

Linux研修オンライン【例題で学ぼう】

Linuxカーネルの構成要素と管理

ZEUS
enterprise

株式会社ゼウス・エンタープライズ
(LPI-Japanプラチナスポンサー・アカデミック認定校)
鯨井貴博

LPI-JAPAN

鯨井貴博(くじらいたかひろ)

- ・LinuCエバンジェリスト
- ・OSSJ(Open Source Summit Japan)運営ボランティアリーダー
- ・KubeCon/CloudNativeCon Japan運営ボランティアリーダー

2000年 大学卒業後、建設業界に就職。

2007年 IT業界に転職し、Linuxを学ぶ。

2008年 OSS界隈で勉強会などのイベントに参加し始める。

LinuC(当時はLPIC)のベータ試験に参加。

2012年 社内講師として活動を開始。

個人ブログ、メルマガ執筆、LinuCセミナー、専門学校での授業など行う。

2015年 ACCEL(Apache Cloudstack Certification Exam by LPI-japan)認定第1号。

2022年 OSSJ運営ボランティアリーダー

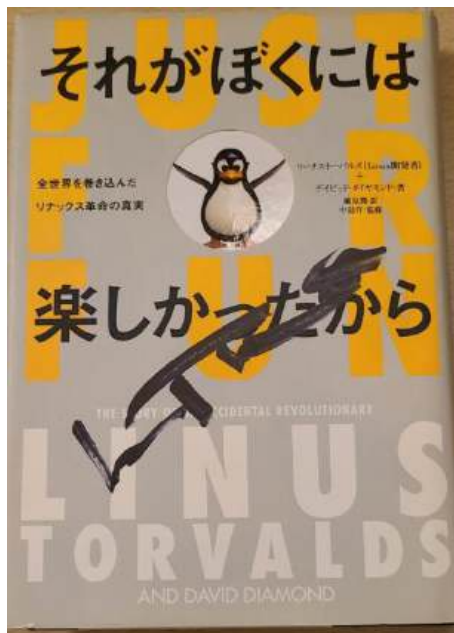
2025年 KubeCon/CloudNativeCon Japan運営ボランティアリーダー



技術を学習する上で大事にしていること

- ・その技術が生まれた背景や開発者の思いの理解
- ・実際に使ってみる

Linux開発者 Linus Torvalds



The Linux Kernel Archives

<https://www.kernel.org/category/releases.html>

LinuxカーネルメンテナGreg Kroah-Hartman



Longterm release kernels

Version	Maintainer	Released	Projected EOL
6.12	Greg Kroah-Hartman & Sasha Levin	2024-11-17	Dec, 2026
6.6	Greg Kroah-Hartman & Sasha Levin	2023-10-29	Dec, 2026
6.1	Greg Kroah-Hartman & Sasha Levin	2022-12-11	Dec, 2027
5.15	Greg Kroah-Hartman & Sasha Levin	2021-10-31	Dec, 2026
5.10	Greg Kroah-Hartman & Sasha Levin	2020-12-13	Dec, 2026
5.4	Greg Kroah-Hartman & Sasha Levin	2019-11-24	Dec, 2025

Open Source Summit Japan

2013年 初参加(Attendee)

2014年 ボランティアメンバーとして運営を手伝う

2022年 ボランティアリーダーとして運営を行う

2025/12/8-10(虎ノ門)のボランティアメンバーは、
絶賛募集中♪

※気になる方は、以下から応募ください！

11/4 23:59まで



Open Source Summit Japan

<https://events.linuxfoundation.org/open-source-summit-japan/>

世界と繋がる、グローバルカンファレンスの舞台裏！ Open Source Summit Japan2024運営ボランティアの体験レポート Part1/Part2

<https://thinkit.co.jp/article/37762>

<https://thinkit.co.jp/article/37777>

KubeCon/CloudNativeCon Japan

2025年 日本初開催

※2026年は7月29/30日 横浜開催



KubeCon/CloudNativeCon Japan2025

<https://www.cncf.io/reports/kubecon-cloudnativecon-japan-2025/>

株式会社ゼウス・エンタープライズ

LINUX・ネットワークなどのスキルを学ぶ、ITスクール。

ZEUS IT TRAINING CENTER
 ゼウスITトレーニングセンター

はじめての方へ

コース紹介

法人様向け

受講者の声

アクセス

資料請求



ネットワークエンジニアを目指したい方、
 サーバやネットワークインフラに興味のある方におすすめ。

Linux & ネットワーク講座

ネットワークエンジニアを目指したい方、サーバやネットワークインフラに興味のある方におすすめの講座一覧です。コマにわかれたコースと連続受講のコースがあります。

講座詳細へ



Kubernetesに関するスキルを身に付けたい方におすすめ。

Kubernetes講座

Kubernetesクラスターを構築・管理するためのコースです。
 また、Kubernetesを管理するために必要なスキルを身に付けることができます。

講座詳細へ



教育案件や人材サービスを探している方、
 ご連絡お待ちしております！

<https://www.zeus-enterprise.co.jp/solution/service/>
<https://www.it-training.tokyo/>

高水準エンジニアによる支援サービス

Network Engineering Service

高い専門スキルを有するエンジニア集団だから可能な質の高いソリューション

ゼウス・エンタープライズは、時代変革の要となるネットワーク・セキュリティ分野に特化したエンジニア集団として、顧客のニーズや課題に迅速かつ確実に応え、満足度の高いIT支援サービスを提供しています。情報通信・官公庁・金融・製造などの様々なクライアント先にてTCP/IPスタックの機器や、Linuxにおける豊富な経験と高度な技術を活用し、ネットワークやセキュリティ分野のパフォーマンスを最大限に引き出します。

主な業務としては、小規模LANから大規模WANまでのネットワーク構築や運用支援。各種アプリケーションの実行基盤やデータベースなど業務サーバーの構築や運用支援。また、オンプレ環境やクラウド環境、ハイブリッドクラウドの環境においても、セキュリティを重視した構築や運用支援を提供しています。

そして、当社は活躍する社員一人ひとりの能力を昇華させるべく、「ゼウスITトレーニングセンター」という教育機関を併設しており、ネットワークやLinuxを中心に、時代のトレンドに沿ったインフラ教育を行っています。

日々変革を遂げるIT業界に伴い、研修にて社員のスキルを底上げし、ネットワーク・セキュリティに特化したスペシャリスト集団として、クライアントの課題解決に貢献いたします。



アジェンダ

- LinuC、およびその学習方法
- 例題と解説
- Linuxカーネルの構成要素と管理について
- 振り返り

本日のゴール

- Linuxカーネル構成要素と管理の理解
- LinuCレベル2資格対策の理解

LinuC、およびその学習方法

■LinuCとは

クラウド／DX時代のITエンジニアに求められるシステム構築から運用管理に必要なスキルを証明する技術者認定です。

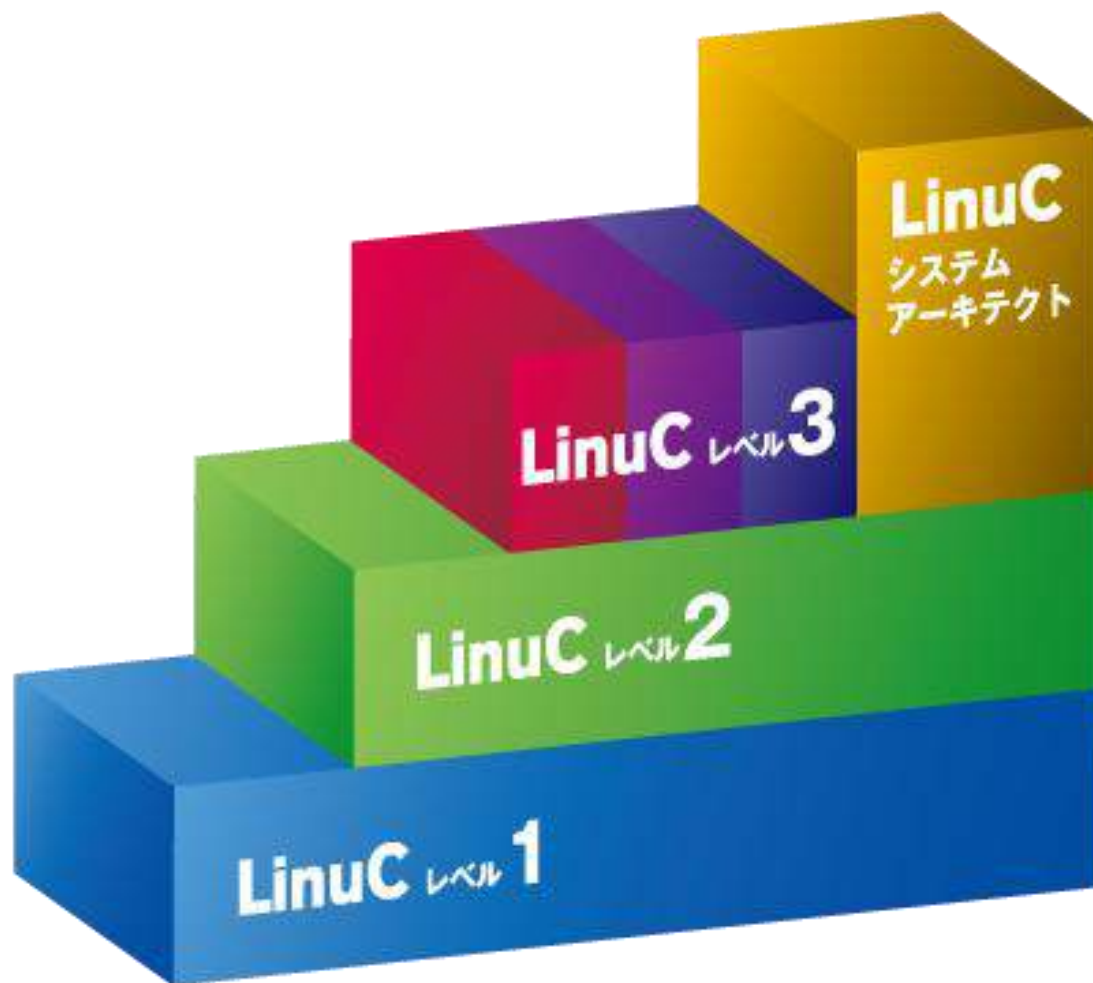
- ✓ クラウド活用に役立つスキルの習得
 - ・ オンプレミス／仮想化・コンテナを問わず様々な環境下でのサーバー構築
 - ・ 他社とのコラボレーションの前提となるオープンソースへの理解
- ✓ 習得できるスキルが実践的

