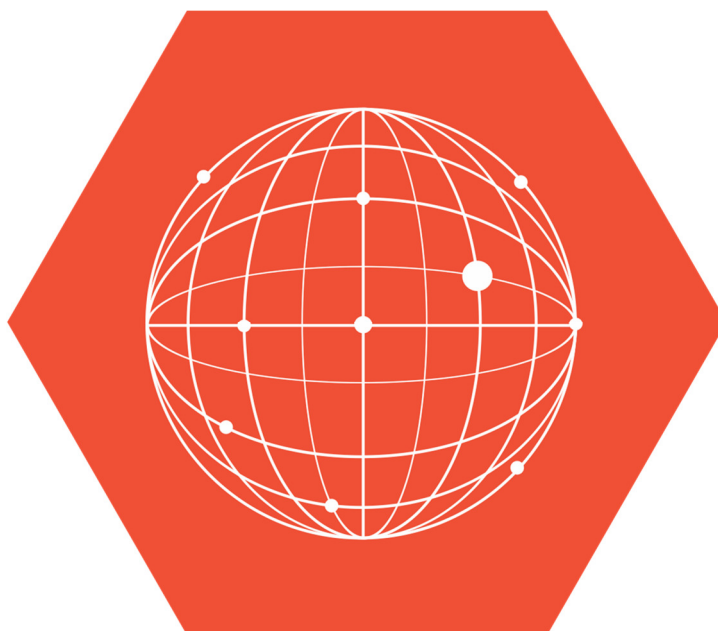




## Unidade 01

### Sistemas Operacionais Livres Software Livre



## Módulo 4

**Disciplina:** Plataforma de Desenvolvimento em Software Livre e Servidores Web

*Prof. Juvêncio Geraldo de Moura*

*Versão 1.2 – março/2021*



## 1 Sistemas Operacionais Livre

Um sistema operacional é um software que gerencia o hardware do computador, bem como fornece um ambiente para programas aplicativos serem executados. Talvez o aspecto mais visível de um sistema operacional seja a interface com o sistema de computação que ele fornece ao usuário humano.

Para que um computador realize seu trabalho de execução de programas, os programas devem estar na memória principal. A memória principal é a única grande área de armazenamento que o processador pode acessar diretamente. Ela é um *array* de bytes, variando em tamanho de milhões a bilhões. Cada byte na memória tem seu próprio



endereço. Normalmente, a memória principal é um dispositivo de armazenamento volátil que perde seu conteúdo quando a energia é desligada ou perdida. A maioria dos sistemas de computação fornece memória secundária como uma extensão da memória principal. A memória secundária fornece um tipo de armazenamento não volátil que pode manter grandes quantidades de dados permanentemente. O dispositivo de memória secundária mais comum é o disco magnético, que fornece armazenamento tanto de programas quanto de dados.

Pode-se dizer também que um sistema operacional é um conjunto de rotinas executado pelo processador, de forma semelhante aos programas dos usuários. A principal função é controlar o funcionamento de um computador, gerenciando a utilização e o compartilhamento dos seus diversos recursos, como processadores, memórias e dispositivos de entrada e saída.

Sem o sistema operacional, um usuário para interagir com o computador deveria conhecer profundamente diversos detalhes sobre hardware do equipamento, o que tornaria seu trabalho lento e com grandes possibilidades de erros. O sistema operacional tem como objetivo funcionar como uma interface entre o usuário e o computador, tornando sua utilização mais simples, rápida e segura.

A grande diferença entre um sistema operacional e aplicações convencionais é a maneira como suas rotinas são executadas em função do tempo. Um sistema operacional não é executado de forma linear como na maioria das aplicações, com início, meio e fim. Suas rotinas são executadas concorrentemente em função de eventos assíncronos, ou seja, eventos que podem ocorrer a qualquer momento.

O nome sistema operacional, apesar de ser o mais empregado atualmente, não é o único para designar esse conjunto de rotinas. Denominações como monitor, executivo, supervisor ou controlador possuem, normalmente, o mesmo significado.

Um sistema operacional possui inúmeras funções, mas antes de começar o estudo dos conceitos e dos seus principais componentes é importante saber primeiramente quais são suas funções básicas. Nesta introdução, as funções de um sistema operacional são resumidas em duas, descritas a seguir:



## Facilidade de acesso aos recursos do sistema:

Para a maioria dos usuários, uma operação como a leitura de um arquivo em disco pode parecer simples. Na realidade, existe um conjunto de rotinas específicas, controladas pelo sistema operacional, responsável pelo acionamento do mecanismo de leitura e gravação da unidade de disco, posicionamento na trilha e setor corretos, transferência dos dados para a memória e, finalmente, informar ao programa a conclusão da operação. Cabe, então, ao sistema operacional servir de interface entre os usuários e os recursos disponíveis no sistema computacional, tornando esta comunicação transparente, além de permitir um trabalho mais eficiente e com menores chances de erros. Este conceito de ambiente simulado, criado pelo sistema operacional, é denominado máquina virtual e está presente na maioria dos sistemas modernos (Figura 1).

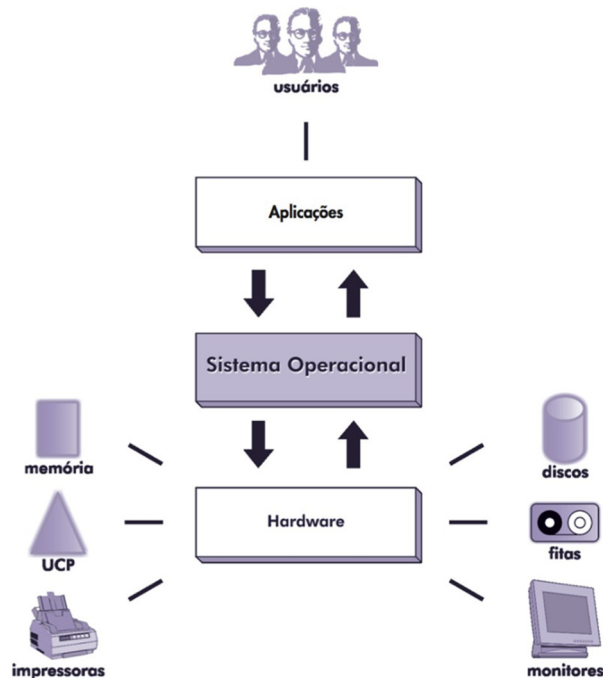


Figura 1 - Visão do sistema operacional



## Compartilhamento de recursos de forma organizada e protegida

Em sistemas onde diversos usuários compartilham recursos do sistema computacional, é necessário controlar o uso concorrente desses recursos. Se imaginarmos uma impressora sendo compartilhada, deverá existir algum tipo de controle para que a impressão de um usuário não interfira nas dos demais. Novamente é o sistema operacional que tem a responsabilidade de permitir o acesso concorrente a esse e a outros recursos de forma organizada e protegida.

Não é apenas em sistema multiusuário que o sistema operacional é importante. Se pensarmos que um computador pessoal nos permite executar diversas tarefas ao mesmo tempo, como imprimir um documento, copiar um arquivo pela Internet ou processar uma planilha, o sistema operacional deve ser capaz de controlar a execução concorrente de todas essas atividades.

O sistema operacional é formado por um conjunto de rotinas que oferece serviços aos usuários e às suas aplicações. Esse conjunto de rotinas é denominado núcleo do sistema, ou *kernel*. A maioria dos sistemas operacionais é fornecida acompanhada de utilitários e linguagem de comandos, que são ferramentas de apoio ao usuário, porém não são parte do núcleo do sistema.

