPL só pode ser distribuído sob essa mesma licença. Essa abordagem visa garantir a preservação dos princípios fundamentais do software livre.

## GNU/Linux

O **Projeto GNU** iniciou com a meta primordial de criar um sistema operacional composto exclusivamente por software livre, com compatibilidade Unix. Em 1984, Richard Stallman e outros programadores dedicados à causa começaram a desenvolver elementos fundamentais desse sistema. Até 1992, o sistema operacional GNU estava **quase pronto**, restando apenas o desenvolvimento do kernel. O projeto estava trabalhando em um kernel chamado Hurd. No entanto, um estudante finlandês optou por licenciar seu próprio kernel de forma livre e compatível com a GPL do GNU.

O kernel, que ficou conhecido como **Linux**, podia utilizar todas as ferramentas desenvolvidas pelo Projeto GNU. Essa colaboração marcou um momento significativo, permitindo, pela primeira vez desde o fechamento do código do Unix, a execução de um sistema operacional totalmente livre. Embora o termo "Linux" seja comumente utilizado, é importante notar que Linux refere-se ao kernel, não ao sistema operacional em sua totalidade. Como Linux não é um sistema operacional, mas sim um Kernel, algumas pessoas chamam o sistema de **GNU/Linux**, enquanto outros abreviaram apenas para Linux. Como Linus Torvalds disse, não importa o nome, o importante é que usem.

Citando algumas das ferramentas desenvolvidas pelo projeto GNU:

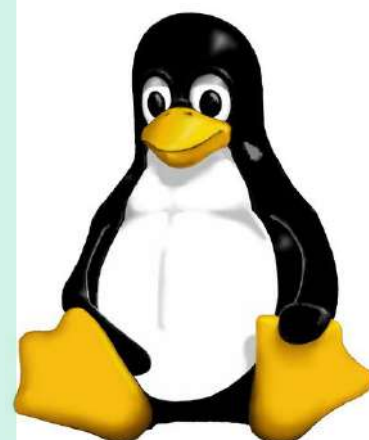
- **GCC (GNU Compiler Collection):** Uma coleção de compiladores para várias linguagens de programação, incluindo C, C++, Fortran, Ada e outros. O GCC é uma ferramenta fundamental para compilar código-fonte em código executável.
- **GDB (GNU Debugger):** Um depurador poderoso usado para analisar e corrigir bugs em programas. Ele oferece recursos como rastreamento de execução, inspeção de variáveis e análise de código.
- **Bash (Bourne Again SHell):** Um shell de linha de comando que fornece uma interface entre o usuário e o sistema operacional. O Bash é um interpretador de comandos poderoso e amplamente utilizado.
- **Emacs:** Um editor de texto altamente extensível e personalizável que oferece recursos avançados de edição. O Emacs é conhecido por suas capacidades de extensão e suporte a várias linguagens de programação.
- **Grep:** Um utilitário de linha de comando para pesquisa, permitindo procurar padrões em arquivos.
- **Coreutils:** Um conjunto de utilitários essenciais que realizam operações básicas no sistema de arquivos, como ls (listagem de diretórios), cp (cópia de arquivos), mv (movimentação de arquivos) e outros.



### Você sabia?

*Linus Torvalds teve a ideia de escolher uma mascote para o sistema Linux, reconhecendo sua importância visual e comercial. A inspiração surgiu quando ele **foi mordido por um pinguim** na Austrália, desenvolvendo um afeto por essas criaturas. Ao decidir adotar um pinguim como mascote, Linus lançou um concurso de logotipos para encontrar a representação perfeita.*

*Quanto ao nome **Tux**, há divergências sobre sua origem, com algumas sugestões como uma combinação de "Torvalds UniX" ou uma referência ao visual de smoking dos pinguins. De qualquer forma, Tux tornou-se sinônimo do carismático pinguim que simboliza o Linux.*



## Distribuições Linux

As **Distribuições Linux**, comumente chamadas de **distros**, são variantes específicas do sistema operacional Linux. Embora todas compartilhem o **mesmo kernel Linux**, cada distribuição pode ter diferentes conjuntos de software, gerenciadores de pacotes, ambientes de desktop, configurações padrão e filosofias de design.

Ao considerar qual distribuição Linux escolher, uma série de fatores importantes precisa ser avaliada. Questões como experiência do usuário, propósito específico (seja para iniciantes, desenvolvedores, desktops ou servidores), influenciam na decisão. Mesmo após responder a essas perguntas, muitas opções permanecem disponíveis, tornando a escolha, em última instância, uma decisão pessoal, alinhada às preferências individuais e aos requisitos específicos de cada usuário.

É possível dividir as distribuições de acordo com uso para Desktop ou Servidor.

**Desktop:** Distribuições voltadas para desktop são projetadas para fornecer uma experiência de usuário rica, amigável e visualmente atraente. Elas incluem ambientes de desktop completos, aplicativos gráficos e drivers para suportar uma ampla variedade de hardware.

**Servidor:** Distribuições voltadas para servidor são otimizadas para oferecer estabilidade, desempenho e segurança em ambientes de servidor. Elas podem não incluir ambientes gráficos por padrão, pois muitos servidores são gerenciados remotamente através da linha de comando.

Algumas vantagens ao empregar uma distribuição linux em servidores:

- **Custo:** A maioria das distribuições Linux é de código aberto e gratuita.
- **Estabilidade e Confiabilidade:** O Linux é conhecido por sua estabilidade e confiabilidade. Muitas distribuições são projetadas para operar 24/7 sem interrupções, o que é essencial para servidores web que precisam estar disponíveis continuamente.
- **Desempenho:** altamente eficiente em termos de desempenho. Sua arquitetura modular e capacidade de ser ajustado para atender a cargas específicas de trabalho permitem um desempenho robusto, especialmente em ambientes de servidor web de alto tráfego.
- **Segurança:** A arquitetura de segurança do Linux, juntamente com uma comunidade ativa de desenvolvedores, contribui para tornar os servidores Linux menos suscetíveis a ameaças de segurança. Além disso, as atualizações regulares e o gerenciamento de pacotes facilitam a manutenção da segurança do sistema.
- **Software Open Source:** Muitas das principais ferramentas e tecnologias usadas em servidores web, como o Apache HTTP Server, NGINX e MySQL, são de código aberto e têm uma longa história de desenvolvimento ativo na comunidade Linux.





## Distribuições para desktop

Algumas das principais distribuições para desktop:

**Ubuntu:** Uma das distribuições mais populares e amigáveis para iniciantes, o Ubuntu é conhecido por sua facilidade de uso e vasta comunidade de suporte.

**Linux Mint:** Baseado no Ubuntu, o Linux Mint oferece uma experiência de desktop elegante e intuitiva, além de ser reconhecido por sua estabilidade.

**Fedora Workstation:** Focada em oferecer software mais recente e inovações tecnológicas, a Fedora é uma escolha para usuários que buscam recursos atualizados.

**Debian:** Conhecido por sua estabilidade e compromisso com o software livre, o Debian é a base para muitas outras distribuições, incluindo o Ubuntu.

## Distribuições para servidor

Algumas das principais distribuições para servidores:

**Red Hat Enterprise Linux (RHEL):** é uma distribuição conhecida por seu suporte corporativo e estabilidade. Empresas como IBM, Amazon Web Services (AWS) e Google Cloud Platform (GCP) usam o RHEL em seus serviços.

**Ubuntu Server:** A versão do Ubuntu projetada para servidores é amplamente utilizada, especialmente em ambientes de nuvem. Empresas como Netflix, PayPal e Wikipedia usam Ubuntu Server.

**SUSE Linux Enterprise Server (SLES):** Oferece uma solução empresarial com foco em confiabilidade e suporte. Empresas como BMW, Bosch e SAP usam o SLES.

**Debian:** Muitas empresas, incluindo a DreamHost e o CERN, utilizam o Debian em servidores devido à sua estabilidade e ao compromisso com o software livre.

**CentOS:** Uma versão gratuita e de código aberto do RHEL.

**Fedora Server:** Embora o Fedora seja mais conhecido como uma distribuição de desktop, a versão Server é usada em alguns ambientes empresariais. Um exemplo de empresa que a utiliza é Lenovo.

**Oracle Linux:** é empacotada e distribuída gratuitamente pela Oracle. Oferece acesso a algumas das inovações mais avançadas do Linux, como Ksplice (extensão do Kernel Linux que permite que patches de segurança sejam aplicados a um kernel em execução sem a necessidade de reinicializações), e DTrace (estrutura de rastreamento dinâmico)

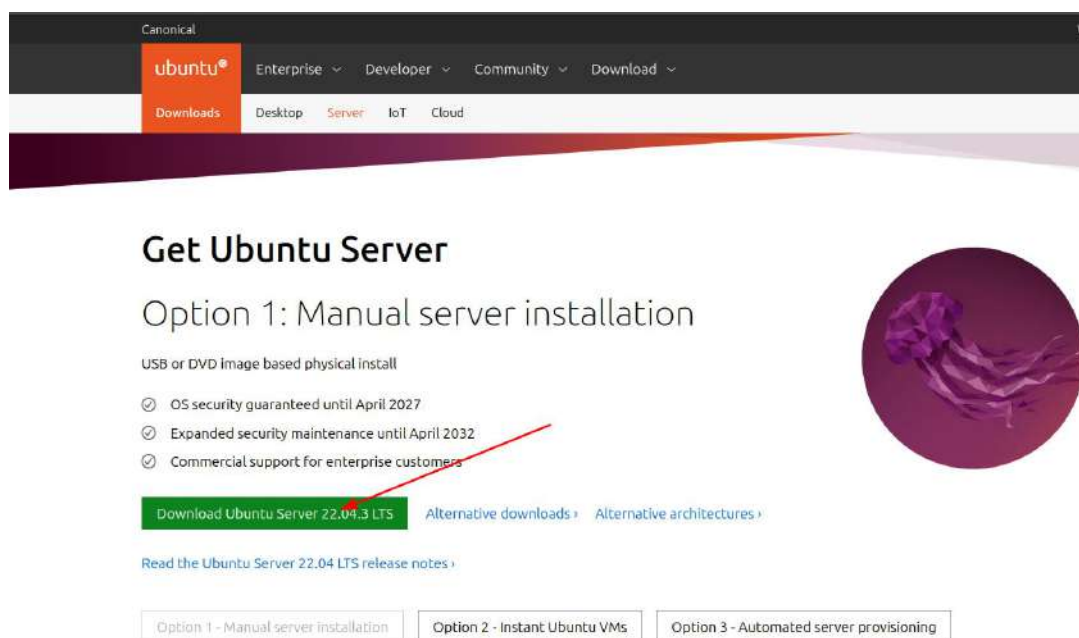
**Arch Linux:** Embora menos comum em ambientes empresariais tradicionais, o Arch Linux é conhecido pela sua simplicidade e flexibilidade, sendo usado por alguns para servidores personalizados.

## Instalação Linux

O objetivo é explicar as principais formas de instalar uma distribuição Linux em uma máquina e ensinar o passo a passo para fazer a instalação de uma máquina virtual que contenha Linux. A versão LTS 24.02 do Linux é a última, mas grande parte dos servidores estão com a versão LTS 22.04. Se é a sua primeira vez lendo esse material, recomenda-se já iniciar o download de uma imagem de disco, isto é, baixar uma ".iso" de Ubuntu server no endereço a seguir:

<https://ubuntu.com/download/server>

Clique na primeira opção (instalação manual) e, em seguida, clicar no botão verde, conforme a imagem abaixo:



Dessa forma, você ganha tempo. Enquanto o download ocorre (2GB), você pode ir estudando este material. Mais à frente precisaremos desse arquivo.

## Formas de instalação

Existem várias abordagens para utilizar Linux no seu dispositivo. A seguir, destacamos as principais opções.

### Máquina virtual

A utilização de uma **Máquina Virtual (Virtual Machine - VM)** implica na criação de um ambiente isolado no sistema, onde é possível instalar e executar uma distribuição Linux como se fosse um programa. Dentre as principais vantagens, destacam-se a facilidade de instalação e remoção, bem como o isolamento do sistema principal. A principal desvantagem reside na possibilidade de aplicação de alguma sobrecarga de desempenho devido à virtualização.



Para mitigar essa sobrecarga dentro da Máquina Virtual, sugere-se a utilizar Ubuntu Server em vez do Ubuntu Desktop, uma vez que o primeiro não possui interface gráfica. Mesmo no caso de possuir um dispositivo com uma distro Linux já instalada, é recomendável instalar uma Máquina Virtual, uma vez que certas atividades aqui no Projeto AlphaEdtech são mais eficientes em um ambiente isolado.

A título de exemplo, considere a situação em que um servidor é instalado utilizando o Nginx. Posteriormente, quando for necessário instalar outro servidor, como o Apache, é possível simplesmente criar uma nova máquina virtual e implementar o servidor dentro dela, de forma isolada. Essa abordagem evita a necessidade de remover ou substituir todas as configurações anteriormente empregadas no Nginx pelas configurações do Apache.

Na imagem abaixo, nota-se duas máquinas virtuais desligadas.



## WSL

O **Windows Subsystem for Linux (WSL)** é uma funcionalidade do sistema operacional Windows que permite a execução de distros Linux diretamente no ambiente Windows. Essa integração proporciona uma ponte entre o mundo Windows e o Linux, permitindo que desenvolvedores e usuários aproveitem as vantagens de ambos os sistemas operacionais.

Existem duas versões do WSL:

- WSL 1: Lançado em 2016, essa versão não fazia uso de uma máquina virtual para executar o sistema Linux. Em vez disso, ela traduzia as "chamadas de sistema" Linux (quando um programa no WSL chama uma função do kernel do Linux) para chamadas de sistema existentes no Windows. Era uma "camada de tradução" entre funções do kernel do Linux e funções do kernel do Windows.
- WSL 2: Lançado em 2019, essa versão executa um Linux real (uma distribuição com kernel Linux) dentro de uma máquina virtual leve (mais leve do que executar uma distro Linux dentro do Virtual Box, por exemplo)

Como o WSL 2 tem um kernel Linux completo em vez de fazer traduções entre chamadas de sistema (como no WSL 1), o WSL 2 dá compatibilidade para mais programas Linux do que o WSL 1.

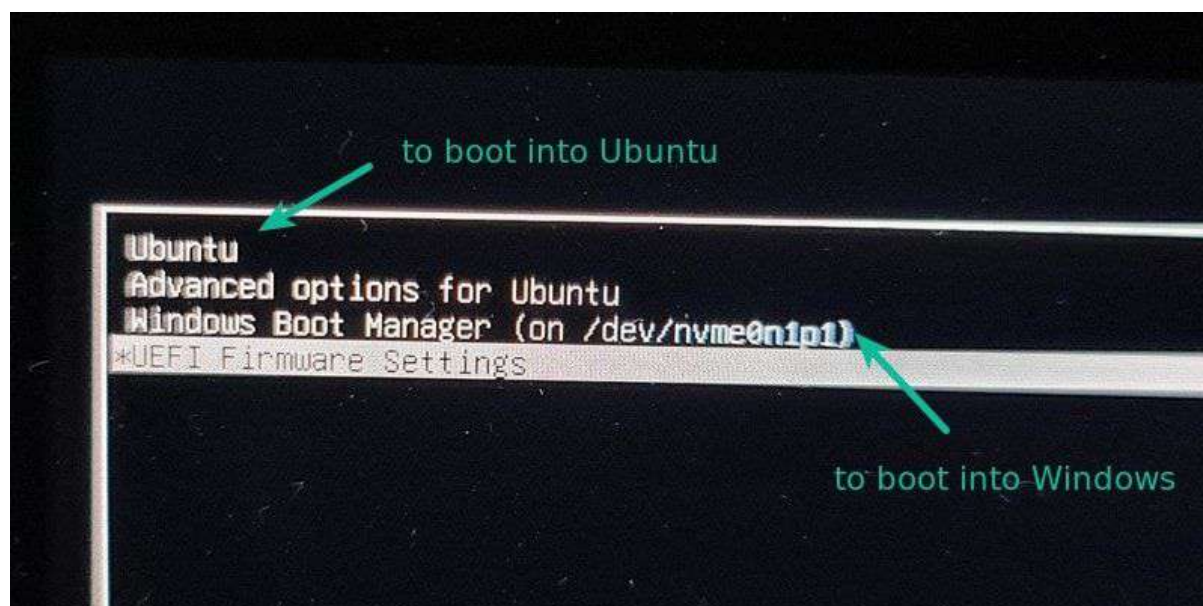
O WSL requer menos recursos (CPU, memória e armazenamento) do que uma máquina virtual completa. Também permite que você execute ferramentas e aplicativos de linha de comando do Linux junto com a linha de comando do Windows, aplicativos de desktop e de loja, e acesse seus arquivos do Windows a partir do Linux. Isso permite que você use aplicativos do Windows e ferramentas de linha de comando do Linux no mesmo conjunto de arquivos, se desejar.

## Dual boot

Trata-se de uma abordagem em que um dispositivo é configurado para inicializar dois sistemas operacionais diferentes, permitindo que o usuário escolha qual sistema operacional usar no momento da inicialização. Portanto, você pode optar por instalar uma distribuição Linux em um dispositivo com outro sistema operacional, como o Windows. Durante o processo de inicialização, um menu é exibido, permitindo a seleção do sistema desejado para aquela sessão.

A principal vantagem dessa abordagem reside na utilização plena do desempenho do sistema, sem os impactos associados à virtualização. É crucial realizar o particionamento adequado do disco, reservando uma porção significativa de memória para cada sistema operacional. Vale ressaltar que o uso de containers Docker pode consumir considerável quantidade de memória física, especialmente devido ao constante download de imagens Docker, embora este seja um tema que faz parte de outra trilha.

A imagem abaixo mostra a inicialização de um dispositivo com Windows e Ubuntu.



Na próxima seção, vamos aprender como instalar uma máquina virtual com Ubuntu Server.

## Instalação de uma VM

Pesquisando pela internet, você encontrará diversos tutoriais de como instalar uma distro Linux numa **Máquina Virtual (virtual Machine - VM)**. O passo a passo que mostraremos aqui é uma adaptação/transcrição do tutorial disponível no link a seguir:

<https://hibbard.eu/install-ubuntu-virtual-box/>

Faça o download da **Oracle VM Virtual Box** de acordo com o seu sistema operacional, a partir desse link:

<https://www.virtualbox.org/wiki/Downloads>



# VirtualBox

- About
- Screenshots
- Downloads
- Documentation
  - End-user docs
  - Technical docs
- Contribute
- Community

## Download VirtualBox

Here you will find links to VirtualBox binaries and its source code.

### VirtualBox binaries

By downloading, you agree to the terms and conditions of the respective license.

### VirtualBox 7.0.14 platform packages

- Windows hosts
- macOS / Intel hosts
- Linux distributions
- Solaris hosts
- Solaris 11 IPS hosts

The binaries are released under the terms of the GPL version 3.

See the [changelog](#) for what has changed.

Considerando que você tenha Windows ou MacOS, basta fazer o download e instalar, sem mistério. Caso seu computador pessoal seja Linux, teremos alguns passos a mais. Por exemplo, o sistema operacional do meu computador é uma distribuição Ubuntu 22.04. Portanto, cliquei em Linux distributions.



# VirtualBox

- About
- Screenshots
- Downloads
- Documentation
  - End-user docs
  - Technical docs
- Contribute
- Community

## Download VirtualBox for Linux Hosts

**Note:** The package architecture has to match the Linux kernel architecture, that is, if you are running a 64-bit kernel, install the appropriate AMD64 package (it does not matter if you have an Intel or an AMD CPU). Mixed installations (e.g. Debian/Lenny ships an AMD64 kernel with 32-bit packages) are not supported. To install VirtualBox anyway you need to setup a 64-bit chroot environment.

The VirtualBox base package binaries are released under the terms of the GPL version 2.

Please choose the appropriate package for your Linux distribution.

### VirtualBox 7.0.14 for Linux

- Oracle Linux 9 / Red Hat Enterprise Linux 9
- Oracle Linux 8 / Red Hat Enterprise Linux 8
- Oracle Linux 7 / Red Hat Enterprise Linux 7 / CentOS 7
- Ubuntu 22.04**
- Ubuntu 20.04
- Ubuntu 18.04 / 18.10 / 19.04
- Debian 12
- Debian 11
- Debian 10
- openSUSE 15.3 / 15.4 / 15.5
- Fedora 36 / 37 / 38 / 39
- Fedora 35
- All distributions (built on EL6 and therefore not requiring recent system libraries)



O passo anterior dá início ao download de um **arquivo .deb**, que é uma extensão usada por distribuições **Debian** ou derivadas dela (o Ubuntu é derivado da distro Debian), e representam pacote de softwares que permite instalar e configurar um programa, nesse caso a Oracle VM VirtualBox.

Com o download finalizado, podemos ir até o diretório Downloads clicar com o botão direito em algum local da pasta e escolher **Open in Terminal**, com o objetivo de abrir o terminal no local onde está o arquivo baixado.

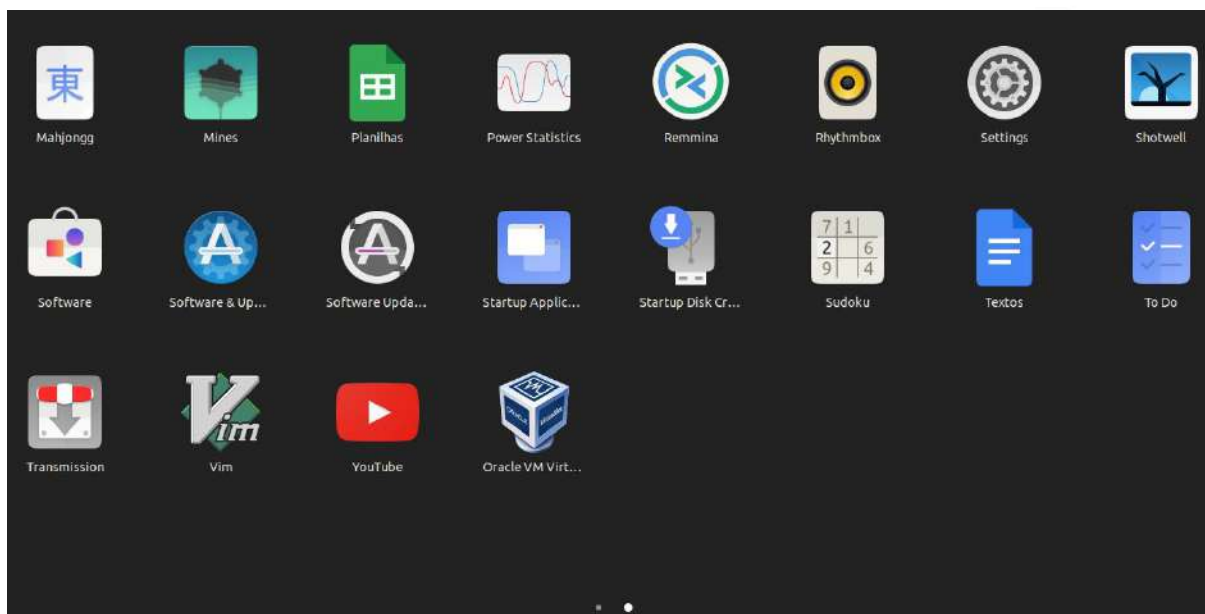
Em seguida, faça a instalação através desse comando:

**sudo apt install ./nome-do-arquivo.deb**



```
letonio.silva@BRRIOLN043879: ~/Downloads
letonio.silva@BRRIOLN043879:~/Downloads $ sudo apt install ./virtualbox-7.0_7.0.14-161095~Ubuntu~jammy_amd64.deb
```

Após clicar no menu no canto inferior esquerdo (Show Applications), podemos clicar no ícone da Oracle VM VirtualBox.



