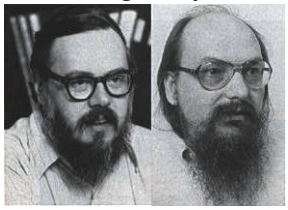
HISTÓRIA DO LINUX

MULTICS

Criado em 1964, foi o primeiro sistema operacional de tempo compartilhado da história, resultado do projeto MAC (Mathematics Computation), projeto cooperativo liderado pelo MIT junto a divisão de produtos para grandes computadores da companhia General Electric e dos Laboratórios Bell da AT&T. Ele foi concebido como um produto comercial para a GE. Os laboratórios Bell abandonaram o projeto em 1969. Em 1970 a empresa GE, incluindo o Multics, foram comprados pela Honeywell.

UNIX

Em 1969 os Laboratórios Bell da AT&T estavam retirando-se como contribuidores do sistema Multics, que depois de ter surgido como uma ótima ideia, parecia não ter um futuro promissor devido a falta de organização e de objetivos comuns entre os desenvolvedores. Com esse afastamento, vários hackers do Bell Labs que trabalhavam no Multics ficaram sem função específica. Foi então que os três programadores, Ken Thompson, Dennis Ritchie e J.F. Ossanna, tentaram reviver o espírito de comunidade do desenvolvimento e programação em grupo, espirito esse que havia sido criado pelo projeto Multics. De início trabalhando apenas no papel, os três se empenharam em desenvolver um novo sistema operacional, aproveitando as ideias de Ken Thompson, que queria criar um sistema de arquivos realmente inovador.



A Bell Labs não se entusiasmou com a ideia dos funcionários, pois o projeto Multics, que tinha sido o maior projeto de um sistema até então, saiu do controle durante o desenvolvimento. Assim, todo pedido de desenvolvimento de um SO era negado imediatamente pela diretoria.

Conformados com a ideia de que não poderiam desenvolver seu sistema operacional, os três assumiram suas funções normais em outros projetos que estavam em andamento na AT&T. Quase que por acidente, eles tiveram que trabalhar com máquinas PDP-7, que apesar de grandes e caras, eram pequenas para os padrões da época.

No seu tempo livre, Ken Thompson travalhava no PDP-7 em um sistema base. Esse sistema lhe permitiria jogar um game de viajem espacial que ele tinha começado a programar ainda na época do Multics. Como o game precisava armazenar dados do jogo, Dennis Ritchie escreveu um pequeno conjunto central de programas (que mais tarde seria conhecido como um kernel) que controlava o sistema de arquivos.

Depois de algum tempo, o sistema de Thompson e Ritchie que rodava na PDP-7 se tornou mais que um simples terminal gráfico. Com isso, finalmente a AT&T Bell Labs deu início ao projeto tão esperado. Originalmente, o projeto tinha o objetivo de desenvolver um sistema operacional multiusuário e multitarefa, que precisasse de pouca memória e que pudesse interagir com os terminais ASCII dos PDP-7. Esse instante marca a criação do Unix.

O Unix foi originalmente escrito em Assembly. Porém, após a adoção do Unix pela DEV (Digital Equipment Corporation), que queria competir com a IBM no mercado de computadores, os hackers da AT&T tiveram acesso ao PDP-11. Como o PDP-11 tinha uma arquitetura bem diferente do seu antecessor, as técnicas de programação necessárias eram outras. Portar o Unix significaria rescreve-lo. Para evitar que isso voltasse a acontecer, os programadores pensaram em uma linguagem de alto nível, para desenvolver os programas de forma algorítmica, onde os detalhes da arquitetura seriam tratados por um compilador. No início, foi usada uma variante da linguagem B, porém logo percebeu-se que os compiladores de B não assimilavam bem os endereçamentos dos bytes da PDP-11, não gerando programas com qualidade satisfatória.

A partir da experiência em redefinir a linguagem B, Thompson e seus colegas criaram outra linguagem, batizada de C. Uma evolução direta do B. Assim, em 1973, o Unix foi rescrito em C. Isso gerou uma revolução, pois até então nunca havia se cogitado a ideia de um sistema operacional totalmente portável.

Thompson então começou a enviar por correio, para todos seus amigos, algumas fitas magnéticas com os fontes do Unix e alguns utilitários. Na correspondência, ele escrevia somente: “Com amor, Ken”. Surge aqui, no início da década de 70, a cultura dos hackers do Unix, compartilhando melhorias e trabalhando no fonte do Unix da Bell Labs.