Existem três maneiras distintas de os usuários se comunicarem com o kernel do sistema operacional. Uma delas é por intermédio das chamadas rotinas do sistema realizadas por aplicações. Além disso, os usuários podem interagir com o núcleo mais amigavelmente por meio de utilitários ou linguagem de comandos. Cada sistema operacional oferece seus próprios utilitários, como compiladores e editores de texto. A linguagem de comandos também é particular de cada sistema, com estruturas e sintaxe próprias.

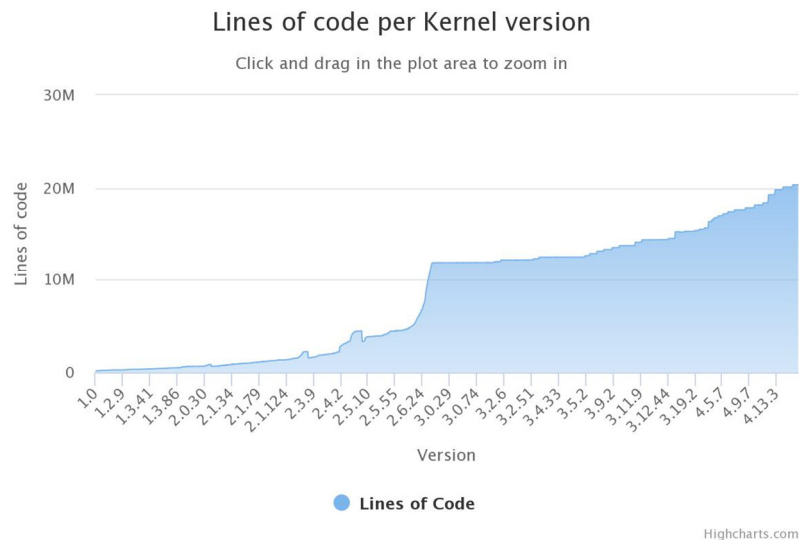


Figura 2 - Linhas de código por versão do Kernel.

Os primeiros sistemas operacionais foram desenvolvidos integralmente em assembly, e o código possuía cerca de um milhão de instruções (IBM OS/360). Com a evolução dos sistemas e o aumento do número de linhas de código para algo perto de 20 milhões (MULTICS), técnicas de programação modular foram incorporadas ao projeto, além de linguagens de alto nível, como PL/I e Algol. Nos sistemas operacionais atuais, o número de linhas de código pode chegar a mais de 40 milhões (Windows 2000). O kernel do Linux 4.15.6 possui mais de 20,3 milhões de linhas de códigos (Figura 2).

Veja mais em curiosidades em [www.linuxcounter.net/statistics/kernel](http://www.linuxcounter.net/statistics/kernel).

O sistema GNU/Linux é um sistema livre que pode ser facilmente obtido na Internet. O Linux é um sistema UNIX-like e emprega muitas soluções comuns a outros sistemas tipo UNIX. Executa em várias plataformas de hardware e é compatível com a maioria dos softwares UNIX existentes. Ele difere da maioria das outras variantes do UNIX pelo fato de ser gratuito, com código-fonte aberto e desenvolvido de maneira cooperativa, com colaborações provenientes de milhares de indivíduos e organizações.



Figura 3 - Linus Torvalds

## 1.1 Um pouco de história

O Linux se originou em 1991 como um projeto de Linus Torvalds (Figura 3), um universitário finlandês. Ele desenvolveu o projeto como uma ramificação do Minix, um sistema operacional modelo escrito por Andrew S. Tanenbaum. Entretanto, o Linux gerou interesse substancial no mundo todo, e o kernel logo ganhou vida própria. Explorando o poder do desenvolvimento cooperativo, Linus foi capaz de empreender uma tarefa muito mais ambiciosa. A versão 1.0 do kernel foi lançada em 1994; a versão estável mais recente do *kernel* do Linux é a 5.0.2 (lançada em 13/03/2019).

O *minix* é um sistema simples, baseado em UNIX, desenvolvido por Andrew Tanenbaum e sua equipe para fins didáticos e amplamente difundido no meio acadêmico no início dos anos 90.

Após a conclusão de uma versão inicial de seu novo sistema operacional, Linus divulgou-o na lista de discussão do próprio *minix*, disponibilizando-o para que outros estudantes o utilizassem como fonte de estudo. Rapidamente, outras pessoas aderiram a esse projeto, oferecendo sua contribuição pessoal para o desenvolvimento de novas funcionalidades, depuração, etc. Essa iniciativa tornou-se o fenômeno Linux que todos conhecemos atualmente. Hoje em dia, milhares de programadores auxiliam a desenvolver não somente o núcleo do Linux, mas também ferramentas que facilitam sua utilização e divulgação, assim como os mais variados programas de sistemas. Linus Torvalds ainda



participa da evolução do Linux como uma espécie de gerente geral de projetos da parte relacionada ao núcleo. Assim surgiu o Linux (Linus' UNIX).

## 1.2 Distribuições Linux

O Linux é um clone do UNIX, derivado em grande parte da família BSD. Técnica-mente, o nome Linux faz referência apenas ao núcleo do sistema operacional, mas, popularmente, este nome designa o sistema operacional como um todo, ou seja, o núcleo, os programas de sistema e aplicativos. Esse conjunto de software é, na sua grande maioria, um software livre oriundo das mais diferentes partes do mundo. Para fazer referência ao sistema operacional completo, o correto seria dizer GNU/Linux para destacar o software (GNU) e o núcleo (Linux), no entanto, as pessoas em geral falam apenas Linux. O conjunto completo de software, GNU e outros, mais o núcleo Linux, constituem o que se chama de **distribuição Linux ou apenas distro**. A maioria das distribuições são gratuitas, já outras podem ser adquiridas, como algumas versões do Red Hat que chegam na casa dos US\$15.000 (Red Hat Enterprise Linux Server For IBM System Z).

FreeBSD, NetBSD e OpenBSD, todos os quais são ramificações do Berkley Software Distribution da UC Berkley, possuem seus próprios e ardorosos seguidores. Esses sistemas operacionais são comparáveis ao Linux em termos de recursos e confiabilidade, embora eles tenham um pouco menos de suporte por parte de fornecedores de software independentes.

Todas as distribuições **oferecem uma versão do núcleo Linux, um conjunto de software e uma interface de instalação**. A diferença entre elas reside justamente nesses dois últimos pontos. O conjunto de software varia de distribuição para distribuição. Elas propõem ainda diferentes métodos de instalação e manutenção do sistema. A escolha de uma distribuição depende basicamente do gosto pessoal e da “intimidade” que o usuário tem com os comandos do sistema UNIX, necessidade de uso, hardware disponível, estabilidade daquela distro, suporte disponível, entre outros. Durante muito tempo, as distribuições Debian e SuSe eram tidas como as distribuições para “iniciantes” em UNIX,





ao passo que a distribuição Red Hat era associada a usuários mais leigos.

Apesar do conjunto de software ser diferente de uma distribuição para outra, existe uma base comum bastante grande. Essa base corresponde às ferramentas desenvolvidas dentro do projeto GNU (*GNU is Not Unix*) da *Free Software Foundation*. Entre essas, pode-se citar o compilador C (GCC), o editor de textos Emacs, o editor gráfico GIMP, o ambiente de janelas GNOME, etc. A lista de softwares GNU é imensa.

