openSUSE Leap 15.7

リファレンス

リファレンス
openSUSE Leap 15.7

発行日: 2026/11/05

SUSE LLC
1800 South Novell Place
Provo, UT 84606
USA
https://documentation.suse.com

Copyright © 2006– 2026 SUSE LLC and contributors. All rights reserved.

訳: SUSE LLC および貢献者が全権利を留保しています。

Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.2 or (at your option) version 1.3; with the Invariant Section being this copyright notice and license. A copy of the license version 1.2 is included in the section entitled 「GNU Free Documentation License」.

訳: この文書を、フリーソフトウェア財団発行の GNU フリー文書利用許諾契約書バージョン 1.2 または (希望すれば) 1.3 が定める条件の下で複製、頒布、あるいは改変することを許可します。ただし、この著作権とライセンス表記については変更不可部分とします。この利用許諾契約書の複製物は、「GNU フリー文書利用許諾契約書」という章に含まれています。

For SUSE trademarks, see https://www.suse.com/company/legal/ . All other third-party trademarks are the property of their respective owners. Trademark symbols (®, ™ etc.) denote trademarks of SUSE and its affiliates. Asterisks (*) denote third-party trademarks.

訳: SUSE 社の商標については、https://www.suse.com/company/legal/ をご覧ください。その他の商標は各所有者の所有物です。商標シンボル(®, ™ など) は、それぞれ SUSE 社およびその関連会社の商標であることを示しています。また、アスタリスク (*) は第三者の商標を示しています。

All information found in this book has been compiled with utmost attention to detail. However, this does not guarantee complete accuracy. Neither SUSE LLC, its affiliates, the authors nor the translators shall be held liable for possible errors or the consequences thereof.

訳: この書籍内にある全ての情報は、細部に至るまで最大限の注意を払って制作されていますが、完全に正確であることを保証するものではありません。SUSE LLC やその関連会社, 著者, 翻訳者のいずれも、本書籍内の誤りとそこから生じる結果について、一切の保証はいたしません。

注記

なお、本文書は原文 (英語) の翻訳文書であり、公式な文書ではありません。あらかじめご了承ください。

目次

このガイドについて xviii
1 利用可能なドキュメンテーション xviii
2 ドキュメンテーションの改善 xix
3 文書規約 xx
4 ソースコードについて xxii
5 謝辞 xxii

I 高度な管理 1

1 テキストモードでの YaST 2
1.1 モジュール内の移動 3
1.2 高度なキーの組み合わせ 4
1.3 キーの組み合わせに対する制限 5
1.4 YaST コマンドラインオプション 6
コマンドラインからのパッケージのインストール 6 • 個別のモジュールでの作業 6 • YaST モジュールでのコマンドラインパラメータ 7

2 コマンドラインツールでのソフトウエア管理 32
2.1 zypper の使用 32
一般的な使用方法 32 • zypper のサブコマンドの使用 34 • zypper を利用したソフトウエアのインストールと削除 34 • zypper を利用したソフトウエアの更新 39 • 削除したファイルを使用しているプロセスやサービスの検出 44 • zypper を利用したリポジトリの管理 46 • zypper を利用したリポジトリやパッケージの問い合わせ 48 • zypper の設定 50 • トラブルシューティング 50 • btrfs ファイルシステムにおける zypper のロールバック機能 51 • さらなる情報 51

2.2 RPM パッケージマネージャ 51
パッケージの発行者の検証 52 • パッケージの管理: インストール／更新／削除 52 • 差分 (デルタ) RPM パッケージ 54 • RPM データベースへの問い合わせ 54 • ソースパッケージのインストールとコンパイル 57 • build を利用した RPM パッケージのコンパイル 59 • RPM 書庫および RPM データベース向けツール 60

3 Snapper によるシステムの復元とスナップショット管理 61

3.1 既定の設定 62
既定の設定 63 • スナップショットの種類 63 • スナップショットから除外されるディレクトリ 64 • 設定のカスタマイズ 65

3.2 Snapper による変更点の巻き戻し 68
YaST や zypper が変更した内容の巻き戻し 70 • ファイルを復元するための Snapper の使用 74

3.3 スナップショットからの起動によるシステムのロールバック 76
ロールバック後のスナップショット 79 • スナップショットの起動項目へのアクセスと識別 79 • 制限事項 81

3.4 ユーザのホームディレクトリに対する Snapper の有効化 82
pam_snapper のインストールとユーザの作成 83 • ユーザの削除 84 • 手作業でのホームディレクトリに対するスナップショットの有効化 84

3.5 Snapper の設定の作成と修正 84
既存の設定の管理 86

3.6 スナップショットの手動作成と管理 89
スナップショットのメタデータ 90 • スナップショットの採取 92 • スナップショットのメタデータの変更 93 • スナップショットの削除 93

3.7 自動的なスナップショットのクリーンアップ 95
番号スナップショットのクリーンアップ 95 • タイムラインスナップショットのクリーンアップ 97 • 違いのないスナップショット対のクリーンアップ 98 • 手作業で作成したスナップショットのクリーンアップ 99 • ディスククォータサポートの追加 99

3.8 スナップショットが占有しているディスク容量の表示 100

3.9 よくある質問とその回答 102

4 VNC によるリモートグラフィカルセッション 104

4.1 vncviewer クライアント 104
vncviewer コマンドラインを利用した接続 105 • vncviewer GUI を利用した接続 105 • 暗号化されていない接続に対する通知 106

4.2 Remmina: リモートデスクトップクライアント 106
インストール 106 • メインウインドウ 106 • リモートセッションの追加 107 • リモートセッションの開始 108 • 保存済みセッションの編集／コピー／削除 109 • コマンドラインからのリモートセッションの実行 110

4.3 VNC サーバ側でのワンタイムセッションの設定 111
利用可能な設定 112 • ワンタイム VNC セッションの開始 113 • ワンタイム VNC セッションの設定 113

4.4 永続 VNC サーバセッションの設定 114
vncmanager を利用した VNC セッションの起動 114

4.5 VNC サーバ側での暗号化の設定 117

4.6 Wayland との互換性について 119

5 [熟練者向けパーティション設定] 120

5.1 [熟練者向けパーティション設定] の使用 120
パーティションテーブル 122 • パーティション 123 • パーティションの編集 127 • 熟練者向けオプション 129 • [tmpfs] 130 • [Bcache デバイス] 130 • 高度なオプション 131 • パーティション設定のヒント 131 • パーティション設定と LVM 134