Até sua 6ª versão, o Unix foi amplamente distribuído. Para universidades a distribuição era gratuita, o que possibilitou a um professor chamado John Lion escrever livros sobre o assunto.

Lions era um professor de ciências da computação da University of New South Wales, na Austrália. Ele acreditava que poderia desenvolver os conhecimentos de programação dos seus alunos possibilitando-os o estudo dos códigos dos melhores programadores. Para isso, ele escreveu dois livros: “A commentary on the Unix

operating system level six” e “Unix operating system souce code level six”. Esses livros foram emblemáticos tanto pelo tamanho exagerado e desengonçado quanto pelas cores chamativas de suas capas.

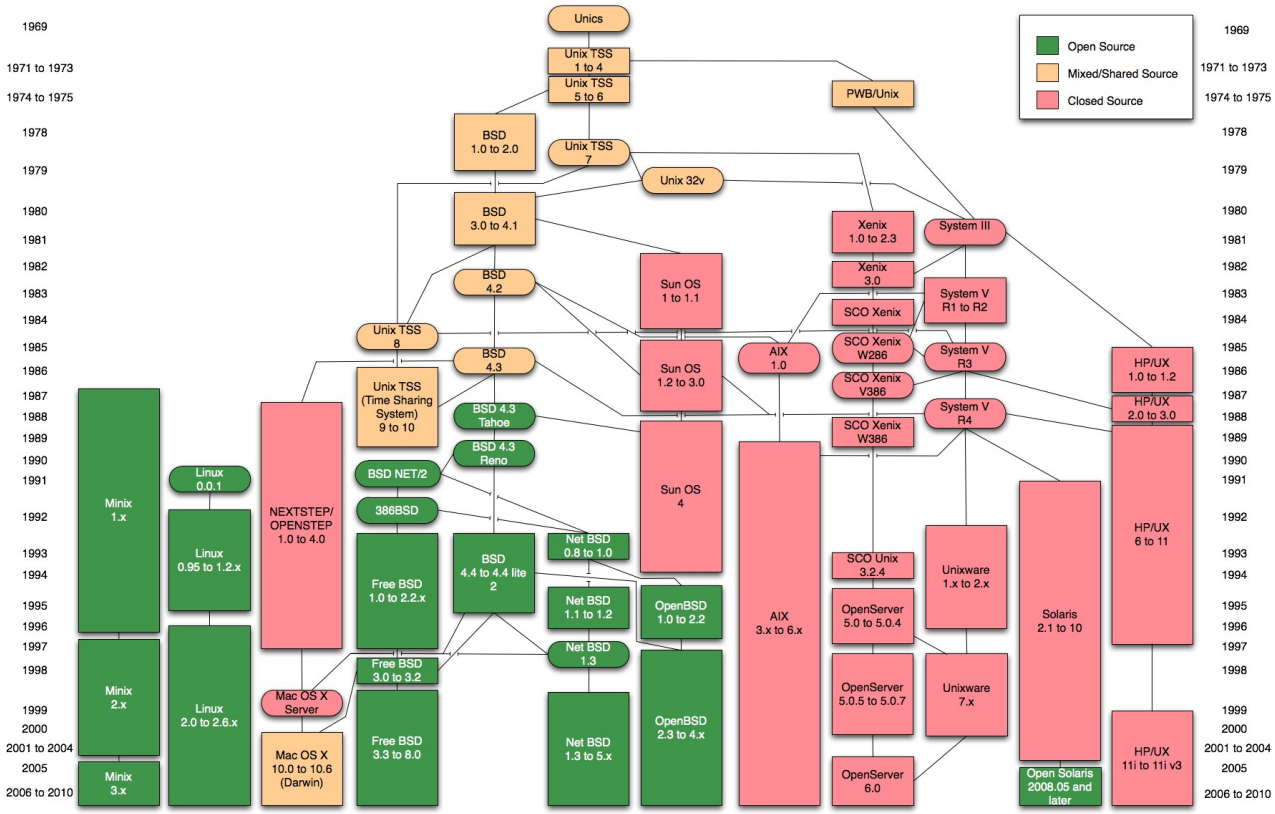
As aulas do professor Lions estavam ficando famosas. A AT&T ficou sabendo o que estava acontecendo e, através de seus advogados, suspendeu a publicação dos livros com a alegação de que eles discutiam assuntos relacionados à códigos proprietários da Bell Labs.