Outro ponto em relação às distribuições GNU/Linux é a questão comercial. Se o GNU/Linux é um sistema livre, por que nós compramos uma distribuição? Na realidade, o que se paga, no momento da compra de uma distribuição, não é o GNU/Linux em si, mas sim uma série de serviços que são disponibilizados pela distribuição.

As distribuições variam em seu foco, suporte e popularidade. Existem várias distribuições como, por exemplo, RedHat, Debian, Ubuntu, Slackware, Suse e Mandriva. Dentre as distribuições com mais destaque em 2018 temos: Linux Mint, Ubuntu, Deepin OS, Kali Linux, Steam OS, Elementary OS, Tails Linux, Cent OS, Fedora, Solus OS, Manjaro OS.



Figura 4 - Red Hat Linux.

O Red Hat Enterprise Linux (Figura 4) é uma base estável e comprovada, versátil o suficiente para implantar novos aplicativos, virtualizar ambientes e criar uma cloud híbrida segura, além de contar com o respaldo do nosso suporte premiado. Esses são apenas alguns dos motivos pelo qual 90% das empresas da Fortune Global 500 escolheram a

Red Hat (RED HAT ENTERPRISE LINUX, 2018).

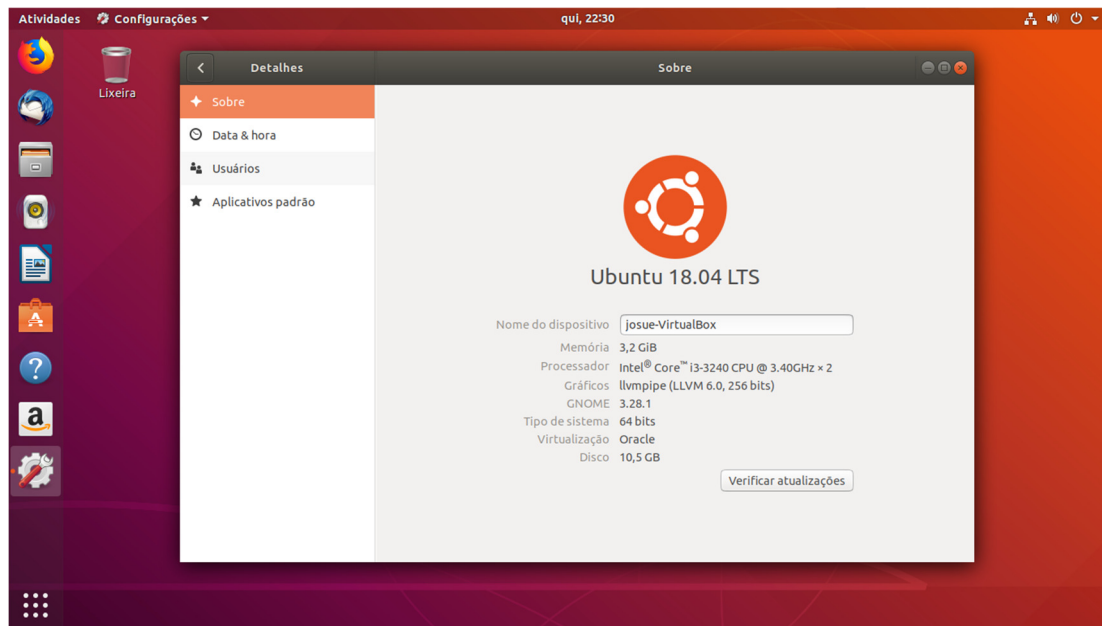


Figura 6 - Tela inicial do Ubuntu 18.04.

Figura 5 - Debian com a interface gráfica XFCE4.

As distribuições Debian (Figura 5) e Ubuntu (Figura 6) mantêm um compromisso ideológico com o desenvolvimento da comunidade e abre acesso, para que nunca existam questões sobre que partes da distribuição são gratuitas ou redistribuíveis. O Debian é mantido pelo Projeto Debian e recebe apoio de outros indivíduos e organizações de todo mundo. É formado por voluntários, e estão geralmente procurando por novos desenvolvedores que tenham algum conhecimento técnico, interesse em software livre e algum tempo livre (veja mais em <https://www.debian.org/devel/>). Já o Ubuntu é uma distribuição Linux produzida pela empresa africana Canonical. Ele é um sistema operacional completo que pode ser instalado em computadores PC e Mac. O sistema é um dos Linux mais populares da atualidade. Por padrão, o Ubuntu usa a interface gráfica (ou ambiente) Unity, formada basicamente por uma barra de tarefas que fica no lado esquerdo da tela e um painel na parte superior.

A maioria das distribuições do Linux pode fazer praticamente tudo o que você



imagina querer fazer com um sistema operacional Linux. Algumas delas podem exigir a instalação de software adicional para serem totalmente funcionais e outras podem facilitar determinadas tarefas. Podem também apresentar melhor desempenho para determinada tarefa ou ser polivalente.

Agora, qual é a distro ideal para desenvolvedores que precisam configurar o Ruby ou NodeJS. Qual é a distribuição Linux que irá executar em um melhor ambiente virtual ou em um hardware antigo? Todas estas, e muitas outras questões, são difíceis de responder. **A resposta que vem logo a mente é que a melhor distribuição Linux sempre é aquela que atenderá suas reais necessidades.** Contudo, exige tempo e dedicação testar e verificar qual distribuição Linux, realmente, atenderá suas necessidades. Portanto, para auxiliar você a escolher a melhor distro; indico a leitura de uma matéria exclusiva sobre esse assunto, disponibilizada no site do Linux Descomplicado (veja no quadro a seguir o endereço e acesse), para você conhecer as principais distribuições Linux que são recomendadas para desenvolvedores *frontend* e *backend*. Ficou curioso(a)? Acesse lá!

**Quais são as distribuições Linux  
recomendadas para desenvolvedores frontend e backend?**

Veja nesse link:

<https://www.linuxdescomplicado.com.br/2016/08/quais-sao-as-distribuicoes-linux-recomendadas-para-desenvolvedores-frontend-e-backend.html>.

Até o final da disciplina iremos conhecer algumas dessas distros e ferramentas para auxiliar o desenvolvimento de aplicações web. É provável que algum estudante desse curso tenha conhecido algumas dessas distribuições. Mas, iremos instalar, configurar e testar tanto uma distro quanto ferramentas, ainda nessa disciplina. Entretanto não se preocupe, você receberá as devidas orientações.

Ao estudar e conhecer mais sobre o Linux e sua história, você irá perceber que o sistema operacional foi desenvolvido originalmente para uso em servidores, compartilhados por muitos usuários. Porém, hoje ele é muito usado em desktop, PDA's, smartphones, dispositivos embarcados, veículos e vários outros tipos de dispositivos, sem, entretanto, abandonar suas origens.



## CURIOSIDADE

O nome do mascote Tux (o pinguim do Linux), foi escolhido por votação, pelo próprio Linus. Segundo ele "gostaria de um pinguim cheio, satisfeito por ter comido muitos peixes". Diz a lenda que o motivo por trás da escolha do Tux como mascote foi uma mordida que Linus levou de um pinguim em um zoológico.

Um servidor é uma máquina que fica o tempo todo ligada, sempre fazendo a mesma coisa. Existem vários tipos de servidores, como servidores web, servidores de arquivos, servidores de impressão, etc., sendo que uma única máquina pode rodar simultaneamente vários serviços, dependendo apenas dos recursos de hardware e da carga de trabalho. A palavra "serviço" indica, nesse caso, um aplicativo destinado a responder requisições dos clientes, como no caso do Apache (servidor web) ou do Samba (servidor de arquivos).