5.2 デバイスの暗号化 134
暗号化方式 134 • パスワードベースの鍵導出関数 136

5.3 LVM 設定 137
物理ボリューム (PV) の作成 138 • ボリュームグループ (VG) の作成 138 • 論理ボリューム (LV) の設定 139

5.4 ソフトウエア RAID 141
ソフトウエア RAID の設定 141 • トラブルシューティング 142 • さらなる情報 143

6 複数バージョンのカーネルのインストール 144
6.1 マルチバージョン対応の有効化と設定 144
不要なカーネルの自動削除 145 • 設定例: 再起動後にのみ古いカーネルを削除する設定 147 • 設定例: 予備として古いカーネルを残す設定 147 • 設定例: 特定バージョンのカーネルを維持する設定 147
6.2 YaST による複数バージョンのカーネルのインストールと削除 148
6.3 zypper による複数バージョンのカーネルのインストールと削除 149
6.4 Kernel:HEAD リポジトリからの最新カーネルのインストール 151
7 グラフィカルユーザインターフェイス 152
7.1 X Window System 152
7.2 フォントのインストールと設定 153
インストール済みのフォントの一覧表示 154 • フォントの表示 155 • フォントの問い合わせ 155 • フォントのインストール 156 • フォントの外観の設定 156
7.3 管理者向けの GNOME 設定 165
dconf システム 165 • システム全体の設定 165 • さらなる情報 166
7.4 SUSE Prime を利用した Intel および NVIDIA Optimus GPU の切り替え 166
事前要件 167 • SUSE Prime のインストールと使用 167 • NVIDIA ドライバのインストール 168
II システム 169
8 64 ビットシステム環境における 32 ビット／ 64 ビットアプリケーション 170
8.1 ランタイムのサポート 170
8.2 カーネルの仕様 171
9 起動処理の紹介 172
9.1 用語説明 172

9.2 Linux の起動処理 173
初期化とブートローダの処理フェーズ 173 • カーネルの処理フェーズ 174 • initramfs での初期化フェーズ 177 • systemd の処理フェーズ 179

10 systemd デーモン 180

10.1 systemd の考え方 181
ユニットファイル 181

10.2 基本的な使い方 182
動作中のシステムにおけるサービスの管理 183 • サービスの恒久的な有効化／無効化 184

10.3 システムの起動とターゲットの管理 186
ターゲットとランレベルの違い 186 • システム起動時のデバッグ 190 • System V との互換性 193

10.4 YaST を利用したサービスの管理 194

10.5 systemd のカスタマイズ 195
ユニットファイルの配置場所 195 • ドロップイン・ファイルによる上書き 195 • 手作業によるドロップイン・ファイルの作成 196 • xinetd から systemd へのサービス変換 198 • 独自のターゲットの作成 199

10.6 高度な使い方 200
一時ディレクトリのクリーニング 200 • システムログ 201 • スナップショット 201 • カーネルモジュールの読み込み 202 • サービス読み込み前の処理の実行 202 • カーネルコントロールグループ (cgroups) 203 • サービスの終了 (シグナルの送信) 204 • D-Bus サービスに対する重要な情報 205 • サービスのデバッグ 205

10.7 systemd タイマーユニット 206
systemd タイマーの種類 207 • systemd とサービスユニット 207 • 具体的な例 207 • systemd タイマーの管理 209

10.8 さらなる情報 209

11 journalctl : systemd ジャーナルへの問い合わせコマンド 210

11.1 ジャーナルの恒久化 210

11.2 journalctl での便利なスイッチ 211

11.3 ジャーナル出力のフィルタリング 212
起動番号をベースにしたフィルタ 212 • 時間範囲をベースにしたフィルタ 212 • 項目をベースにしたフィルタ 213

11.4 systemd のエラー調査 214

11.5 journald の設定 215
ジャーナルのサイズ制限の変更 215 • ジャーナルから /dev/ttyX への転送 215 • ジャーナルから syslog への転送 216

11.6 YaST を利用した systemd ジャーナルのフィルタリング 216

11.7 GNOME でのログ表示 217

12 ブートローダ GRUB 2 218

12.1 GRUB Legacy と GRUB 2 との主な違い 218

12.2 設定ファイルの構造 218
/boot/grub2/grub.cfg ファイル 219 • /etc/default/grub ファイル 220 • /etc/grub.d 内にあるスクリプト 223 • BIOS ドライブと Linux デバイスのマッピングについて 225 • 起動処理中でのメニュー項目の編集 225 • 起動パスワードの設定 227 • 起動メニュー項目に対するアクセス保護 228

12.3 YaST によるブートローダの設定 229
ブートローダの場所とブートコードのオプション 230 • ディスクの順序の調整 232 • 高度なオプションの変更 232

12.4 便利な GRUB 2 コマンド 235

12.5 レスキューモード 237

12.6 さらなる情報 237

13 ネットワークの基礎 238

13.1 IP アドレスとルーティング 241
IP アドレス 241 • ネットマスクとルーティング 242

13.2 IPv6—次世代インターネット 244
利点 244 • アドレスの種類と構造 246 • IPv4 と IPv6 の共存 250 • IPv6 の設定 251 • さらなる情報 251

13.3 名前解決 252

13.4 YaST を利用したネットワークの設定 253
YaST を利用したネットワークカードの設定 254

13.5 NetworkManager 265
NetworkManager と `wicked` の違い 265 • NetworkManager の機能と設定ファイル 266 • NetworkManager の機能制御とロックダウン 267

13.6 ネットワーク接続の手動管理 267
`wicked` ネットワーク設定 267 • 設定ファイル 274 • 設定のテスト 285 • ユニットファイルと起動スクリプト 290