Os livros se tornaram raros, sendo fotocopiados a exaustão e distribuídos em segredo por mais de uma década, educando uma geração inteira de hackers de sistemas operacionais. Por não poder usar os livros legalmente nas aulas, os professores e alunos se encontravam de madrugada em salas de aula vazias para estudá-los.

A portabilidade era uma das maiores características do Unix, criando uma cultura baseada no compartilhamento do código fonte, que passava de mão em mão tendo seus erros corrigidos e funcionalidades desenvolvidas. Nesse momento, qualquer um que distribuísse sistemas binários sem os fontes seria ridicularizado nessa comunidade, acostumada com a liberdade do acesso ao código.

Em 1973, Thompson e Ritchie apresentaram o Unix no Simpósio de Princípios de Sistemas Operacionais, na Universiade de Purdue, em Indiana, nos Estados Unidos. Acadêmicos da Universidade de Berkley se interessaram em obter uma cópia do sistema para testes em sua instituição. Esse contato iniciou uma colaboração entre a AT&T e a Universidade de Berkley que durou até 1976. Depois do fim do desenvolvimento conjunto, o código que ficou em Berkley continuou a ser desenvolvido parelelamente ao que permaneceu na Bell Labs.

Em uma sucessão atordoante de fatos, passando por processos de direitos autorais, que ficaram conhecidos como “Unix War”, a AT&T e a Universidade de Berkley se enfrentaram por vários anos. Esses fatos culminaram na separação completa dos códigos gerados em cada instituição, gerando sistemas distintos em código, porém muito semelhantes em arquitetura e funcionalidades. O Unix da AT&T continuou sendo desenvolvido até 1988, quando já era chamado de Unix System V. Ele ainda deu origem a vários outros sistemas operacionais. Entre eles podemos citar o Solaris e OpenSolaris (Sun), AIX (IBM) e o HP-UX. Já os sistemas BSD (Berkley Software Distribuition) deram origem ao FreeBSD, OpenBSD, NetBSD e vários outros. A maioria gozando da livre distribuição garantida pela licença BSD. Posteriormente, o MacOSX nasceu de uma derivação do NetBSD. Mas isso já é uma outra história.



PROJETO GNU

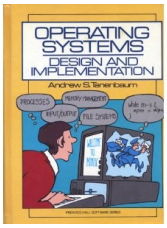
O laboratório de inteligência artificial do MIT era o campo de testes dos computadores PDP da DEC. Lá foi desenvolvido o sistema operacional ITS. Esse sistema era escrito em linguagem de baixo nível e feito sob medida para o PDP-10. Não havia a possibilidade de portabilidade para outros hardwares. Assim, seu fim chegou quando a DEC anunciou a descontinuação do PDP-10 e a adoção do Unix da AT&T como sistema operacional da PDP-11. Depois disso, os esforços concentrados no desenvolvimento de outros sistemas operacionais não portáveis não tiveram bons resultados e o MIT estava tomando outros rumos. Agora, no final da década de 70, o MIT desenvolvia tecnologias para suprir as demandas da indústria japonesa de eletrônicos. Nessa época era comum ver os projetos de domínio público do MIT se tornarem produtos proprietários no oriente. Além disso, empresas de softwares comerciais atraíram muitos dos brilhantes programadores do MIT, negociando acordos de confidencialidade para proteger seus segredos.

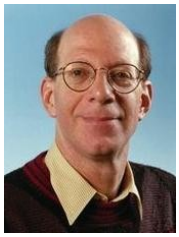
Revoltado com o comércio massivo de sua comunidade de hackers, o programador do MIT Richard Stallman resolveu mergulhar no mundo Unix. Ele possuía suas próprias fotocópias dos livros do Prof. Lions e começou a escrever um S.O. em C. O sistema deveria ser portável e ter uma licença que garantisse que ele sempre seria de propriedade de uma comunidade livre de desenvolvedores. Com a publicação do manifesto de criação do projeto GNU, que significa “GNU is Not Unix”, em setembro de 1983, Stallman iniciou um movimento para produzir e distribuir softwares que transmitia sua filosofia. Dentre os vários produtos gerados por essa iniciativa, o mais importante para a comunidade foi a General Public License. A GPL faz uso dos direitos de Copyright para manter o sistema publicamente disponível sob qualquer circunstância e mediante qualquer alteração do seu código. Basicamente, determina que todo trabalho derivado de uma obra licenciada sob GPL também torne-se GPL. Em janeiro de 1984, Richard Stallman deixou seu trabalho no MIT e passou a se dedicar ao Projeto GNU, criando em 1985 a Free Software Foundation, que é uma entidade com a missão de promover a liberdade de usuários de computadores e de defender os direitos de todos os usuários de software livre.