Até o final da disciplina, iremos conhecer um pouco mais sobre os principais serviços utilizados para aplicações Web, instalação e configuração.

[illegible]



kernel tem um mantenedor que monitora seu desenvolvimento.

Os vários programadores, trabalhando para manter o kernel, estão em uma hierarquia destacada.

Programadores experientes, que contribuíram mais e se tornaram voluntários para essa responsabilidade são designados para trabalhos mais importantes. A autoridade máxima sobre todos esses programadores é o mantenedor oficial da árvore.

O mantenedor oficial da árvore é como o diretor de uma peça – ele designa tarefas para os outros programadores, dando, às vezes, exemplos de direção que desejam que o código tome.

Eles trabalham para fazer o Kernel atingir sua visão do que ele deveria ser.

Pode então aparecer a seguinte dúvida: como um mantenedor de árvore decide sobre a direção do Kernel; de onde vem sua visão? Dos usuários, é a resposta. Os usuários pedem por funções, reportam erros que interferem em seu trabalho e geralmente provêm um retorno sobre o que o Kernel está fazendo para eles.

Baseados nisso e em motivação pessoal, os mantenedores decidem em um conjunto de objetivos concretos para o Kernel. Essa é sua visão. Por exemplo, Linus Torvalds foi para versão 2.4.x, Alan Cox para a 2.2.x e David Weinehall para a 2.0.x.

A direção do desenvolvimento do Kernel não é estática através da vida de um braço específico do código. Quando as necessidades dos usuários mudam, os objetivos para a árvore mudam. A discussão dessas mudanças se dá na lista de e-mail do Kernel Linux. A lista recebe quase todas as discussões públicas entre os vários desenvolvedores assim como relatórios de erros gerais, pedidos de funções e patches.

Apenas alguns dias após o lançamento da série de kernel 5.0 do Linux, o desenvolvedor e mantenedor do kernel Linux, Greg Kroah-Hartman, anunciou a disponibilidade da primeira versão pontual, o kernel Linux 5.0.2.

O kernel Linux 5.0 foi lançado no domingo, 03 de março de 2018, por Linus Torvalds, e atualmente é a série de kernel mais avançada disponível para sistemas operacionais baseados em Linux.

Todos os fornecedores de SO Linux agora são encorajados a adotar a mais recente série de kernel do Linux 5.0 para seus sistemas operacionais em arquiteturas suportadas,



pois ele traz vários novos recursos, melhorias e drivers atualizados para melhor suporte de hardware.

Com o lançamento do Kernel Linux 5.0, veio algumas novidades que já estavam sendo preparadas e que agora chegaram em seu mainline. Algumas novidades trazidas foram:

- Suporte para o AMD Radeon FreeSync;
- Suporte para a nova VegaM;
- Suporte para o NVIDIA Xavier
- Melhoramento nos gráficos do Intel Icelake Gen11
- Suporte inicial para os SoCs NXP i.MX8;
- Suporte para Allwinner T3, Qualcomm QCS404 e NXP Layerscape LX2160A;
- Intel VT-d Scalable Mode com suporte para o Scalable I/O Virtualization;
- Novos drivers Intel Stratix 10 FPGA;
- Correções para F2FS, EXT4 e XFS;
- Btrfs file-system com suporte de restauração dos arquivos de swap;
- AgFscrypt Adiantum da Google agora é suportado com ajuda a criptografia rápida de dados em hardware low-end. Isso substitui o algoritmo Speck pela NSA;
- Melhorias no driver Realtek R8169;
- Suporte de alta resolução para rolagens da Logitech;
- Driver para tela sensível ao toque de Raspberry Pi;
- Melhoria aos drivers de notebooks com arquitetura x86;
- Aprimoramento de segurança para o Thunderbolt;
- Suporte para a placa Chameleon96 Intel FPGA;
- Melhor gerenciamento de energia;

No comunicado, Linus Torvalds disse que está contente com o lançamento e que a próxima janela de desenvolvimento está aberta, para a versão 5.1, e que já tem várias solicitações chegando para analisar e processar. Mas o que chamou a atenção, foi essa



declaração no final do comunicado na lista de discussão do projeto, em que ele diz o seguinte:

*“As mudanças gerais para todas as versões do “5.0” são muito maiores. Mas, eu gostaria de ressaltar (mais uma vez) que não fazemos lançamentos baseados em recursos, e que o “5.0” não significa nada mais do que isso. Os números para a série 4.x estavam ficando grandes o suficiente para que eu ficasse sem dedos na mão e dos pés para contar.”*  
(DIOLINUX, 2019)

A versão mais atual do Kernel Linux é a 5.11, lançada no dia 14/02/2021. Segundo Alecrim (2021), apesar de ter poucas novidades, traz um conjunto de pequenos aprimoramentos que a tornam bastante relevante, como suporte nativo ao Wi-Fi 6E, às GPU's Nvidia RTX 3000 e aos gráficos Intel Iris Xe.

## Teste os conhecimentos adquiridos!

O kernel é o componente principal que faz parte da distribuição Linux.  
Defina os termos sublinhados.

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---





## 2. Software Livre

Software Livre (Free Software) é o software disponível com a permissão para qualquer um usá-lo, copiá-lo, e distribuí-lo, seja na sua forma original ou com modificações, seja gratuitamente ou com custo. A possibilidade de modificações implica em que o código fonte esteja disponível. Se um programa é livre, potencialmente ele pode ser incluído em um sistema operacional também livre.

### Software Livre != Software Grátis

É importante não confundir software livre com software grátis porque a liberdade associada ao software livre de copiar, modificar e redistribuir, independe de gratuidade. Existem programas que podem ser obtidos gratuitamente mas que não podem ser modificados, nem redistribuídos.

#### ALGUNS CONCEITOS

**Software Livre:** é o software que pode ser livremente copiado e que possui código-fonte disponível para quem quiser vê-lo e alterá-lo.

**Open source (código aberto):** é aquele no qual o usuário pode modificar o código, de acordo com o que deseja usar. Porém, o desenvolvedor original do programa determina as condições de uso e de distribuição. Por exigir mais conhecimentos técnicos, os códigos abertos são geralmente manipulados por programadores

**Freeware:** é o software que é gratuito apenas, ou seja, embora não se pague nada para usá-lo, não se tem acesso ao código-fonte e não pode ser alterado.

**Shareware:** é uma modalidade de distribuição e comercialização de software, onde o software pode ser livremente copiado, mas ele funciona em um modo “demonstração” por determinado período de tempo.

O movimento do software livre criou milhares de projetos de código-fonte aberto, inclusive sistemas operacionais. Graças a esses projetos, os estudantes podem usar o



código-fonte como ferramenta de aprendizado. Eles podem modificar programas e testá-los, ajudar a encontrar e corrigir bugs, ou então explorar sistemas operacionais maduros e completos, compiladores, ferramentas, interfaces de usuário e outros tipos de programas.