A tela inicial da Oracle VM VirtualBox é assim:

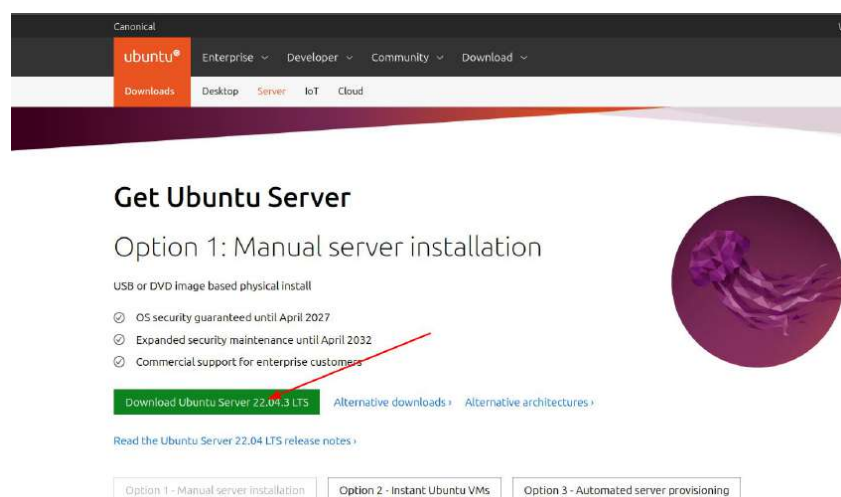


Vamos mantê-la aberta, pois precisamos ter um **arquivo .iso**, isto é, uma imagem de disco que representa o sistema operacional que desejamos instalar.

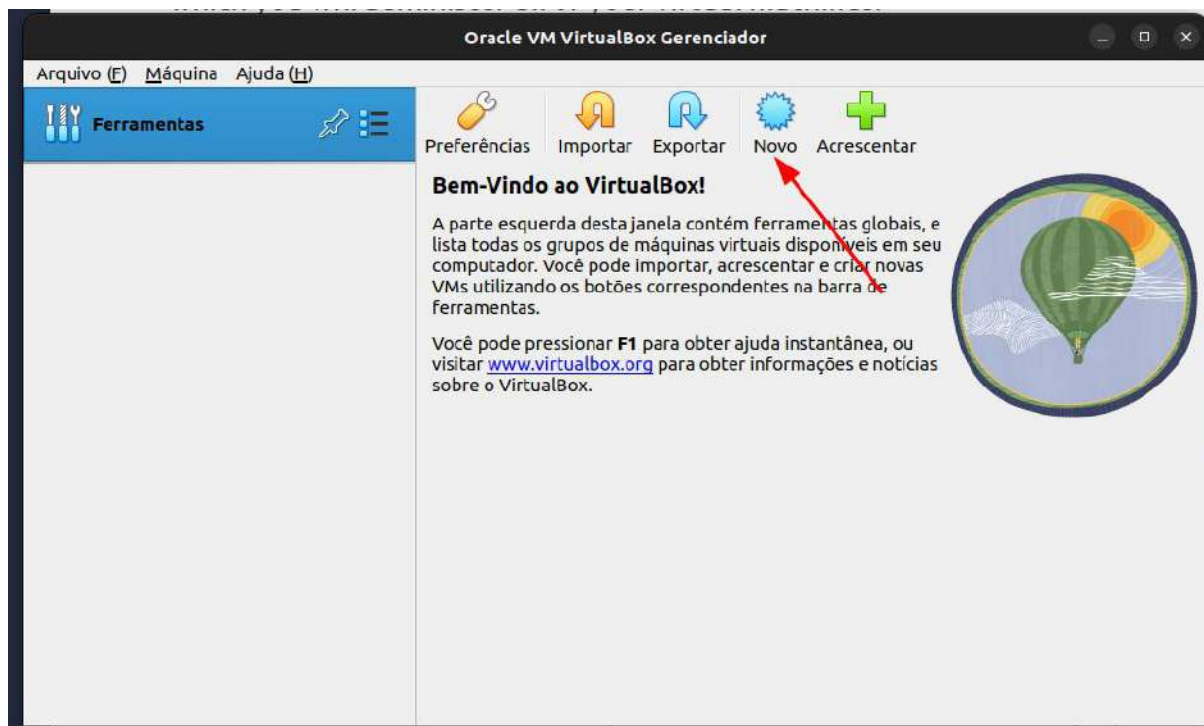
Caso ainda não o tenha feito, o próximo passo é baixar uma imagem “.iso” do Ubuntu Server 22.04, através desse site:

<https://ubuntu.com/download/server>

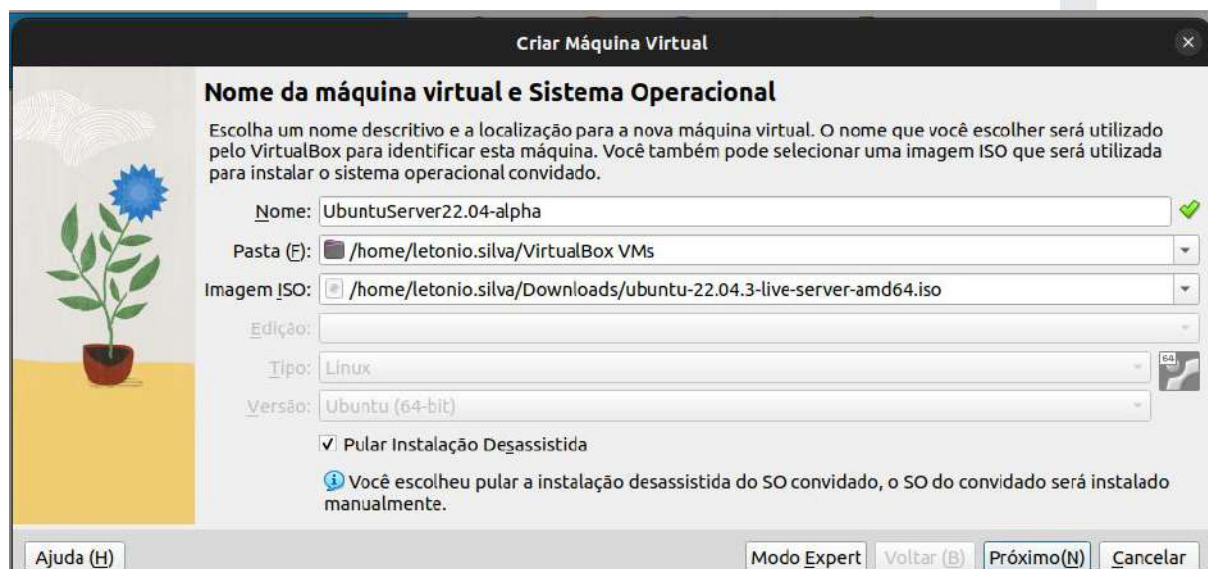
Escolha a primeira opção (Manual Server Installation). Em seguida, clique no botão verde de download, exatamente como na imagem abaixo:



Com o download completo, podemos inicializar nossa primeira máquina virtual, clique no ícone de Novo (New), conforme ilustrado na imagem a seguir:

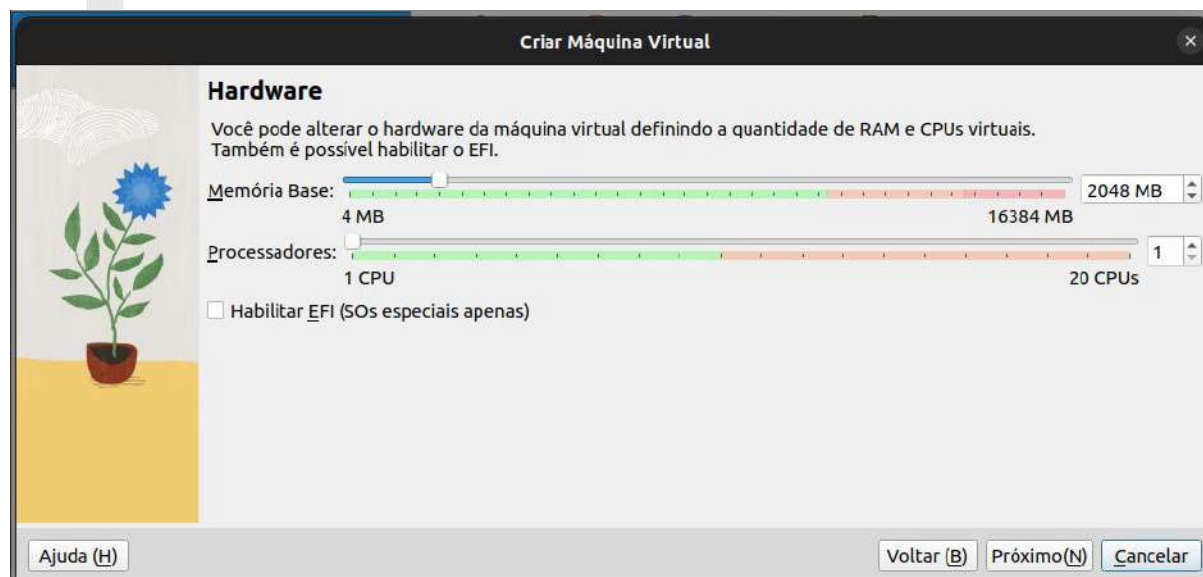


Dê um nome para sua Máquina Virtual (Virtual Machine - VM). O gerenciador vai detectar o tipo de sistema operacional da sua máquina. No input Imagem ISO procure pelo arquivo ISO que você baixou. Além disso, marque a opção **Pular Instalação Desassistida**, desse modo poderemos instalar o novo SO de forma manual.

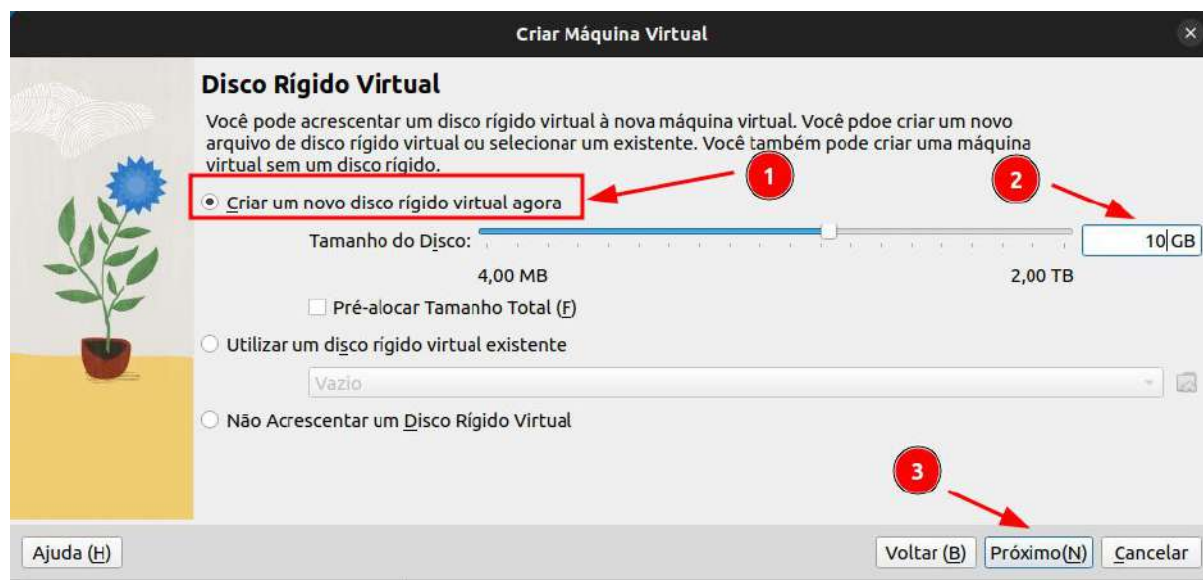




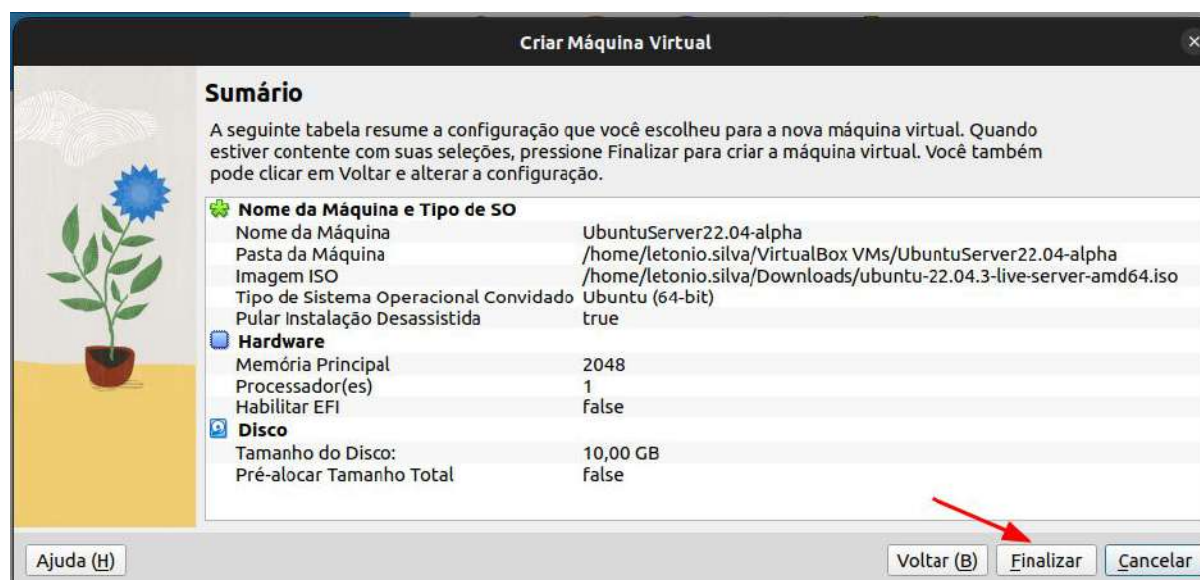
Clique em Próximo (Next). Você deve selecionar a quantidade de memória (RAM) em megabytes a ser alocada para a máquina virtual. Por enquanto, 2 GB (2.048 megabytes) já é mais que suficiente para as operações que vão ser realizadas. Além disso, caso no futuro você sinta necessidade de aumentar esse valor, lembre-se que você pode criar/remover máquinas virtuais.



Clique em Próximo (Next). A etapa seguinte é adicionar um disco rígido virtual à nova máquina. Escolha a opção **Criar um novo disco rígido virtual agora (create a virtual hard disk now)**. Defina a quantidade que deseja destinar (10GB é suficiente) e pressione o botão Criar (Create).

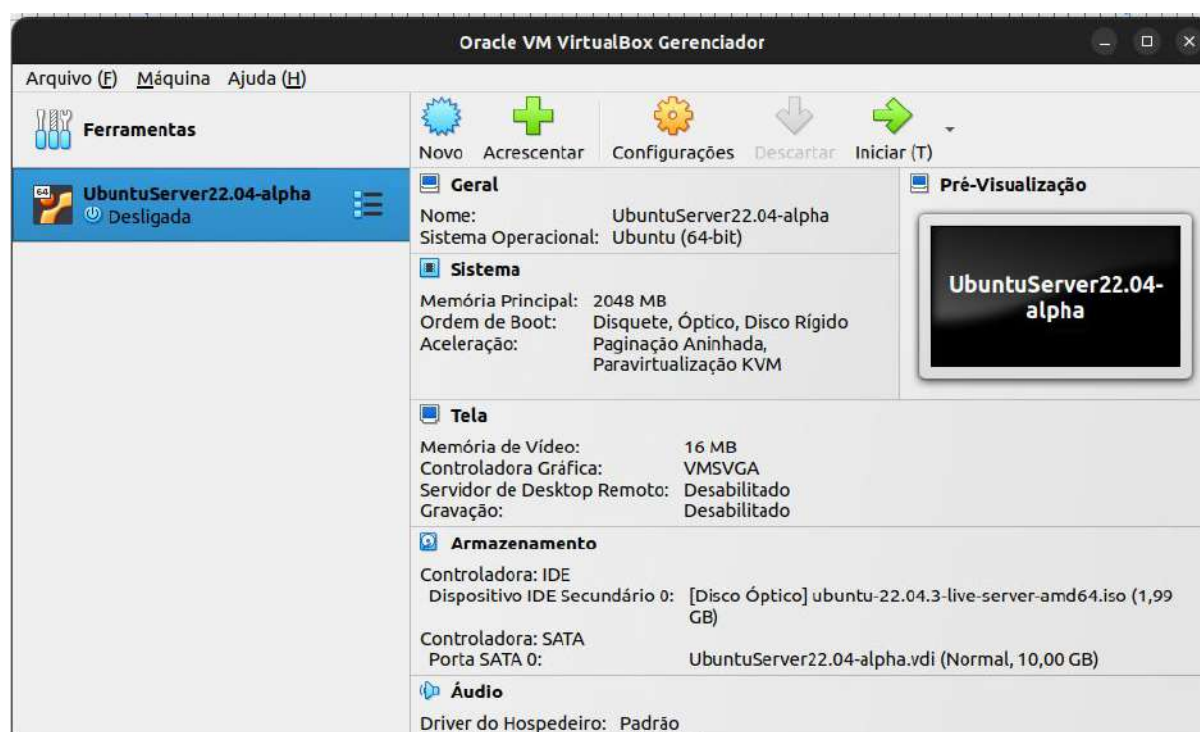


A próxima tela mostra um resumo das opções escolhidas. Clique em Finalizar.



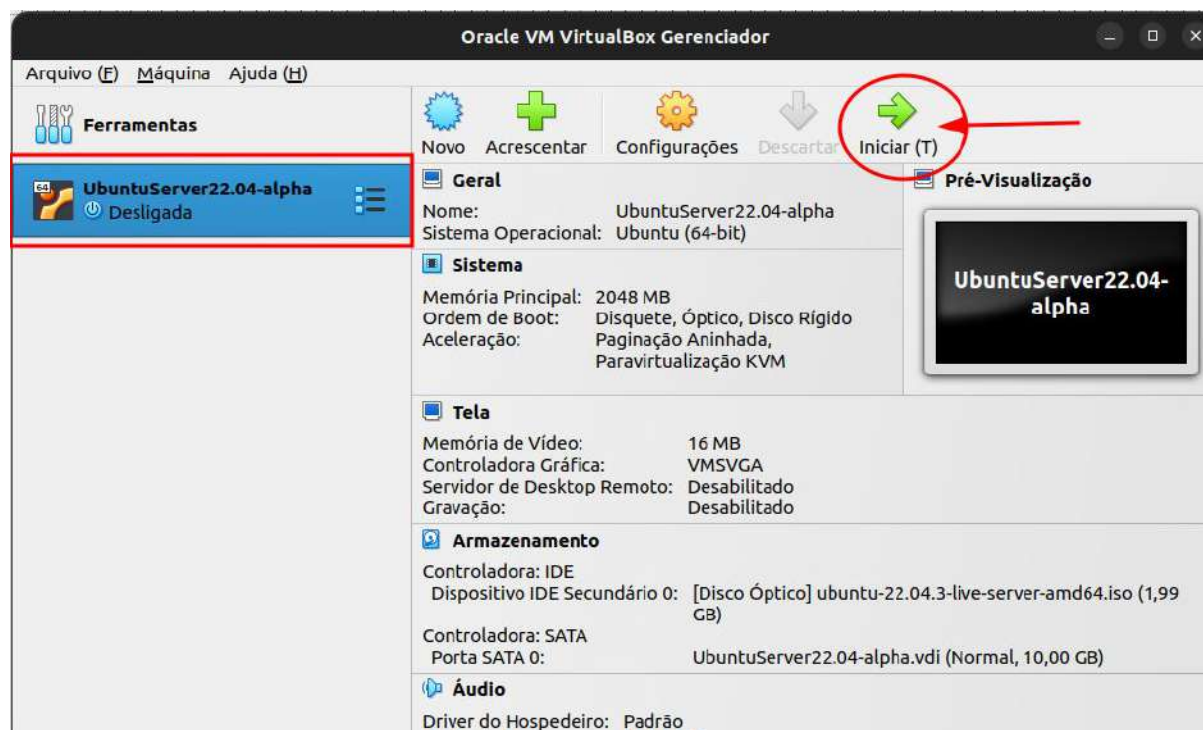
Na próxima tela, pergunta-se: o novo disco rígido virtual deve crescer à medida que é usado (alocado dinamicamente) ou se deve ser criado em seu tamanho máximo? Certifique-se de que Alocado dinamicamente (Dynamically allocated) esteja selecionado e clique em Avançar.

O disco rígido agora deve ser criado e você deve voltar ao VirtualBox Manager. Você verá sua máquina virtual recém-criada listada à esquerda.