13.7 基本的なルータの構築 291

13.8 ボンドデバイスの設定 292
ボンドポートのホットプラグ 295 • 予測可能なインターフェイス命名方式 296

13.9 ネットワークチーミング によるチームデバイス (チーミング／ボンディング) の設定 297
使用例: ネットワークチーミング による負荷分散 301 • 使用例: ネットワークチーミング による冗長構成 302 • 使用例: チーミングデバイスでの VLAN 303

13.10 ネットワークブリッジデバイスの設定 305
手作業によるネットワークブリッジの作成 305

13.11 Open vSwitch を利用したソフトウエア定義型ネットワーク 306
Open vSwitch の利点 307 • Open vSwitch のインストール 307 • Open vSwitch デーモンとユーティリティの概要 308 • Open vSwitch を利用したブリッジの作成 309 • KVM による Open vSwitch の直接使用 310 • `libvirt` での Open vSwitch の使用 312 • さらなる情報 312

14 UEFI (Unified Extensible Firmware Interface) 313

14.1 Secure Boot 313
openSUSE Leap での実装 314 • MOK (マシン所有者鍵; Machine Owner Key) 316 • 独自のカーネルの起動 317 • 非同梱ドライバの使用について 319 • 機能と制限 320

14.2 Secure Boot 失効リスト 321
オンラインの失効リスト更新の適用 321 • オンラインの失効リスト更新の適用 322

14.3 さらなる情報 323

15 特殊なシステム機能 324

15.1 特殊なソフトウエアパッケージについての情報 324
bash パッケージと /etc/profile の関係 324 • cron パッケージ 325 • cron の状態メッセージの停止 326 • ログファイル: logrotate パッケージ 326 • locate コマンド 326 • ulimit コマンド 327 • free コマンド 328 • マニュアルページと info ページ 328 • man コマンドでのマニュアルページの選択 328 • GNU Emacs の設定 329

15.2 仮想コンソール 330

15.3 キーボードマッピング 330

15.4 言語および国固有の設定 331
システム全体の言語設定 332 • いくつかの例 333 • ~/.i18n 内でのロケール (言語) 設定 334 • 言語に対応するための設定 334 • さらなる情報 335

16 udev による動的なカーネルデバイス管理 336

16.1 /dev ディレクトリ 336

16.2 カーネルの uevent と udev 336

16.3 ドライバ／カーネルモジュール／デバイス 337

16.4 システムの起動と初期デバイス設定 337

16.5 動作中の udev デーモンの監視 338

16.6 udev ルールによるカーネルのデバイスイベント処理の調整 339
udev ルール内での演算子の使用 341 • udev ルール内での置換の使用 342 • udev マッチキーの使用 343 • udev 代入キーの使用 344

16.7 デバイスの命名の恒久化 346

16.8 udev で使用されるファイル 346

16.9 さらなる情報 347

III サービス 348

17 SLP 349
17.1 SLP フロントエンド `slptool` 349
17.2 SLP を介したサービス情報の提供 350
SLP インストールサーバの構築 352
17.3 さらなる情報 352

18 NTP を利用した時刻同期 353
18.1 YaST を利用した NTP クライアントの設定 354
NTP デーモンの開始 354 • 設定元の種類 355 • タイムサーバの設定 355
18.2 ネットワーク内での NTP の手動設定 356
18.3 NTS の設定 357
18.4 `chronyc` による動作中の `chronyd` の設定 360
18.5 動作中の動的な時刻同期 361
18.6 ローカル参照時計の設定 361

19 ドメインネームシステム 363
19.1 DNS で使用される用語 363
19.2 インストール 364
19.3 YaST での設定 364
ウイザードによる設定 365 • 熟練者向け設定 368
19.4 BIND ネームサーバの起動 376
19.5 /etc/named.conf 設定ファイル 378
主要な設定オプション 379 • ログ 380 • ゾーンの項目 381
19.6 ゾーンファイル 382
19.7 ゾーンデータの動的な更新 386

19.8 トランザクションの暗号化 386
19.9 DNS セキュリティ 387
19.10 さらなる情報 388
20 DHCP 389
20.1 YaST を利用した DHCP サーバの設定 390
初期設定 (ウイザード) 390 • DHCP サーバ設定 (熟練者向け) 395
20.2 DHCP ソフトウエアパッケージ 401
20.3 DHCP サーバ dhcpd 402
固定 IP アドレスのクライアント 404 • openSUSE Leap 版について 404
20.4 さらなる情報 405
21 Samba 406
21.1 用語 406
21.2 Samba サーバのインストール 408
21.3 Samba の開始と停止 408
21.4 Samba サーバの設定 408
YaST を利用した Samba サーバの設定 409 • 手作業によるサーバの設定 411
21.5 クライアントの設定 416
YaST での Samba クライアントの設定 416 • クライアント側での SMB1/CIFS 共有のマウント 416
21.6 ログインサーバとしての Samba 417
21.7 Active Directory ネットワーク環境での Samba サーバ 418
Active Directory を管理するための `realmd` の使用 420
21.8 高度なトピック 420
`systemd` を利用した CIFS ファイルシステムの自動マウント 421 • btrfs による透過型ファイル圧縮 422 • スナップショット 423
21.9 さらなる情報 432

22 NFS によるファイル共有 433
22.1 概要 433
22.2 NFS サーバのインストール 435
22.3 NFS サーバの設定 435
YaST を利用したファイルシステムの公開 435 • 手作業によるファイルシステムの公開 437 • Kerberos を併用する NFS 440
22.4 クライアントの設定 441
YaST を利用したファイルシステムの取り込み 442 • 手作業によるファイルシステムの取り込み 443 • Parallel NFS (pNFS) 445
22.5 ファイアウオールを介した NFS サーバと NFS クライアントの通信 446
NFS 4. x 446 • NFS 3 447
22.6 NFSv4 を介したアクセス制御リスト (ACL) の管理 449
22.7 さらなる情報 450
22.8 NFS のトラブルシューティングにおける情報収集 451
一般的なトラブルシューティング 451 • 高度な NFS デバッグ 453
23 autofs によるオンデマンド型のマウント 455
23.1 インストール 455
23.2 設定 455
マスターマップファイル 456 • マップファイル 458
23.3 操作とデバッグ 458
autofs サービスの制御 459 • automounter の問題調査 459
23.4 NFS 共有の自動マウント 460
23.5 高度なトピック 461
/net マウントポイント 461 • 自動マウントサブディレクトリでのワイルドカードの使用 462 • CIFS ファイルシステムの自動マウント 462
24 Apache HTTP サーバ 463
24.1 クイックスタート 463
要件 463 • インストール 464 • 起動 464