## 2.1 As quatro liberdades permitidas pelo software livre

Por “software livre” devemos entender aquele software que respeita a liberdade e senso de comunidade dos usuários. Grosso modo, isso significa que os usuários possuem a liberdade de executar, copiar, distribuir, estudar, mudar e melhorar o software. Assim sendo, “software livre” é uma questão de liberdade, não de preço. (GNU, 2018)

Com essas liberdades, os usuários (tanto individualmente quanto coletivamente) controlam o programa e o que ele faz por eles. Quando os usuários não controlam o programa, o programa controla os usuários. O desenvolvedor controla o programa e, por meio dele, controla os usuários. Esse programa não livre é “proprietário”. (GNU, 2018)

Um programa é software livre se os usuários possuem as quatro liberdades essenciais:

- **Liberdade 0:** A liberdade de executar o programa como você desejar, para qualquer propósito.
- **Liberdade 1:** A liberdade de estudar como o programa funciona, e adaptá-lo às suas necessidades. Para tanto, acesso ao código-fonte é um pré-requisito.
- **Liberdade 2:** A liberdade de redistribuir cópias de modo que você possa ajudar outros.
- **Liberdade 3:** A liberdade de distribuir cópias de suas versões modificadas a outro. Desta forma, você pode dar a toda comunidade a chance de beneficiar de suas mudanças. Para tanto, acesso ao código-fonte é um pré-requisito.

Um programa é software livre se ele dá aos usuários todas essas liberdades de forma adequada. Do contrário, ele é não livre. Em qualquer cenário, essas liberdades devem ser aplicadas em qualquer código do qual planejamos fazer uso, ou que levamos outros a fazer uso. Por exemplo, considere um programa A que automaticamente inicia um programa B



para lidar com alguns casos. Se nós planejamos distribuir A como está, isso significa que usuários precisarão de B, de forma que nós precisamos julgar se tanto A quanto B são livres. Porém, se nós planejamos modificar A de forma que ele não use B, apenas A precisa ser livre; B não é pertinente àquele planejamento.

“Software livre” não significa “não comercial”. Um programa livre deve estar disponível para uso comercial, desenvolvimento comercial e distribuição comercial. Desenvolvimento comercial de software livre deixou de ser incomum; tais software livre comerciais são muito importantes. Você pode ter pago dinheiro por suas cópias de software livre, ou você pode tê-las obtido a custo zero, mas independentemente de como você conseguiu suas cópias, você sempre deve ter a liberdade para copiar e mudar o software, ou mesmo para vender cópias. (GNU, 2018)

## **A liberdade de executar o programa como você desejar**

A liberdade de executar o programa significa que qualquer tipo de pessoa ou organização é livre para usá-lo em qualquer tipo de sistema computacional, ou para qualquer tipo de trabalho e propósito, sem que seja necessário comunicar ao desenvolvedor ou qualquer outra entidade específica.

Nessa liberdade, é o propósito do usuário que importa, não aquele do desenvolvedor; você, como usuário, é livre para rodar o programa para seus propósitos e, caso você o distribua a outra pessoa, ela também será livre para executá-lo com os propósitos dela, mas você não é intitulado a impor seus propósitos sobre ela.

A liberdade de executar o programa como você deseja significa que você não está proibido ou impedido de executá-lo. Isso não tem nada a ver com qual funcionalidade o programa possui, se ele é tecnicamente capaz de funcionar em qualquer ambiente dado ou se ele é útil para alguma atividade computacional específica.

## **A liberdade de estudar o código-fonte e fazer alterações**

Para que as liberdades 1 e 3 (a liberdade de modificar e a liberdade de publicar versões modificadas) façam sentido, você deve ter acesso ao código-fonte do programa. Consequentemente, acesso ao código-fonte é uma condição necessária para o software



livre. Código-fonte “obscurecido” não é código-fonte real e não conta como código-fonte.

A liberdade 1 inclui a liberdade de usar sua versão modificada em lugar da original. Se um programa é entregue num produto projetado para rodar a versão de outra pessoa, mas se recusa a rodar a sua — prática conhecida como “travamento” ou ainda (na terminologia perversa de seus praticantes) como “boot seguro” — a liberdade 1 se torna pretensão vazia ao invés de realidade prática. Esses binários não são software livre mesmo que o código-fonte a partir do qual foram compilados seja livre.

Uma maneira importante de modificar um programa é agregar a ele módulos e sub-rotinas livres. Se a licença do programa diz que você não pode agregar a ele um módulo com uma licença adequada — por exemplo, se ele requer que você seja o detentor dos direitos autorais de qualquer código que adicionar — então essa licença é muito restritiva para ser qualificada como livre.

Se uma modificação constitui ou não um aperfeiçoamento é uma questão subjetiva. Se o seu direito de modificar um programa é limitado, fundamentalmente, a mudanças que outra pessoa considere um aperfeiçoamento, o programa não é livre.

## **A liberdade de redistribuir se assim desejar: requisitos básicos**

Liberdade para distribuir (liberdades 2 e 3) significam que você é livre para redistribuir cópias, modificadas ou não, gratuitamente ou cobrando uma taxa pela distribuição, a qualquer um, em qualquer lugar. Ser livre para fazer tudo isso significa (entre outras coisas) que você não deve ter que pedir ou pagar pela permissão para fazê-lo.

Você também deve ter a liberdade de fazer modificações e usá-las privativamente ou em seu trabalho ou lazer, sem sequer mencionar que eles existem. Se publicar suas modificações, você não deve ser obrigado a avisar ninguém em particular, ou de qualquer modo em particular.

A liberdade 3 inclui a liberdade de publicar quaisquer versões modificadas como software livre. Uma licença livre também pode permitir outras maneiras de liberá-las; em outras palavras, ela não tem que ser uma licença *copyleft*. No entanto, a licença que requer que modificações sejam não livres não se qualifica como uma licença livre.

A liberdade de redistribuir cópias deve incluir formas executáveis ou binárias do



programa, bem como o código-fonte, tanto da versão modificada quanto da inalterada. (Distribuir programas em formato executável é necessário para sistemas operacionais livres e convenientemente instaláveis.) Não há problemas se não for possível produzir uma forma binária ou executável (pois algumas linguagens de programação não suportam este recurso), mas deve ser concedida a liberdade de se redistribuir nessas formas caso seja desenvolvido um meio de criá-las.

O GNU/Linux e o BSD UNIX são sistemas operacionais de código-fonte aberto. As vantagens do software livre e do código-fonte aberto devem aumentar a quantidade e a qualidade de projetos de código-fonte aberto, levando a um acréscimo no número de indivíduos e empresas que usam esses projetos.

A Figura 7 apresenta algumas distribuições GNU/Linux que podem ser instaladas no










Distribuição	Descrição breve
	Dragora GNU/Linux-Libre, uma distribuição de GNU/Linux independente baseada em conceitos de simplicidade.
	Dyne:bolic, uma distribuição GNU/Linux com ênfase especial na edição de áudio e vídeo. Essa é uma distro "estática", normalmente executada a partir de um "live CD". Como ele não receberá atualizações de segurança, ele deve ser usado desconectado.
	gNewSense, uma distribuição de GNU/Linux baseada no Debian, patrocinada pela FSF.
	Guix System Distribution é uma distro GNU/Linux avançada construída sobre o GNU Guix (pronunciado "geeks"), um gerenciador de pacotes puramente funcional para o sistema GNU.
	Musix, uma distribuição GNU+Linux com ênfase especial na produção de áudio. Essa é uma distro "estática", normalmente executada a partir de um "live CD". Como ele não receberá atualizações de segurança, ele deve ser usado desconectado.
	Parabola GNU/Linux-libre, uma distribuição baseada no Arch que prioriza o fácil gerenciamento de pacotes e do sistema.
	PureOS, uma distribuição de GNU baseada no Debian com um foco na privacidade, segurança e conveniência.
	Trisquel, uma distribuição de GNU/Linux baseada no Ubuntu que é destinada a pequenas empresas, usuários domésticos e centros educacionais.
	Ututo S, uma distribuição 100% livre de GNU/Linux. Foi o primeiro sistema GNU/Linux totalmente livre reconhecido pelo Projeto GNU.