問題作成にはトップエンジニアも参加するコミュニティ内の意見を取り込むことで、本当に必要な内容を網羅的に盛り込んでいます。
- ✓ 上流工程を担うアーキテクトの領域までカバー

システムの運用管理からアーキテクチャ設計までの4つのレベルをひとつずつ習得していくことで、活躍できるエンジニアとして必要なスキルを網羅的に身につけていくことができます



LinuCは、サーバーの運用管理からアーキテクト設計まで、システム開発・運用に必要な知識とスキルを体系立てて習得することができます。



LinuC システムアーキテクト

ITプロジェクトを成功に導く上級エンジニア

SA01試験

SA02試験

LinuC レベル3

高度な技術力を備えた特定分野のスペシャリスト

304試験 (仮想化&高可用性)

300試験
(混在環境)

303試験
(セキュリティ)

LinuC レベル2

仮想マシン・コンテナを含むLinuxシステム、ネットワークの設計・構築

201試験

202試験

LinuC レベル1

物理/仮想Linuxサーバーの操作・運用

101試験

102試験

学習の具体的な進め方(2~3か月程度)

試験範囲の確認(LinuxC HP)

12月頃、受験しようという目標を立てる



LinuxC認定教材の購入・1週目読込

<https://lpi.or.jp/linuc1/book.shtml>
<https://lpi.or.jp/linuc2/book.shtml>



LinuxC認定教材・LinuxC技術解説動画を参考に、実機操作(サーバ構築やコマンド操作)を試す
 ※操作やトラブルシュートで力が身に付く！

<https://linuc.org/measures/movie/>



LinuxC認定教材 2週目読込

12月23日頃、受験しようと決める



問題集やメルマガサンプル問題で理解力確認
 ※理解不足箇所の洗い出し

<https://linuc.org/study/samples/>



LinuxC認定教材 3週目
 ※弱点の補強



受験申込

受験日の変更も可能なので安心



問題を8~9割以上、正解となるまで繰り返し解く
 苦手な部分を重点的に復習



受験

- ・受験まで継続して学習すること
- ・繰り返し学習し、理解度/問題正解率を高めた状態で受験すること

- ① 出題範囲の内容について調べてみる
公式ドキュメント・技術書など
- ② 実際に操作してみる
これが大事！
- ③ 学習の補助教材などを利用する
 - ・メールマガジン
 - ・標準教科書
 - ・過去のセミナー資料
 詳細は、<https://lpi.or.jp/learning/>



メールマガジンでコツコツと

学習に役立つメールマガジン

LPI-Japanでは、試験レベルごとの例題解説など、
学習に役立つメールマガジンを無料でお届けしています。

過去のメールマガジンの
例題解説をまとめています。

LPI-Japanでは、試験レベルごとの例題解説など、
学習に役立つメールマガジンを無料でお届けしています。

LPI-Japanが開発した
大人気の教科書で
Linuxを効率的に学ぶ

- Linux 標準教科書
- Linux サーバー標準教科書
- 高信頼システム構築標準教科書
- Linux セキュリティ標準教科書
- Linux システム管理標準教科書

Linux豆知識
Linuxを学習する上で出てくる素朴な疑問や
便利なテクニックなどを紹介しています。

Linux初心者のための入門編と
中級者向けのネットワーク編の
Linux解説コラム

- Linux 道場入門編
- Linux 道場ネットワーク編
- Linux 道場 Linux学習環境構築編



人気の技術解説無料
セミナーも活用して

LPI-Japanでは、『LinuCレベル1～新出題範囲における
受験準備とポイント解説』など、レベル別の
技術解説無料セミナーを開催しています。
学習の仕方で迷ったら是非足を運んでみてください。
他の受験者の方と意見交換もでき、モチベーションもあがります！

過去のセミナー資料のダウンロードはこちら④

④過去セミナーの動画

<https://linuc.org/measures/movie/>

動画で学ぶピンポイント技術解説
～LinuC技術解説セミナー動画～

LinuCレベル1 LinuCレベル2 202試験

「LinuC技術解説セミナー」では出題範囲について学ぶ上でわかりにくいテーマや技術について掘り下げて講師が解説しています。動画なので、ピンポイントで知りたいことで学習できます。技術書の学習と組み合わせ、自己学習に役立ててください。

毎月開催される技術解説セミナーのスケジュールやそこで配布される資料などは、[セミナーページ](#)から確認することができます。

LinuCレベル1

LinuCレベル2



201試験

主題	セミナー動画	講師
201全般	LinuCレベル2 201試験範囲解説 サーバー構築のできるLinuxエキスパートへの道	藤井 貴博 氏 (株式会社ゼウス・エンタープライズ)
2.01 : システムの起動とLinuxカーネル	システムの起動とLinuxカーネル	藤井 貴博 氏 (株式会社ゼウス・エンタープライズ)
2.02 : ファイルシステムとストレージ管理	ファイルシステムとストレージ管理	末永 貴一 氏 (エスディーテック株式会社)
2.03 : ネットワーク構成	ネットワーク構成	藤井 貴博 氏 (株式会社ゼウス・エンタープライズ)
2.04 : システムの保守と運用管理	リソースの使用状況の把握/死活監視と運用監視ツール 『Ansible』による環境構築の自動化	福田 浩之 氏 植草 克友 氏 (株式会社カシリアル)
2.05 : 仮想化サーバー	仮想化サーバー	末永 貴一 氏 (エスディーテック株式会社)
	コンテナ技術・Dockerの解説 (Linux学習)	藤井 貴博 氏 (株式会社ゼウス・エンタープライズ)
2.06 : コンテナ	コンテナの仕組み/Dockerコンテナとコンテナイメージの管理 コンテナ技術について	藤井 貴博 氏 (株式会社ゼウス・エンタープライズ) 藤井 貴博 氏 (株式会社ゼウス・エンタープライズ)

202試験

主題	セミナー動画	講師
202全般	LinuCレベル2 202試験範囲解説 サーバー構築のできるLinuxエキスパートへの道	藤井 貴博 氏 (株式会社ゼウス・エンタープライズ)
2.07 : ネットワーククライアントの管理	ネットワーククライアントの管理	末永 貴一 氏 (エスディーテック株式会社)
2.08 : ドメインネームサーバー	DNSの役割を理解しよう！ (BIND、ゾーン情報、他)	藤井 貴博 氏 (株式会社ゼウス・エンタープライズ)
2.09 : HTTPサーバーとプロキシサーバー	サーバー構築ハンズオン Web編/nginx/apache/Squid	藤井 貴博 氏 (株式会社ゼウス・エンタープライズ)
2.10 : 電子メールサービス	電子メールサービスの仕組み (Postfix/Dovecotの設定方法を理解する) NFSサーバーの設定と管理	藤井 貴博 氏 (株式会社ゼウス・エンタープライズ) 末永 貴一 氏 (エスディーテック株式会社)
2.11 : ファイル共有サービス	Sambaの設定と管理 ファイル共有サービス、システムのセキュリティ	藤井 貴博 氏 (株式会社ゼウス・エンタープライズ) 藤井 貴博 氏 (株式会社ゼウス・エンタープライズ)
2.12 : システムのセキュリティ	OpenSSHサーバーの設定と管理 ファイル共有サービス、システムのセキュリティ (一部解説)	藤井 貴博 氏 (株式会社ゼウス・エンタープライズ) 藤井 貴博 氏 (株式会社ゼウス・エンタープライズ)
2.13 : システムアーキテクチャ	システムアーキテクチャ	濱野 賢一朗 氏

⑤例題&解説 ※レベル1・3・SAもあり

<https://linuc.org/study/samples/>



試験概要 ▾
受験の手引き
受験申込み
学習コンテンツ

LinuCレベル2

201試験の例題と解説
>

90個の例題 / 2025年6月11日更新

主題2.01 システムの起動とLinuxカーネル

主題2.02 ファイルシステムとストレージ管理

主題2.03 ネットワーク構成

主題2.04 システムの保守と運用管理

主題2.05 仮想化サーバー

主題2.06 コンテナ

202試験の例題と解説
>

93個の例題 / 2025年7月15日更新

主題2.07 ネットワーククライアント管理

主題2.08 ドメインネームサーバー

主題2.09 HTTPサーバーとプロキシサーバー

主題2.10 電子メールサービス

主題2.11 ファイル共有サービス

主題2.12 システムのセキュリティ

主題2.13 システムアーキテクチャ

LinuCレベル1(101)

- 1.01 : Linuxのインストールと仮想マシン・コンテナの利用
 - 1.01.1 Linuxのインストール、起動、接続、切断と停止
 - 1.01.2 仮想マシン・コンテナの概念と利用
 - 1.01.3 ブートプロセスとsystemd
 - 1.01.4 プロセスの生成、監視、終了
 - 1.01.5 デスクトップ環境の利用
- 1.02 : ファイル・ディレクトリの操作と管理
 - 1.02.1 ファイルの所有者とパーミッション
 - 1.02.2 基本的なファイル管理の実行
 - 1.02.3 ハードリンクとシンボリックリンク
 - 1.02.4 ファイルの配置と検索
- 1.03 : GNUとUnixのコマンド
 - 1.03.1 コマンドラインの操作
 - 1.03.2 フィルタを使ったテキストストリームの処理
 - 1.03.3 ストリーム、パイプ、リダイレクトの使用
 - 1.03.4 正規表現を使用したテキストファイルの検索
 - 1.03.5 エディタを使った基本的なファイル編集の実行
- 1.04 : リポジトリとパッケージ管理
 - 1.04.1 apt コマンドによるパッケージ管理
 - 1.04.2 Debian パッケージ管理
 - 1.04.3 yum コマンドによるパッケージ管理
 - 1.04.4 RPM パッケージ管理
- 1.05 : ハードウェア、ディスク、パーティション、ファイルシステム
 - 1.05.1 ハードウェアの基礎知識と設定
 - 1.05.2 ハードディスクのレイアウトとパーティション
 - 1.05.3 ファイルシステムの作成と管理、マウント

<https://linuc.org/linuc1/range/101.html>

LinuCレベル2(201)