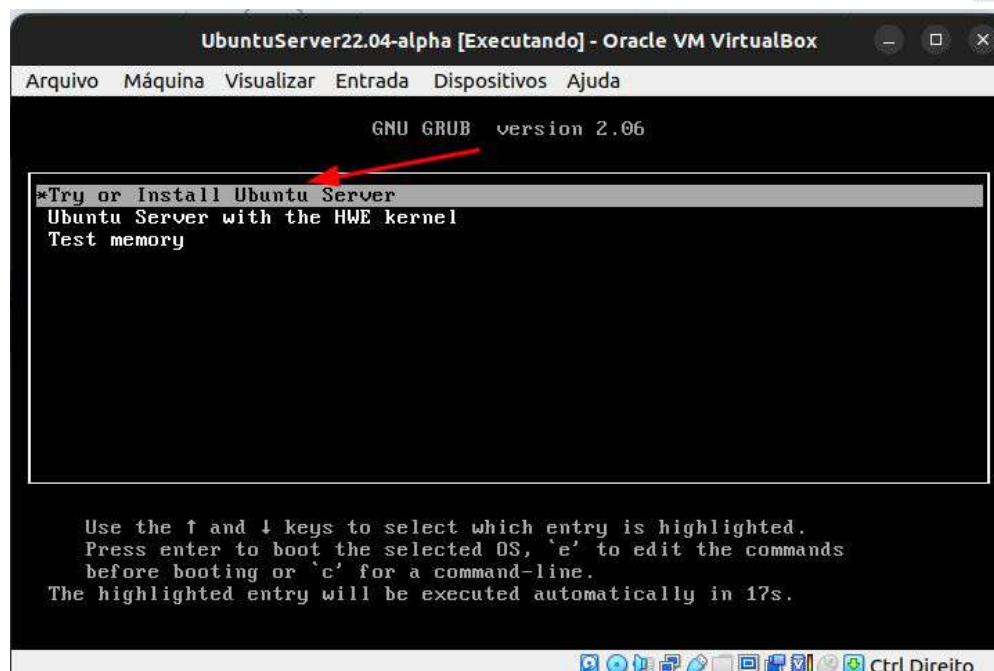


## Instalação do Ubuntu Server na VM

Certifique-se de que sua máquina virtual esteja selecionada e pressione **Iniciar (Start)**. O VirtualBox Manager solicitará que você selecione um arquivo de disco virtual ou uma unidade óptica física para iniciar a máquina virtual. Clique na pasta com a seta para cima no lado direito da caixa de diálogo, selecione o arquivo ISO que você baixou anteriormente e pressione Iniciar (Start).



Na tela seguinte, pressione Enter (mantenha a opção Try or Install Ubuntu Server).

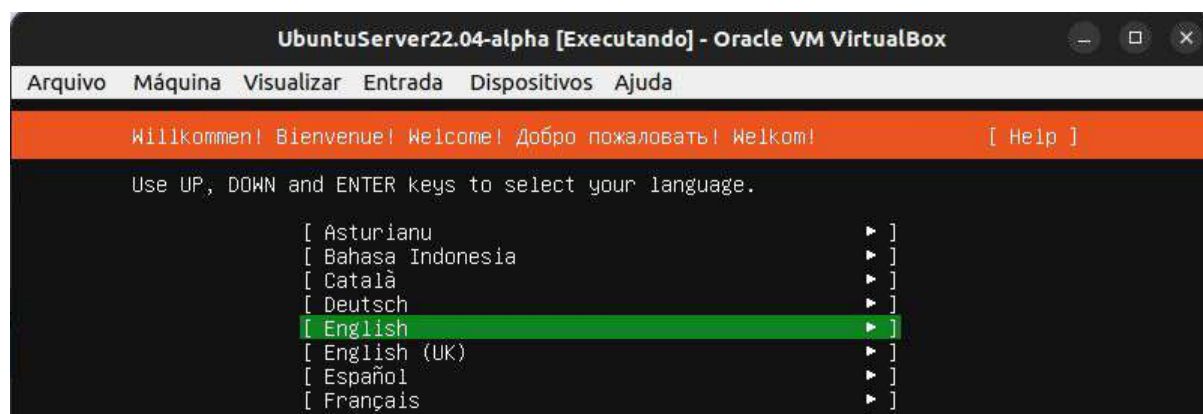


O processo de instalação do Ubuntu começará agora.



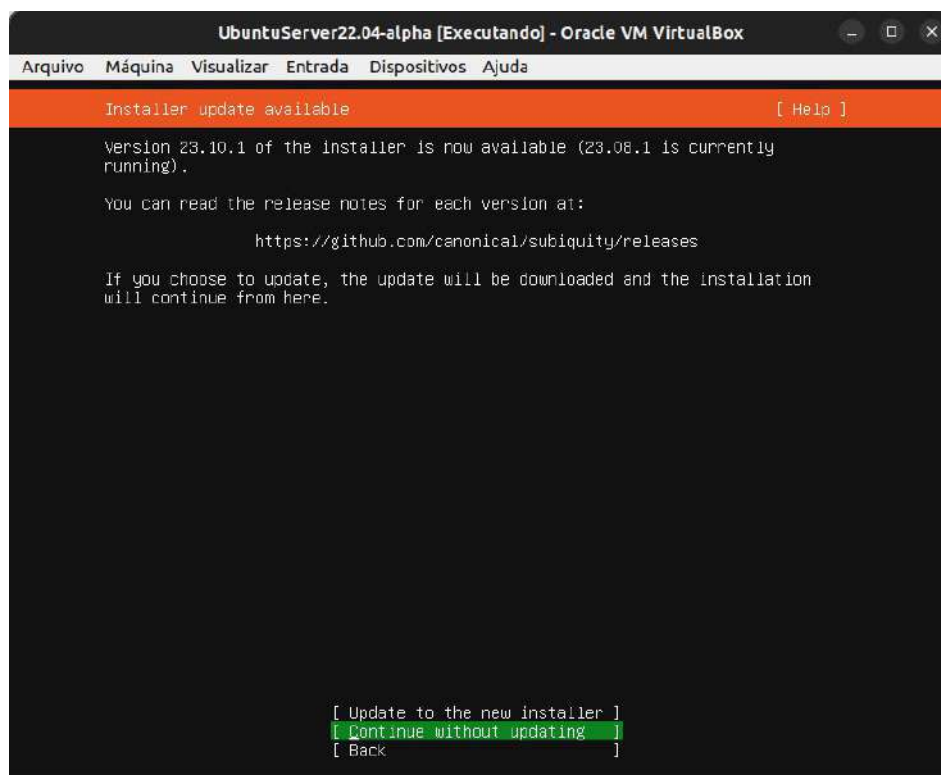
## A tela de boas-vindas

Aqui você deve selecionar seu idioma preferido. Optei por **English**.



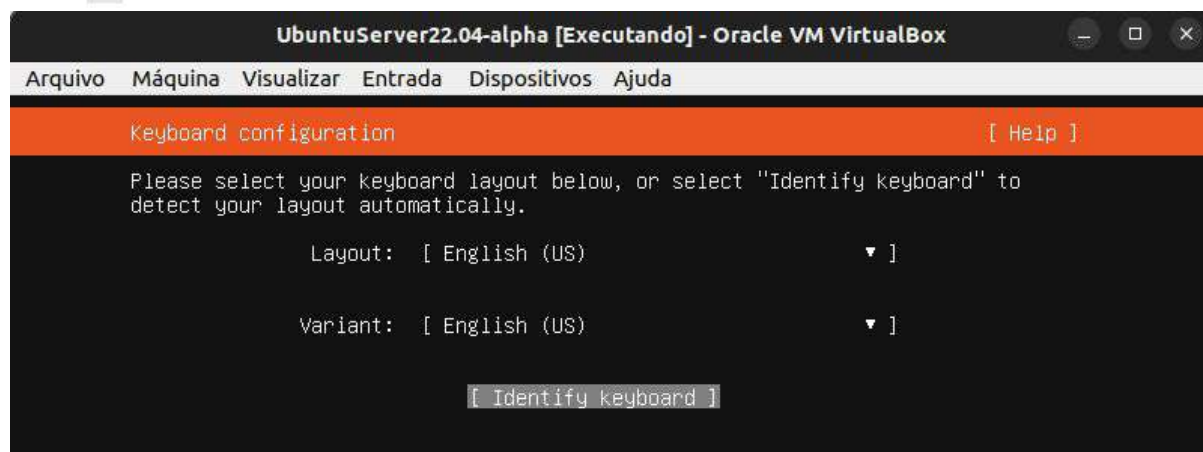
## Atualizar o instalador

Se uma atualização do instalador estiver disponível, você poderá escolher instalá-la ou ignorá-la. Optei por ignorar (Continue without updating). Caso escolha atualizar, o download da atualização começará e o instalador será reiniciado.

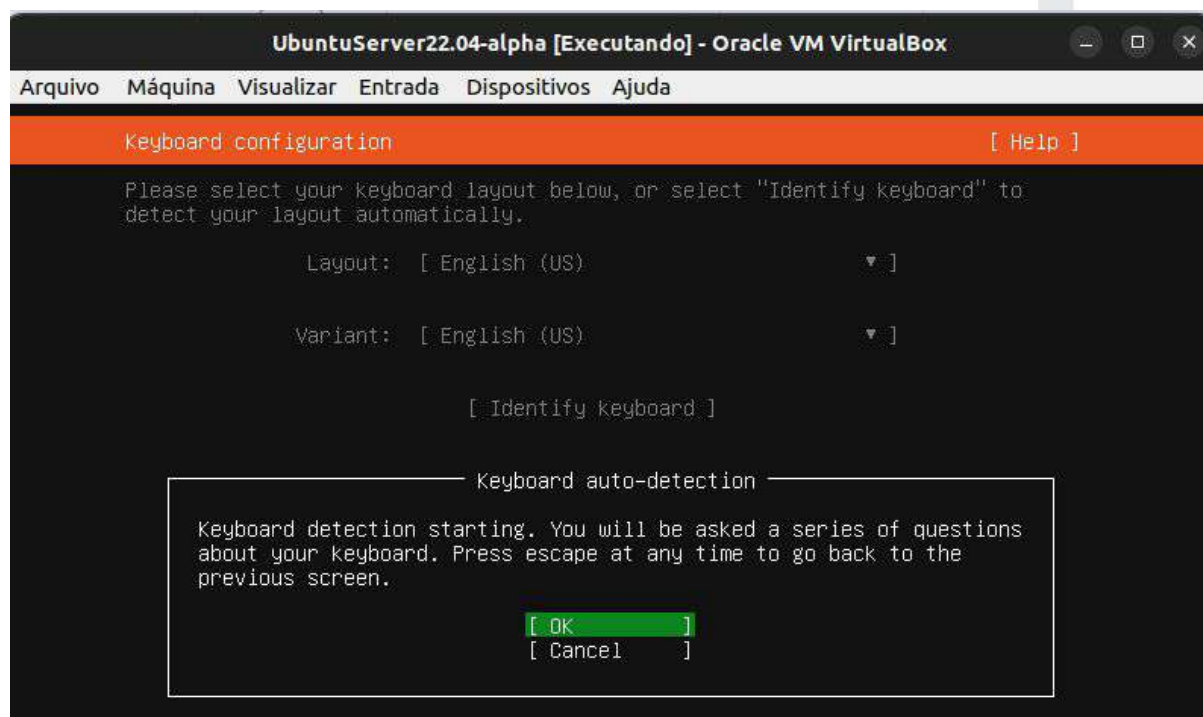


## Configuração do teclado

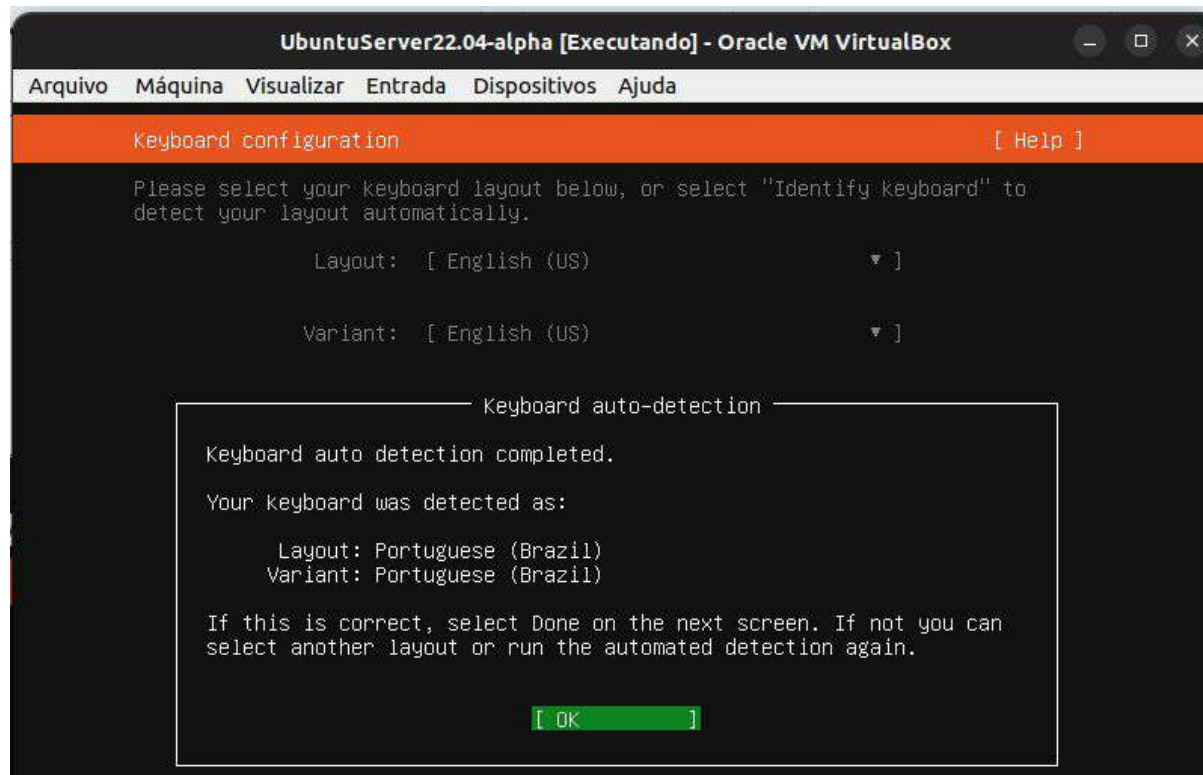
Aqui você deve selecionar um layout de teclado. Você pode pedir para o Ubuntu detectar o seu teclado (Identify keyboard). Nesse caso, ele fará algumas perguntas simples, onde você apertará as teclas correspondentes ao que está sendo mostrado.



Na imagem abaixo, o processo de identificação de teclado está prestes a começar.

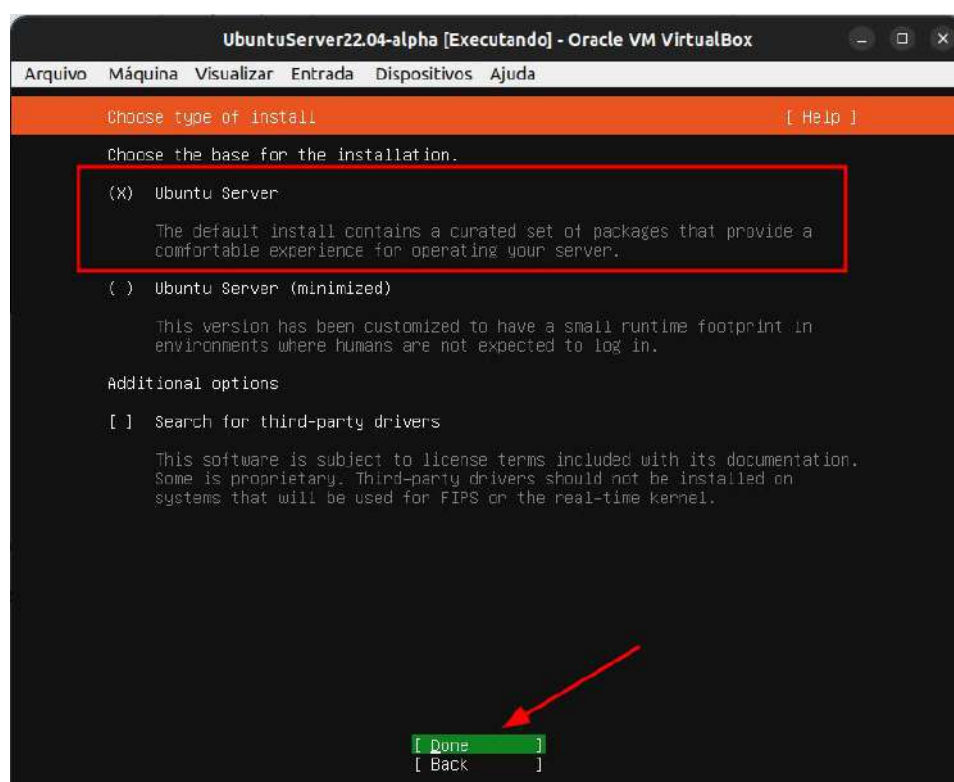


Após algumas perguntas, ele identificou que o meu teclado é PT-BR.



## Tipo de instalação

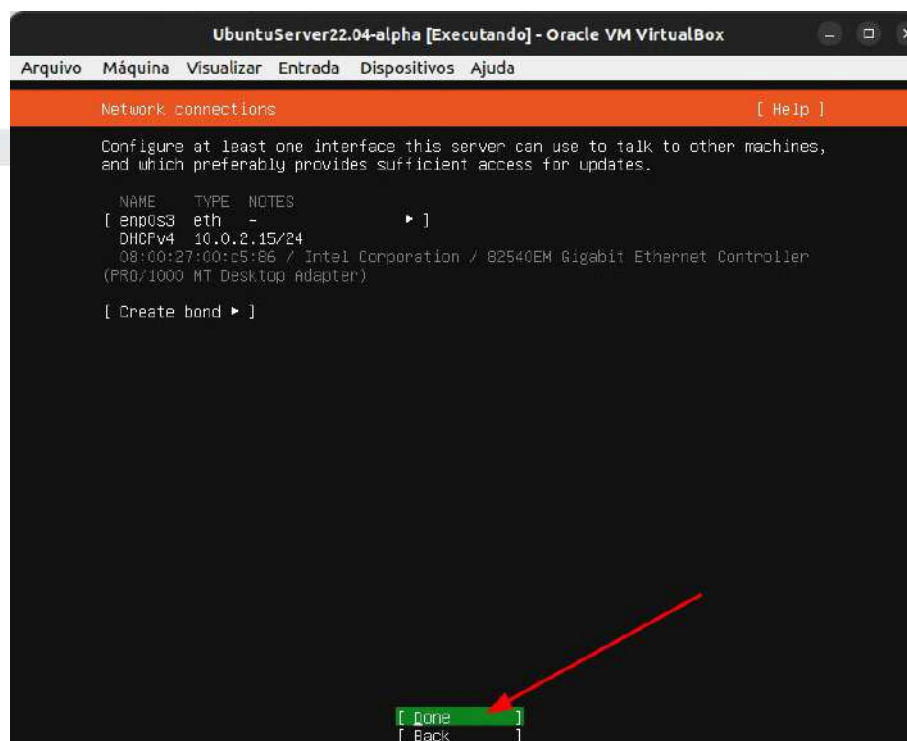
Em seguida, escolha a instalação padrão que contém um conjunto selecionado de pacotes.





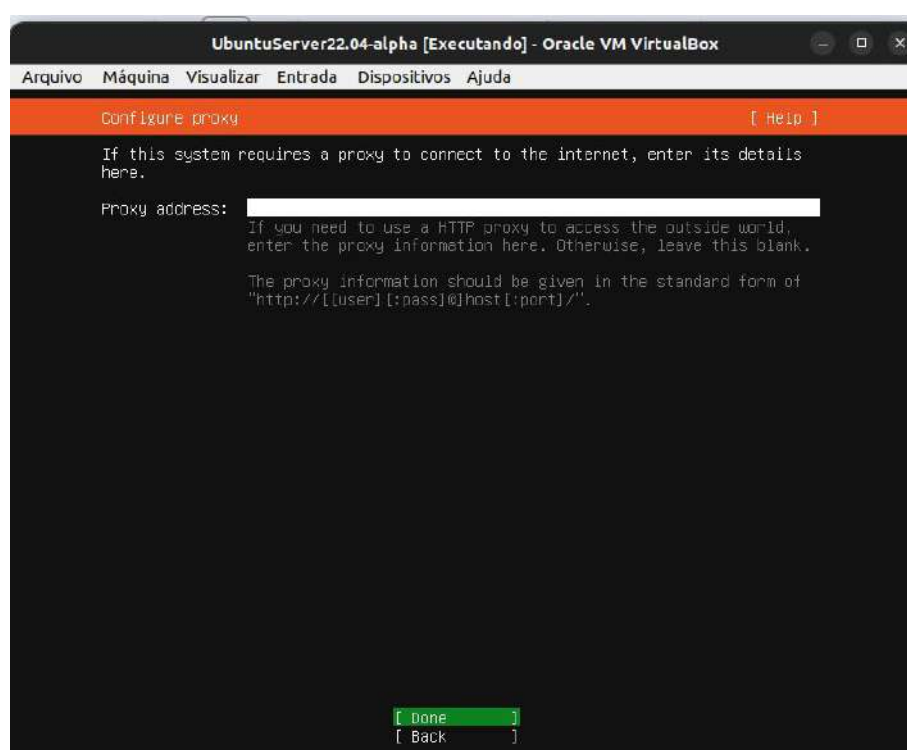
## Conexões de rede

Aqui o Ubuntu tentará configurar a interface de rede padrão. Normalmente, você pode aceitar o padrão e selecionar **Done**.



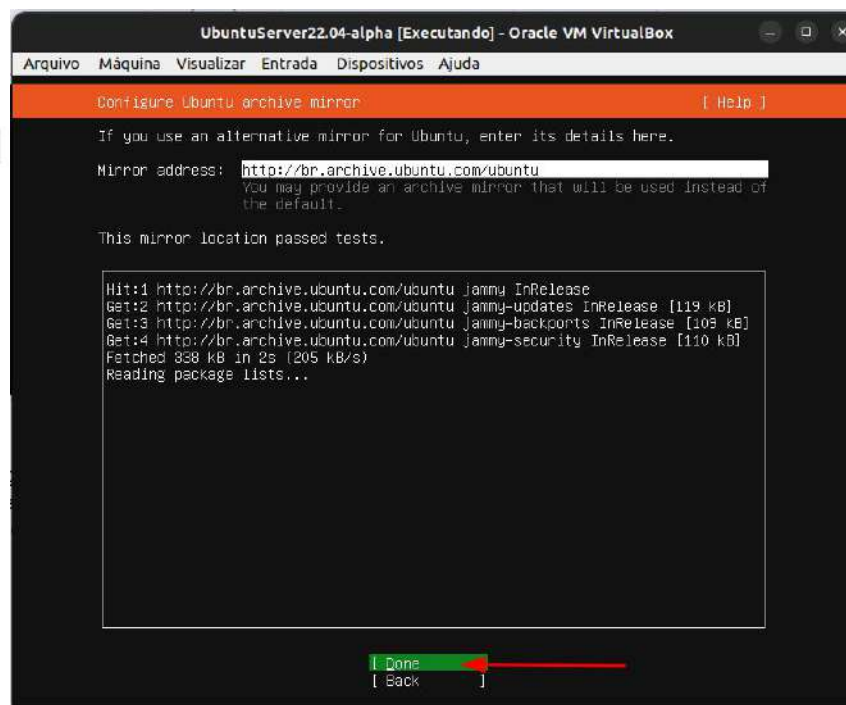
## Configurar proxy

Se o seu sistema exigir um proxy para se conectar à Internet (o meu não exige), insira seus detalhes na próxima caixa de diálogo. No nosso caso, vamos deixar em branco e apenas clicar em Concluído (Done).