Figura 7- Distribuições apresentadas no site do Projeto GNU.



disco rígido do computador e/ou podem ser executadas para conhecer sem ter a necessidade de instalar no computador.

## 2.2 Projeto GNU

GNU (*GNU is not UNIX*) é um sistema operacional que é software livre – que é, o respeito a liberdade de seus usuários. O sistema operacional GNU consiste em pacotes GNU (programas especificamente lançados pelo Projeto GNU) bem como software livre lançado por terceiros. O desenvolvimento do GNU tornou possível o uso de um computador sem um software que ameace sua liberdade.

GNU é o nome de um SO completo e compatível com o UNIX escrito a partir de 1983 por Richard Stallman (Fundador da *Free Software Foundation*) e inúmeros hackers da comunidade de software livre espalhados pela Internet. A licença GPL (GNU Public License) fornece as 4 liberdades básicas do software livre: A liberdade de executar o programa como você desejar, para qualquer propósito (liberdade 0); A liberdade de estudar como o programa funciona, e adaptá-lo às suas necessidades (liberdade 1). Para tanto, acesso ao código-fonte é um pré-requisito; A liberdade de redistribuir cópias de modo que você possa ajudar outros (liberdade 2); A liberdade de distribuir cópias de suas versões modificadas a outros (liberdade 3). Desta forma, você pode dar a toda comunidade a chance de beneficiar de suas mudanças. Para tanto, acesso ao código-fonte é um pré-requisito.





Figura 8 - Richard Stallman em visita ao Brasil em 2017.

O objetivo principal e contínuo do GNU é oferecer um sistema compatível com o Unix que seria 100% software livre. Não 95% livre, não 99,5%, mas 100%. O nome do sistema, GNU, é um acrônimo recursivo que significa “*GNU's Not Unix*” (em português, “GNU Não é Unix”) — uma maneira de homenagear as ideias técnicas do Unix, enquanto ao mesmo tempo diz que o GNU é algo diferente. Tecnicamente, o GNU é como o Unix. Mas, ao contrário do Unix, o GNU dá liberdade a seus usuários.

Milhares de pessoas se juntaram para tornar o GNU o sucesso que é hoje, e há muitas formas de contribuir, tanto técnicas como não técnicas. Os desenvolvedores do GNU se reúnem de tempos em tempos em GNU *Hackers Meetings* (“Reuniões de Hackers do GNU”), às vezes como parte das conferências do LibrePlanet, a maior comunidade de software livre.

O GNU sido apoiado de várias maneiras pela *Free Software Foundation* (FSF), a organização sem fins lucrativos também fundada pelo Stallman para defender ideais de software livre. Entre outras coisas, a FSF aceita atribuições de copyrights e isenções de responsabilidade, para que possa atuar no tribunal em nome dos programas GNU. (Para ser claro, contribuir com um programa para o GNU, não exige a transferência do copyright para a FSF. Se você atribuir o *copyright*, a FSF aplicará a GPL para o programa se alguém violá-la; se você mantiver o *copyright*, a aplicação dependerá de você.)



O objetivo final é fornecer software livre para fazer todos os trabalhos que os usuários de computadores desejam fazer — e assim fazer o software proprietário uma coisa do passado.

## 2.3 Tipos de Licenças

Ao desenvolver um software, o desenvolvedor ou empresa proprietária do mesmo pode decidir torná-lo livre. Para tanto, é necessário estabelecer como ele deverá ser distribuído, ou seja, como as pessoas que adquirirem o produto devem fazer uso do mesmo. Este tipo de informação é definido através da licença escolhida.

Hoje, existe inúmeras licenças para a distribuição de softwares livres, porém elas são classificadas em **licenças permissivas** e **licenças recíprocas totais** e **licenças recíprocas parciais**, voltadas para a redistribuição de um trabalho e pela criação de um novo produto através da derivação de um trabalho. (Sabino, 2011)

Para fazê-lo software livre, você precisa lançá-lo sob uma licença de software livre. Normalmente é usada a Licença Pública Geral GNU (GNU GPL), especificando a versão 3 ou qualquer outra versão posterior, mas ocasionalmente pode-se usar outras licenças de software livre.

A documentação para software livre deve ser documentação livre, então as pessoas podem redistribuí-la e melhorá-la como o software é descrito. Para elaborar uma documentação livre, você precisa lançá-la sob uma licença de documentação livre. Para tanto, existe a Licença de Documentação Livre GNU (GNU FDL), mas ocasionalmente pode-se registrar o documento com outras licenças de documentação livre.

### 2.3.1 Licenças permissivas

É um tipo de licença em que se pretende atingir um grande número de pessoas. É conhecida como licença acadêmica, uma vez que esse tipo de licença impõe poucas restrições para o uso de um software e de seu código, permitindo que seja desenvolvido um produto derivado do código e esse pode até ter seu código fechado e comercializado.

As principais licenças deste grupo são: BSD, MIT e Apache.





## **BSD (*Berkeley Software Distribution*)**

A licença BSD é uma licença de código aberto inicialmente utilizada nos sistemas operacionais. Tal licença impõe poucas restrições quando comparada aquelas impostas por outras licenças, o que a aproxima do domínio público. O texto da licença é considerado como de domínio público e pode ser modificado sem nenhuma restrição, mas nesse caso deve ser informado o nome do indivíduo ou organização que realizou a modificação. A licença BSD permite que o software distribuído sob a licença, seja incorporado a produtos proprietários. Também é possível que softwares sejam distribuídos pela licença BSD junto de outra licença.

## **Licença MIT (criada pelo Massachusetts Institute of Technology)**

A licença MIT permite que o software seja tratado sem restrições para o uso, modificação e distribuição. Desta forma, pode ser utilizada tanto em projetos de software livre, quanto em projeto de software proprietário. No texto desta licença não existe copyright, desta forma outros grupos podem modificar a licença, com o objetivo de atender as suas necessidades.

## **Licença Apache 2.0**

Não compensa o esforço para usar *copyleft* para a maioria dos programas pequenos. É usado 300 linhas como marca de referência: quando o código-fonte do pacote de um software é menor do que isso, os benefícios fornecidos pelo *copyleft* é geralmente pequeno demais para justificar a inconveniência de se certificar que uma cópia de licença sempre acompanhe o software.

Para aqueles programas, é recomendado a **Licença Apache 2.0**. Esta é uma licença de software permissiva (não *copyleft*) que possui termos que evitam contribuidores e distribuidores de processar por violação de patente. Isso não torna o software imune a ameaças de patentes (uma licença de software não pode fazer isso), mas evita detentores de patentes prepararem uma "armadilha" na qual eles lançam um software sob termos livres



e, então, exigem que o destinatário concorde com termos não livres em uma licença de patente.

A *Apache License* é uma licença para software livre de autoria da *Apache Software Foundation*. Tal licença permite ser usada em qualquer projeto, desde que sejam obedecidos os termos e condições contidos em seu texto. Ela permite o uso e distribuição do código-fonte tanto no software livre, quanto no proprietário. Entretanto, exige a inclusão do aviso de *copyright* (direito autoral) e do termo de responsabilidade (informa os direitos do leitor e as responsabilidades assumidas e não assumidas pelo autor) no produto.