- 2.01 : システムの起動とLinuxカーネル
 - 2.01.1 ブートプロセスとGRUB
 - 2.01.2 システム起動のカスタマイズ
 - 2.01.3 Linux カーネルの構成要素
 - 2.01.4 Linuxカーネルのコンパイル
 - 2.01.5 カーネル実行時における管理とトラブルシューティング
- 2.02 : ファイルシステムとストレージ管理
 - 2.02.1 ファイルシステムの設定とマウント
 - 2.02.2 ファイルシステムの管理
 - 2.02.3 論理ボリュームマネージャの設定と管理
- 2.03 : ネットワーク構成
 - 2.03.1 基本的なネットワーク構成
 - 2.03.2 高度なネットワーク構成
 - 2.03.3 ネットワークの問題解決
- 2.04 : システムの保守と運用管理
 - 2.04.1 makeによるソースコードからのビルドとインストール
 - 2.04.2 バックアップとリストア
 - 2.04.3 ユーザへの通知
 - 2.04.4 リソース使用状況の把握
 - 2.04.5 死活監視、リソース監視、運用監視ツール
 - 2.04.6 システム構成ツール
- 2.05 : 仮想化サーバー
 - 2.05.1 仮想マシンの仕組みとKVM
 - 2.05.2 仮想マシンの作成と管理
- 2.06 : コンテナ
 - 2.06.1 コンテナの仕組み
 - 2.06.2 Dockerコンテナとコンテナイメージの管理

<https://linuc.org/linuc2/range/201.html>

主題2.01 : システムの起動とLinuxカーネル

2.01.3 Linux カーネルの構成要素 (重要度 2)

概要 特定のハードウェア、ハードウェアドライバ、システムリソース、およびさまざまな要求に必要なカーネルの構成要素を利用する。これには、異なる種類のカーネルイメージを実装すること、安定版および開発版のカーネルとパッチを区別すること、カーネルモジュールを利用することなども含まれる。

詳細

Kernel の配布形式。

bzImage , xz データ圧縮

Kernelのモジュールとドキュメント。

/usr/src/linux/

/usr/src/linux/Documentation/

主題2.01：システムの起動とLinuxカーネル

2.01.4 Linuxカーネルのコンパイル (重要度 2)

概要 Linuxカーネルの特定の機能を必要に応じて取り込んだり無効化するために、カーネルを適切に構成できる。また、必要に応じてLinuxカーネルをコンパイルし、新しいカーネルに変更点を書き込み、initrdイメージを作成し、新しいカーネルをインストールできる。

詳細

/usr/src/linux/

/usr/src/linux/.config

カーネルの Makefile

Kernel 2.6.x、3.x、4.x、5.x のmakeのターゲット。

all, config, xconfig, menuconfig, gconfig, oldconfig, mrproper, bzImage, modules, modules_install, rpm-pkg, binrpm-pkg, deb-pkg

カーネル構成をカスタマイズする。

新しいカーネルおよび適切なカーネルモジュールを構築する。

/lib/modules/kernel-version/, gzip, bzip2

新しいカーネルおよび必要なモジュールをインストールする。

module tools, depmod

ブートマネージャが新しいカーネルおよび関連付けられたファイルを探せるようにする。

モジュールの構成ファイル

DKMS を使用してカーネルのモジュールをコンパイルする。

dkms

initrd を構成する。

Dracut, mkinitrd, mkinitramfs

主題2.01：システムの起動とLinuxカーネル

2.01.5 カーネル実行時における管理とトラブルシューティング (重要度 3)

概要 2.6.x、3.x、4.x、5.x カーネルとそのロード可能なモジュールについての管理や照会ができる。ブートおよび実行時の一般的な問題を特定および修正することができる。udevを使用したデバイスの検知と管理について理解する。これには、udevルールのトラブルシューティングが含まれる。

詳細

コマンドラインユーティリティを使用して、現在実行中のカーネルおよびカーネルモジュールに関する情報を取得する。

depmod, modinfo

手作業でカーネルモジュールをロードおよびアンロードする。

modprobe, insmod, lsmod, rmmod

モジュールをアンロードできるタイミングを判断する。

モジュールが受け取るパラメータを判断する。

/etc/modprobe.d/

カーネルのバージョンを確認する。

uname

/lib/modules/kernel-version/modules.dep

モジュールをファイル名ではなく別の名前でロードできるようにシステムを設定する。

/proc ファイルシステム

/proc/sys/kernel/

/etc/sysctl.conf, /etc/sysctl.d/, /sbin/sysctl

/sys ファイルシステム (sysfs)

/bootおよび /lib/modules の内容

利用可能なハードウェアに関する情報を分析するツール

dmesg, lspci, lsdev, lsusb, journalctl

udevルール

udevmonitor, udevadm monitor, /etc/udev/

/etc内のモジュール設定ファイル

カーネルダンプを取得する。設定は含まない。

kdump, kexec

例題と解説1

例題1

以下のうち、Linuxカーネルとして適切でないものはどれですか？

1. longtermは長期的なサポートを提供する。
2. mainlineは最新の機能がいち早く導入される。
3. stableはMainlineよりも常にパフォーマンスが高い。
4. mainlineは新しいハードウェアへの対応が含まれることが多い。

解説1

正解は、「3. StableはMainlineよりも常にパフォーマンスが高い。」です。

Linuxカーネル情報は、"<https://www.kernel.org/>"で公開されており、Mainline(開発版)は新しいハードウェアへの対応や新機能が含まれる、Stable(安定版)バグフィックスなどが行われるのが特徴です。

また、UbuntuやAlmaLinuxなど様々なLinuxディストリビューションでは、Longtermと呼ばれる長期サポートが行われるカーネルが採用されています。

Linuxカーネルの開発状況については以下で確認することが出来、日々世界中から更新コードなどへの貢献が行われています。

<https://lore.kernel.org/>

なお、安定版のLinuxカーネルであるからといって、必ずしもパフォーマンスが高くなるとは限りません。

What's Linux kernel ?

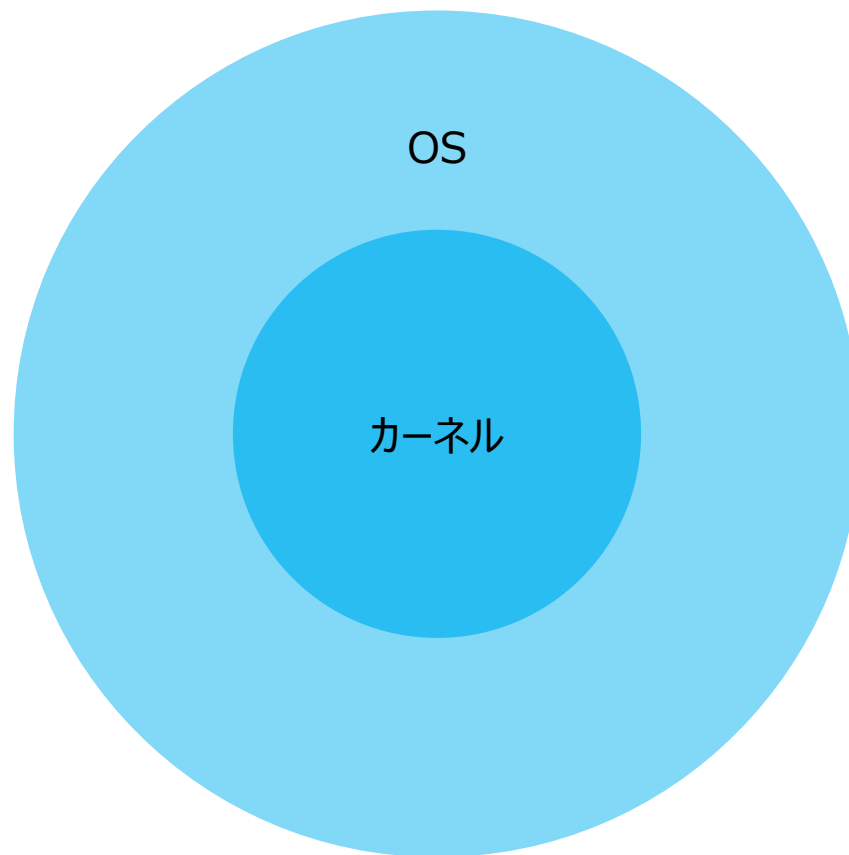


<https://www.kernel.org/>

What's Linux kernel ?

カーネル = OSの核(コア)となるプログラム

- ハードウェア制御(デバイス管理)
- メモリマネージメント
- プロセス処理
- ファイルシステム
- アプリケーション実行
- モジュール管理
などを担当する。



Linuxカーネルの歴史（1991年8月25日）

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 (torv...@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 :-).

<https://groups.google.com/forum/#!original/comp.os.minix/dlNtH7RRrGA/SwRavCzVE7gJ>

Linuxカーネルの歴史

Linuxカーネルの歴史をたどる 2020 Linux Kernel History Report を公開

By The Linux Foundation 8月25, 2020



過去数十年にわたりLinuxは着実に成長し、最も広く使われているオペレーティングシステムカーネルになりました。センサーからスーパーコンピューターまで、宇宙船、自動車、スマートフォン/ウォッチなど、さまざまなデバイスでLinuxが使用されています。Linux Foundationは、2008年にLinux Kernel Development Reports (Linuxカーネル開発レポート) の発行を開始して以来、各時点における進捗を監視してきました。

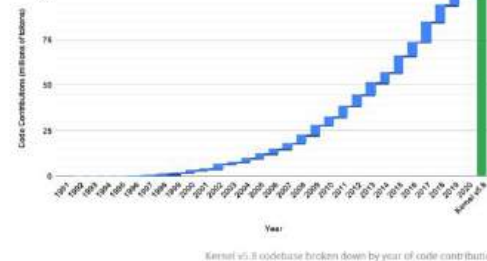
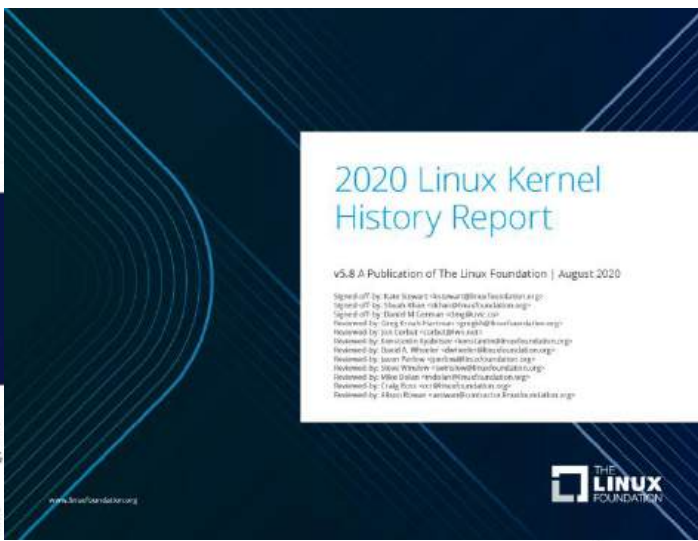
1991年の最初のリリース以来、Linuxは2万人を超えるコントリビューターを擁する歴史上最も成功したコラボレーションの一つになりました。バージョン5.8の最近の発表がこれまで最も大きいカーネルの一つであることを考えると、最新のリリースで1時間あたり10+コミットという新記録を示すなど、開発がスローダウンする兆しは見えません。