24.2 Apache の設定 465
Apache の設定ファイル 465 • Apache の手作業での設定 469 • YaST による Apache の設定 473

24.3 Apache の開始と停止 480

24.4 モジュールのインストール／有効化／設定 482
モジュールのインストール 483 • 有効化と無効化 483 • 基本モジュールと拡張モジュール 484 • マルチプロセッシングモジュール (MPM) 487 • 外部モジュール 488 • コンパイル 489

24.5 CGI スクリプトの有効化 490
Apache の設定 491 • サンプルスクリプトの実行 491 • CGI トラブルシューティング 492

24.6 SSL を利用した暗号化機能付き Web サーバの設定 492
SSL 証明書の作成 493 • SSL を利用する場合の Apache の設定 497

24.7 同一サーバ内での複数 Apache インスタンスの起動 499

24.8 セキュリティ問題の回避 502
最新版のソフトウエアの使用 502 • DocumentRoot のアクセス権 503 • ファイルシステムのアクセス権 503 • CGI スクリプト 503 • ユーザディレクトリ 504

24.9 トラブルシューティング 504

24.10 さらなる情報 505
Apache 2.4 505 • Apache モジュール 505 • 開発 506

25 YaST による FTP サーバの設定 507

25.1 FTP サーバの開始 508

25.2 FTP の一般的な設定 508

25.3 FTP パフォーマンス設定 509

25.4 認証 509

25.5 詳細設定 510

25.6 さらなる情報 511

26 squid: キャッシュ機能付きプロキシサーバ 512
26.1 プロキシサーバについて 513
squid とセキュリティ 513 • 複数のキャッシュ 513 • インターネットオブジェクトのキャッシュ 514
26.2 システム要件 515
メモリ 515 • CPU 515 • ディスクキャッシュのサイズ 516 • ハードディスク/SSD の構成 516
26.3 squid の基本的な使い方 516
squid の開始 517 • squid が動作しているかどうかの確認 517 • squid の停止／再読み込み／再起動 519 • squid の削除 520 • ローカル DNS サーバ 520
26.4 YaST squid モジュール 521
26.5 squid の設定ファイル 522
一般的な設定オプション 523 • アクセス制御用オプション 526
26.6 透過型プロキシの設定 528
26.7 squid キャッシュマネージャ CGI インターフェイスの使用 (cachemgr.cgi) 530
26.8 Calamaris によるキャッシュレポートの生成 532
26.9 さらなる情報 533
IV モバイルコンピュータ 534
27 Linux でのモバイルコンピューティング 535
27.1 ラップトップ 535
電源管理 535 • 利用環境の変更作業 536 • ソフトウエアオプション 538 • データセキュリティ 543
27.2 モバイルハードウエア 544
27.3 モバイルデバイス (スマートフォンやタブレットなど) 545
28 NetworkManager の使用 546
28.1 NetworkManager の使用例 546

28.2 NetworkManager の有効化と無効化 546

28.3 ネットワーク接続の設定 547
有線ネットワーク接続の管理 549 • 無線ネットワーク接続の管理 549 • Wi-Fi/Bluetooth カードに対するアクセスポイントとしての設定方法 550 • NetworkManager と VPN 550

28.4 NetworkManager とセキュリティ 552
ユーザ接続とシステム接続 552 • パスワードなどの認証情報の保存 553

28.5 よくある質問 553

28.6 トラブルシューティング 554

28.7 さらなる情報 555

29 電源管理 556

29.1 電源管理機能 556

29.2 Advanced Configuration and Power Interface (ACPI) 557
CPU の性能制御 558 • トラブルシューティング 558

29.3 ハードディスクの休止 560

29.4 トラブルシューティング 562
CPU の周波数制御が動作しない 562

A ネットワーク例 563

B GNU ライセンス 564

このガイドについて

このマニュアルでは、openSUSE® Leap に関する一般的な理解を深めるための情報が書かれています。主にシステム管理者や一般家庭のユーザに対して、基本的なシステム管理知識を提供します。また、このマニュアルの様々な箇所には、日々の作業で必要となるアプリケーションや、高度なインストールや設定を行うためのより深い説明などが書かれています。

高度な管理
: YaST のテキストモードでの使用や、コマンドラインからのソフトウエア管理など、より高度な管理方法を説明しています。また、openSUSE Leap に用意されている snapper を利用したシステムのロールバック (巻き戻し) や、高度なストレージ技術なども説明しています。

システム
: お使いの Linux システムにおける各種のコンポーネント (構成部品) と、それらの動作に関するより深い説明を行っています。

サービス
: openSUSE Leap に付属している様々なネットワークサービスやファイルサービスについて、その設定方法を説明しています。

モバイルコンピュータ
: openSUSE Leap でのモバイルコンピューティングに関する紹介と、無線 LAN や電源管理機能を使用する際の様々な方法について説明しています。

1 利用可能なドキュメンテーション

オンラインドキュメンテーション
: ドキュメンテーションは https://doc.opensuse.org で公開されています。ここから直接読むこともできますし、様々な形式でダウンロードを行うこともできます。

> **注記: 最新の更新について**
> ドキュメンテーションの最新版は通常、英語版が先に作成されます。

SUSE ナレッジベース
: 何らかの問題に直面した場合は、https://www.suse.com/support/kb/ からアクセスできる技術情報文書 (TID) をご確認ください。ここからお客様からの問い合わせで発生した、様々な SUSE 社の知識情報が検索できます。

お使いのシステム内

オフライン環境での利用を想定して、お使いのシステム内の `/usr/share/doc/release-notes` ディレクトリにはリリースノートが保存されているほか、各パッケージに対するドキュメンテーションが `/usr/share/doc` 内に存在しています。
また、多くのコマンドに対して、マニュアルページ も用意されています。マニュアルページは `man` コマンドで表示することができます。このコマンドの後ろに参照したいコマンド名を指定してください。なお、`man` コマンドがインストールされていない場合は、`sudo zypper install man` を実行してインストールしてください。