### 2.3.2 Licenças Recíprocas Totais e Parciais

As Licenças Recíprocas Totais têm como condição que todo software livre deve se manter livre, portanto, todo trabalho gerado a partir de um software sob uma licença recíproca deve ser redistribuído e disponibilizado sob os mesmos termos da licença original, que é conhecida como *copyleft*, ou seja, impede que sejam acrescentadas restrições em cima de versões derivadas. A principal licença desta categoria é a GPL.

As Licenças Recíprocas Parciais têm os mesmos princípios das recíprocas totais, entretanto, se as modificações forem utilizadas como componente de outro projeto de software, este projeto não precisa, necessariamente, ser disponibilizado sob a mesma licença. As principais licenças desta categoria são a LGPL e a MPL.

### Licença Pública Geral GNU

A Licença Pública Geral GNU (*GNU General Public License*) é frequentemente chamada abreviadamente de GNU GPL; ela é utilizada pela maioria dos programas GNU, assim como mais da metade de todos os outros programas de software livre. A versão mais recente é a versão 3.

A Licença Pública Geral GNU está disponível nesses formatos: HTML, texto plano, ODF, Docbook v4 ou v5, Texinfo, LaTeX, Markdown, e RTF. Esses documentos não são formatados para serem publicados isoladamente, e a intenção é que sejam incluídos em outro documento.



## **Licença Pública Geral Menor GNU**

A Licença Pública Geral Menor GNU (*GNU Lesser General Public License*) é utilizada em algumas (mas não todas) as bibliotecas GNU. A versão mais recente é a versão 3.

O texto da Licença Pública Geral Menor GNU está disponível nesses formatos: HTML, texto plano, Docbook, Texinfo, Markdown, ODF, e RTF. Esses documentos não são formatados para serem publicados isoladamente, e a intenção é que sejam incluídos em outro documento.

## **Licença Pública Geral Affero GNU**

A Licença Pública Geral Affero GNU (*GNU Affero General Public License*) é baseada na GNU GPL, mas tem um termo adicional que permite os usuários que interagir com o software licenciado em uma rede receber o fonte daquele programa. Nós recomendamos que as pessoas considere usar a GNU AGPL para qualquer software que irá comumente ser executada em rede. A última versão é a 3.

O texto da Licença Pública Geral Affero GNU está disponível nesses formatos: HTML, texto plano, Docbook, Texinfo, LaTeX, Markdown, ODF, e RTF. Esses documentos não são formatados para serem publicados isoladamente, e a intenção é que sejam incluídos em outro documento.

## **Licença de Documentação Livre GNU**

A Licença de Documentação Livre GNU (*GNU Free Documentation License*) é uma forma de copyleft criada para uso em manuais, livros ou outros documentos para garantir que qualquer um tem a real liberdade de copiar e redistribuí-los, com ou sem modificações, tanto comercial quanto não comercialmente. A versão mais recente é versão 1.3.

O texto da Licença de Documentação Livre GNU está disponível nesses formatos: HTML, texto plano, Docbook v4 ou v5, Texinfo, LaTeX, Markdown, ODF, e RTF. Esses documentos não são formatados para serem publicados isoladamente, e a intenção é que sejam incluídos em outro documento.



## **Copyleft**

*Copyleft* é um método legal de tornar um programa em software livre e exigir que todas as versões modificadas e estendidas do programa também sejam software livre. O modo mais simples de tornar um programa em software livre é colocá-lo sob o domínio público, sem *copyright*. Isto permite às pessoas compartilharem o programa e suas melhorias, se elas assim o desejarem. Mas, isto também permite as pessoas não cooperativas converterem o programa em software proprietário. Elas podem fazer modificações, muitas ou poucas, e distribuir o resultado como um produto proprietário. As pessoas que recebem o programa nesta versão modificada não têm as liberdades que o autor original deu a elas; o intermediário as retirou.

No Projeto GNU, a intenção é dar a todos os usuários a liberdade de redistribuir e modificar o software GNU. Se intermediários pudessem retirar as liberdades, nós poderíamos ter muitos usuários, mas não teríamos liberdade. Por isso, em vez de colocar o GNU em domínio público, nós o tornamos “*copyleft*”. *Copyleft* diz que qualquer um que redistribui o software, com ou sem modificações, tem que passar adiante as liberdades de fazer novas cópias e modificá-las. **O *Copyleft* garante que todos os usuários tenham liberdade.**

*Copyleft* também fornece um incentivo para outros programadores adicionarem recursos ao software livre. Programas livres importantes como o compilador GNU C++ só existem por causa disso.

O *Copyleft* também ajuda os programadores que desejam contribuir com melhorias para o software livre a obterem permissão de fazê-lo. Esses programadores frequentemente trabalham para empresas ou universidades que fariam qualquer coisa para ganhar mais dinheiro. Um programador pode desejar contribuir com modificações para a comunidade, mas seu empregador pode desejar transformar as mudanças em um produto de software proprietário.

Para tornar um programa em *copyleft*, primeiro afirma que ele está sob *copyright*; depois adiciona termos de distribuição, que são um instrumento legal que concede a todos o direito de usar, modificar, e redistribuir o código-fonte do programa ou qualquer outro programa derivado dele, mas somente se os termos de distribuição permanecerem



inalterados. Assim, o código e as liberdades se tornam legalmente inseparáveis.

Desenvolvedores de software proprietário utilizam o *copyright* para retirar as liberdades das pessoas; o projeto GNU utiliza o *copyright* para garantir essas liberdades. É por isso que o nome foi invertido, mudando “*copyright*” (“direitos de cópia”, “cópia direita”) para “*copyleft*” (“esquerda de cópia”, “cópia esquerda”).

O *Copyleft* é um conceito genérico; existem vários modos de preencher os detalhes. No Projeto GNU, os termos específicos de distribuição que são utilizados estão contidos na Licença Pública Geral GNU, na Licença Pública Geral Menor GNU e na Licença de Documentação Livre GNU.

A GNU GPL foi desenhada de modo que ela seja facilmente aplicável ao seu próprio programa se você é o detentor do *copyright*. Você não tem que modificar a GNU GPL para fazer isso, apenas adicione notas ao seu programa que se refiram adequadamente à GPL. O texto completo da GPL deve ser utilizado. Ele é um conjunto indivisível, e cópias parciais não são permitidas. (O mesmo vale para a LGPL, AGPL e FDL).

Utilizar os mesmos termos de distribuição para vários programas diferentes torna mais fácil copiar código entre eles. Já que eles têm os mesmos termos de distribuição, não há necessidade de verificar se os termos são compatíveis. A LGPL inclui termos que deixa você alterar os termos de distribuição para a GPL original, de modo que você possa copiar código de um programa LGPL para um programa coberto pela GPL.

## **Mozilla (MPL)**

A *Mozilla Public License* é uma licença para software livre de código aberto. Esta licença define que o código-fonte copiado ou alterado sob ela deve continuar sob a mesma licença. Entretanto, é permitido que este código seja combinado em um software com arquivos proprietários. Além disso, é possível criar uma versão proprietária de um código sob a licença Mozilla. Esta licença também permite a redistribuição do código produzido, mas obriga a inclusão de citação do autor.