本レポートは、Linuxの全歴史をカバーしています。Linuxの分析は、初期リリースおよび1991年9月17日の最初のカーネルリリースから2020年8月2日までの期間のBitKeeperとgitの開発者コミュニティによるコミットに基づいています。2020年8月2日にLinux 5.8がリリースされ、5.9のマージウィンドウが完了した現在、過去29年間記録されたLinuxカーネルヒストリーの100万件を超えるコミットを分析することが可能になりました。

レポートでは、Linuxカーネルの歴史と、最も重要なソフトウェア コラボレーションの一つであるLinux開発を可能にするために出現したベストプラクティスとツリーリング インフラストラクチャの影響について振り返ります。

- 1991年 0.01リリース
- 1994年 1.0リリース
- 1996年 2.0リリース
- 2003年 2.6リリース
- 2011年 3.0リリース
- 2015年 4.0リリース
- 2019年 5.0リリース
- 2022年 6.0リリース

<https://www.linuxfoundation.jp/blog/2020/08/download-the-2020-linux-kernel-history-report/>
https://www.linuxfoundation.jp/wp-content/uploads/2020/08/2020_kernel_history_report_082720.pdf



As Table 1 illustrates, over half of the code in the 5.8 kernel was written in the last seven years, but traces exist from all the prior years contribute to the code in version 5.8 even today.

Year	# of Tokens	Proportion	Accumulated
1991	2,364	0.00%	0.00%
1992	30,740	0.03%	0.03%
1993	46,970	0.04%	0.07%
1994	38,443	0.03%	0.10%
1995	95,488	0.09%	0.19%
1996	204,572	0.19%	0.38%
1997	370,193	0.34%	0.72%
1998	436,941	0.42%	1.14%
1999	775,706	0.70%	1.84%
2000	1,460,450	1.38%	3.23%
2001	828,510	0.79%	4.02%
2002	2,475,002	2.34%	6.36%
2003	1,310,588	1.19%	7.55%
2004	2,023,705	1.83%	9.38%
2005	2,575,195	2.33%	11.71%
2006	3,448,966	3.22%	14.93%
2007	3,318,829	2.93%	17.86%
2008	4,410,921	4.00%	21.86%
2009	5,463,340	4.95%	26.81%
2010	4,805,525	4.35%	31.16%
2011	5,309,906	4.81%	35.97%
2012	6,072,494	5.50%	41.47%
2013	6,440,436	5.84%	47.31%
2014	6,991,508	6.32%	53.63%
2015	6,958,298	6.32%	60.34%
2016	7,622,786	6.91%	67.25%
2017	11,028,463	9.91%	77.16%
2018	9,348,269	8.38%	85.54%
2019	10,843,056	9.83%	95.37%
2020*	5,248,662	4.76%	100.00%

Table 1. Year of Origin of Tokens in Linux v5.8 *as of 8/2/2020



Linuxカーネル

The Linux Kernel Archives

[About](#)

[Contact us](#)

[FAQ](#)

[Releases](#)

[Signatures](#)

[Site news](#)

Protocol

[HTTP](#)

[GIT](#)

[RSYNC](#)

Location

<https://www.kernel.org/pub/>

<https://git.kernel.org/>

<rsync://rsync.kernel.org/pub/>

Latest Release

6.17.3

mainline:	6.18-rc1	2025-10-12	[tarball]	[patch]	[view diff]	[browse]			
stable:	6.17.3	2025-10-15	[tarball]	[pgp]	[patch]	[inc. patch]	[view diff]	[browse]	[changelog]
stable:	6.16.12 [EOL]	2025-10-12	[tarball]	[pgp]	[patch]	[inc. patch]	[view diff]	[browse]	[changelog]
longterm:	6.12.53	2025-10-15	[tarball]	[pgp]	[patch]	[inc. patch]	[view diff]	[browse]	[changelog]
longterm:	6.6.112	2025-10-15	[tarball]	[pgp]	[patch]	[inc. patch]	[view diff]	[browse]	[changelog]
longterm:	6.1.156	2025-10-15	[tarball]	[pgp]	[patch]	[inc. patch]	[view diff]	[browse]	[changelog]
longterm:	5.15.194	2025-10-02	[tarball]	[pgp]	[patch]	[inc. patch]	[view diff]	[browse]	[changelog]
longterm:	5.10.245	2025-10-02	[tarball]	[pgp]	[patch]	[inc. patch]	[view diff]	[browse]	[changelog]
longterm:	5.4.300	2025-10-02	[tarball]	[pgp]	[patch]	[inc. patch]	[view diff]	[browse]	[changelog]
linux-next:	next-20251017	2025-10-17						[browse]	

mainline (開発版)
約3か月のインターバルで新機能を開発する
x.y-rc1 → x.y-rc2 → x.y-rc3...x.y-rc12

stable (安定版)
開発完了した新機能・バグ修正が行われる
 $x.y \rightarrow x.y.1 \rightarrow x.y.2 \dots$

longterm (長期保守版)
重要なバグ修正のみが行われる
 $x.y \rightarrow x.y.1 \rightarrow x.y.2 \dots$

<https://www.kernel.org/>
<https://www.kernel.org/category/releases.html>

Linuxカーネル



index : kernel/git/torvalds/linux.git

Linux kernel source tree

master switch

Linus Torvalds

about summary refs log tree commit diff stats

log msg search

Mode	Name	Size			
-rw-r--r--	.clang-format	16544	log	stats	plain
-rw-r--r--	.cocciconfig	59	log	stats	plain
-rw-r--r--	.get_maintainer.ignore	71	log	stats	plain
-rw-r--r--	.gitattributes	62	log	stats	plain
-rw-r--r--	.gitignore	1852	log	stats	plain
-rw-r--r--	.mailmap	17556	log	stats	plain
-rw-r--r--	COPYING	496	log	stats	plain
-rw-r--r--	CREDITS	99788	log	stats	plain
d-----	Documentation	3324	log	stats	plain
-rw-r--r--	Kbuild	1327	log	stats	plain
-rw-r--r--	Kconfig	555	log	stats	plain
d-----	LICENSES	141	log	stats	plain
-rw-r--r--	MAINTAINERS	570461	log	stats	plain
-rw-r--r--	Makefile	63210	log	stats	plain
-rw-r--r--	README	727	log	stats	plain
d-----	arch	836	log	stats	plain
d-----	block	2787	log	stats	plain
d-----	certs	374	log	stats	plain
d-----	crypto	5623	log	stats	plain
d-----	drivers	4494	log	stats	plain

<https://git.kernel.org/pub/scm/linux/kernel/git/torvalds/linux.git/tree/?h=v5.9-rc6>

ソースコードディレクトリから見るLinuxカーネルの持つ役割

```
[root@centos8test src]# ls
-rw-r--r--. 1 root root 108M  7月 22 16:44 linux-5.7.10.tar.xz
[root@centos8test src]# cd linux-5.7.10
[root@centos8test linux-5.7.10]# ls
COPYING      Kbuild  MAINTAINERS arch  crypto  include kernel net  security  usr
CREDITS      Kconfig Makefile block drivers init  lib  samples  sound  virt
Documentation LICENSES README  certs fs    ipc  mm  scripts  tools
```

```
.
linux-5.7.10/arch/riscv/
linux-5.7.10/arch/riscv/Kbuild
linux-5.7.10/arch/riscv/Kconfig
linux-5.7.10/arch/riscv/Kconfig.debug
linux-5.7.10/arch/riscv/Kconfig.socs
linux-5.7.10/arch/riscv/Makefile
.
```

```
.
linux-5.7.10/fs/ext4/
linux-5.7.10/fs/ext4/Kconfig
linux-5.7.10/fs/ext4/Makefile
.
```

```
.
linux-5.7.10/drivers/gpu/drm/radeon/r600.c
linux-5.7.10/drivers/gpu/drm/radeon/r600_blit_shaders.c
linux-5.7.10/drivers/gpu/drm/radeon/r600_blit_shaders.h
linux-5.7.10/drivers/gpu/drm/radeon/r600_cs.c
linux-5.7.10/drivers/gpu/drm/radeon/r600_dma.c
linux-5.7.10/drivers/gpu/drm/radeon/r600_dpm.c
linux-5.7.10/drivers/gpu/drm/radeon/r600_dpm.h
linux-5.7.10/drivers/gpu/drm/radeon/r600_hdmi.c
linux-5.7.10/drivers/gpu/drm/radeon/r600_reg.h
linux-5.7.10/drivers/gpu/drm/radeon/r600d.h
.
```

```
.
linux-5.7.10/kernel/dma/
linux-5.7.10/kernel/dma/Kconfig
linux-5.7.10/kernel/dma/Makefile
.
```

security (セキュリティ)
virt (仮想化関連)
mm (メモリ管理)
lib (ライブラリ)
crypto (暗号化)
sound (サウンド)
Documentation
(ドキュメント)
など

unameコマンド（Linuxカーネルバージョンの確認）

```
[root@centos8test ~]# uname -r
```

```
5.7.10
```

```
[root@centos8test ~]# uname -a
```

```
Linux centos8test 5.7.10 #2 SMP Thu Jul 23 23:00:48 JST 2020 x86_64 x86_64 x86_64 GNU/Linux
```

```
[root@centos8test boot]# ls -l /boot
```

```
合計 210472
```

```
lrwxrwxrwx. 1 root root    23  7月 24 10:08 System.map -> /boot/System.map-5.7.10
```

```
-rw-----. 1 root root 3909996  5月  8 20:07 System.map-4.18.0-193.el8.x86_64
```

```
-rw-r--r--. 1 root root 4600996  7月 24 10:08 System.map-5.7.10
```

```
-rw-r--r--. 1 root root  187648  5月  8 20:07 config-4.18.0-193.el8.x86_64
```

```
drwxr-xr-x. 3 root root   4096  7月 23 11:14 efi
```

```
drwx-----. 4 root root   4096  7月 23 11:17 grub2
```

```
-rw-----. 1 root root 54334239  7月 23 11:16 initramfs-0-rescue-ab3015baa6bb48818a10a264e966f18c.img
```

```
-rw-----. 1 root root 29419458  7月 23 11:18 initramfs-4.18.0-193.el8.x86_64.img
```

```
-rw-----. 1 root root 19588423  7月 23 11:24 initramfs-4.18.0-193.el8.x86_64kdump.img
```

```
-rw-----. 1 root root 77062114  7月 24 10:09 initramfs-5.7.10.img
```

```
drwxr-xr-x. 3 root root   4096  7月 23 11:15 loader
```

```
drwx-----. 2 root root   16384  7月 23 11:13 lost+found
```

```
lrwxrwxrwx. 1 root root    20  7月 24 10:08 vmlinuz -> /boot/vmlinuz-5.7.10
```

```
-rwxr-xr-x. 1 root root  8913656  7月 23 11:16 vmlinuz-0-rescue-ab3015baa6bb48818a10a264e966f18c
```