2 ドキュメンテーションの改善

このドキュメンテーションに対するご意見だけでなく、改善や追記などの貢献をいただければ幸いです。それぞれ下記のチャンネル経由でお送りいただくことができます:

バグ報告

ドキュメンテーション内に誤記などを見つけた場合は、https://bugzilla.opensuse.org/ で問題を報告してください。
なお、この文書の HTML 版の各章には、問題を報告するための［Report a bug］というアイコンが用意されていますので、こちらをお使いのうえ報告をお願いいたします。これにより、Bugzilla で対象の製品や分類、そしてリンク先などをそれぞれ自動で設定するようになっています。あとは問題点の説明を記述するだけです。
なお、報告には Bugzilla のアカウントが必要となるほか、英語でのやり取りが必要となりますので、あらかじめご了承ください。

貢献

このドキュメンテーションに対して追記もしくは更新すべき具体的な内容をお持ちの場合は、この文書の HTML 版の各章に用意されている［EDIT SOURCE］のリンクをお使いください。これにより、GitHub 内にある 英語版原文の ソースコードを表示し編集することができますので、作業が終わり次第 pull request を送信してください。
なお、GitHub のアカウントが必要となります。

注記: ［Edit Source］は英語版のみに提供される件について

［EDIT SOURCE］のリンクは各文書の英語版原文を編集する目的でのみご利用いただけます。その他の言語については［Report a bug］でお知らせください。

なお、本文書で使用しているドキュメンテーション環境の情報については、リポジトリ内の README をお読みください。

電子メール

本製品のドキュメンテーションに対するフィードバックは、`doc-team@suse.com` でも受け付けております。なお、文書のタイトルと製品のバージョン、およびドキュメンテーションの発行日付をそれぞれご記入ください。また、問題点の報告や記述の追加に関するご提案は、それぞれ概要と対応するセクション番号、およびページ (もしくは URL) をご記入ください。

ヘルプ

openSUSE Leap に対するさらなる支援をご希望の場合は、https://ja.opensuse.org/Portal:Support をご覧ください。

3 文書規約

この文書内では、下記のような記述ルールを使用しています:

- `/etc/passwd` : ディレクトリ名やファイル名を示しています
- `PLACEHOLDER` : `PLACEHOLDER` の箇所は、実際の値に置き換えるべきものであることを示しています
- `PATH` : 環境変数であることを示しています
- `ls` , `--help` : コマンドやオプション、パラメータであることを示しています
- `user` : ユーザ名やグループ名であることを示しています
- **`パッケージ名`** : ソフトウエアのパッケージ名を示しています
- `Alt` , `Alt`–`F1` : キー入力や組み合わせキー入力を示しています; キーはキーボードに書かれているとおりに大文字で示されます
- [ファイル] , [ファイル] › [名前を付けて保存] : メニュー項目やボタンなどを示しています
- 第1章「章のタイトル」: 本ガイド内の他の箇所への参照を示しています。
- 下記は `root` ユーザの権限で実行しなければならないコマンドを示しています。一般ユーザから実行する場合は、これらのコマンドの前に `sudo` を付けることで、`root` で実行できるようになります:

```
# コマンド
> sudo コマンド
```

- 下記は一般ユーザで実行できるコマンドを示しています:

```
> コマンド
```

- また、行末にバックスラッシュ文字（ `\` ）を付けることで、コマンドを複数行に分けて記述している場合もあります。バックスラッシュ文字は、シェルに対して、これ以降にもコマンドが続くことを示す文字になります:

```
> echo a b \
c d
```

- このほか、行頭にプロンプトが書かれたコマンド行に続いて、そのコマンドを実行した場合の出力例を示す場合もあります:

```
> コマンド
出力
```

- 各種の情報について

警告: 警告

実際に実施したりする前に、注意しておかなければならない、きわめて重要な情報を記述しています。セキュリティ面の問題のほか、データを失ってしまう可能性への告知、ハードウエアの損傷の可能性や物理的な障害が発生する可能性を示しています。

重要: 重要な情報

実際に実施する前に注意すべき点を説明しています。

注記: 一般的な情報

一般的な補足情報を示しています。たとえばソフトウエアバージョン間での違いなどを説明しています。

ヒント: その他のヒント

ガイドラインや実践的なアドバイスなど、ヒントとなる情報を示しています。

- 簡潔な補足情報

一般的な補足情報を示しています。たとえばソフトウエアバージョン間での違いなどを説明しています。

ガイドラインや実践的なアドバイスなど、ヒントとなる情報を示しています。

4 ソースコードについて

openSUSE Leap のソースコードは一般的に公開されています。ダウンロード先や詳しい情報については、https://ja.opensuse.org/%E3%82%BD%E3%83%BC%E3%82%B9%E3%82%B3%E3%83%BC%E3%83%89 をお読みください。

5 謝辞

多数の無償貢献のお陰で、Linux 開発者はその開発にあたってグローバルな協力を行うことができています。我々は彼らのそのような努力に感謝します。彼らの貢献がなければ本ディストリビューションは存在していませんでした。もちろん Linus Torvalds 氏には特に感謝しています。

I 高度な管理

1 テキストモードでの YaST 2

2 コマンドラインツールでのソフトウエア管理 32

3 Snapper によるシステムの復元とスナップショット管理 61

4 VNC によるリモートグラフィカルセッション 104

5 [熟練者向けパーティション設定] 120

6 複数バージョンのカーネルのインストール 144

7 グラフィカルユーザインターフェイス 152

1 テキストモードでの YaST

改訂履歴

2024-05-13

ncurses ベースの擬似端末型 YaST インターフェイスは主に、X サーバを動作させないサーバ用途でのシステム管理の際に使用します。このテキストモードでは、通常の GUI と比較すると、いくつかの利点があります。テキストモードではキーボードのみを利用してインターフェイスを利用することができるほか、全てのインターフェイス要素に対してキーボードショートカットを使用することができます。またリソースはそれほど消費しない軽量型の構造であり、新しいハードウエアではより高速に動作するようになっています。また SSH 経由で接続して ncurses ベースの YaST を利用することもできますので、遠隔作業にも便利です。ただし、YaST を動作させるのに必要な端末エミュレータの最小サイズは、80x25 文字 (英数字) であることに注意してください。