## Configurar o espelho de arquivo do Ubuntu

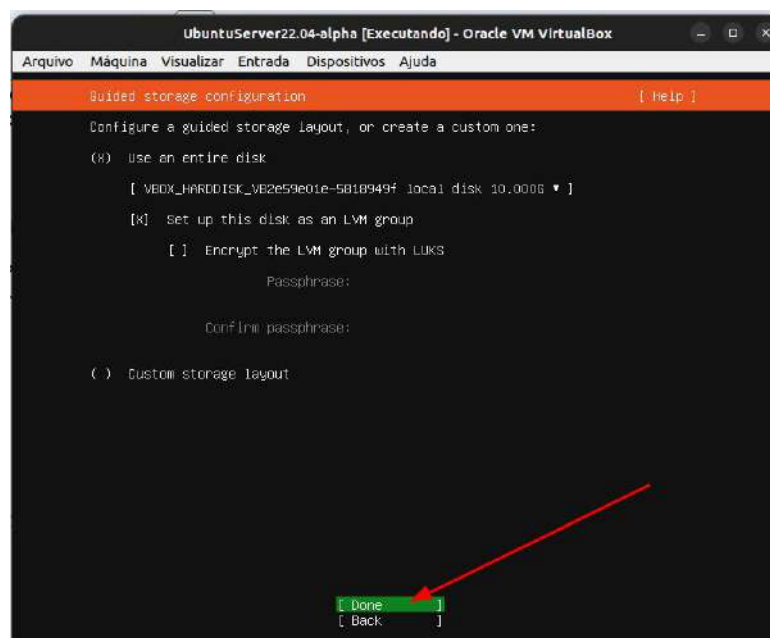
Se desejar usar um espelho alternativo para Ubuntu, você pode inserir os detalhes aqui. Caso contrário, aceite o espelho padrão selecionando Concluído (Done). Aceitei o padrão.



## Configuração de armazenamento guiada

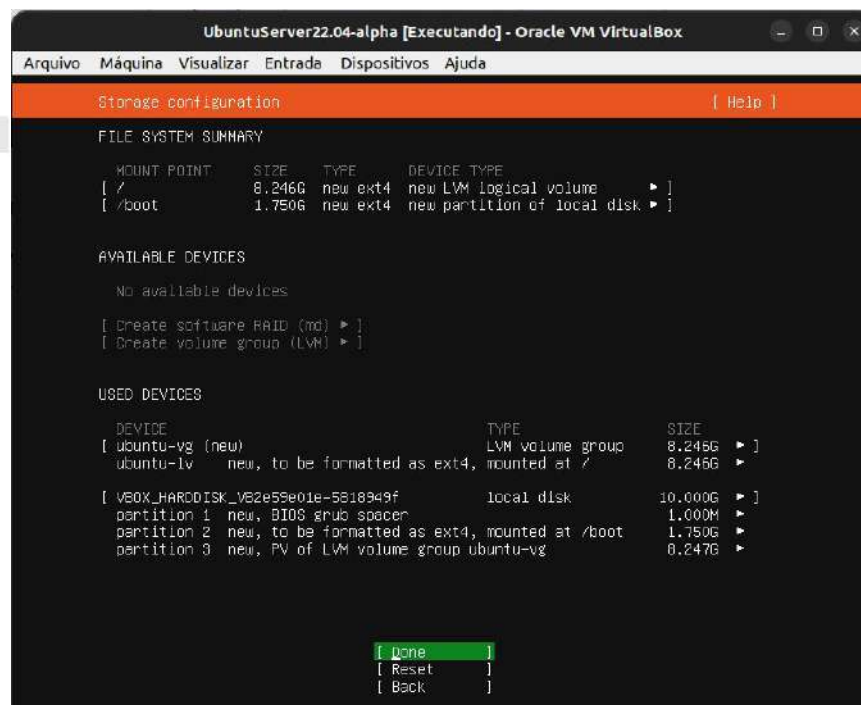
O instalador pode orientá-lo no particionamento de um disco inteiro (entire disk) ou, se preferir, você pode fazer isso manualmente. Se você optar por particionar um disco inteiro, você ainda terá a chance de revisar e modificar os resultados antes da instalação do Ubuntu. Optei por usar um disco inteiro.

Opcionalmente, você pode instruir o instalador a configurar o disco como um grupo LVM, bem como criptografá-lo usando LUKS. Optei pela configuração do LVM, pois o LVM oferece vários benefícios, como permitir backups mais fáceis de um servidor em execução.

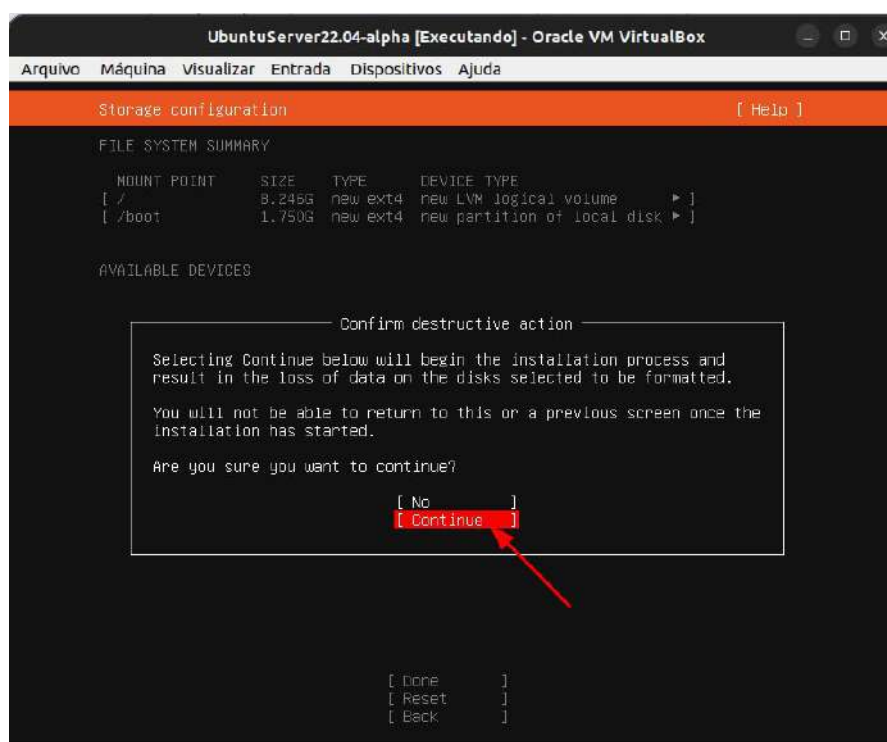


## Resumo da configuração de armazenamento

A próxima tela resume as escolhas que você fez na etapa anterior. Se você estiver satisfeito com tudo, selecione Concluído (Done).



Como esta é uma “ação destrutiva”, é necessário confirmar sua escolha de que deseja **Continuar**.



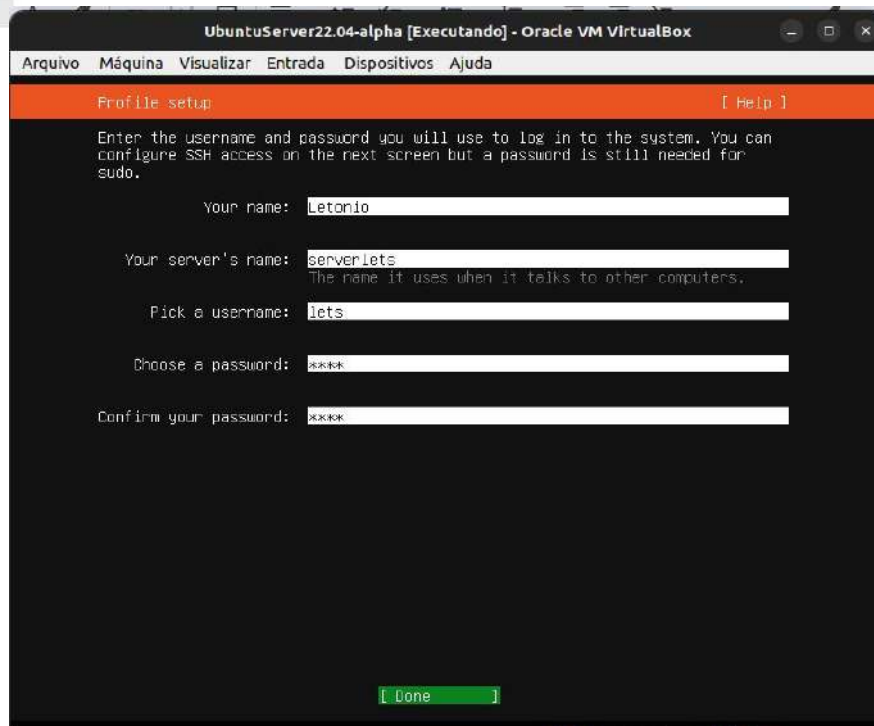


## Configuração de perfil

Aqui você é obrigado a inserir:

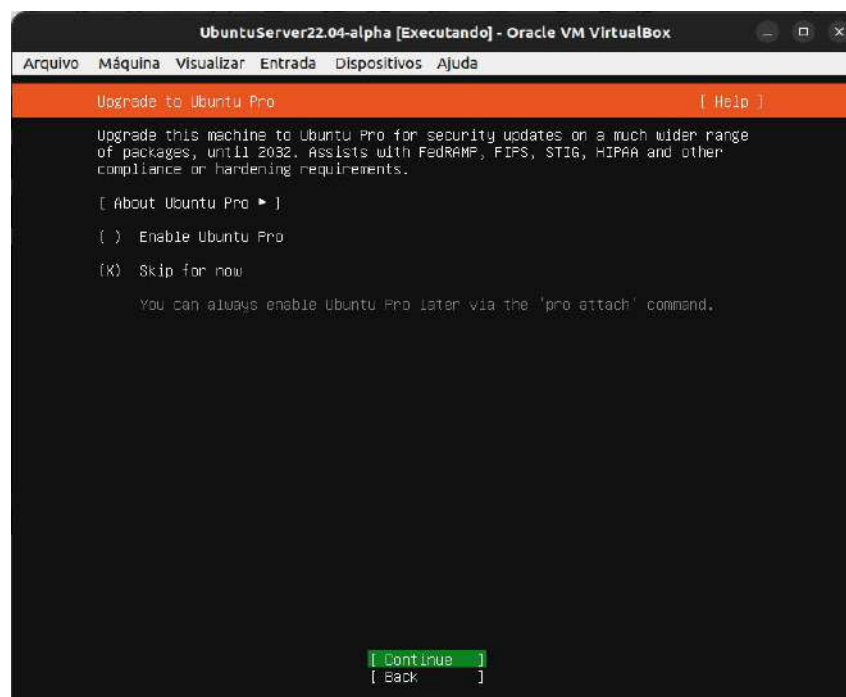
- Seu nome verdadeiro
- O nome do seu servidor
- Seu nome de usuário
- Senha

Preencha esses detalhes conforme achar adequado e, em seguida, aperte Concluído (Done).



## Upgrade pro

Não faça upgrade! Escolha **Skip for now** e, em seguida, pressione Concluído (Done).

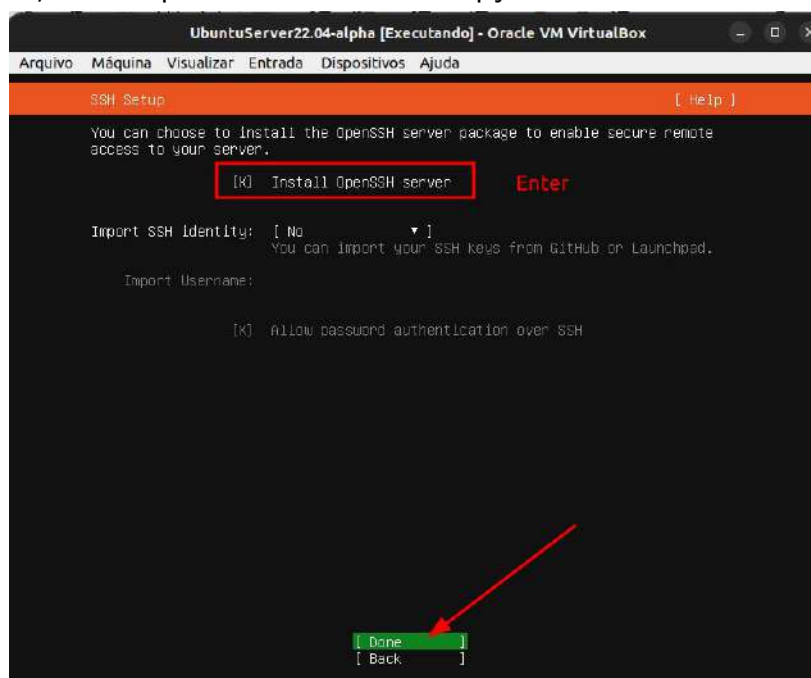


## Configuração do SSH

Nesse ponto, temos a oportunidade de instalar o pacote OpenSSH Server. Precisaremos disso para nos conectarmos à máquina virtual via SSH posteriormente, portanto, certifique-se de selecionar essa opção.

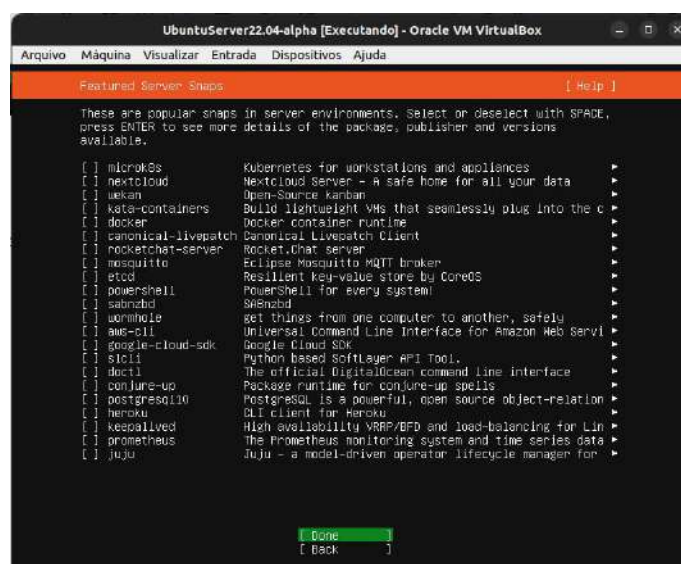
Essa tela também permite adicionar chave SSH do github, mas **não recomendo fazer isso**. Se você nunca configurou chave SSH do seu github, não se preocupe. Vamos aprender ainda a fazer isso, mas sinta-se livre para procurar mais sobre o assunto ou até mesmo fazer a instalação. Normalmente, as empresas exigem que seu github esteja com essa configuração de chaves, além da verificação de dois fatores, por critérios de segurança.

Para marcar a opção **Install OpenSSH Server** basta mover o cursor para essa opção e pressionar Enter. Ao fazer isso, um "X" aparecerá em cima dessa opção.



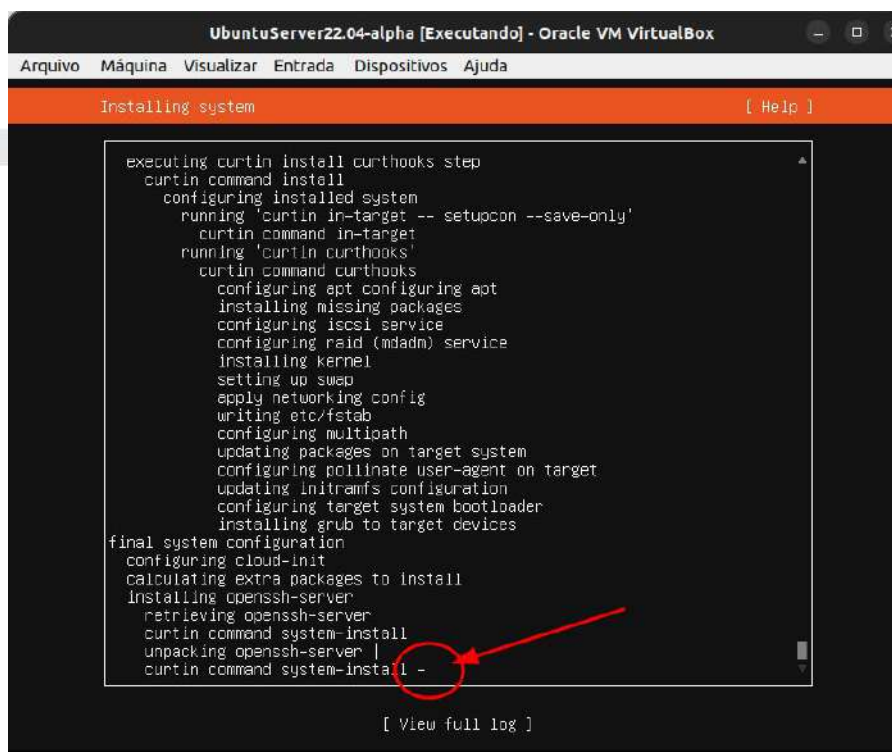
## Snaps de servidor em destaque

Aqui você pode selecionar em uma lista de snaps populares para instalar em seu sistema. **Snaps** são pacotes de software independentes que funcionam em diversas distribuições Linux. Recomendo deixar **vazio**.



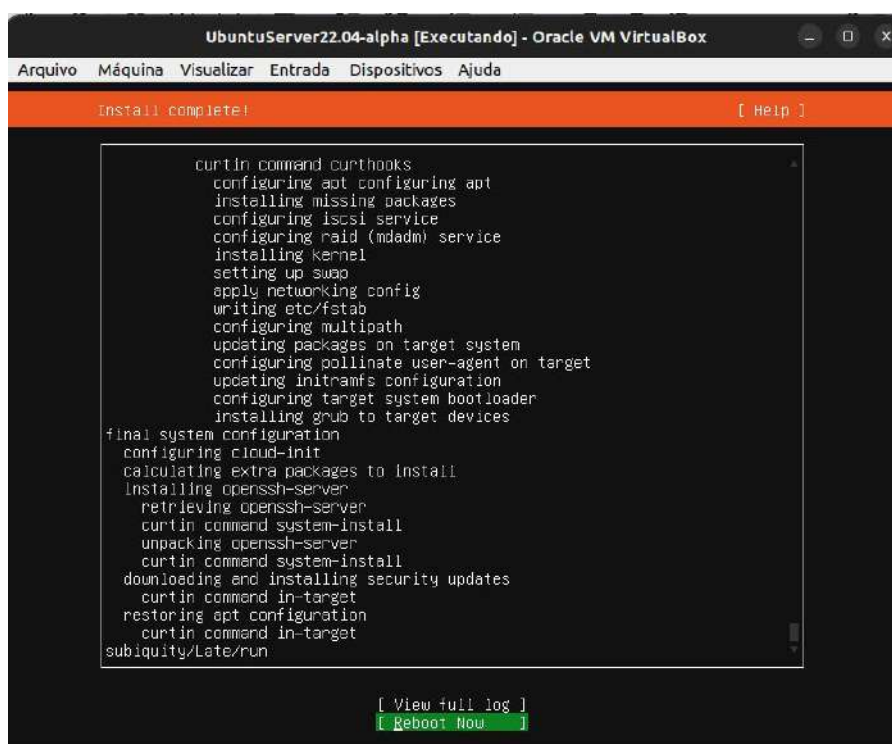
## Instalar e reiniciar

Após pressionar Concluído (Done), a instalação vai começar. Essa é a parte mais demorada e dura alguns minutos. Aproveite esse tempo para se levantar um pouco, beber uma água, etc.



```
executing curtin install curthooks step
curtin command install
configuring installed system
running 'curtin in-target -- setupcon --save-only'
curtin command in-target
running 'curtin curthooks'
curtin command curthooks
  configuring apt configuring apt
  installing missing packages
  configuring iscsi service
  configuring raid (mdadm) service
  installing kernel
  setting up swap
  apply networking config
  writing etc/fstab
  configuring multipath
  updating packages on target system
  configuring pollinate user-agent on target
  updating initramfs configuration
  configuring target system bootloader
  installing grub to target devices
final system configuration
  configuring cloud-init
  calculating extra packages to install
  installing openssh-server
  retrieving openssh-server
  curtin command system-install
  unpacking openssh-server
  curtin command system-install -
[ View full log ]
```