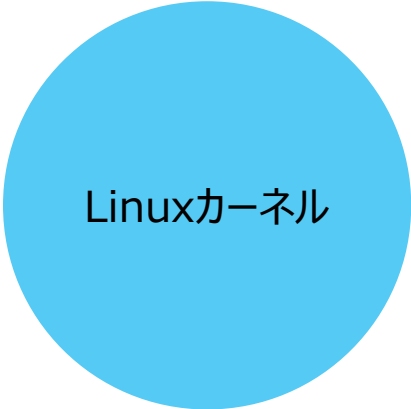
```
-rwxr-xr-x. 1 root root  8913656  5月  8 20:07 vmlinuz-4.18.0-193.el8.x86_64
```

```
-rw-r--r--. 1 root root  8542464  7月 24 10:08 vmlinuz-5.7.10
```

Linux = Linuxカーネル？

Linux（狭義）＝

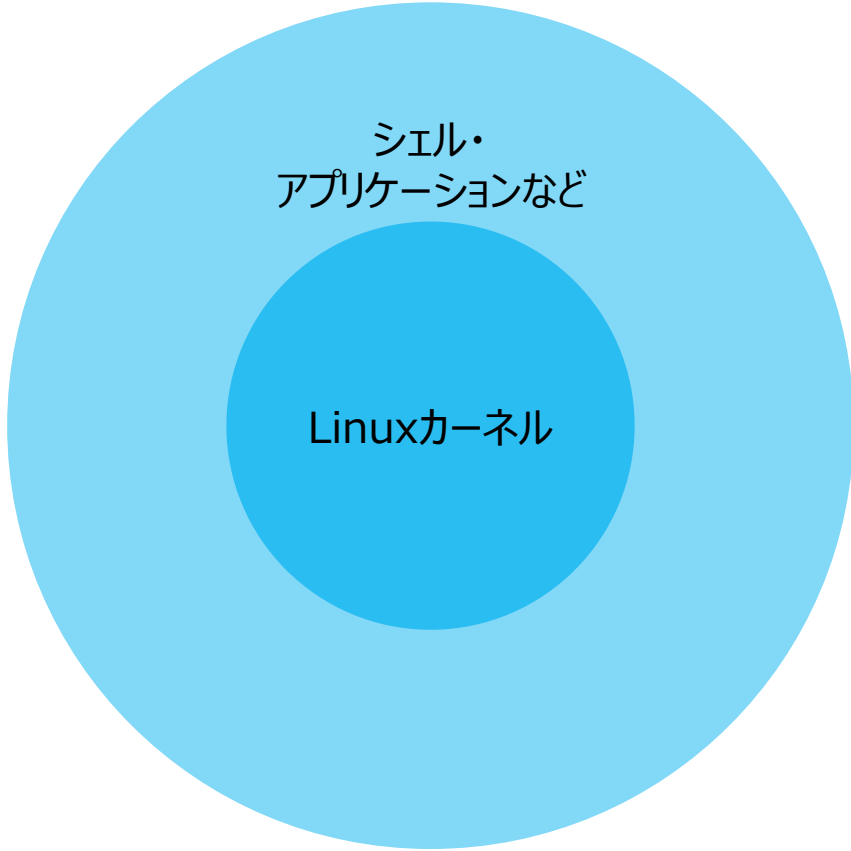
Linuxカーネル



Linux（広義、ディストリビューション）＝

シェル・
アプリケーションなど

Linuxカーネル



例題と解説2

例題2

initrd(initial ramdisk)、またはinitramfs(initial RAM filesystem)の主な目的として、最も適切なものはどれですか。

1. ユーザーがログインした後に使用する、最終的なルートファイルシステムとして機能する
2. ルートファイルシステムをマウントするために必要な一時的なカーネルモジュール（ドライバ）やスクリプトを格納する
3. ブートローダ（GRUBなど）の設定ファイルを格納する
4. システムで利用可能なすべてのカーネルモジュールを格納する

解説2

正解は、「2.ルートファイルシステムをマウントするために必要な、一時的なカーネルモジュール（ドライバ）やスクリプトを格納する」です。

initrdやinitramfsは、システムが起動する初期段階でのみ使用される一時的なファイルシステムです。最終的なルートファイルシステムではありません。initramfsの役割は、実際のルートファイルシステムがマウントされるまでの橋渡しをすることです。ルートファイルシステムがマウントされると、initramfsはメモリから解放され、制御は実際のルートファイルシステムに移ります。

また、その他の選択肢は、正しくは以下となります。

initrdやinitramfsは、最終的なルートファイルシステムではなく、システムが起動する初期段階でのみ使用される一時的なファイルシステムです。

ブートローダ（GRUB）の設定ファイルは、通常、/bootディレクトリ内や、ブートローダがインストールされているパーティションに直接格納されます。initrdやinitramfsの内部に含まれるわけではありません。ブートローダは、カーネルイメージとinitramfsイメージをメモリに読み込む役割を担います。

initrdやinitramfsには、システムが起動し、ルートファイルシステムをマウントするのに最低限必要なカーネルモジュールのみが格納されます。

<https://linuc.org/study/samples/7547/>

Linux kernelの再構築

- ①ソースコードのダウンロード
- ②ダウンロードファイルの解凍
- ③カーネル設定(.config)の生成
- ④カーネルに組み込む機能のコンパイル
- ⑤動的モジュールとして組み込む機能のコンパイル
- ⑥動的モジュールのインストール
- ⑦カーネルのインストール
- ⑧再起動

Opensourcetechnologyブログ

OpensourcetechnologyによるNGINX/Zabbix/Neo4j/Linuxなどオープンソース技術に関するブログです。

2020-07-24

CentOS8のカーネル再構築(Linux kernel 5.7.10)

[Linux](#)
[kernel](#)
[カーネル](#)
[CentOS](#)
[CentOS8](#)
[make](#)
[make config](#)
[make menuconfig](#)
[make bzImage](#)
[make modules](#)
[make modules_install](#)
[make install](#)
[.config](#)

LinuxCエバンジェリストの藤井昌博@opensourcetechnologyです。

Linux Kernelの再構築方法の紹介です。

CentOS8のデフォルトカーネル(4.18.0-193.el8.x86_64)を新しいカーネルにしてみます。

<https://www.kernel.org/>

The Linux Kernel Archives

[About](#)
[Contact us](#)
[FAQ](#)
[Releases](#)
[Signatures](#)
[Site news](#)

Protocol	Location
HTTP	https://www.kernel.org/pub/
GIT	https://git.kernel.org/
RSYNC	rsync://rsync.kernel.org/pub/

Latest Release

5.7.10

mainline:	5.8-rc6	2020-07-19	[tarball]	[patch]	[inc. patch]	[view diff]	[browse]
stable:	5.7.10	2020-07-22	[tarball]	[pgp]	[patch]	[inc. patch]	[view diff] [browse] [changelog]
longterm:	5.4.53	2020-07-22	[tarball]	[pgp]	[patch]	[inc. patch]	[view diff] [browse] [changelog]
longterm:	4.19.134	2020-07-22	[tarball]	[pgp]	[patch]	[inc. patch]	[view diff] [browse] [changelog]
longterm:	4.14.189	2020-07-22	[tarball]	[pgp]	[patch]	[inc. patch]	[view diff] [browse] [changelog]
longterm:	4.9.231	2020-07-22	[tarball]	[pgp]	[patch]	[inc. patch]	[view diff] [browse] [changelog]
longterm:	4.4.231	2020-07-22	[tarball]	[pgp]	[patch]	[inc. patch]	[view diff] [browse] [changelog]
linux-next:	next-20200722	2020-07-22					[browse]

CentOS8のカーネル再構築(Linux kernel 5.7.10)

<https://www.opensourcetechnology.tokyo/?page=1596349636>



35

ソースコードのダウンロード(wgetなど)
解凍(tar/xzなど)

The Linux Kernel Archives



[About](#)
[Contact us](#)
[FAQ](#)
[Releases](#)
[Signatures](#)
[Site news](#)

Protocol	Location
HTTP	https://www.kernel.org/pub/
GIT	https://git.kernel.org/
RSYNC	rsync://rsync.kernel.org/pub/

Latest Release
5.8.10


mainline:	5.9-rc5	2020-09-13	[tarball]	[patch]	[inc. patch]	[view diff]	[browse]
stable:	5.8.10	2020-09-17	[tarball]	[pgp]	[patch]	[inc. patch]	[view diff]
stable:	5.7.19 [EOL]	2020-08-27	[tarball]	[pgp]	[patch]	[inc. patch]	[view diff]
longterm:	5.4.66	2020-09-17	[tarball]	[pgp]	[patch]	[inc. patch]	[view diff]
longterm:	4.19.146	2020-09-17	[tarball]	[pgp]	[patch]	[inc. patch]	[view diff]
longterm:	4.14.198	2020-09-12	[tarball]	[pgp]	[patch]	[inc. patch]	[view diff]
longterm:	4.9.236	2020-09-12	[tarball]	[pgp]	[patch]	[inc. patch]	[view diff]
longterm:	4.4.236	2020-09-12	[tarball]	[pgp]	[patch]	[inc. patch]	[view diff]
linux-next:	next-20200918	2020-09-18					[browse]