図 1.1: テキストモードでの YAST のメインウインドウ

YaST の ncurses 版を起動するには、端末を開いて `sudo yast2` と入力して実行します。あとは `<Tab>` キーと矢印キーを利用してメニュー項目やフィールド、ボタンなどを選択していきます。YaST 内での全てのメニュー項目とボタンは、対応するファンクションキーやキーボードショートカットでも選択することができます。たとえば現在の操作をキャンセルしたい場合は `F9` を、変更点を確定させるには `F10` を押します。YaST の ncurses ベースのインターフェイスでは、それぞれのラベル内にハイライト表示された文字がありますが、これがショートカットとして設定されている文字を表します。たとえば [終了 (Q)] というボタンの場合、`Q` という文字がハイライト表示されていますので、`Alt`–`Alt+Q` でそれを押すことができます。

ヒント: YaST ダイアログの再表示について

YaST ダイアログの表示がおかしくなってしまったような場合 (たとえば端末ウインドウのサイズを変更した場合など) は、 Ctrl – L を押してください。これで画面が再表示されて表示が見えるようになります。

1.1 モジュール内の移動

下記に示す YaST モジュールの操作方法の説明は、全てのファンクションキーと Alt キーの組み合わせが動作し、他のグローバル機能などに割り当てられていない前提で記述しています。よくある例外については、1.3項「キーの組み合わせに対する制限」をお読みください。

ボタンや選択リスト間での移動

ボタンや選択リストを含むフレーム間での移動には、 <Tab> を使用します。逆順で移動したい場合は、 Alt – <Tab> もしくは Shift – <Tab> の組み合わせをお使いください。

選択リスト内での移動

選択リストを含むフレームを選択していて、個別の要素間を移動したい場合は、 ↑ と ↓ を使用します。要素が枠内に入りきっていない場合は、 Shift – → または Shift – ← を押すことで、それぞれ左右にスクロールすることができます。矢印キーでは選択が他のフレームに移動してしまうような場合は、 Ctrl – E や Ctrl – A を使用することもできます。

ボタン／ラジオボタン／チェックボックスでの作業

空の四角形が描かれたボタン (チェックボックス) や空の括弧が描かれたボタン (ラジオボタン) を選択するには、 Space または Enter を押します。それ以外にも、ラジオボタンやチェックボックスは Alt –ライト表示された を押しても選択することができます。この場合は、 Enter を押す必要はありません。また、 <Tab> で項目間を移動して Enter を押すと、選択したアクションの実行や関連するメニュー項目の表示などを行うことができます。

ファンクションキー

ファンクションキー (F1 ... F12) は、様々なボタンに対する素早いアクセス機能を提供するものです。利用可能なファンクションキーの組み合わせ (FX) は、YaST 画面の最下部に表示されます。YaST のモジュールによって様々なボタン ([詳細] , [情報] , [追加] , [削除] など) が用意されていることから、ファンクションキーの割り当てもモジュールごとに異なります。一般的には F10 が [了解] , [OK] , [次へ] , [完了] のいずれかに割り当てられています。また F1 を押すと、YaST のヘルプにアクセスすることができます。

ナビゲーションツリーの使用

いくつかの YaST モジュールでは、ウインドウの左側でツリー構造を表示して、右側にそれぞれの設定ダイアログを選択するナビゲーションツリー形式のものがあります。このツリー表示では、矢印 (カーソル) キー (↑ および ↓) を利用して選択し、ツリーの内容を開くには Space を押してください。なお、ncurses モードでは、選択したダイアログを開くには、 Enter を押さなければならないことに注意してください。これは再描画に伴う時間消費を削減するためのもので、敢えて実装している仕組みです。

ソフトウエアインストールモジュールでのソフトウエアのインストール

左側のフィルタを利用して、表示されるパッケージ数を制限することができます。インストール済みのパッケージには i という文字が現れます。パッケージの状態を変更するには、 Space もしくは Enter を押してください。それ以外にも、［アクション］メニューから必要な状態変更 (インストール／削除／更新／禁止／ロック (施錠)) を行うこともできます。