Para atingir o sonho de criar um sistema operacional totalmente livre, Stallman primeiro focou em desenvolver ferramentas portáveis que viabilizassem esse sistema. Considerado um feito surpreendente para um só programador, ele criou o GCC (GNU C Compiler), que é o compilador mais robusto e eficiente já criado. Além do GCC, várias outras ferramentas foram criadas, mas faltava ainda um kernel para seu sistema operacional.

Chegada a hora de ter um kernel, o Projeto GNU decidiu adotar o microkernel Mach. Depois de muitas dificuldades em adaptar esse kernel, outras opções começaram a ser avaliadas. O BSD Unix de Berkley estava travando brigas judiciais e o Unix era um sistema operacional enorme, concebido para mainframes de alto custo e pouco acessíveis. Outra opção seria usar o Minix, cujo uso deveria ser licenciado. Apesar do licenciamento ser extremamente barato para os padrões da época, o tipo de licença fugia da filosofia do projeto. O esforço foi concentrado no desenvolvimento do kernel GNU Hurd, que ficaria pronto em alguns anos. Tempo demais...

MINIX





O Minix (Mini Unix) é uma ferramenta desenvolvida pelo professor Andrew S. Tanenbaum, da Vrije Universiteit em Amsterdam, para ensinar os princípios de sistemas operacionais para seus alunos. Em 1987, Tanenbaum escreveu um livro chamado "Operating Systems Design and Implementation", um abreviado das 12.000 linhas de código do kernel, do gerenciador de memória e do sistema de arquivos do MINIX 1.0. Apesar de ter sido fortemente baseado no Unix, o Minix da época não era eficiente e sequer chegava perto do que deveria ser um sistema operacional, mas era bastante pequeno e didático, oferecendo uma bela noção das funções que um S.O. deveria desempenhar. Desde abril de 2000, o Minix, que atualmente está na versão 3, tornou-se Open Source. Adotou a licença BDS, que se fosse a licença usada na época em que o projeto GNU estava à procura de um kernel, provavelmente o Linux não seria o que conhecemos hoje. Possivelmente nem existiria ou seria apenas um simples e desconhecido projeto acadêmico.

LINUX

Aos 21 anos, Linus Torvalds cursava o 2º ano de ciências da computação da Universidade de Helsinki, na Finlândia. Ele convivia diariamente com as ineficiências do Minix. A curiosidade em sistemas operacionais e a frustração a respeito da licença do Minix, que só permitia uso educacional, impedindo que melhorias fossem redistribuídas, fizeram com que Linus resolvesse escrever seu próprio kernel. Nessa época, programadores ao redor do mundo eram inspirados pelo projeto GNU, um movimento para fornecer software livre e de qualidade. Stallman era reverenciado como um herói no mundo da informática. Nesse contexto, Linus começou a escrever seu kernel seguindo os padrões GNU, enquanto estudava e aprendia a desenvolver na linguagem C.

Em 25 de agosto de 1991, ele mandou uma mensagem para a lista de discussão do Minix, informando sobre seu projeto e solicitando sugestões de novas funcionalidades.

> ***From:*** *torvalds@klaava.Helsinki.FI (Linus Benedict Torvalds)*
> ***Newsgroups:*** *comp.os.minix*
> ***Date:*** *25 Aug 91 20:57:08 GMT*
> ***Organization:*** *University of Helsinki*
> ***Subject:*** *What would you like to see most in minix?*
>
> *Hello everybody out there using minix -*
>
> *I'm doing a (free) operating system (just a hobby, won't be big and professional like gnu) for 386(486) AT clones. This has been brewing since april, and is starting to get ready. I'd like any feedback on things people like/dislike in minix, as my OS resembles it somewhat (same physical layout of the file-system (due to practical reasons) among other things).*
> *I've currently ported bash(1.08) and gcc(1.40), and things seem to work. This implies that I'll get something practical within a few months, and I'd like to know what features most people would want. Any suggestions are welcome, but I won't promise I'll implement them :-)*
>
> *Linus (torvalds@kruuna.helsinki.fi)*

PS. Yes - it's free of any minix code, and it has a multi-threaded fs. It is NOT protable (uses 386 task switching etc), and it probably never will support anything other than AT-harddisks, as that's all I have :-(.