Other resources

Cgit	Documentation	Wikis
Bugzilla	Patchwork	Kernel Mailing Lists
Mirrors	Linux.com	Linux Foundation

Social

- [Site Atom feed](#)
- [Releases Atom Feed](#)
- [Kernel Planet](#)

<https://www.kernel.org/>

カーネル設定(.config)の生成

[root@centos8test linux-5.7.10]# **make help** ※一部ターゲットを抜粋

Configuration targets:

- config** - Update current config utilising a line-oriented program ...テキスト形式でパラメータを選択
- nconfig** - Update current config utilising a ncurses menu based program
- menuconfig** - Update current config utilising a menu based program...メニュー形式でパラメータを選択
- xconfig** - Update current config utilising a Qt based front-end
- gconfig** - Update current config utilising a GTK+ based front-end
- oldconfig** - Update current config utilising a provided .config as base...既存設定を使用
- localmodconfig** - Update current config disabling modules not loaded
- localyesconfig** - Update current config converting local mods to core
- defconfig** - New config with default from ARCH supplied defconfig
- savedefconfig** - Save current config as ./defconfig (minimal config)
- allnoconfig** - New config where all options are answered with no
- allyesconfig** - New config where all options are accepted with yes
- allmodconfig** - New config selecting modules when possible
- alldefconfig** - New config with all symbols set to default
- randconfig** - New config with random answer to all options
- yes2modconfig** - Change answers from yes to mod if possible
- mod2yesconfig** - Change answers from mod to yes if possible
- listnewconfig** - List new options
- helpnewconfig** - List new options and help text
- olddefconfig** - Same as oldconfig but sets new symbols to their...既存設定を使用 + 追加されているパラメータを選択
default value without prompting

各コマンド実行時、
不足するパッケージがあれば追加しましょう。

make menuconfig (メニュー形式)

```
.config - Linux/x86 5.7.10 Kernel Configuration

Linux/x86 5.7.10 Kernel Configuration
Arrow keys navigate the menu. <Enter> selects submenus ---> (or empty submenus ----).
Highlighted letters are hotkeys. Pressing <Y> includes, <N> excludes, <M> modularizes
features. Press <Esc><Esc> to exit, <?> for Help, </> for Search. Legend: [*] built-in [ ]
excluded <M> module < > module capable

*** Compiler: gcc (GCC) 8.3.1 20191121 (Red Hat 8.3.1-5) ***
General setup --->
[*] 64-bit kernel
Processor type and features --->
Power management and ACPI options --->
Bus options (PCI etc.) --->
Binary Emulations --->
Firmware Drivers --->
[*] Virtualization --->
General architecture-dependent options --->
[*] Enable loadable module support --->
-* Enable the block layer --->
IO Schedulers --->
Executable file formats --->
Memory Management options --->
[*] Networking support --->
Device Drivers --->
File systems --->
Security options --->
-* Cryptographic API --->
Library routines --->

1(+)
```

<Select> <Exit> <Help> <Save> <Load>

make config (テキスト形式)

```
[root@centos8test linux-5.7.10]# make config
HOSTCC scripts/kconfig/conf.o
HOSTLD scripts/kconfig/conf
scripts/kconfig/conf --oldaskconfig Kconfig
#
# using defaults found in /boot/config-4.18.0-193.el8.x86_64
#
/boot/config-4.18.0-193.el8.x86_64:1089:warning: symbol value 'm' invalid for NF_CT_PROTO_GRE
/boot/config-4.18.0-193.el8.x86_64:2949:warning: symbol value 'm' invalid for ISDN_CAPI
*
* Linux/x86 5.7.10 Kernel Configuration
*
*
* Compiler: gcc (GCC) 8.3.1 20191121 (Red Hat 8.3.1-5)
*
*
* General setup
*
Compile also drivers which will not load (COMPILE_TEST) [N/y/?]
```

make olddefconfig (既存設定を踏襲 + 追加された設定はデフォルト値)

```
[root@centos8test linux-5.7.10]# make olddefconfig
scripts/kconfig/conf --olddefconfig Kconfig
#
# using defaults found in /boot/config-4.18.0-193.el8.x86_64
#
/boot/config-4.18.0-193.el8.x86_64:1089:warning: symbol value 'm' invalid for NF_CT_PROTO_GRE
/boot/config-4.18.0-193.el8.x86_64:2949:warning: symbol value 'm' invalid for ISDN_CAPI
#
# configuration written to .config
#
```

make listnewconfig (追加されている設定をリストアップ)

```
[root@centos8test linux-5.7.10]# make listnewconfig
scripts/kconfig/conf --listnewconfig Kconfig
#
# using defaults found in /boot/config-4.18.0-193.el8.x86_64
#
/boot/config-4.18.0-193.el8.x86_64:1089:warning: symbol value 'm' invalid for NF_CT_PROTO_GRE
/boot/config-4.18.0-193.el8.x86_64:2949:warning: symbol value 'm' invalid for ISDN_CAPI
CONFIG_BUILD_SALT=""
CONFIG_SCHED_THERMAL_PRESSURE=n
CONFIG_IKHEADERS=n
CONFIG_UCLAMP_TASK=n
CONFIG_TIME_NS=y
CONFIG_BOOT_CONFIG=n
CONFIG_BPF_LSM=n
.
```

.config（設定ファイル）の内容

```
[root@centos8test linux-5.7.10]# cat -n .config
```

```

1  #
2  # Automatically generated file; DO NOT EDIT.
3  # Linux/x86 5.7.10 Kernel Configuration
4  #
5
6  #
7  # Compiler: gcc (GCC) 8.3.1 20191121 (Red Hat 8.3.1-5)
8  #
9  CONFIG_CC_IS_GCC=y ← カーネルに組み込む
10 CONFIG_GCC_VERSION=80301
11 CONFIG_LD_VERSION=230000000
12 CONFIG_CLANG_VERSION=0
13 CONFIG_CC_CAN_LINK=y
14 CONFIG_CC_HAS_ASM_GOTO=y

```

.
 .
 省略

```

.
.
8173 CONFIG_TEST_BPF=m ← 動的モジュールとして利用する
8174 # CONFIG_TEST_BLACKHOLE_DEV is not set

```

make bzImage (カーネルに組み込む機能のコンパイル)

※.configで「パラメータ名=y」となっている項目

```
[root@centos8test linux-5.7.10]# make bzImage
SYSTBL arch/x86/include/generated/asm/syscalls_32.h
SYSHDR arch/x86/include/generated/asm/unistd_32_ia32.h
SYSHDR arch/x86/include/generated/asm/unistd_64_x32.h
SYSTBL arch/x86/include/generated/asm/syscalls_64.h
HYPERCALLS arch/x86/include/generated/asm/xen-hypercalls.h
SYSHDR arch/x86/include/generated/uapi/asm/unistd_32.h
SYSHDR arch/x86/include/generated/uapi/asm/unistd_64.h
SYSHDR arch/x86/include/generated/uapi/asm/unistd_x32.h
HOSTCC arch/x86/tools/relocs_32.o
.
.
CC arch/x86/boot/video-vesa.o
CC arch/x86/boot/video-bios.o
LD arch/x86/boot/setup.elf
OBJCOPY arch/x86/boot/setup.bin
OBJCOPY arch/x86/boot/vmlinux.bin
HOSTCC arch/x86/boot/tools/build
BUILD arch/x86/boot/bzImage
Setup is 15068 bytes (padded to 15360 bytes).
System is 8328 kB
CRC da731b84
Kernel: arch/x86/boot/bzImage is ready (#2)
```

make modules (動的モジュールとして利用する機能のコンパイル)

※.configで「パラメータ名=m」となっている項目

```
[root@centos8test linux-5.7.10]# make modules
```

```
CALL scripts/checksyscalls.sh
CALL scripts/atomic/check-atomics.sh
DESCEND objtool
CHK include/generated/compile.h
CC [M] arch/x86/events/amd/power.o
CC [M] arch/x86/events/intel/uncore.o
CC [M] arch/x86/events/intel/uncore_nhmex.o
CC [M] arch/x86/events/intel/uncore_snb.o
CC [M] arch/x86/events/intel/uncore_snbep.o
LD [M] arch/x86/events/intel/intel-uncore.o
CC [M] arch/x86/events/intel/cstate.o
LD [M] arch/x86/events/intel/intel-cstate.o
CC [M] arch/x86/events/rapl.o
CC [M] arch/x86/kernel/cpu/mce/inject.o
LD [M] arch/x86/kernel/cpu/mce/mce-inject.o
AS [M] arch/x86/crypto/twofish-x86_64-asm_64.o
CC [M] arch/x86/crypto/twofish_glue.o
LD [M] arch/x86/crypto/twofish-x86_64.o
.
```

make modules_install (make modulesでコンパイルした動的モジュールのインストール)

```
[root@centos8test linux-5.7.10]# make modules_install
INSTALL arch/x86/crypto/blowfish-x86_64.ko
INSTALL arch/x86/crypto/camellia-aesni-avx-x86_64.ko
INSTALL arch/x86/crypto/camellia-aesni-avx2.ko
INSTALL arch/x86/crypto/camellia-x86_64.ko
INSTALL arch/x86/crypto/cast5-avx-x86_64.ko
INSTALL arch/x86/crypto/cast6-avx-x86_64.ko
INSTALL arch/x86/crypto/chacha-x86_64.ko
INSTALL arch/x86/crypto/crc32-pclmul.ko
INSTALL arch/x86/crypto/crc32c-intel.ko
INSTALL arch/x86/crypto/crct10dif-pclmul.ko
INSTALL arch/x86/crypto/des3_edc-x86_64.ko
INSTALL arch/x86/crypto/ghash-clmulni-intel.ko
INSTALL arch/x86/crypto/poly1305-x86_64.ko
INSTALL arch/x86/crypto/serpent-avx-x86_64.ko
INSTALL arch/x86/crypto/serpent-avx2.ko
INSTALL arch/x86/crypto/serpent-sse2-x86_64.ko
INSTALL arch/x86/crypto/sha512-ssse3.ko
```

.