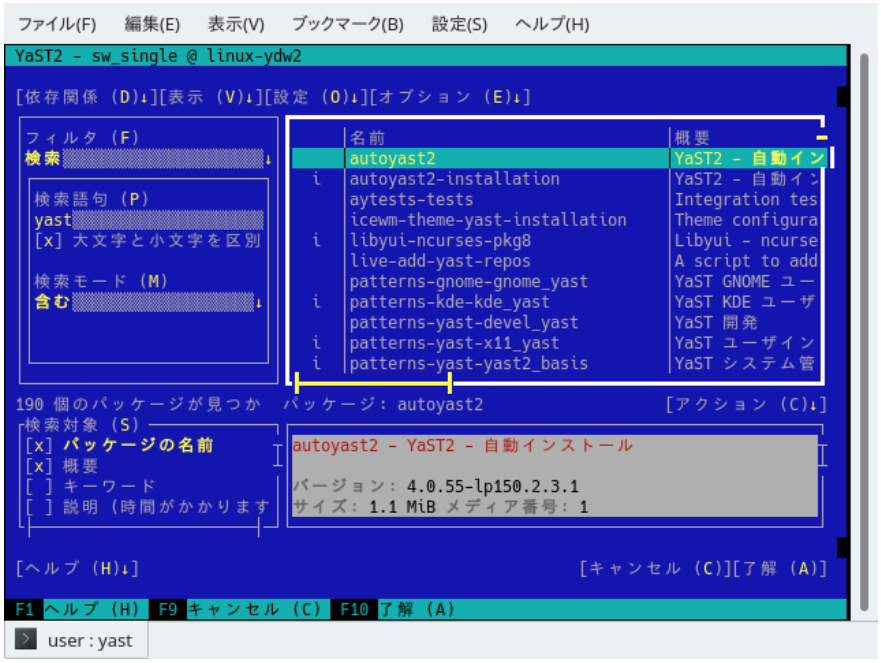


図 1.2: ソフトウエアインストールモジュール

1.2 高度なキーの組み合わせ

ncurses モードでの YaST では、キー入力の組み合わせで高度な操作を行うことができます。

Shift – F1

高度なホットキーの一覧を表示します。

`Shift` – `F4`

色スキーマを変更します。

`Ctrl` – `Q`

アプリケーションを終了します。

`Ctrl` – `L`

画面を再表示します。

`Ctrl` – `D` `F1`

高度なホットキーの一覧を表示します。

`Ctrl` – `D` `Shift` – `D`

ダイアログをスクリーンショットとしてログファイルに保存します。

`Ctrl` – `D` `Shift` – `Y`

YDialogSpy を開いてウイジェットの階層構造を確認します。

1.3 キーの組み合わせに対する制限

お使いのウインドウマネージャがグローバルショートカットとして `Alt` の組み合わせを使用するような場合、YaST での `Alt` の操作が動作しない場合があります。また、端末側でも `Alt` や `Shift` が占有されてしまうことがあります。

`Alt` の `Esc` への置き換えについて

`Alt` を押すショートカットの場合、`Alt` を `Esc` に置き換えて実行することができます。たとえば `Alt` – `H` を `Esc` – `H` に置き換えることができます。この場合、`Esc` を押してから、その後に `H` を押します。

前後に進む操作についても、`Ctrl` – `F` と `Ctrl` – `B` に置き換えることができます。

`Alt` と `Shift` の組み合わせがウインドウマネージャや端末側に占有されてしまっている場合は、`Ctrl` – `F` (前) および `Ctrl` – `B` (後ろ) を代替として使用することができます。

ファンクションキーの制限について

ファンクションキー (`F1` ... `F12`) を利用して様々な機能にアクセスすることもできます。ただし、端末側で特定のファンクションキーが占有されてしまっていて、YaST 側で利用できない場合があります。しかしながら、`Alt` の組み合わせやファンクションキーは、テキストコンソールであれば全て利用することができます。

1.4 YaST コマンドラインオプション

YaST はテキストモードのインターフェイスだけでなく、コマンドラインインターフェイスも用意しています。YaST のコマンドラインオプションの一覧を取得するには、下記のように入力して実行します:

```
> sudo yast -h
```

1.4.1 コマンドラインからのパッケージのインストール

パッケージ名がわかっていて、かつ有効なリポジトリからそのパッケージが提供されている場合は、コマンドラインオプション `-i` を指定して、パッケージをインストールすることができます:

```
> sudo yast -i パッケージ名
```

もしくは、下記のように実行してもかまいません:

```
> sudo yast --install -i パッケージ名
```

`パッケージ名` には単一のパッケージ名を指定します (例: `gvim`) 。この際、依存関係で確認できたパッケージも自動的にインストールすることができます。また、RPM パッケージのフルパスを指定することもできます。この場合は、依存関係の確認は行わずにインストールします。

YaST では、コマンドラインによるソフトウエア管理機能は基本的なものしか提供されていません。より高度なパッケージ管理作業を行いたい場合は、zypper をお使いください。zypper に関する詳細は2.1項「zypper の使用」で説明しています。

1.4.2 個別のモジュールでの作業

時間の節約のため、YaST では個別のモジュールを直接起動することもできます。モジュールを指定するには、下記のように実行します:

```
> sudo yast モジュール名
```

お使いのシステムで利用可能な全てのモジュールを一覧表示するには、`yast -l` もしくは `yast --list` のように実行します。

1.4.3 YaST モジュールでのコマンドラインパラメータ

YaST の機能をスクリプト内から使用できるようにする目的で、YaST にはモジュールごとにコマンドラインサポートが提供されています。ただし、全てのモジュールがコマンドラインに対応しているわけではありません。モジュールで利用可能なオプションを一覧表示するには、下記のように入力して実行します:

```
> sudo yast モジュール名 help
```

モジュールがコマンドラインに対応していない場合は、下記のように表示されます:

```
この YaST モジュールは、コマンドラインインターフェイスには対応していません。
```

下記には、コマンドラインに対応している全ての YaST モジュールに関する説明が書かれています。また、利用可能な全てのコマンドとオプションについて、簡潔な説明も書いてあります。

1.4.3.1 一般的な YaST モジュールのコマンド

全ての YaST モジュールに対して、下記のコマンドが用意されています:

help
: モジュール側で対応しているコマンドと、その説明の一覧を表示します:

```
> sudo yast lan help
```

longhelp
: `help` と同じですが、各コマンドのオプションに関する詳細と説明が追加されます:

```
> sudo yast lan longhelp
```

xmlhelp
: `longhelp` と同じですが、出力が XML 文書として構造化されます。ファイルに出力して使用します:

```
> sudo yast lan xmlhelp xmlfile=/tmp/yast_lan.xml
```

interactive
: 対話 モードに入ります。これにより、いちいち `sudo yast` 等と前置きすることなく、モジュールのコマンドを直接 (`sudo yast ...` 無しで) 実行できるようになります。対話モードを終了するには `exit` と入力してください。

1.4.3.2 yast add-on

指定したパスから新しいアドオン製品を追加します:

```
> sudo yast add-on http://server.name/directory/Lang-AddOn-CD1/
```

ソースパスの指定には、http:// ftp:// nfs:// disk:// cd:// dvd:// などのプロトコルを指定することができます。

1.4.3.3 yast audit-laf

Linux 監査 (Audit) フレームワークの設定を行うことができます。詳しくは『セキュリティ強化ガイド』をお読みください。`yast audit-laf` では、下記のコマンドに対応しています:

set
: オプションを設定するには、下記のように実行します:

  ```
  > sudo yast audit-laf set log_file=/tmp/audit.log
  ```

  全てのオプションの一覧を取得するには、`yast audit-laf set help` と入力して実行します。

show
: オプションの設定値を表示するには、下記のように実行します:

  ```
  > sudo yast audit-laf show diskspace
  space_left: 75
  space_left_action: SYSLOG
  admin_space_left: 50
  admin_space_left_action: SUSPEND
  action_mail_acct: root
  disk_full_action: SUSPEND
  disk_error_action: SUSPEND
  ```