As sugestões foram chegando e Linus passou a dedicar grande parte do seu tempo em implementar algumas delas. Depois de pouco mais de um mês da primeira mensagem, em 5 de outubro de 1991, Linus enviou uma outra mensagem para a lista, dessa vez informando sobre a disponibilidade dos seus códigos para download, marcando o início do modelo colaborativo de desenvolvimento usado pelo Linux até os dias atuais.

From: *torvalds@klaava.Helsinki.FI (Linus Benedict Torvalds)*
Newsgroups: *comp.os.minix*
Date: *5 Oct 91 05:41:06 GMT*
Organization: *University of Helsinki*
Subject: *Free minix-like kernel sources for 386-AT*

Do you pine for the nice days of minix-1.1, when men were men and wrote their own device drivers?
Are you without a nice project and just dying to cut your teeth on a OS you can try to modify for your needs? Are you finding it frustrating when everything works on minix? No more all-nighters to get a nifty program working? Then this post might be just for you :-)

As I mentioned a month(?)ago, I'm working on a free version of a minix-lookalike for AT-386 computers. It has finally reached the stage where it's even usable (though may not be depending on what you want), and I am willing to put out the sources for wider distribution. It is just version 0.02 (+1 (very small) patch already), but I've successfully run bash/gcc/gnu-make/gnu-sed/compress etc under it.

Sources for this pet project of mine can be found at nic.funet.fi (128.214.6.100) in the directory /pub/OS/Linux.
The directory also contains some README-file and a couple of binaries to work under linux (bash, update and gcc, what more can you ask for :-). Full kernel source is provided, as no minix code has been used. Library sources are only partially free, so that cannot be distributed currently. The system is able to compile "as-is" and has been known to work.
Heh. Sources to the binaries (bash and gcc) can be found at the same place in /pub/gnu.

ALERT! WARNING! NOTE! These sources still need minix-386 to be compiled (and gcc-1.40, possibly 1.37.1, haven't tested), and you need minix to set it up if you want to run it, so it is not yet a standalone system for those of you without minix. I'm working on it. You also need to be something of a hacker to set it up (?), so for those hoping for an alternative to minix-386, please ignore me. It is currently meant for hackers interested in operating systems and 386's with access to minix.

The system needs an AT-compatible harddisk (IDE is fine) and EGA/VGA. If you are still interested, please ftp the README/RELNOTES, and/or mail me for additional info.

I can (well, almost) hear you asking yourselves "why?". Hurd will be out in a year (or two, or next month, who knows), and I've already got minix. This is a program for hackers by a hacker. I've enjouyed doing it, and somebody might enjoy looking at it and even modifying it for their own needs. It is still small enough to understand, use and modify, and I'm looking forward to any comments you might have.

I'm also interested in hearing from anybody who has written any of the utilities/library functions for minix. If your efforts are freely distributable (under copyright or even public domain), I'd like to hear from you, so I can add them to the system. I'm using Earl Chews estdio right now (thanks for a nice and working system Earl), and similar works will be very wellcome. Your (C)'s will of course be left intact. Drop me a line if you are willing to let me use your code.

Linus

PS. to PHIL NELSON! I'm unable to get through to you, and keep getting "forward error - strawberry unknown domain" or something.

Imediatamente, Linus começou a receber patches e correções que garantiam novas características ao código. Todos esses patches eram licenciados sob GPL, o que forçou o Linux a se tornar também GPL.

Inicialmente o Linux se chamava Freax (Freak Unix), porém foi rebatizado pelo administrador do FTP onde o código foi disponibilizado. Apesar de considerar um nome egocêntrico, Linus acabou consentindo a mudança. Já o Tux, o pinguim símbolo do Linux, foi escolhido por Linus devido a um incidente divertido ocorrido em uma viagem de férias na Austrália, onde um pinguim “assassino” o mordeu.

Desde o início, o kernel Linux era distribuído junto com as ferramentas GNU. Isso originou o que conhecemos hoje como o sistema operacional GNU/Linux, que creditou a Linus Torvalds tudo que Richard Stallman desejou que seu sistema GNU fosse um dia.

Essa união viabilizou um sistema operacional completo e usável. Vários grupos de usuários começaram a adicionar ferramentas, principalmente para gerenciamento de softwares, criando o que hoje conhecemos como “Distribuições Linux”. A primeira distribuição baseada no S.O. GNU/Linux foi o Slackware, criada em junho de 1993 por Peter Volkerding. Depois disso, várias outras distribuições apareceram, cada uma escrevendo sua própria página na história da computação.

LINUX DISTRO TIMELINE