make install (make bzImageでコンパイルしたもののインストール)

```
[root@centos8test linux-5.7.10]# make install
sh ./arch/x86/boot/install.sh 5.7.10 arch/x86/boot/bzImage ¥
System.map "/boot"
```

```
[root@centos8test boot]# ls -l /boot
合計 210472
lrwxrwxrwx. 1 root root    23  7月 24 10:08 System.map -> /boot/System.map-5.7.10
-rw-----. 1 root root 3909996  5月  8 20:07 System.map-4.18.0-193.el8.x86_64
-rw-r--r--. 1 root root 4600996  7月 24 10:08 System.map-5.7.10
-rw-r--r--. 1 root root  187648  5月  8 20:07 config-4.18.0-193.el8.x86_64
drwxr-xr-x. 3 root root   4096  7月 23 11:14 efi
drwx-----. 4 root root   4096  7月 23 11:17 grub2
-rw-----. 1 root root 54334239  7月 23 11:16 initramfs-0-rescue-ab3015baa6bb48818a10a264e966f18c.img
-rw-----. 1 root root 29419458  7月 23 11:18 initramfs-4.18.0-193.el8.x86_64.img
-rw-----. 1 root root 19588423  7月 23 11:24 initramfs-4.18.0-193.el8.x86_64kdump.img
-rw-----. 1 root root 77062114  7月 24 10:09 initramfs-5.7.10.img
drwxr-xr-x. 3 root root   4096  7月 23 11:15 loader
drwx-----. 2 root root  16384  7月 23 11:13 lost+found
lrwxrwxrwx. 1 root root    20  7月 24 10:08 vmlinuz -> /boot/vmlinuz-5.7.10
-rwxr-xr-x. 1 root root 8913656  7月 23 11:16 vmlinuz-0-rescue-ab3015baa6bb48818a10a264e966f18c
-rwxr-xr-x. 1 root root 8913656  5月  8 20:07 vmlinuz-4.18.0-193.el8.x86_64
-rw-r--r--. 1 root root 8542464  7月 24 10:08 vmlinuz-5.7.10
```

再起動（reboot）後、GRUBの起動メニューに構築したカーネルが選択されるようになっている。

```
CentOS Linux (5.7.10) 8 (Core)
CentOS Linux (4.18.0-193.el8.x86_64) 8 (Core)
CentOS Linux (0-rescue-ab3015baa6bb48818a10a264e966f18c) 8 (Core)

Use the ↑ and ↓ keys to change the selection.
Press 'e' to edit the selected item, or 'c' for a command prompt.
```

“e”をタイプすると、
起動メニューの編集が可能

起動時に使用されるパラメータを変更して、「Ctrl+x」でそのパラメータで起動

```
load_video
set gfx_payload=keep
insmod gzio
linux ($root)/vmlinuz-5.7.10 root=/dev/mapper/cl-root ro crashkernel=auto resume=/dev/mapper/cl-swap rd.lvm.lv=cl/root rd.lvm.lv=cl/swap
initrd ($root)/initramfs-5.7.10.img $tuned_initrd
```

Press Ctrl-x to start, Ctrl-c for a command prompt or Escape to discard edits and return to the menu. Pressing Tab lists possible completions.

```
load_video
set gfx_payload=keep
insmod gzio
linux ($root)/vmlinuz-5.7.10 root=/dev/mapper/cl-root rw init=/bin/sh crashkernel=auto resume=/dev/mapper/cl-swap rd.lvm.lv=cl/root rd.lvm.lv=cl/swap
initrd ($root)/initramfs-5.7.10.img $tuned_initrd
```

例：起動出来ないトラブルなどの際、
シングルユーザモードにする

CentOS7をシングルユーザモードで起動する方法

<https://www.opensourcetech.tokyo/entry/20200308/1583595134>

grub2-mkconfig

全体設定 (/etc/default/grub) とカスタム設定 (/etc/grub.d/配下の設定ファイル) からGRUB設定 (grub.cfg) を生成するコマンド
標準出力に出力されるが、-oオプションでファイル出力可能

```
[root@centos8test ~]# ls -l /etc/default/grub
-rw-r--r--. 1 root root 311  7月 23 11:17 /etc/default/grub
[root@centos8test ~]# ls -l /etc/grub.d/
合計 84
-rwxr-xr-x. 1 root root 8958  4月 14 23:53 00_header
-rwxr-xr-x. 1 root root 1043 12月 12  2019 00_tuned
-rwxr-xr-x. 1 root root 1240  4月 14 23:53 01_menu_auto_hide
-rwxr-xr-x. 1 root root  232  4月 14 23:53 01_users
-rwxr-xr-x. 1 root root 13434  4月 14 23:53 10_linux
-rwxr-xr-x. 1 root root 11696  4月 14 23:53 20_linux_xen
-rwxr-xr-x. 1 root root 2559  4月 14 23:53 20_ppc_terminfo
-rwxr-xr-x. 1 root root 10670  4月 14 23:53 30_os-prober
-rwxr-xr-x. 1 root root 1412  4月 14 23:53 30_uefi-firmware
-rwxr-xr-x. 1 root root  214  4月 14 23:53 40_custom
-rwxr-xr-x. 1 root root  216  4月 14 23:53 41_custom
-rw-r--r--. 1 root root 483  4月 14 23:53 README
[root@centos8test ~]# grub2-mkconfig -o /var/tmp/grub.cfg
Generating grub configuration file ...
done
[root@centos8test ~]# ls -l /var/tmp/
合計 44
-rw-r--r--. 1 root root 36617  9月 22 00:08 dmesg.log
-rw-r--r--. 1 root root 5121  9月 26 23:20 grub.cfg
```

例題と解説3

例題3

カーネルモジュールのバージョンや依存関係など、詳細情報を表示するためのコマンドで正しいものを一つ選択してください。

1. lsmod
2. modprobe
3. depmod
4. modinfo

解説3

正解は、「4. modinfo」です。

カーネルモジュールとは、デバイスドライバなどカーネル機能の一部をモジュール化したものです。カーネルから分離し、必要な時のみ読み込んで使用します。カーネルモジュールによりカーネル本体のサイズを小さくできるため、起動時間を短縮することができます。

カーネルモジュールの詳細情報を表示するコマンドは「modinfo」です。

ロードされていないカーネルモジュールの情報も表示することができるため、手動でカーネルモジュールをロードする際に、依存関係の確認などに利用します。

以下はカーネルモジュールの「drm_kms_helper」の情報を表示した場合の例です。

```
$ modinfo drm_kms_helper
filename:      /lib/modules/3.10.0-327.28.3.el7.x86_64/kernel/drivers/gpu/drm/drm_kms_helper.ko
license:      GPL and additional rights
description:   DRM KMS helper
author:       David Airlie, Jesse Barnes
rhelversion:   7.2
srcversion:    A0481B1796DF7AB221B0ABC
depends:       drm,i2c-core
intree:       Y
vermagic:     3.10.0-327.28.3.el7.x86_64 SMP mod_unload modversions
signer:       CentOS Linux kernel signing key
sig_key:      15:64:6F:1E:11:B7:3F:8C:2A:ED:8A:E2:91:65:5D:52:58:05:6E:E9
sig_hashalgo: sha256
parm:         edid_firmware:Do not probe monitor, use specified EDID blob from built-in data or /lib/firmware instead. (string)
parm:         poll:bool
parm:         dp_aux_i2c_transfer_size:Number of bytes to transfer in a single I2C over DP AUX CH message, (1-16, default 16)
(int)
```

よって「4. modinfo」が正解となります。

1. lsmod

ロードされているモジュールを一覧で確認するためのコマンドです。

「lsmod」コマンドを実行すると、ロードされているモジュールをリストで表示します。以下のように、モジュール名、モジュールのサイズ、依存しているモジュールの数と依存モジュール名を表示します。

```
$ lsmod
Module                Size  Used by
binfmt_misc           17468  1
sg                    40721  0
e1000                 149323  0
i2c_piix4             22106  0
ppdev                 17671  0
pcspkr                12718  0
parport_pc            28165  0
parport               42348  2 ppdev,parport_pc
:
```

(以下略)

2. modprobe

依存関係を解決してモジュールをロードもしくはアンロードするコマンドです。

モジュールのロードはinsmodコマンド、アンロードはrmmodコマンドでも実行することができます。しかし、insmodやrmmodのコマンドを利用する場合、依存関係を考慮する必要があります。

modprobeコマンドは、依存関係を自動で解決してくれるため、容易にモジュールの管理を行うことができます。

3. depmod

モジュールの依存関係情報を更新するコマンドです。

選択肢2のmodprobeコマンドで利用する依存関係情報を、depmodコマンドを用いて更新することができます。

カーネルモジュールに関するコマンドは、ブート時の問題に対処するために利用することがあります。どのコマンドがどのような処理を行うためのコマンドなのか、再度確認し利用できるようにしておきましょう。