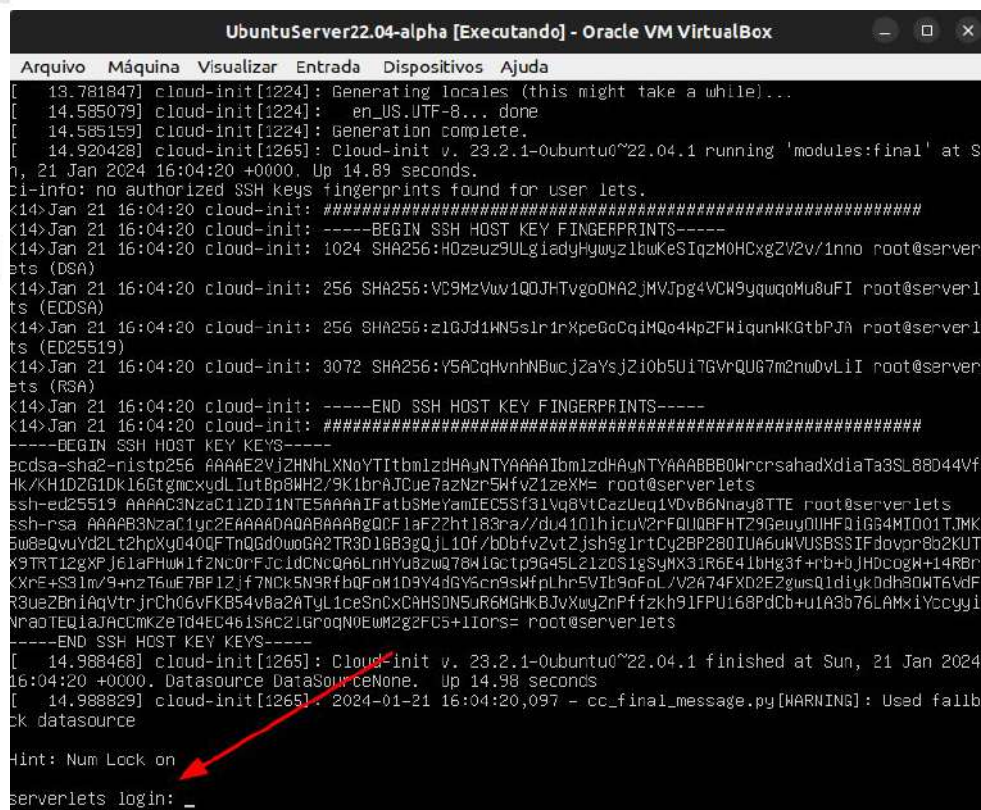
Após alguns minutos, a instalação estará concluída. Escolha a opção Reiniciar agora (Reboot Now).



```
curtin command curthooks
  configuring apt configuring apt
  installing missing packages
  configuring iscsi service
  configuring raid (mdadm) service
  installing kernel
  setting up swap
  apply networking config
  writing etc/fstab
  configuring multipath
  updating packages on target system
  configuring pollinate user-agent on target
  updating initramfs configuration
  configuring target system bootloader
  installing grub to target devices
final system configuration
  configuring cloud-init
  calculating extra packages to install
  installing openssh-server
  retrieving openssh-server
  curtin command system-install
  unpacking openssh-server
  curtin command system-install
  downloading and installing security updates
  curtin command in-target
  restoring apt configuration
  curtin command in-target
subiquity/Late/run
[ View full log ]
[ Reboot Now ]
```



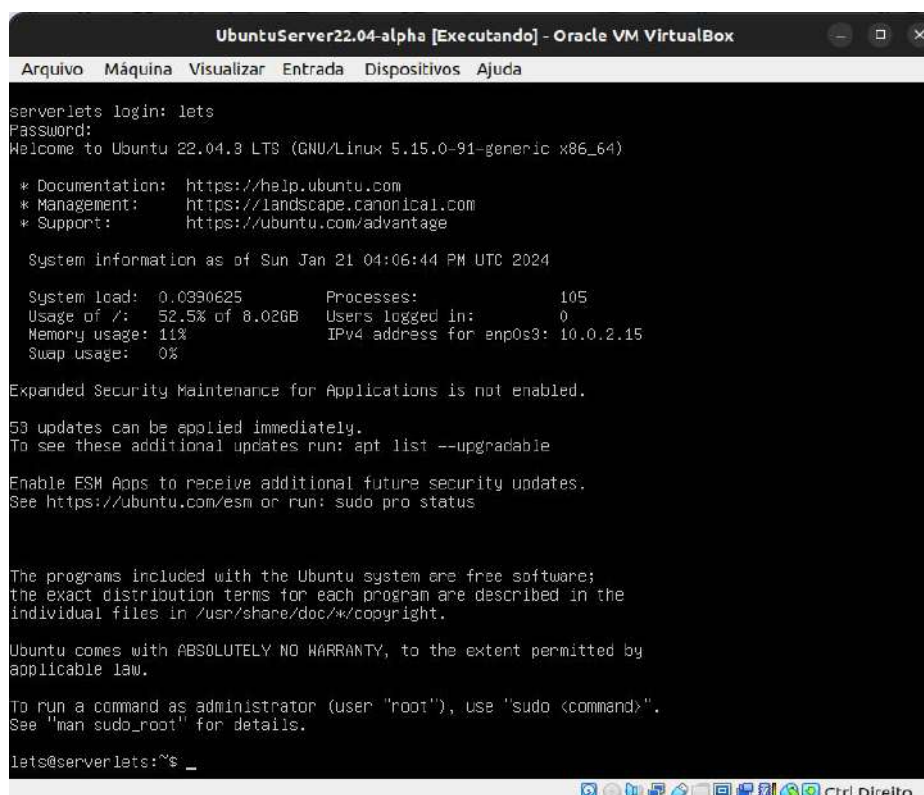
Após reiniciar, você deve fazer login. Caso não esteja visualizando o pedido de login na tela toda preta, pressione **Enter**. A imagem abaixo ilustra um exemplo onde a tela está toda poluída, mas ao pressionar Enter, o pedido de login apareceu. Um detalhe importante: **ao digitar a senha, não aparece asteriscos (\*)**, o cursor continua no mesmo lugar. Após digitar a senha, pressione enter. Caso tenha inserido a senha corretamente, aparecerá uma mensagem de bem-vindo.



```
UbuntuServer22.04-alpha [Executando] - Oracle VM VirtualBox
Arquivo  Máquina  Visualizar  Entrada  Dispositivos  Ajuda

[ 13.781847] cloud-init[1224]: Generating locales (this might take a while)...
[ 14.585079] cloud-init[1224]:   en_US.UTF-8... done
[ 14.585159] cloud-init[1224]: Generation complete.
[ 14.920428] cloud-init[1265]: Cloud-init v. 23.2.1-0ubuntu0~22.04.1 running 'modules:final' at Su
n, 21 Jan 2024 16:04:20 +0000. Up 14.89 seconds.
ci-info: no authorized SSH keys fingerprints found for user lets.
K14>Jan 21 16:04:20 cloud-init: #####
K14>Jan 21 16:04:20 cloud-init: -----BEGIN SSH HOST KEY FINGERPRINTS-----
K14>Jan 21 16:04:20 cloud-init: 1024 SHA256:HDzeuz9ULgiadyHywyz1bwKeSIqzM0HCxgZV2v/1nno root@server1
ets (DSA)
K14>Jan 21 16:04:20 cloud-init: 256 SHA256:VC9MzVwv1Q0JHtVgo0HA2jMVJpg4VCH9yquqoMu8uFI root@server1e
ts (ECDSA)
K14>Jan 21 16:04:20 cloud-init: 256 SHA256:21GJd1WN5s1r1nXpeGcQimQo4Hp2FWiqunWK0tbPJh root@server1e
ts (ED25519)
K14>Jan 21 16:04:20 cloud-init: 3072 SHA256:Y5ACqHvnhNBucjZaYsjZi0b5Ui7GvRQU67m2nwDvLiI root@server1
ets (RSA)
K14>Jan 21 16:04:20 cloud-init: -----END SSH HOST KEY FINGERPRINTS-----
K14>Jan 21 16:04:20 cloud-init: #####
-----BEGIN SSH HOST KEY KEYS-----
ecdsa-sha2-nistp256 AAAAE2VjZHNhLXNoYTItbmlzdHAuNTYAAAAIbmlzdHAuNTYAAABBB0WrcrsahadXdiaTa3SL88D44VfC
HK/KH1DZG1DK16GtgmcydLlUt8p8WH2/9K1brAJCue7azNzr5Wfv21zeXM= root@serverlets
ssh-ed25519 AAAAC3NzaC1l2D11NTE5AAAAIFatbSMeYamIEC5Sf31Vq8vtCezUeq1VDvB6Nnay8TTE root@serverlets
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAQGCFlafZ2ht183ra//du4101h1cuV2rFQU0BFHTZ9Geuy0UHFQIG54MI001TJMK+
5u8eQduYd2L2t2hpXy040QFtNQGdOmoGA2TR3D16B3gQJL10f/bDbfv2vtZjsh9glrtCy2BP280IUA6uWVUSBSIFdowar8b2KUTC
X9TRT12gXPJ61aFHW1f2NcorFJcIdCNCQA6LhYU82wQ78W1GctP9G45L21Z0S1gSyMX31R6E41bhg3f+rb+bJHdcogW+14RBre
XrE+S3lm/9+nzT6wE78P12jf7Nck5N9RfbQFm1D9Y4dGYscn9sWfplHr5Vib9oFoL/V2474FXD2E2gwsQ1diykDdh8QWT6vdf
R3ue2BniAqVtrjrcCh06vFKB54v8a2ATyL1ceSnCvCAHS0N5JR6MGHkBJvXuyZnPfzkh91FPU168PdCb+u1A3b76LAXxiYccyyiI
NpaoTEU1aJACmKzeId4EC461SAC2LGroqN0EwM2g2FC5+1Iors= root@serverlets
-----END SSH HOST KEY KEYS-----
[ 14.988468] cloud-init[1265]: Cloud-init v. 23.2.1-0ubuntu0~22.04.1 finished at Sun, 21 Jan 2024
16:04:20 +0000. DataSource DataSourceNone. Up 14.98 seconds
[ 14.988829] cloud-init[1265]: 2024-01-21 16:04:20,097 - cc_final_message.py[HARNING]: Used fallback
ck datasource
Hint: Num Lock on
serverlets login: _
```

Na imagem abaixo, fiz login com sucesso.



```
UbuntuServer22.04-alpha [Executando] - Oracle VM VirtualBox
Arquivo  Máquina  Visualizar  Entrada  Dispositivos  Ajuda

serverlets login: lets
Password:
Welcome to Ubuntu 22.04.3 LTS (GNU/Linux 5.15.0-91-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/advantage

System information as of Sun Jan 21 04:06:44 PM UTC 2024

System load:  0.0390625      Processes:            105
Usage of /:   52.5% of 8.026B Users logged in:          0
Memory usage: 11%           IPv4 address for enp0s3: 10.0.2.15
Swap usage:   0%

Expanded Security Maintenance for Applications is not enabled.

53 updates can be applied immediately.
To see these additional updates run: apt list --upgradable

Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

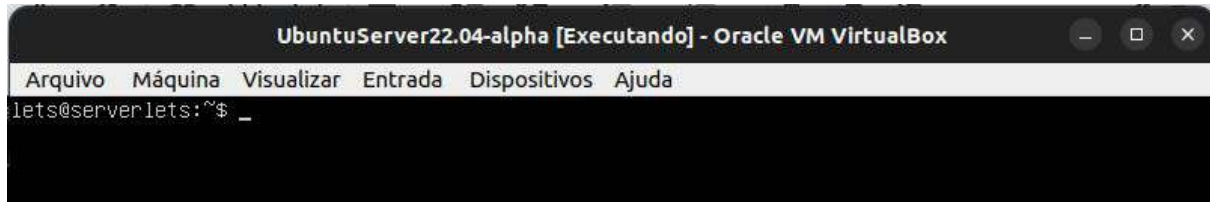
To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

lets@serverlets:~$
```

## Primeiros comandos

Há duas certezas na vida. Uma delas é de que ao fazer login e se deparar com a tela poluída de boas-vindas, o primeiro comando que você vai utilizar é:

**clear**



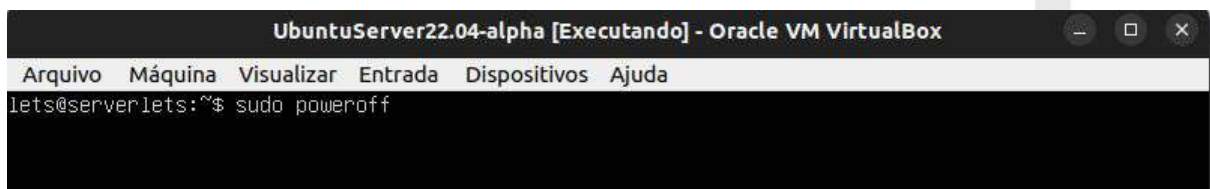
```
UbuntuServer22.04-alpha [Executando] - Oracle VM VirtualBox
Arquivo  Máquina  Visualizar  Entrada  Dispositivos  Ajuda
lets@serverlets:~$ _
```

O cachorro é o melhor amigo do homem e o comando clear é o do usuário de Linux.

Vamos falar um pouco sobre o **lets@serverlets**, presente na imagem anterior. o primeiro lets (antes do @) é o nome do usuário que está acessando o sistema nesse momento. Em outra aula, aprenderemos como criar novos usuários. O **serverlets** é o nome que defini para o meu servidor.

Podemos desligar a máquina com esse comando:

**sudo poweroff**



```
UbuntuServer22.04-alpha [Executando] - Oracle VM VirtualBox
Arquivo  Máquina  Visualizar  Entrada  Dispositivos  Ajuda
lets@serverlets:~$ sudo poweroff
```

Podemos confirmar que a máquina está desligada observando o menu à esquerda da VirtualBox.

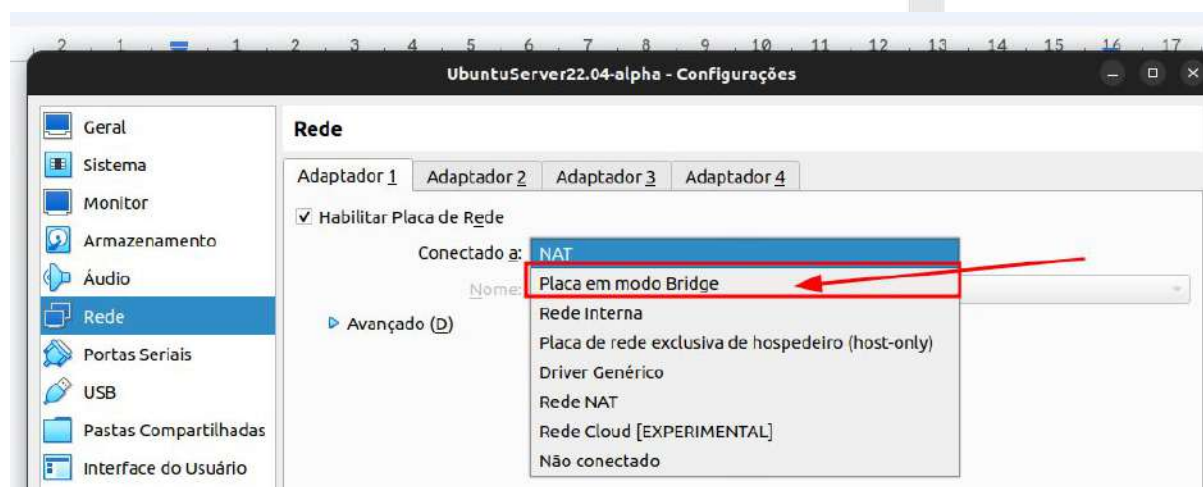


## Configurar o IP da VM

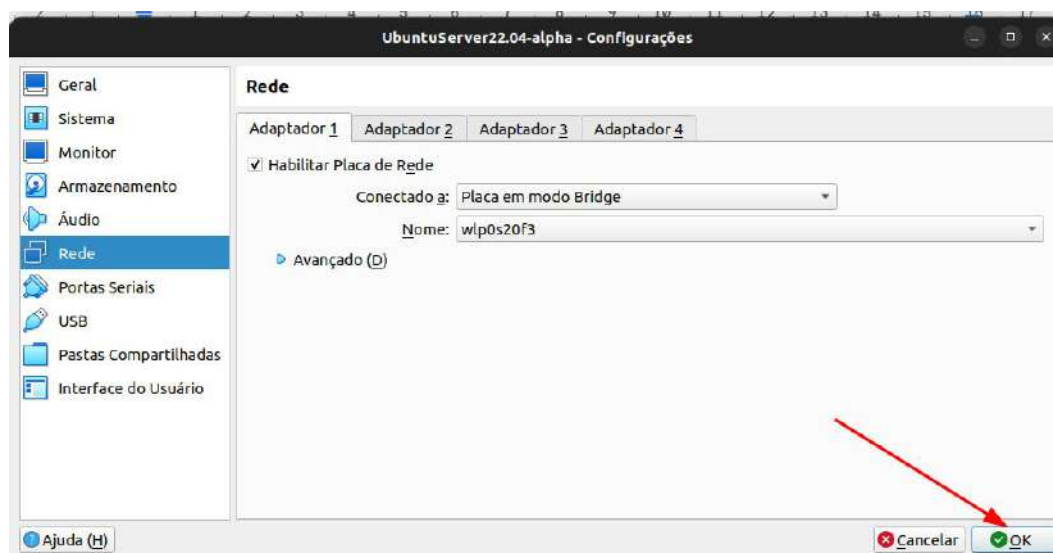
A próxima etapa é fornecer ao seu servidor Ubuntu um endereço IP na sua rede local. Com a máquina virtual selecionada vá em configurações (Settings).



Clique em Rede (Network) no menu à esquerda. Em seguida, em Adapter 1 (Adaptador 1), mude de NAT para Bridged Adapter (Placa em modo Bridge). Por fim, clique em OK.

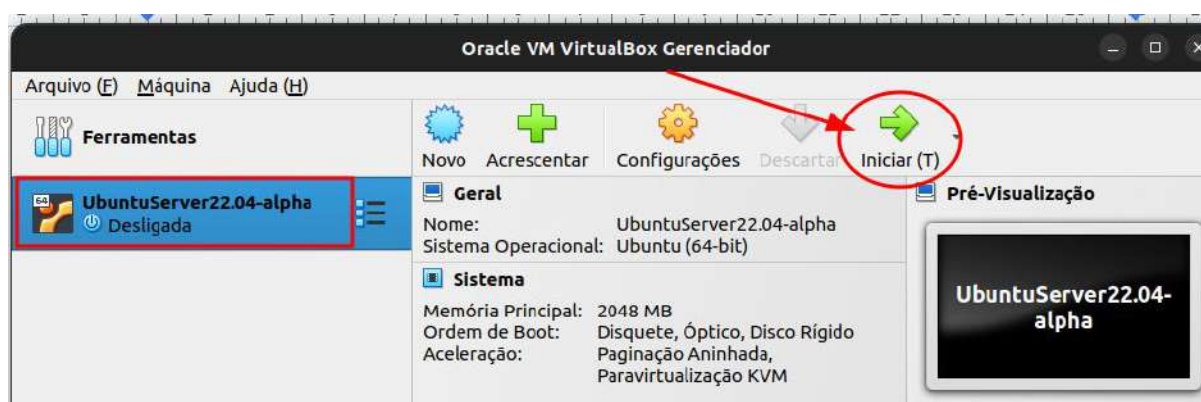


A imagem abaixo ilustra o clique em OK.





Na tela inicial da VirtualBox, mantenha a VM selecionada e clique em Iniciar (Start). Faça login normalmente. A seguir, aprenderemos um comando para visualizar o endereço IP da VM.



Para visualizar o IP, basta aplicar esse comando:

**ip address**

O comando **ip address** pode ser abreviado como:

**ip a**

Na imagem abaixo, nota-se que o endereço IP está após inet. No caso apresentado, o IP é apenas a parte antes de “barra 24” (/24), ou seja, 192.168.0.102. Lembre-se de anotar o IP da sua VM.

```
Arquivo  Máquina  Visualizar  Entrada  Dispositivos  Ajuda
lets@serverlets:~$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
        valid_lft forever preferred_lft forever
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 08:00:27:00:c5:86 brd ff:ff:ff:ff:ff:ff
    inet 192.168.0.102/24 metric 100 brd 192.168.0.255 scope global dynamic enp0s3
        valid_lft 86158sec preferred_lft 86158sec
    inet6 fe80::a00:27ff:fe00:c586/64 scope link
        valid_lft forever preferred_lft forever
lets@serverlets:~$
```

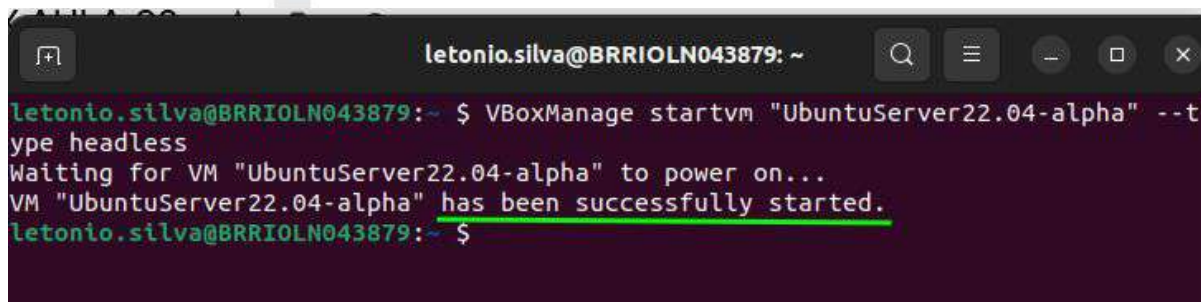
Se você já tentou passar o cursor do mouse sobre o terminal da VM, deve ter notado que ele some. O principal problema que isso acarreta é a impossibilidade de copiar e colar comandos no terminal. Para contornar esse problema, vamos aprender a usar SSH para fazer o acesso remoto à VM utilizando seu computador pessoal.

## Iniciar e parar a VirtualBox em Headless Mode

Se você acha trabalhoso ficar abrindo o software da VirtualBox, para depois clicar em iniciar. Há uma maneira mais direta, onde você utiliza o terminal do seu computador pessoal para iniciar a VM. Abra o terminal do seu desktop. Ao usar o [Headless Mode](#), sua máquina é ligada em background, sem a necessidade de abrir a interface da Oracle VirtualBox. Nos próximos comandos, lembre-se de substituir "UbuntuServer22.04-alpha" pelo nome da sua VM.

Para ligar, use esse comando:

**VBoxManage startvm "UbuntuServer22.04-alpha" --type headless**

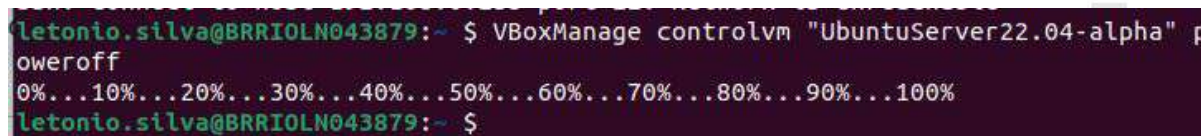


```
letonio.silva@BRRIOLN043879: ~  
letonio.silva@BRRIOLN043879:~ $ VBoxManage startvm "UbuntuServer22.04-alpha" --type headless  
Waiting for VM "UbuntuServer22.04-alpha" to power on...  
VM "UbuntuServer22.04-alpha" has been successfully started.  
letonio.silva@BRRIOLN043879:~ $
```

Pronto, sua máquina já está ligada. Em breve, aprenderemos como acessá-la via SSH.