## RESUMINDO!

A licença de software do tipo **permissiva** (Apache, MIT/X11, BSD e assemelhadas), o software é livre, mas pode ser relicenciado sem permissão adicional do autor. As **recíprocas totais** (GPL e assemelhadas), o software é livre, deve permanecer livre e trabalhos derivados devem ser também livres. Já o tipo **recíprocas parciais** (Apache, MIT/X11, BSD e assemelhadas) o software é livre e deve permanecer livre, mas trabalhos derivados não precisam ser livres.



## **Pesquise e pense!**

**1) Existem outras licenças para software livre. Hoje encontramos diversos softwares que possuem a licença Open Source. Como é essa licença? Comente sobre ela.**

---

---

---

---

---

---

---

---

---

---

---

---

**2) Acredito que já tenha desenvolvido alguma aplicação, seja ela simples ou complexa. Já pensou em disponibilizar sob uma licença de software livre? Qual delas você escolheria? Justifique.**

---

---

---

---

---

---

---

## **2.4 Viabilidade econômica e investimento das empresas**

Se você pesquisou, pensou e refletiu no final do tópico anterior, deve ter pensado: Como vou ganhar dinheiro com o meu software livre? Claro que deve ter pensado no lado social também, certo? Mas, mesmo publicando o meu software com uma licença de software livre existem diversas maneiras para angariar recursos. Já parou para pensar ou pesquisar como isso é possível?

Quem realmente precisa de suporte pelo sistema, vai pagar por ele. Quem não



precisa vai usar de qualquer maneira, independente da licença permitir isso ou não. Publicação de livros, ganham o autor e as editoras. Diversas empresas utilizam software livre para: Gerar lucros; Reduzir custos; Aprimorar sua equipe.

O diretor-executivo da Linux Foundation, Jon “Maddog” Hall, no Fórum Internacional de Software Livre de 2012 apresentou dicas de como ganhar dinheiro usando o software livre. Muitas pessoas já se tornaram milionárias com o software livre, mas para Jon Hall, tudo depende do seu empenho: “Muitos procuram um emprego no software livre para ganhar dinheiro, mas a maioria das pessoas escolhe o caminho errado. Não existe fórmula mágica, para ganhar dinheiro com software livre é preciso muito esforço, como em qualquer trabalho.” (TechTudo, 2018)

Ele ainda citou o caso da Caixa Econômica Federal, que usava um software proprietário para gerenciar seu sistema de loterias, pagando 1 milhão de dólares por mês pela licença. Dinheiro não era problema, mas se a Caixa queria incluir um novo jogo no sistema, isto levava cerca de 10 meses. A solução foi simples, contratar três programadores que desenvolveram uma solução usando software livre, e hoje em dia é possível incluir um novo jogo em apenas 3 semanas. “Qual o valor deste software? Usando o software livre você pode ter uma solução adequada ao seu problema, que pode ser personalizada da noite para o dia para atender as necessidades do seu cliente, o que tem um valor imenso”, completou Hall. (TechTudo, 2018)

Ainda nesse evento, Hall contou o caso da ProjectDotNet, uma empresa que tinha mais lucros dando suporte e treinamentos do que vendendo seu programa. Eles disponibilizaram o aplicativo como software livre, e isto aumentou muito a quantidade de clientes, o que gerou muitos lucros em suporte, fazendo a empresa se tornar mais lucrativa do que nunca. “Esta é a promessa do software livre”, concluiu Jon “Maddog” Hall. (TechTudo, 2018)

Todos os serviços de hospedagem e data centers disponibilizam softwares de código aberto, incluindo o Azure da Microsoft. Aliás, a Microsoft aderiu a comunidade Linux em novembro de 2016. Quem diria? (Efagundes, 2018)

Para quem trabalha com Big Data, Analytics e IoT, as soluções de software livre são naturais. O Hadoop e sua família de softwares permitem análises de grandes volumes de





dados. Obviamente, se você quiser um suporte profissional será necessário contratar uma empresa que tenha profissionais qualificados para garantir a disponibilidade e add-on que melhorem o desempenho do software. (Efagundes, 2018)

As empresas estão cada vez mais utilizando softwares livres com suporte profissional. Empresas de serviços de hospedagem oferecendo plataformas com software livre com suporte profissional incluído. Startups especializadas em serviços para empresas utilizando software livre como plataforma para seus desenvolvimentos e soluções. (Efagundes, 2018) Empresas que continuarem a gastar rios de dinheiros pagando licenças de softwares proprietários tendem a perder competitividade e flexibilidade para se adaptar as novas regras de mercado.

***O software livre pode trazer benefícios para sua carreira pela consequência natural da comunidade valorizar o seu conhecimento.***



## Teste os conhecimentos adquiridos!

**1) A GPL é a licença para software livre com maior utilização. Ela se baseia em 4 liberdades. Comente sobre cada uma delas.**

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

**2) Faça uma pesquisa e explique como é realizada a escolha e definição de uma licença para um determinado software livre. Cite dois softwares livres com as suas respectivas licenças.**

---

---

---

---

---

---

---

---

---

---



### 3. Referências

ALECRIM, Emerson. Linux 5.11 é lançado com suporte ao Wi-Fi 6E e às GPUs RTX 3000. Tecnoblog. Disponível em <https://tecnoblog.net/412401/kernel-linux-5-11-suporte-wi-fi-6e-nvidia-rtx-3000-intel-iris-xe/>. Acessado em 13/03/2021.

ANUNCIAÇÃO, Heverton. Linux Total e Software Livre. Rio de Janeiro: Editora Ciência Moderna Ltda, 2007.

DIOLINUX. Disponível em <https://www.diolinux.com.br/2018/08/novidades-kernel-linux-418.html>. Acessado em 19/08/2018.

DIOLINUX. Disponível em <https://www.diolinux.com.br/2019/03/kernel-linux-50-lancado-mas-voce-realmente-precisa-atualizar.html>. Acessado em 14/03/2019.

EFAGUNDES. Disponível em <http://efagundes.com/artigos/da-para-confiar-e-ganhar-dinheiro-com-software-livre/>. Acessado em 10/09/2018.

EVANGELISTA, Rafael. O movimento software livre do Brasil: política, trabalho e hacking. Horiz. antropol. [online]. 2014, vol.20, n.41, pp.173-200. ISSN 0104-7183. <http://dx.doi.org/10.1590/S0104-71832014000100007>.

GNU. Disponível em <https://www.gnu.org/philosophy/free-sw.pt-br.html>. Acessado em 21/08/2018.

MATERA. Licença de software Livre. Disponibilizado em <http://www.matera.com/blog/post/licencas-de-software-livres>. Acessado em 21/08/2018.

NEMETH, Evi; SNYDER, Garth; HEIN, Trent R. Manual Completo do Linux: guia do administrador. São Paulo: Pearson Education, 2007.

RED HAT ENTERPRISE LINUX. Disponível em <https://www.redhat.com/pt-br/technologies/linux-platforms/enterprise-linux>. Acessado em 10/08/2018.

SABINO, V. C. Um estudo sistemático de licenças de software livre. Dissertação (Mestrado em Ciência da Computação) – Instituto de Matemática e Estatística, Universidade de São Paulo, São Paulo, 2011. Disponível em: [www.teses.usp.br/teses/disponiveis/45/45134/tde-14032012-003454/pt-br.php](http://www.teses.usp.br/teses/disponiveis/45/45134/tde-14032012-003454/pt-br.php).

SILBERSCHATZ, Abraham; GALVIN, Peter Baer; GAGNE, Greg. Fundamentos de sistemas operacionais. 9 ed. Rio de Janeiro: LTC, 2015.