使用されているモジュールの一覧表示

```
[root@localhost ~]# lsmod
```

```
Module              Size Used by
binfmt_misc         28672 1
uinput              24576 0
snd_seq_dummy       12288 0
snd_hrtimer          12288 1
nft_fib_inet         12288 0
nft_fib_ipv4         12288 1 nft_fib_inet
nft_fib_ipv6         12288 1 nft_fib_inet
nft_fib              12288 3
nft_fib_ipv6,nft_fib_ipv4,nft_fib_inet
nft_reject_inet      12288 0
nf_reject_ipv4       16384 1 nft_reject_inet
nf_reject_ipv6       24576 1 nft_reject_inet
nft_reject           12288 1 nft_reject_inet
nft_ct               24576 0
nft_chain_nat        12288 0
nf_nat               65536 1 nft_chain_nat
nf_conntrack         229376 2 nf_nat,nft_ct
nf_defrag_ipv6       24576 1 nf_conntrack
nf_defrag_ipv4       12288 1 nf_conntrack
8021q                 53248 0
garp                 16384 1 8021q
mrp                  20480 1 8021q
bonding              282624 0
tls                  159744 1 bonding
bridge               417792 0
stp                  12288 2 bridge,garp
llc                  16384 3 bridge,stp,garp
ip_set                69632 0
rfkill               40960 3
nf_tables             356352 8
nft_ct,nft_reject_inet,nft_fib_ipv6,nft_fib_ipv4,nft_chain_n
at,nft_reject,nft_fib,nft_fib_inet
nfnetlink             20480 2 nf_tables,ip_set
qrtr                  57344 2
```

```
sunrpc               892928 1
intel_rapl_msr        20480 0
intel_rapl_common     57344 1 intel_rapl_msr
intel_uncore_frequency 12288 0
intel_uncore_frequency_common 16384 1
intel_uncore_frequency
intel_pmc_core_pltdrv 12288 0
intel_pmc_core        122880 0
intel_vsec            20480 1 intel_pmc_core
pmt_telemetry         16384 1 intel_pmc_core
pmt_class             16384 1 pmt_telemetry
intel_powerclamp      24576 0
kvm_intel             446464 0
snd_intel8x0          53248 2
kvm                   1404928 1 kvm_intel
snd_ac97_codec        200704 1 snd_intel8x0
ac97_bus              12288 1 snd_ac97_codec
snd_seq               131072 7 snd_seq_dummy
snd_seq_device        16384 1 snd_seq
snd_pcm               192512 2
snd_intel8x0,snd_ac97_codec
snd_timer             53248 3
snd_seq,snd_hrtimer,snd_pcm
rapl                  24576 0
snd                   155648 12
snd_seq,snd_seq_device,snd_intel8x0,snd_timer,snd_ac97
_codec,snd_pcm
intel_cstate          24576 0
intel_uncore          270336 0
pcspkr                12288 0
i2c_piix4             28672 0
soundcore             16384 1 snd
xfs                   2686976 2
libcrc32c             12288 4
nf_conntrack,nf_nat,nf_tables,xfs
vmwgfx                475136 1
```

```
drm_ttm_helper        16384 2 vmwgfx
ttm                   114688 2 vmwgfx,drm_ttm_helper
drm_kms_helper        266240 2 vmwgfx,drm_ttm_helper
sr_mod                28672 0
sd_mod                90112 3
cdrom                 90112 1 sr_mod
sg                    53248 0
ahci                  49152 3
ata_generic           16384 0
drm                   811008 6
vmwgfx,drm_kms_helper,drm_ttm_helper,ttm
libahci               61440 1 ahci
e1000                 196608 0
ata_piix              45056 0
libata                520192 4 ata_piix,libahci,ahci,ata_generic
crct10dif_pclmul      12288 1
crc32_pclmul          12288 0
crc32c_intel          24576 1
ghash_clmulni_intel   16384 0
video                 77824 0
wmi                   45056 1 video
serio_raw             16384 0
dm_mirror             28672 0
dm_region_hash        28672 1 dm_mirror
dm_log                24576 2 dm_region_hash,dm_mirror
dm_mod               245760 9 dm_log,dm_mirror
fuse                  212992 5
```

モジュール名	主要な機能	分野/カテゴリ
ext4	第4拡張ファイルシステム。Linuxで広く使用されるジャーナリングファイルシステムの実装。	ファイルシステム
btrfs	B-tree File System。スナップショットやチェックサムなどの高度な機能を持つ新世代ファイルシステム。	ファイルシステム
nfs	Network File System (NFS) のクライアント/サーバー機能。ネットワーク経由でのファイル共有・アクセスを可能にする。	ネットワーク/ファイルシステム
i915	Intel統合グラフィックス (iGPU) のデバイスドライバ。ディスプレイ出力、3Dアクセラレーション機能を提供。	デバイスドライバ/グラフィックス
nvidia / amdgpu	NVIDIA または AMD の専用GPUドライバ。高性能なグラフィックス処理やGPGPU機能を提供。	デバイスドライバ/グラフィックス
e1000 / r8169	Ethernetネットワークアダプタ(例: Intel, Realtek) のデバイスドライバ。有線ネットワーク通信を可能にする。	デバイスドライバ/ネットワーク
usbhid	USB Human Interface Device (HID) のサポート。USB接続のキーボード、マウス、ゲームパッドなどの入力を処理。	デバイスドライバ/USB
loop	ループデバイスのサポート。ファイル(ディスクイメージなど)をブロックデバイスとしてマウントし、ファイルシステムとしてアクセスできるようにする。	仮想デバイス
nf_conntrack	Netfilter接続追跡機能。ファイアウォールがパケットの状態を追跡し、ステートフルフィルタリングを可能にする。	ネットワーク/セキュリティ
kvm	Kernel-based Virtual Machine の中核機能。ハードウェア仮想化支援を利用した高性能な仮想マシン実行を可能にする。	仮想化

モジュールの読み込み(依存関係の自動解決をしてくれる)

```
[root@localhost ~]# modprobe kvm_intel
[root@localhost ~]#
```

モジュールの詳細確認

```
[root@localhost ~]# modinfo ext4
filename:      /lib/modules/5.14.0-570.17.1.el9_6.x86_64/kernel/fs/ext4/ext4.ko.xz
softdep:      pre: crc32c
license:      GPL
description:   Fourth Extended Filesystem
author:       Remy Card, Stephen Tweedie, Andrew Morton, Andreas Dilger,
Theodore Ts'o and others
alias:        fs-ext4
alias:        ext3
alias:        fs-ext3
alias:        ext2
alias:        fs-ext2
rhelversion:   9.6
srcversion:    414CA771338855FD8AA67C4
depends:       jbd2,mbcache
retpoline:    Y
intree:       Y
name:         ext4
vermagic:     5.14.0-570.17.1.el9_6.x86_64 SMP preempt mod_unload
modversions
sig_id:       PKCS#7
signer:       Rocky kernel signing key
sig_key:      F5:03:24:D1:25:4A:DE:82:57:F2:1C:EE:7C:D6:C7:14:28:E3:FF
sig_hashalgo: sha256
signature:    61:BC:53:9F:C0:C1:2A:2D:B2:34:DF:10:67:4E:50:C6:10:A5:13:AB:
2E:2B:4D:A7:EE:AD:A3:55:E7:39:D4:25:67:53:8B:1E:0B:8F:0F:05:
90:48:2A:CA:64:CD:DF:62:5D:18:2B:9A:B2:D2:D1:6A:9B:18:DF:60:
8B:12:FF:F0:DC:FB:1B:10:EE:F8:3C:FB:3E:BC:58:4C:5F:12:47:C3:
4B:85:BD:A0:90:F2:36:E6:D7:A2:A2:60:59:79:DF:B5:EA:CF:16:7A:
AF:D6:56:0D:5E:E4:34:16:0C:98:F2:46:5F:47:62:69:C8:0E:97:48:
F8:03:00:DA:B9:CC:ED:40:C9:60:40:6D:24:17:AD:32:DE:9A:A6:DC:
94:A6:A2:D4:7F:6D:0C:7C:B9:DC:8D:26:80:B1:BF:C0:37:15:F8:C5:
46:53:8A:81:40:D4:02:A2:0F:DD:76:C9:F4:DE:70:16:6D:1B:E0:C9:
4D:C5:3D:4F:EE:B6:15:41:F7:EC:00:59:8E:9A:A4:38:DA:F7:52:1C:
B2:1A:F8:D9:74:D3:32:78:80:BE:B6:F7:E0:04:46:E4:00:25:74:AF:
3C:F7:1D:7E:F1:9D:4B:27:B0:35:98:76:C8:66:4D:C2:BC:9E:54:32:
5D:3C:1B:67:80:E8:1D:6D:14:E2:E0:31:FD:5A:1B:A8:88:6C:3F:92:
B4:4B:B8:E5:D3:D0:7D:BD:22:76:57:20:28:62:55:B2:8A:0C:92:F3:
57:E0:05:BF:95:92:23:79:36:C2:AA:11:EE:21:8F:23:9C:7E:7E:23:
92:0C:B4:3C:DD:D5:85:6B:B7:97:64:E8:59:92:61:88:7D:93:C7:3F:
4F:48:BD:4C:35:F6:05:A4:B1:4E:61:1C:36:38:57:03:40:74:57:80:
A4:6E:74:BA:F6:10:E7:29:73:6E:AE:53:2A:C8:00:E5:01:DD:98:B4:
49:7A:0E:1F:B9:39:0C:03:24:5B:97:EE:E3:11:25:98:1D:41:37:5B:
1C:9D:0D:F7
```

振り返り

アジェンダ

- LinuC、およびその学習方法
- 例題と解説
- Linuxカーネルの構成要素と管理について
- 振り返り

本日のゴール

- Linuxカーネル構成要素と管理の理解
- LinuCレベル2資格対策の理解

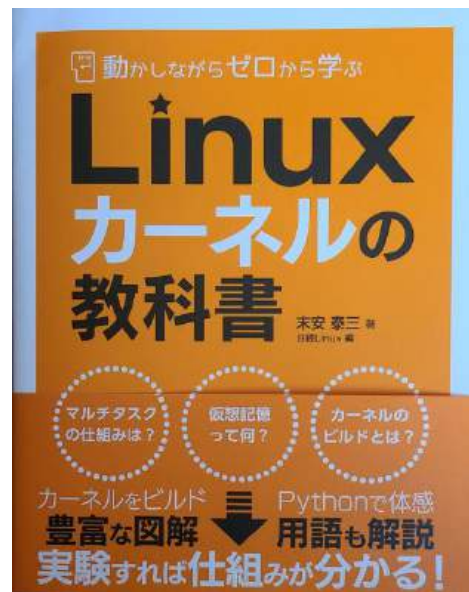
Linuxカーネルを知るための書籍・イベント



それがぼくには楽しかったから
全世界を巻き込んだリナックス革命の真実



改訂3版 Linuxエンジニア養成読本
(Software Design plus)



動かしながらゼロから学ぶ
Linuxカーネルの教科書

Keynote: Linus Torvalds, Creator of Linux & Git, in Conversation with Dirk Hohndel, Head of the Open Source Program Office, Verizon

Tuesday December 9, 2025 10:00 - 10:30 JST
Main Hall (SF)

Speakers

Dirk Hohndel
Head of the Open Source Program Office, Verizon
Dirk is the Head of the Open Source Program Officer at Verizon. Prior to that, Dirk was VMware's Chief Open Source Officer, where he lead the company's Open Source Program Office, directing the efforts and strategy around use of and contribution to open-source projects and driving... [Read More](#)

Linus Torvalds
Creator, Linux & Git
Linus was born on December 28, 1969, in Helsinki, Finland. He enrolled at the University of Helsinki in 1988, graduating with a master's degree in computer science. His M.Sc. thesis was titled "Linux: A Portable Operating System" and was the genesis for what would become the... [Read More](#)

● Keynote Sessions

Open Source Summit Japan
<https://events.linuxfoundation.org/open-source-summit-japan/program/schedule/>

ご清聴ありがとうございました！