Quando for necessário desligar a VM, aplique esse comando:

**VBoxManage controlvm "UbuntuServer22.04-alpha" poweroff**



```
letonio.silva@BRRIOLN043879:~ $ VBoxManage controlvm "UbuntuServer22.04-alpha" poweroff  
0%...10%...20%...30%...40%...50%...60%...70%...80%...90%...100%  
letonio.silva@BRRIOLN043879:~ $
```

## Acesso via SSH

Para acessar o servidor dentro da máquina virtual vamos usar [Secure Shell \(SSH\)](#).

Durante a instalação da VM, se você marcou a opção de instalar o [OpenSSL Server](#), não precisa instalar mais nada, a sua VM é capaz de receber conexões SSH. Caso contrário, **se você não instalou**, vai ser necessário fazer login na sua VM e aplicar esse comando:

**sudo apt install openssh-server**

Se o sistema operacional do seu computador pessoal é Linux ou MacOS, ele já vem com um client SSH embutido para permitir fazer conexões. As versões a partir do Windows 10 também ganharam essa capacidade. No entanto, versões anteriores ao Windows 10 não tem client SSH instalado, sendo necessário utilizar algum software que funcione como client. O software mais utilizado é o Putty (usava bastante no meu falecido Windows 7). Caso você se enquadre nessa situação, vá para a próxima página, na subseção [Usando Putty](#), para ver como usá-lo.

A sintaxe para acessar o servidor na VM via ssh é a seguinte:

**ssh username@endereco-ip**

```
letonio.silva@BRRIOLN043879:~ $ ssh lets@192.168.0.102
ED25519 key fingerprint is SHA256:zLGJd1WN5slr1rXpeGoCqIMQo4WpZFWiqunWKGtbPJA.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
```

Será necessário informar a senha referente ao usuário que você está usando para fazer o acesso. Em outra aula, aprendemos como fazer autenticação sem o uso de senha. Caso tenha ficado curioso, procure pelo assunto “chaves SSH”.

Após o login feito com sucesso, você recebe uma mensagem de boas-vindas. A principal vantagem de se conectar utilizando o seu terminal é que você pode copiar e colar comandos nele.

Para abandonar a conexão você pode aplicar esse comando:

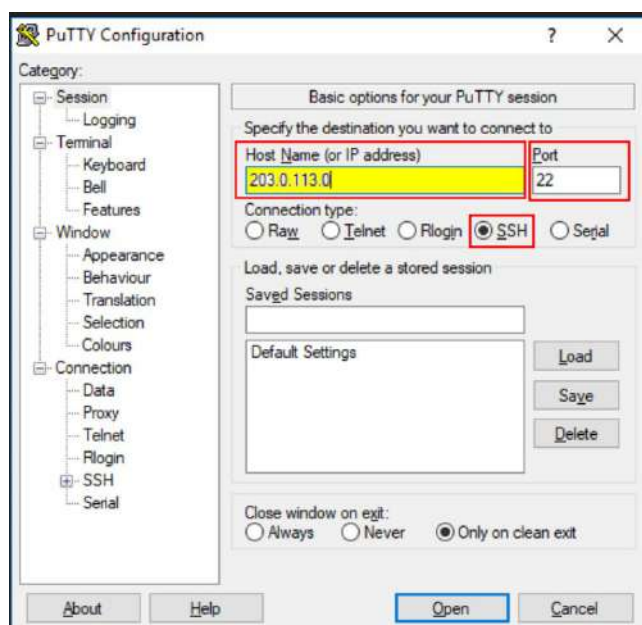
**exit**

```
letonio.silva@BRRIOLN043879: ~
lets@serverlets:~$ exit
logout
Connection to 192.168.0.102 closed.
letonio.silva@BRRIOLN043879:~$
```

## Usando Putty

Caso você tenha um Windows anterior ao 10, será necessário instalar o **Putty** para conseguir fazer conexões via SSH. Putty é um cliente de terminal open-source, leve e gratuito que oferece suporte a vários protocolos de rede, sendo mais conhecido pelo suporte ao protocolo SSH (Secure Shell). Ele é amplamente utilizado em sistemas operacionais Windows para estabelecer conexões seguras com servidores remotos.

Após abrir o Putty, você deve passar algumas informações: Endereço IP do Servidor; Porta SSH (Padrão é 22); Protocolo (SSH); Tipo de Conexão (SSH). Após inserir tudo, clique em Open para abrir a conexão.





## Resumo

Para ligar a VM headless mode:

**`VBoxManage startvm "UbuntuServer22.04-alpha" --type headless`**

Para acessar pelo terminal via SSH (substitua lets pelo usuário que você criou ao instalar o Ubuntu na VM, e coloque o endereço IP da sua máquina virtual (caso não saiba, use esse comando: ip a)):

**`ssh username@ip_do_servidor`**  
**`ssh lets@192.168.0.102`**

Para fazer logout do servidor:

**`exit`**

Para desligar a VM:

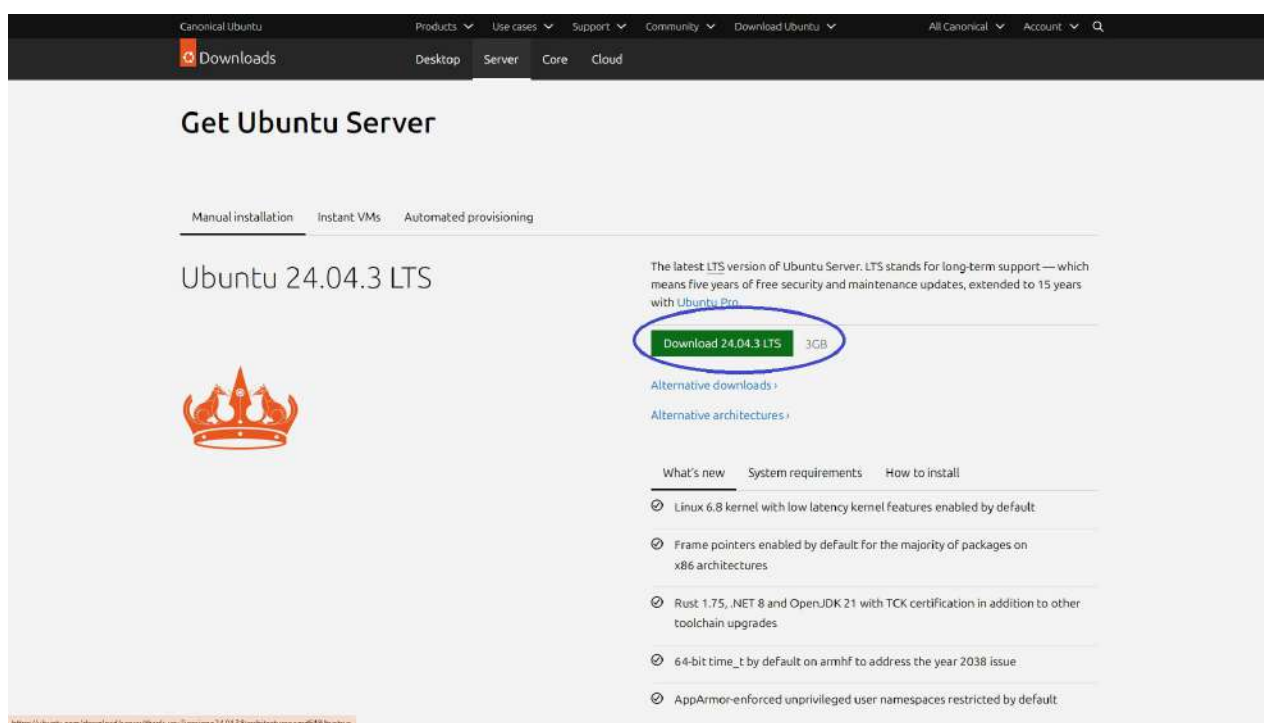
**`VBoxManage controlvm "UbuntuServer22.04-alpha" poweroff`**

## Diferenças com as versões atuais

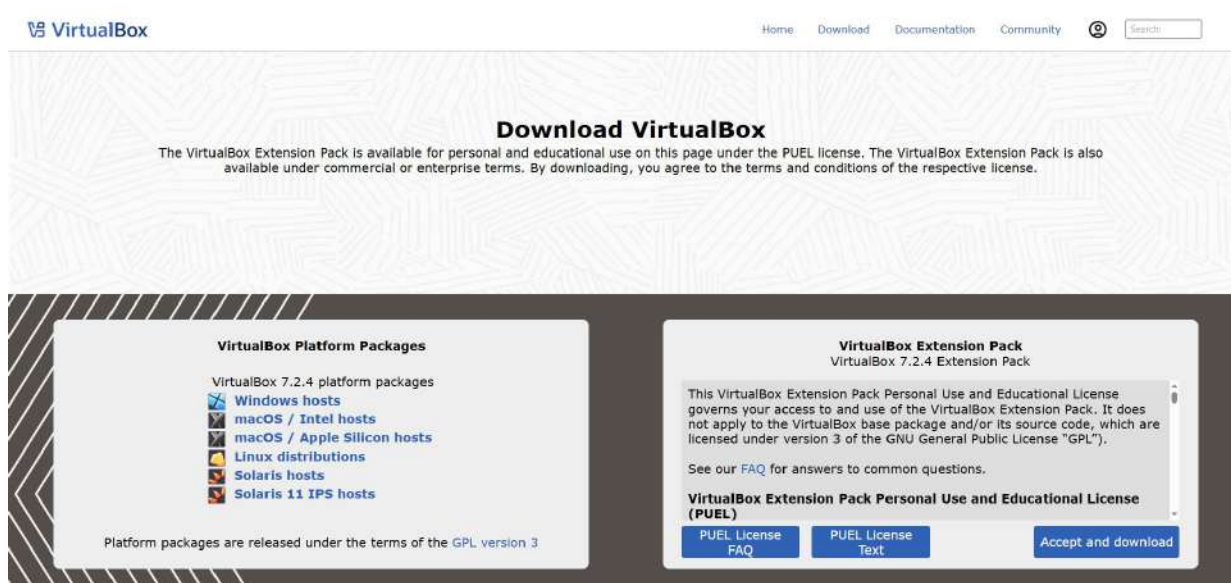
Com as versões atuais dos softwares, mudam algumas telas, que são descritas a seguir.

Observar também, que foi criado um usuário "**lets**" e aqui vamos utilizar um usuário "**aluno**" e o nome do servidor também foi alterado.

A versão mais atual e estável do Ubuntu Server LTS geralmente lançada no mês 4 de cada ano par, atualmente é a Versão 24.04.3 LTS, baixada no site do Ubuntu:

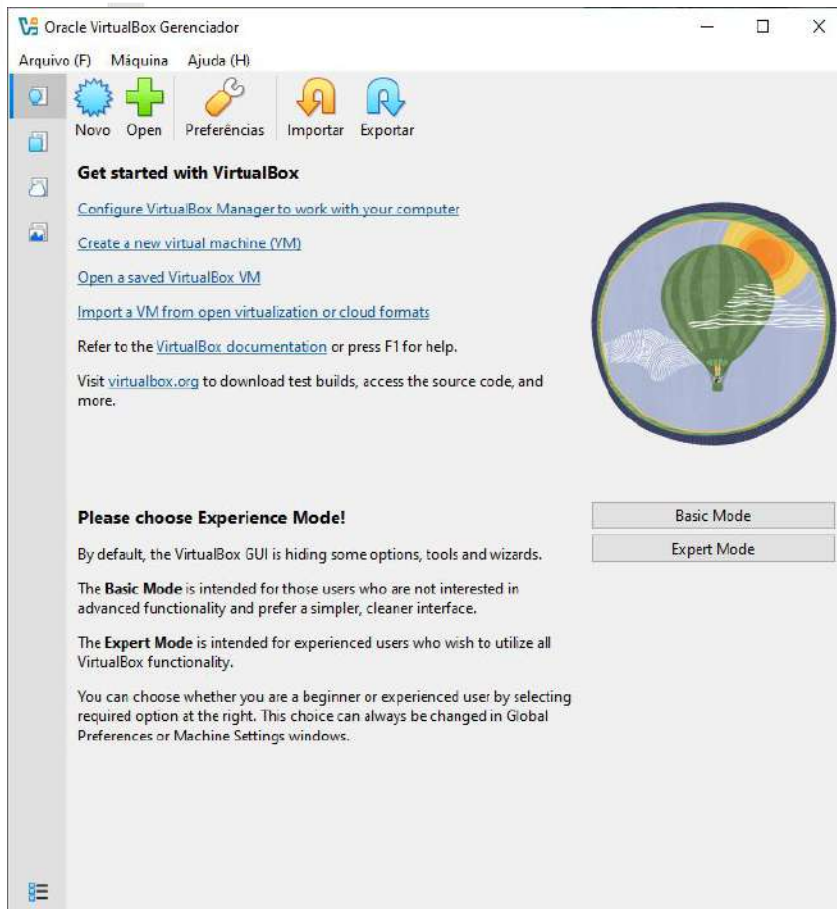


Também a versão mais atual do Virtual Box pode ser encontrada no site:

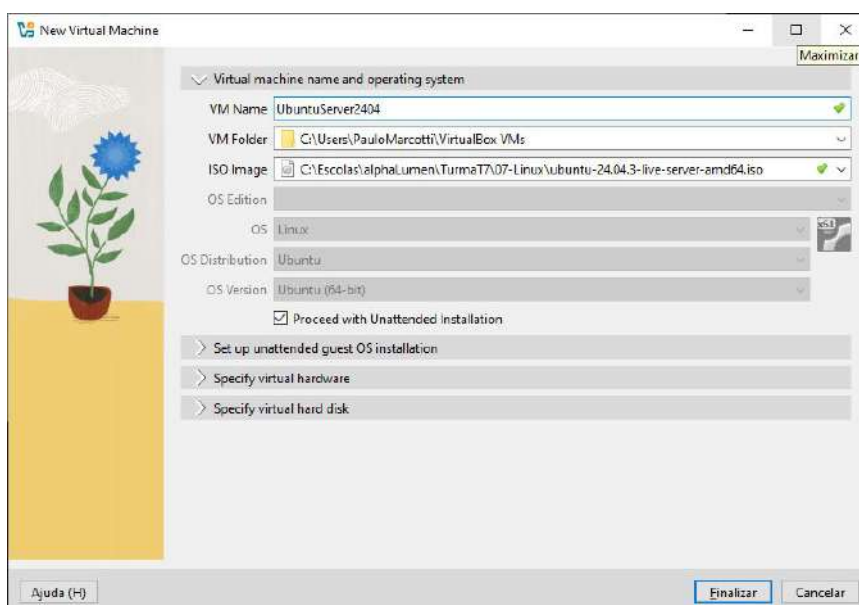


## Diferenças com as versões atuais

A tela de instalação do VirtualBox mudou, mas com as mesmas funcionalidades:



Criando uma nova máquina virtual com o nome "UbuntuServer2404" e usando o arquivo .iso baixado com o Linux 24.04.3:





## Diferenças com as versões atuais

Aqui foi criado um usuário "aluno" com a password "aluno", no Host Name "UbuntuServer2404".

Deste modo, os exemplos de acessar diretamente o protocolo SSH desse servidor, mudará para:

Para ligar a VM headless mode:

**VBoxManage startvm "UbuntuServer2404" --type headless**

Para acessar pelo terminal via SSH (substitua aluno pelo usuário que você criou ao instalar o Ubuntu na VM, e coloque o endereço IP da sua máquina virtual (caso não saiba, use esse comando: ip a)):

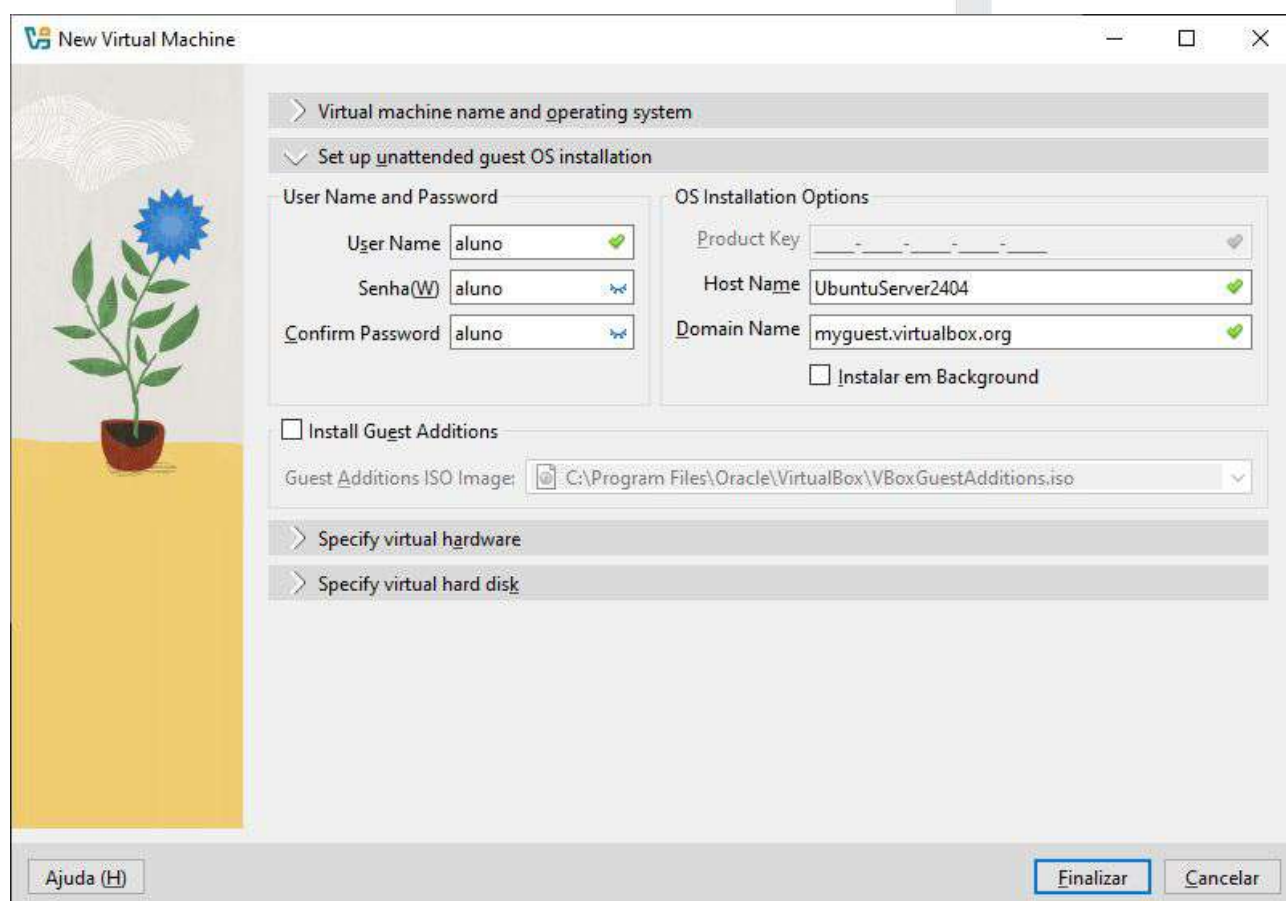
**ssh username@ip\_do\_servidor**  
**ssh aluno@192.168.0.102**

Para fazer logout do servidor:

**exit**

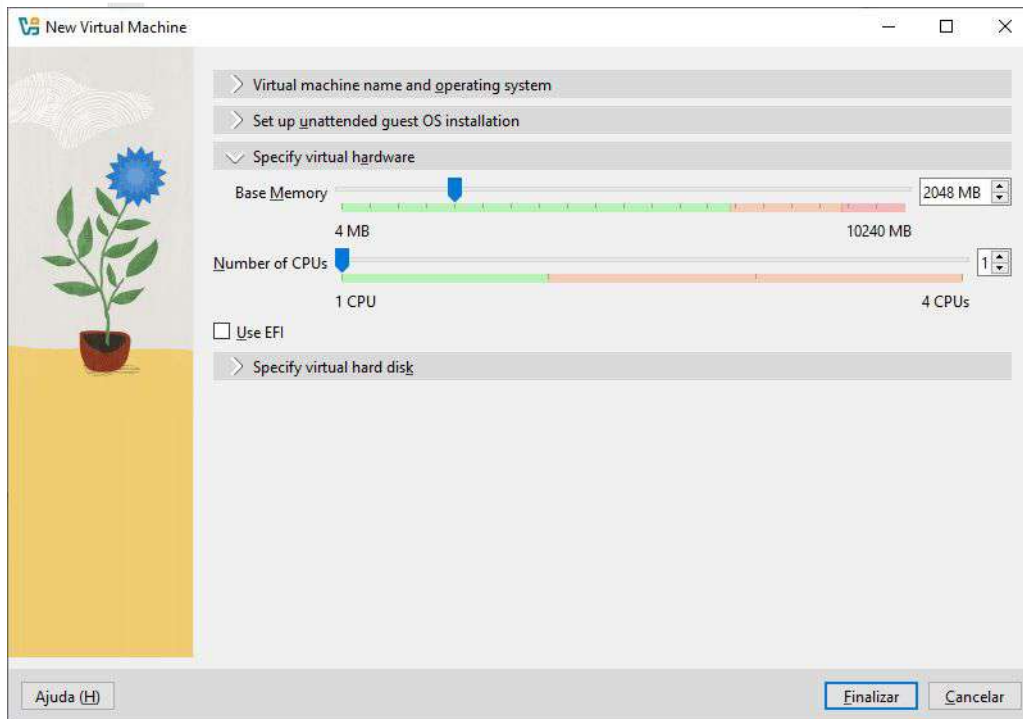
Para desligar a VM:

**VBoxManage controlvm "UbuntuServer2404" poweroff**

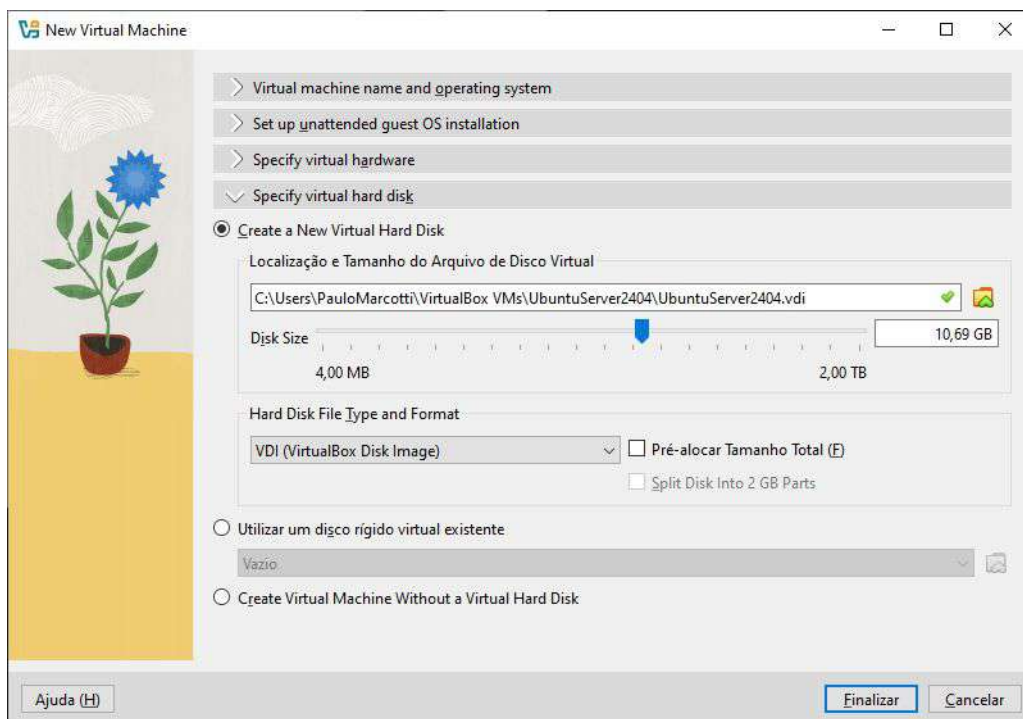


## Diferenças com as versões atuais

A seguir define a memória a ser utilizada pela VM:



Criando a seguir um Disco Virtual para a VM:

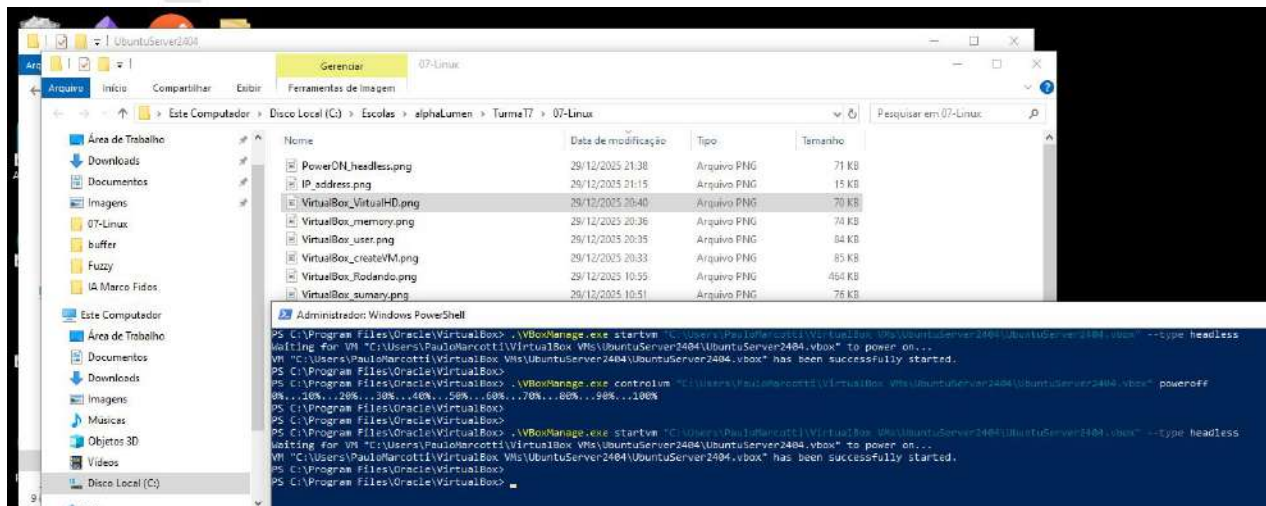


## Diferenças com as versões atuais

Para ligar a VM headless mode:

**VBoxManage startvm "UbuntuServer2404" --type headless**

Para acessar pelo terminal via SSH (com o usuário "aluno"):



Outras mudanças são simples de serem adaptadas.



## Comandos do Linux

O terminal do Linux é uma **interface de linha de comando** (Command Line Interface - CLI) que permite aos usuários interagirem com o sistema operacional por meio de comandos de texto.

Executar comandos no terminal significa enviar instruções específicas ao sistema operacional, geralmente por meio de linhas de texto. Os comandos são palavras-chave ou códigos que realizam uma variedade de tarefas, desde navegar pelos diretórios até instalar programas e realizar operações de sistema.

As flags, argumentos ou **opções de linha de comando**, são parâmetros adicionais que podem ser fornecidos junto com um comando para modificar seu comportamento. Elas são indicadas por um traço (-) seguido de uma letra ou uma sequência de letras. Por exemplo, em um comando como ls, que lista os arquivos em um diretório, você pode adicionar a flag -l (letra "l" minúscula) para exibir informações detalhadas sobre os arquivos. Na aula de hoje, vamos mergulhar de cabeça e aprender sobre os principais comandos que todo usuário de Linux deve conhecer.

## Comando pwd

No sistema Linux, o diretório **raiz (root)**, representado por /, é o ponto de partida para toda a estrutura de diretórios. Ainda na aula de hoje, vamos conhecer mais sobre essa estrutura. O comando **pwd** (present working directory) é usado para mostrar o caminho absoluto da raiz (/) até o diretório de trabalho atual. Lembre-se que o nome do usuário "**letonio.silva**" foi mudado por você na instalação, talvez para "**aluno**".

**pwd**

No exemplo a seguir, a saída obtida após a execução do comando pwd indica que o caminho da raiz até o diretório atual de trabalho (/exemplo/) é:

**/home/letonio.silva/Documentos/exemplo**

```
letonio.silva@BRRIOLN043879:~/Documentos/exemplo $ pwd
/home/letonio.silva/Documentos/exemplo
letonio.silva@BRRIOLN043879:~/Documentos/exemplo $
```

## Comando clear

O comando **clear** é utilizado para limpar a tela do terminal, removendo o conteúdo anterior e proporcionando uma tela limpa para facilitar a leitura de novas informações ou comandos.

**ANTES:**

```
letonio.silva@BRRIOLN043879:~/Documentos/exemplo $ pwd
/home/letonio.silva/Documentos/exemplo
letonio.silva@BRRIOLN043879:~/Documentos/exemplo $ clear
```

**DEPOIS:**

```
letonio.silva@BRRIOLN043879:~/Documentos/exemplo $
```

## Comando ls

O comando **ls** (list) é utilizado em sistemas Linux para listar os arquivos e diretórios no diretório atual (branco representa arquivo, azul representa diretórios, vermelho arquivos compactados, etc). Aplicar o comando com a opção **l** (L minúsculo), isto é, **ls -l** exibe uma listagem detalhada, que inclui informações como permissões, proprietário, grupo, tamanho, data de modificação e nome do arquivo. A listagem detalhada também indica se um item é um diretório (começa com a letra d) ou um arquivo.

Aplicar o comando com a flag **a** (**ls -a**) lista todos os arquivos, incluindo os ocultos. Os arquivos ocultos são aqueles cujos nomes começam com um ponto (.). Por padrão, o **ls** não exibe esses arquivos (na imagem abaixo note que tem um arquivo oculto `.env`).

Aplicar o comando com as duas flags combinadas (**ls -a -l** ou **ls -la** ou **ls -al**) exibe uma listagem detalhada que inclui todos os arquivos, inclusive os ocultos.

```
letonio.silva@BRRIOLN043879:~/Documentos/exemplo $ ls
css hello.txt html README.md
letonio.silva@BRRIOLN043879:~/Documentos/exemplo $ ls -l
total 16
drwxrwxr-x 2 letonio.silva letonio.silva 4096 jan 21 16:20 css
-rw-rw-r-- 1 letonio.silva letonio.silva 11 jan 21 16:16 hello.txt
drwxrwxr-x 2 letonio.silva letonio.silva 4096 jan 21 16:21 html
-rw-rw-r-- 1 letonio.silva letonio.silva 40 jan 21 16:19 README.md
letonio.silva@BRRIOLN043879:~/Documentos/exemplo $ ls -a
. .. css .env hello.txt html README.md
letonio.silva@BRRIOLN043879:~/Documentos/exemplo $ ls -la
total 28
drwxrwxr-x 4 letonio.silva letonio.silva 4096 jan 21 16:25 .
drwxr-xr-x 9 letonio.silva letonio.silva 4096 jan 21 16:11 ..
drwxrwxr-x 2 letonio.silva letonio.silva 4096 jan 21 16:20 css
-rw-rw-r-- 1 letonio.silva letonio.silva 8 jan 21 16:25 .env
-rw-rw-r-- 1 letonio.silva letonio.silva 11 jan 21 16:16 hello.txt
drwxrwxr-x 2 letonio.silva letonio.silva 4096 jan 21 16:21 html
-rw-rw-r-- 1 letonio.silva letonio.silva 40 jan 21 16:19 README.md
letonio.silva@BRRIOLN043879:~/Documentos/exemplo $
```

Podemos exibir o conteúdo das pastas e subpastas do diretório atual de forma recursiva usando a flag **-R** (recursive).

### ls -R

```
letonio.silva@BRRIOLN043879:~/Documentos/exemplo $ ls -R
.:
css dir-teste hello.txt html README.md teste

./css:
style.css

./dir-teste:

./html:
index.html

./teste:
arquivo.txt dir-com-arq

./teste/dir-com-arq:
arq.txt
letonio.silva@BRRIOLN043879:~/Documentos/exemplo $
```



## Comando cd

O comando **cd** (change directory) é usado no terminal para mudar o diretório atual, isto é, permite navegar para outro diretório. Quando estiver escrevendo o nome de um diretório que você deseja acessar, você pode escrever o início do nome do diretório e apertar TAB para ativar o **autocomplete**. O autocomplete é case sensitive, isto é, diferencia letras maiúsculas de minúsculas. Portanto, não adianta escrever **d** (minúsculo) e esperar que ele complete com **Documentos**.

A sintaxe básica é:

**cd <caminho\_ate\_o\_diretorio\_desejado>**

Podemos fazer uso de **caminho absoluto** e **caminho relativo**. Em alguns casos, é mais fácil informar o caminho (path) completo a partir da raiz (root - /). No exemplo a seguir, passamos o caminho absoluto até o diretório /exemplo:

**cd /home/letonio.silva/Documentos/exemplo**

```
letonio.silva@BRRIOLN043879:~/Documentos $ cd /home/letonio.silva/Documentos/exemplo/
letonio.silva@BRRIOLN043879:~/Documentos/exemplo $
letonio.silva@BRRIOLN043879:~/Documentos/exemplo $
```

Alternativamente, poderíamos passar o caminho relativo, que leva em consideração o diretório atual e, portanto, basta informar para onde devemos ir a partir dele. A seguir apresenta-se duas sintaxes válidas:

**cd exemplo/**  
**cd ./exemplo/**

O ponto (.) indica faz referência ao diretório atual. Suponha que você não sabe exatamente onde está o subdiretório que deseja acessar. Nesse caso, você pode ir combinando cd e ls, para ver os subdiretórios e decidir qual deseja acessar.

```
letonio.silva@BRRIOLN043879:~/Documentos $ cd ./exemplo/
letonio.silva@BRRIOLN043879:~/Documentos/exemplo $ pwd
/home/letonio.silva/Documentos/exemplo
letonio.silva@BRRIOLN043879:~/Documentos/exemplo $
```

**cd** : Sem argumentos, simplesmente retorna para o diretório inicial do usuário. Isso é geralmente o diretório pessoal do usuário, como /home/seu-usuario.

**cd ~** : Muda para o diretório pessoal do usuário. Essa notação com o til (~) é uma maneira rápida de voltar para o diretório inicial do usuário (/home/seu-usuario). Portanto, cd sem argumentos ou com til (~) são equivalentes.

```
letonio.silva@BRRIOLN043879:~/Documentos/exemplo $ cd
letonio.silva@BRRIOLN043879:~ $ pwd
/home/letonio.silva
letonio.silva@BRRIOLN043879:~ $ cd Documentos/exemplo/
letonio.silva@BRRIOLN043879:~/Documentos/exemplo $ pwd
/home/letonio.silva/Documentos/exemplo
letonio.silva@BRRIOLN043879:~/Documentos/exemplo $ cd ~
letonio.silva@BRRIOLN043879:~ $ pwd
/home/letonio.silva
letonio.silva@BRRIOLN043879:~ $
```



**cd .** : Representa o diretório atual. Isso pode ser útil em alguns scripts ou sequências de comandos, mas, em geral, **cd .** não tem um impacto perceptível, já que você permanece no diretório atual.

```
letonio.silva@BRRRIOLN043879:~ $ cd ./Documentos/exemplo/  
letonio.silva@BRRRIOLN043879:~/Documentos/exemplo $ pwd  
/home/letonio.silva/Documentos/exemplo  
letonio.silva@BRRRIOLN043879:~/Documentos/exemplo $ cd .  
letonio.silva@BRRRIOLN043879:~/Documentos/exemplo $
```

**cd ..** : Volta para o diretório pai (um nível acima na hierarquia de diretórios).

```
letonio.silva@BRRRIOLN043879:~/Documentos/exemplo $ cd ..  
letonio.silva@BRRRIOLN043879:~/Documentos $ cd ..  
letonio.silva@BRRRIOLN043879:~ $ pwd  
/home/letonio.silva  
letonio.silva@BRRRIOLN043879:~ $ cd Documentos/exemplo/  
letonio.silva@BRRRIOLN043879:~/Documentos/exemplo $ cd ../..  
letonio.silva@BRRRIOLN043879:~ $ pwd  
/home/letonio.silva  
letonio.silva@BRRRIOLN043879:~ $
```

**cd /** : O comando **cd /** é usado para mudar para o diretório **raiz** (root) do sistema de arquivos. O diretório raiz é o ponto de partida fundamental de toda a hierarquia de diretórios no sistema Linux. Mais tarde vamos falar sobre alguns diretórios da raiz.

```
letonio.silva@BRRRIOLN043879:~ $ pwd  
/home/letonio.silva  
letonio.silva@BRRRIOLN043879:~ $ cd /  
letonio.silva@BRRRIOLN043879:/ $ ls  
bin  etc  lib32  log      mnt  root  snap  sys  var  
boot home lib64  lost+found  opt  run  srv   tmp  
dev  lib  libx32 media  proc sbin  swap.img  usr  
letonio.silva@BRRRIOLN043879:/ $ pwd  
/  
letonio.silva@BRRRIOLN043879:/ $
```

## Comando cat

O comando **cat** (concatenate) é usado para exibir o conteúdo de um ou mais arquivos de texto no terminal. Passar a flag **-n** aparece a numeração das linhas. Se usar **tac** em vez de **cat**, o conteúdo do arquivo é impresso na ordem inversa (da última para a primeira linha).

```
letonio.silva@BRRRIOLN043879:~/Documentos/exemplo $ cat hello.txt  
olá, mundo!!  
Oi, pessoal!  
letonio.silva@BRRRIOLN043879:~/Documentos/exemplo $ cat -n hello.txt  
1 olá, mundo!!  
2 Oi, pessoal!  
letonio.silva@BRRRIOLN043879:~/Documentos/exemplo $ tac hello.txt  
Oi, pessoal!  
olá, mundo!!
```

## Comando echo

O comando **echo** é usado para imprimir texto no terminal. A sintaxe básica é

**echo "algum texto"**

Pode ser utilizado em conjunto dos operadores **>** e **>>** para redirecionar o texto e, consequentemente, salvar o conteúdo dentro de um arquivo, em vez de imprimir na saída do terminal. Na imagem abaixo, **echo "Oi, pessoal!"** imprime o texto no terminal. Além disso, o comando **echo > nome\_do\_arquivo** cria um arquivo vazio. Para provar, usamos **cat** para mostrar o conteúdo do arquivo e notamos que nada foi impresso no terminal.

```
letonio.silva@BRRIOLN043879:~/Documentos/exemplo $ echo "Oi, pessoal!"
Oi, pessoal!
letonio.silva@BRRIOLN043879:~/Documentos/exemplo $ echo > arquivo-vazio.txt
letonio.silva@BRRIOLN043879:~/Documentos/exemplo $ cat arquivo-vazio.txt

letonio.silva@BRRIOLN043879:~/Documentos/exemplo $
```

O comando **echo "algum texto" > arquivo** cria o arquivo caso não exista e adiciona dentro dele "algum texto". Caso o arquivo já exista, seu conteúdo será substituído por "algum texto".

O comando **echo "outro texto" >> arquivo** cria o arquivo caso não exista e adiciona dentro dele "outro texto". Caso o arquivo já exista, "outro texto" será adicionado ao final do arquivo. Portanto, o uso de **>>** adiciona o texto à linha seguinte no arquivo, preservando o conteúdo existente.

```
letonio.silva@BRRIOLN043879:~/Documentos/exemplo $ echo "primeiro texto" > arquivo-novo.txt
letonio.silva@BRRIOLN043879:~/Documentos/exemplo $ cat arquivo-novo.txt
primeiro texto
letonio.silva@BRRIOLN043879:~/Documentos/exemplo $ echo "segundo texto" > arquivo-novo.txt
letonio.silva@BRRIOLN043879:~/Documentos/exemplo $ cat arquivo-novo.txt
segundo texto
letonio.silva@BRRIOLN043879:~/Documentos/exemplo $ echo "terceiro texto" >> arquivo-novo.txt
letonio.silva@BRRIOLN043879:~/Documentos/exemplo $ cat arquivo-novo.txt
segundo texto
terceiro texto
letonio.silva@BRRIOLN043879:~/Documentos/exemplo $
```

## Comando touch

O comando **touch** é usado para criar novos arquivos vazios ou atualizar a data de modificação de arquivos existentes.

```
letonio.silva@BRRIOLN043879:~/Documentos/exemplo $ touch teste.txt
letonio.silva@BRRIOLN043879:~/Documentos/exemplo $ ls
css  hello.txt  html  README.md  teste.txt
letonio.silva@BRRIOLN043879:~/Documentos/exemplo $
```



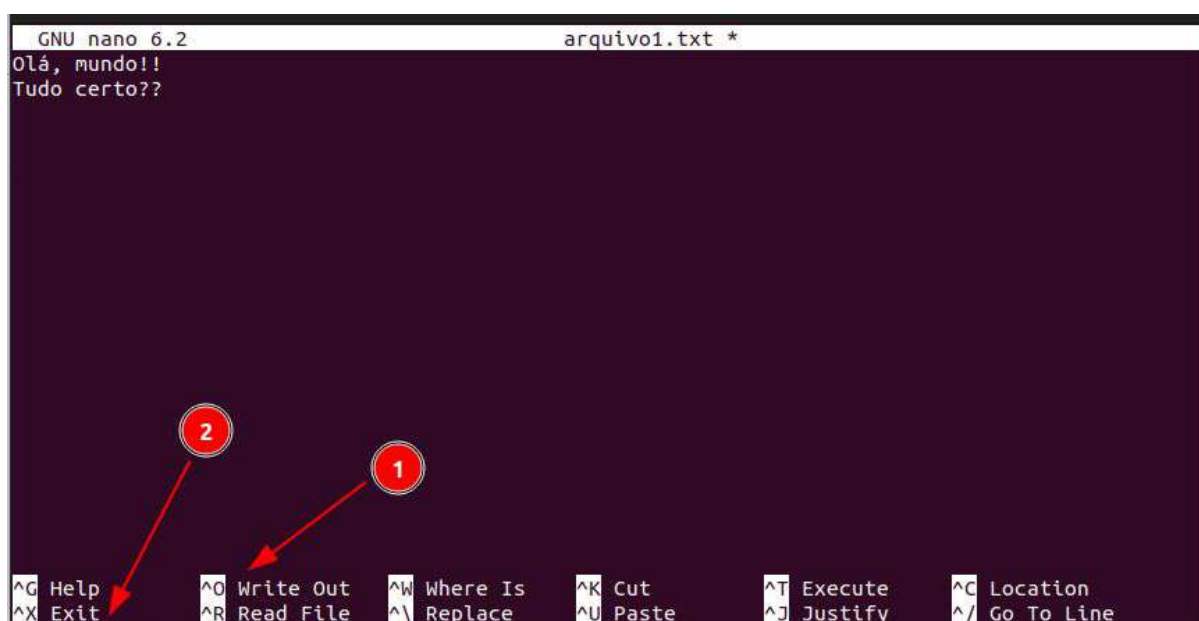
## Comando nano

O **Nano** é um editor de texto simples e leve que é incluído nas distribuições Linux, como o Ubuntu. Ele é projetado para ser fácil de usar e oferece uma interface de usuário intuitiva, tornando-o uma escolha popular para usuários iniciantes. Podemos criar um arquivo vazio e acessá-lo através do comando a seguir:

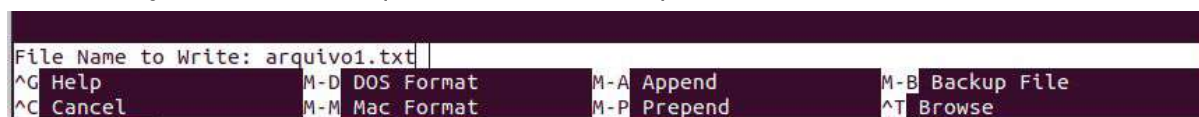
**nano arquivo1.txt**

```
letonio.silva@BRRIOLN043879:~/Documentos/alphaedtech/examples $ nano arquivo1.txt
```

O Nano abre uma janela de edição com o cursor no início do arquivo.



Após adicionar algum texto, você pode salvar as modificações pressionando **CTRL+O**. Será solicitada uma confirmação do nome do arquivo. Pressione **Enter** para confirmar.



Para sair do Nano, pressione **Ctrl + X**. Caso você tenha feito outras alterações, será perguntado se deseja salvar as alterações antes de sair. Se você já salvou, simplesmente pressione Enter para sair.

Podemos visualizar o conteúdo do arquivo usando cat, conforme mostrado na imagem abaixo.

```
letonio.silva@BRRIOLN043879:~/Documentos/alphaedtech/examples $ nano arquivo1.txt
letonio.silva@BRRIOLN043879:~/Documentos/alphaedtech/examples $ cat arquivo1.txt
Olá, mundo!!
Tudo certo??
letonio.silva@BRRIOLN043879:~/Documentos/alphaedtech/exanples $
```



## Comando mkdir

O comando **mkdir** é uma abreviação de **make directory** (criar diretório, em inglês) e sua função principal é criar diretórios (pastas). A sintaxe básica é:

**mkdir <nome\_do\_diretorio>**

Podemos criar mais de um diretório de uma vez. Por exemplo, o comando a seguir cria dois subdiretórios vazios dentro do diretório atual

**mkdir dir-1 dir-2**

A flag **-p** permite a criação de diretórios pai, se eles não existirem. Por exemplo, aplicando comando **mkdir dir-pai/dir-filho**, se "dir-pai" não existir, o comando criará tanto o diretório pai quanto o filho.

```
letonio.silva@BRRIOLN043879:~/Documentos/exemplo $ ls
css hello.txt html README.md
letonio.silva@BRRIOLN043879:~/Documentos/exemplo $ mkdir novo-dir
letonio.silva@BRRIOLN043879:~/Documentos/exemplo $ ls
css hello.txt html novo-dir README.md
letonio.silva@BRRIOLN043879:~/Documentos/exemplo $ mkdir dir-1 dir-2
letonio.silva@BRRIOLN043879:~/Documentos/exemplo $ ls
css dir-1 dir-2 hello.txt html novo-dir README.md
```

A flag **-p** permite a criação de diretórios pai, se eles não existirem. Por exemplo, aplicando comando **mkdir dir-pai/dir-filho**, se "dir-pai" não existir, o comando criará tanto o diretório pai quanto o filho.

```
letonio.silva@BRRIOLN043879:~/Documentos/exemplo $ mkdir -p dir-pai/dir-filho
letonio.silva@BRRIOLN043879:~/Documentos/exemplo $ ls
css dir-1 dir-2 dir-pai hello.txt html novo-dir README.md
letonio.silva@BRRIOLN043879:~/Documentos/exemplo $ cd dir-pai
letonio.silva@BRRIOLN043879:~/Documentos/exemplo/dir-pai $ ls
dir-filho
letonio.silva@BRRIOLN043879:~/Documentos/exemplo/dir-pai $
```

## Comando rm

O comando **rm** é utilizado para remover arquivos ou diretórios. A sintaxe básica para remover arquivo é:

**rm <arquivo>**

Para remover diretório use:

**rm -r <diretorio>**

A opção **-r** vem de recursive e permite deletar o diretório completamente, incluindo arquivos e subdiretórios presentes nele. Tentar remover um diretório sem essa flag **-r** gera uma mensagem.

```
letonio.silva@BRRIOLN043879:~/Documentos/exemplo/teste $ ls
arquivo.txt dir-teste
letonio.silva@BRRIOLN043879:~/Documentos/exemplo/teste $ rm arquivo.txt
letonio.silva@BRRIOLN043879:~/Documentos/exemplo/teste $ ls
dir-teste
letonio.silva@BRRIOLN043879:~/Documentos/exemplo/teste $ rm dir-teste/
rm: cannot remove 'dir-teste/': Is a directory
letonio.silva@BRRIOLN043879:~/Documentos/exemplo/teste $ rm -r dir-teste
letonio.silva@BRRIOLN043879:~/Documentos/exemplo/teste $ ls
letonio.silva@BRRIOLN043879:~/Documentos/exemplo/teste $
```

Pode ser necessário a flag **-f** (force) para forçar a remoção e suprimir eventuais mensagens de confirmação de que deseja realmente excluir o arquivo. Por exemplo, esse comando

**rm -rf <diretorio-com-conteudo>**

apaga de forma recursiva e forçada o diretório e todo seu conteúdo sem exibir mensagem de confirmação.

```
letonio.silva@BRRRIOLN043879:~/Documentos/exemplo/teste $ ls -R
.:
arquivo.txt  dir-com-arq

./dir-com-arq:
arq.txt
letonio.silva@BRRRIOLN043879:~/Documentos/exemplo/teste $ rm -rf dir-com-arq/
letonio.silva@BRRRIOLN043879:~/Documentos/exemplo/teste $ ls -R
.:
arquivo.txt
letonio.silva@BRRRIOLN043879:~/Documentos/exemplo/teste $ rm arquivo.txt
letonio.silva@BRRRIOLN043879:~/Documentos/exemplo/teste $ ls -R
.:
letonio.silva@BRRRIOLN043879:~/Documentos/exemplo/teste $
```

## Comando mv

O comando **mv** (move) é usado para mover arquivos e diretórios. A sintaxe básica é:

**mv <origem> <destino>**

```
letonio.silva@BRRRIOLN043879:~/Documentos/exemplo $ ls -R
.:
css  hello.txt  html  README.md

./css:
style.css

./html:
index.html
letonio.silva@BRRRIOLN043879:~/Documentos/exemplo $ mv hello.txt ./css/
letonio.silva@BRRRIOLN043879:~/Documentos/exemplo $ ls -R
.:
css  html  README.md

./css:
hello.txt  style.css

./html:
index.html
letonio.silva@BRRRIOLN043879:~/Documentos/exemplo $
```

A próxima imagem mostra o diretório html (que tem um arquivo index.html dentro) sendo movido para dentro do diretório ./css.

```
letonio.silva@BRRIOLN043879:~/Documentos/exemplo $ ls -R
.:
css  html  README.md

./css:
hello.txt  style.css

./html:
index.html
letonio.silva@BRRIOLN043879:~/Documentos/exemplo $ mv html ./css
letonio.silva@BRRIOLN043879:~/Documentos/exemplo $ ls -R
.:
css  README.md

./css:
hello.txt  html  style.css

./css/html:
index.html
letonio.silva@BRRIOLN043879:~/Documentos/exemplo $
```

Ao mover um arquivo ou diretório, podemos modificar seu nome no local de destino, ou seja, renomear na saída.

```
letonio.silva@BRRIOLN043879:~/Documentos/exemplo $ ls -R
.:
css  html  README.md

./css:
hello.txt  style.css

./html:
index.html
letonio.silva@BRRIOLN043879:~/Documentos/exemplo $ mv README.md ./css/RENAME-README.md
letonio.silva@BRRIOLN043879:~/Documentos/exemplo $ ls -R
.:
css  html

./css:
hello.txt  RENAME-README.md  style.css

./html:
index.html
letonio.silva@BRRIOLN043879:~/Documentos/exemplo $
```



## Comando cp

O comportamento do comando **cp** é bem parecido com o mv, porém ele faz uma cópia. A sintaxe básica:

**cp <origem> <destino>**

Além disso, ao copiar diretórios é importante usar **-r** para que a cópia seja feita de forma recursiva (ou seja, copiar também subdiretórios do diretório selecionado).

```
letonio.silva@BRRIOLN043879:~/Documentos/exemplo $ cp README.md ./css
letonio.silva@BRRIOLN043879:~/Documentos/exemplo $ ls -R
.:
css  html  README.md

./css:
hello.txt  README.md  style.css

./html:
index.html
letonio.silva@BRRIOLN043879:~/Documentos/exemplo $
```

A próxima imagem mostra o diretório **html** sendo copiado para o diretório **/css** de forma recursiva e renomeando-o no destino.

```
letonio.silva@BRRIOLN043879:~/Documentos/exemplo $ cp -r html ./css/re-html
letonio.silva@BRRIOLN043879:~/Documentos/exemplo $ ls -R
.:
css  html  README.md

./css:
hello.txt  re-html  style.css

./css/re-html:
index.html

./html:
index.html
letonio.silva@BRRIOLN043879:~/Documentos/exemplo $
```

## Comando which

O comando **which** é usado para mostrar o caminho completo de um executável no sistema. Ele ajuda a identificar a localização exata de um programa ou comando que será executado quando você digita seu nome no terminal. A sintaxe básica é:

**which <nome-do-executavel>**

Por exemplo, podemos ver o caminho completo para o comando ls utilizando:

**which ls**

```
letonio.silva@BRRIOLN043879:~/Documentos/alphaedtech/examples $ which ls
/usr/bin/ls
letonio.silva@BRRIOLN043879:~/Documentos/alphaedtech/examples $
```

## Estrutura de pastas do Linux

Abra o terminal e aplique os comandos `cd /` e `ls`. Deve aparecer no terminal algo parecido com isso:

```
letonio.silva@BRRIOLN043879:~/Documentos/exemplo $ cd /
letonio.silva@BRRIOLN043879:/ $ ls
bin  dev  home  lib32  libx32  lost+found  mnt  proc  run  snap  swap.img  tmp  var
boot  etc  lib  lib64  log  media  opt  root  sbin  srv  sys  usr
```

Vamos aprender um pouco sobre alguns desses diretórios, pois a estrutura de diretórios do Linux é muito bem definida.

Para manter a organização, desenvolvedores de distribuições Linux e softwares diversos seguem o **Filesystem Hierarchy Standard** (padrão para sistema de arquivos hierárquico, em português), ou **FHS**, uma espécie de referência que padroniza quais pastas do sistema recebem determinados tipos de arquivo.

### Diretório raiz: /

Todos os elementos do sistema Linux presentes no computador têm sua origem em um único ponto: o diretório raiz. Mesmo se estiverem armazenados em dispositivos físicos distintos, o acesso a eles é possível a partir do diretório raiz, simbolizado pela barra (/).

### Binários executáveis: /bin

Os binários executáveis essenciais, acessíveis a todos os usuários do sistema, encontram-se no diretório /bin. Esses executáveis consistem em comandos fundamentais para manipulação de arquivos, textos e operações básicas de rede, incluindo, mas não se limitando a, cp, mv, ping e grep.

### Binários do sistema: /sbin

Similar ao diretório /bin, esta pasta aloja executáveis, com uma distinção crucial: trata-se de aplicativos destinados a administradores de sistema para executar funções de manutenção e tarefas correlatas. Dentre os comandos disponíveis, incluem-se o ifconfig, utilizado para configurar e gerenciar interfaces de rede TCP/IP, e o fdisk, que possibilita a partição de discos rígidos, entre outras funcionalidades.

### Programas diversos: /usr

Caso não localize um comando nos diretórios /bin ou /sbin, é altamente provável que esteja presente no diretório /usr. O /usr abriga executáveis, bibliotecas e até mesmo documentação de softwares utilizados por usuários e administradores do sistema. Adicionalmente, ao compilar e instalar um programa a partir do código-fonte, a instalação ocorrerá neste diretório.

### Configurações do sistema: /etc

Os arquivos de configuração, utilizados por diversos softwares, e scripts especiais para inicializar ou encerrar módulos e programas estão armazenados no diretório /etc. O arquivo resolv.conf, que contém uma lista de servidores DNS acessíveis pelo sistema juntamente com seus parâmetros correspondentes, é um exemplo de componente encontrado nesse diretório.

## Bibliotecas: /lib

Nesta parte do sistema de arquivos, encontram-se as bibliotecas utilizadas pelos comandos presentes em /bin e /sbin. Geralmente, os arquivos de bibliotecas têm início com os prefixos ld ou lib e possuem uma "extensão" so.

## Opcionais: /opt

No diretório /opt são armazenados aplicativos adicionais que não são cruciais para o funcionamento básico do sistema. Um exemplo de aplicativo nesse diretório pode ser um software de terceiros, como um ambiente de desenvolvimento integrado (IDE) ou uma ferramenta especializada que não faz parte do conjunto fundamental do sistema operacional.

## Arquivos pessoais: /home

Os arquivos pessoais, como documentos e fotografias, são armazenados no diretório /home, organizados em pastas com os nomes dos respectivos usuários. Por exemplo, **/home/joao/Documentos**, é o caminho absoluto até o diretório de documentos do usuário **joao**. É importante observar que o diretório pessoal do administrador não está localizado no mesmo diretório, mas sim em /root.

## Inicialização: /boot

Arquivos relacionados à inicialização do sistema, ou seja, o processo de boot do Linux, quando o computador é ligado, ficam em /boot.

## Volumes: /mnt

A fim de acessar os arquivos de um CD, pendrive ou disco rígido localizado em outra máquina da rede, é necessário "montar" esse conteúdo no sistema de arquivos local, tornando-o acessível como se fosse apenas mais um diretório no sistema.

## Mídias: /media

No diretório /media, são montadas todas as mídias removíveis, como dispositivos USB e DVDs de dados. Enquanto isso, o diretório /mnt é reservado para administradores que necessitam temporariamente montar um sistema de arquivos externo.

## Serviços: /srv

Dados de servidores e serviços em execução no computador ficam armazenados dentro desse diretório.

## Arquivos de dispositivos: /dev

No universo do Linux, toda informação é representada na forma de arquivos. Ao conectar um pendrive ao computador, por exemplo, um arquivo será gerado no diretório /dev, agindo como uma interface para acessar ou gerenciar o dispositivo USB. Dentro desse diretório, são encontrados caminhos análogos para interagir com terminais e diversos dispositivos conectados ao computador, como o mouse e até mesmo modems.



## Arquivos variáveis: /var

Arquivos que crescem em tamanho ao longo do tempo são armazenados no diretório de arquivos variáveis. Um exemplo ilustrativo são os logs do sistema, registros textuais que documentam atividades no Linux, como os registros de logins ao longo dos meses.

## Processos do sistema: /proc

Recorda-se da ideia de que tudo no Linux funciona como um arquivo? O diretório /proc é a confirmação disso. Nele, são encontrados arquivos que fornecem informações sobre os recursos e processos em execução no sistema. Quer um exemplo? Para verificar o tempo decorrido desde a última inicialização do Linux, basta consultar o arquivo /proc/uptime.

## Arquivos temporários: /tmp

Este diretório é destinado a armazenar temporariamente arquivos e diretórios criados tanto pelo sistema quanto pelos usuários. Muitos desses itens são excluídos sempre que o computador é reiniciado.

# Monitoramento de recursos

O monitoramento de recursos no sistema operacional Linux é uma prática essencial para administradores de sistemas e desenvolvedores, visando garantir o desempenho adequado, a estabilidade e a eficiência dos servidores e máquinas. Diversas ferramentas estão disponíveis para acompanhar diferentes aspectos, como o uso de CPU, memória, espaço em disco, tráfego de rede e outros indicadores vitais do sistema.

## Comando top

O comando top exibe informações em tempo real sobre o uso do sistema e os processos em execução. Ele fornece uma lista dinâmica de processos e atualiza as informações regularmente. Para iniciar o top, basta digitar top no terminal e pressionar Enter.

**top**

```
top - 20:11:20 up 1:44, 1 user, load average: 1,27, 1,33, 1,00
Tasks: 401 total, 2 running, 399 sleeping, 0 stopped, 0 zombie
%Cpu(s): 4,1 us, 1,9 sy, 0,0 ni, 93,9 id, 0,1 wa, 0,0 hi, 0,0 si, 0,0 st
MiB Mem : 15674,4 total, 4840,4 free, 5202,3 used, 5631,7 buff/cache
MiB Swap: 4096,0 total, 4096,0 free, 0,0 used. 8913,4 avail Mem
```

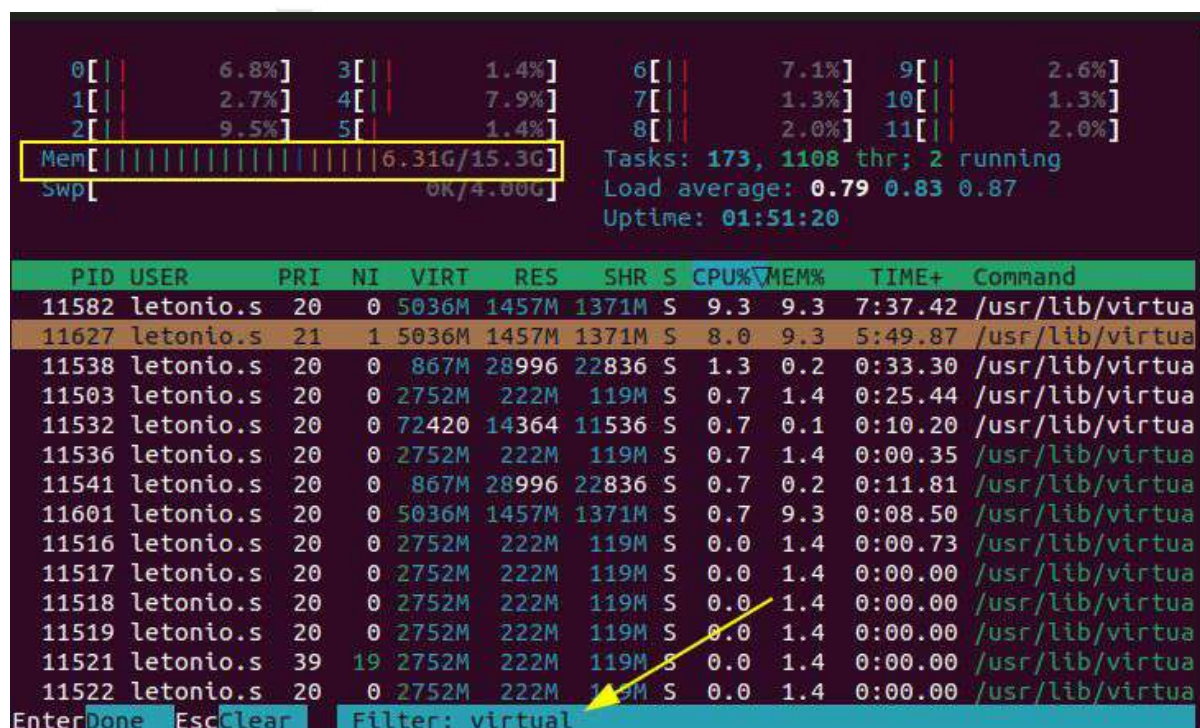
PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
5413	letonio+	20	0	12,2g	590816	221580	S	18,9	3,7	14:22.11	firefox
5958	letonio+	20	0	3674452	807352	139716	R	15,3	5,0	9:24.21	Isolated Web Co
4190	letonio+	20	0	6468908	287280	122532	S	12,3	1,8	5:32.13	gnome-shell
3948	letonio+	20	0	2063268	136260	90032	S	10,6	0,8	4:57.14	Xorg
11582	letonio+	20	0	5157780	1,4g	1,3g	S	8,0	9,3	7:01.35	VirtualBoxVM
10678	letonio+	20	0	894724	58192	41860	S	3,0	0,4	0:12.62	gnome-terminal-
1311	root	20	0	0	0	0	S	0,7	0,0	0:15.47	kcs-evdefer/0
1323	root	20	0	0	0	0	S	0,7	0,0	0:18.61	kcs-file/0
1324	root	20	0	0	0	0	S	0,7	0,0	0:18.50	kcs-file/1
1642	root	20	0	1949376	43728	31404	S	0,7	0,3	0:12.19	containerd
2456	root	20	0	3255700	88580	56232	S	0,7	0,6	0:09.90	dockerd
11503	letonio+	20	0	2818812	227692	122556	S	0,7	1,4	0:23.30	VirtualBox
11538	letonio+	20	0	888728	28996	22836	S	0,7	0,2	0:30.13	VBoxSVC
12100	root	20	0	0	0	0	I	0,7	0,0	0:01.76	kworker/u24:9-kcryptd/+
14	root	20	0	0	0	0	I	0,3	0,0	0:09.18	rcu_sched
1217	root	20	0	1958376	28512	8796	S	0,3	0,2	0:16.81	falcon-sensor
1227	custo...	20	0	14822	5824	5128	S	0,3	0,0	0:11.05	custo...

## Comando htop

O comando **htop** é uma versão aprimorada do comando **top**. Ele fornece uma visão mais amigável e interativa das informações do sistema. Caso a sua distribuição não venha com esse recurso, pode ser instalado através desse comando:

**sudo apt install htop**

Digite **htop** no terminal e pressione Enter para iniciar a interface interativa. Você pode usar F4 para filtrar por palavra chave.



The screenshot shows the htop interface in a terminal window. At the top, there are CPU usage bars for each core (0-11) and system statistics: Tasks: 173, 1108 thr; 2 running; Load average: 0.79 0.83 0.87; Uptime: 01:51:20. The memory section shows Mem: 6.31G/15.3G and Swap: 0K/4.00G. Below this is a table of running processes. The first two processes are highlighted in orange. A yellow arrow points to the 'Filter: virtual' bar at the bottom.

PID	USER	PRI	NI	VIRT	RES	SHR	S	CPU%	MEM%	TIME+	Command
11582	letonio.s	20	0	5036M	1457M	1371M	S	9.3	9.3	7:37.42	/usr/lib/virtua
11627	letonio.s	21	1	5036M	1457M	1371M	S	8.0	9.3	5:49.87	/usr/lib/virtua
11538	letonio.s	20	0	867M	28996	22836	S	1.3	0.2	0:33.30	/usr/lib/virtua
11503	letonio.s	20	0	2752M	222M	119M	S	0.7	1.4	0:25.44	/usr/lib/virtua
11532	letonio.s	20	0	72420	14364	11536	S	0.7	0.1	0:10.20	/usr/lib/virtua
11536	letonio.s	20	0	2752M	222M	119M	S	0.7	1.4	0:00.35	/usr/lib/virtua
11541	letonio.s	20	0	867M	28996	22836	S	0.7	0.2	0:11.81	/usr/lib/virtua
11601	letonio.s	20	0	5036M	1457M	1371M	S	0.7	9.3	0:08.50	/usr/lib/virtua
11516	letonio.s	20	0	2752M	222M	119M	S	0.0	1.4	0:00.73	/usr/lib/virtua
11517	letonio.s	20	0	2752M	222M	119M	S	0.0	1.4	0:00.00	/usr/lib/virtua
11518	letonio.s	20	0	2752M	222M	119M	S	0.0	1.4	0:00.00	/usr/lib/virtua
11519	letonio.s	20	0	2752M	222M	119M	S	0.0	1.4	0:00.00	/usr/lib/virtua
11521	letonio.s	39	19	2752M	222M	119M	S	0.0	1.4	0:00.00	/usr/lib/virtua
11522	letonio.s	20	0	2752M	222M	119M	S	0.0	1.4	0:00.00	/usr/lib/virtua

Enter Done Esc Clear Filter: virtual

Dessa forma, aprendemos sobre os principais que compõe o Linux. Vimos: o sistema de arquivos; como navegar entre os arquivos e diretórios, além de manipulá-los; e os processos que estão em execução no dispositivo.