  全てのオプションの一覧を取得するには、`yast audit-laf show help` と入力して実行します。

1.4.3.4 yast dhcp-server

DHCP サーバの管理と設定を行います。`yast dhcp-server` では、下記のコマンドに対応しています:

disable
: DHCP サーバサービスを無効化します。

enable
: DHCP サーバサービスを有効化します。

host
: 個別のホストに対して設定を行います。

interface
: どのネットワークインターフェイスでサービスを提供するのかを指定します:

```
> sudo yast dhcp-server interface current
選択したインターフェイス: eth0
その他のインターフェイス: bond0, pbu, eth1
```

: 全てのオプションの一覧を取得するには、`yast dhcp-server interface help` と入力して実行します。

options
: グローバルな DHCP オプションを管理します。全てのオプションの一覧を取得するには、`yast dhcp-server options help` と入力して実行します。

status
: DHCP サービスの状態を表示します。

subnet
: DHCP サブネットのオプションを管理します。全てのオプションの一覧を取得するには、`yast dhcp-server subnet help` と入力して実行します。

1.4.3.5 yast dns-server

DNS サーバの設定を行います。`yast dns-server` では、下記のコマンドに対応しています:

acls
: アクセス制御リストの設定を表示します:

```
> sudo yast dns-server acls show
ACL:
-----
 名前          種類     値
 ----------------------------
 any         事前定義
 localips    事前定義
 localnets   事前定義
 none        事前定義
```

dnsrecord
: ゾーンのレコードを設定します:

```
> sudo yast dnsrecord add zone=example.org query=office.example.org type=NS value=ns3
```

全てのオプションの一覧を取得するには、`yast dns-server dnsrecord help` と入力して実行します。

forwarders
: DNS フォワーダの設定を行います:

```
> sudo yast dns-server forwarders add ip=10.0.0.100
> sudo yast dns-server forwarders show
[...]
フォワーダ IP
------------
10.0.0.100
```

全てのオプションの一覧を取得するには、`yast dns-server forwarders help` と入力して実行します。

host
: 「A」レコードと対応する「PTR」レコードを一括で処理します:

```
> sudo yast dns-server host show zone=example.org
```

全てのオプションの一覧を取得するには、`yast dns-server host help` と入力して実行します。

logging
: ログ関連の設定を行います:

```
> sudo yast dns-server logging set updates=no transfers=yes
```

全てのオプションの一覧を取得するには、`yast dns-server logging help` と入力して実行します。

mailserver
: ゾーンのメールサーバを設定します:

```
> sudo yast dns-server mailserver add zone=example.org mx=mx1 priority=100
```

全てのオプションの一覧を取得するには、`yast dns-server mailserver help` と入力して実行します。

nameserver
: ゾーンのネームサーバを設定します:

```
> sudo yast dns-server nameserver add zone=example.com ns=ns1
```

全てのオプションの一覧を取得するには、`yast dns-server nameserver help` と入力して実行します。

soa
: start of authority (SOA) レコードの設定を行います:

```
> sudo yast dns-server soa set zone=example.org serial=2006081623 ttl=2D3H20S
```

全てのオプションの一覧を取得するには、`yast dns-server soa help` と入力して実行します。

startup
: DNS サーバサービスの管理を行います:

```
> sudo yast dns-server startup atboot
```

全てのオプションの一覧を取得するには、`yast dns-server startup help` と入力して実行します。

transport
: ゾーンの転送ルールを設定します。全てのオプションの一覧を取得するには、`yast dns-server transport help` と入力して実行します。

zones
: DNS ゾーンを管理します:

```
> sudo yast dns-server zones add name=example.org zonetype=master
```

全てのオプションの一覧を取得するには、`yast dns-server zones help` と入力して実行します。

1.4.3.6 yast disk

全てのディスクやパーティションに対する情報を表示します。対応するコマンドは `list` だけで、下記のいずれかのオプションを後ろに指定します:

disks
: システム内で設定済みの全てのディスクを一覧表示します:

```
> sudo yast disk list disks
```

```
Device   | Size       | FS Type | Mount Point | Label | Model
---------+------------+---------+-------------+-------+-------------
/dev/sda | 119.24 GiB |         |             |       | SSD 840
/dev/sdb |  60.84 GiB |         |             |       | WD1003FBYX-0
```

partitions

システム内の全てのパーティションを一覧表示します:

```
> sudo yast disk list partitions
Device         | Size       | FS Type | Mount Point | Label | Model
---------------+------------+---------+-------------+-------+------
/dev/sda1      |   1.00 GiB | Ext2    | /boot       |       |
/dev/sdb1      |   1.00 GiB | Swap    | swap        |       |
/dev/sdc1      | 698.64 GiB | XFS     | /mnt/extra  |       |
/dev/vg00/home | 580.50 GiB | Ext3    | /home       |       |
/dev/vg00/root | 100.00 GiB | Ext3    | /           |       |
[...]
```

1.4.3.7 yast ftp-server

FTP サーバの設定を行います。`yast ftp-server` では、下記のコマンドに対応しています:

SSL, TLS

SSL や TLS を利用した機密保持接続を制御します。SSL オプションは `vsftpd` でのみ使用することができます。

```
> sudo yast ftp-server SSL enable
> sudo yast ftp-server TLS disable
```

access

アクセス権を設定します:

```
> sudo yast ftp-server access authen_only
```

全てのオプションの一覧を取得するには、`yast ftp-server access help` と入力して実行します。

anon_access

匿名ユーザに対するアクセス権を設定します:

```
> sudo yast ftp-server anon_access can_upload
```

全てのオプションの一覧を取得するには、`yast ftp-server anon_access help` と入力して実行します。

anon_dir
: 匿名ユーザに対するディレクトリを指定します。ディレクトリはサーバ内に既に存在しているものでなければなりません:

```
> sudo yast ftp-server anon_dir set_anon_dir=/srv/ftp
```

全てのオプションの一覧を取得するには、`yast ftp-server anon_dir help` と入力して実行します。

chroot
: change root (chroot) 環境を制御します:

```
> sudo yast ftp-server chroot enable
> sudo yast ftp-server chroot disable
```

idle-time
: 現在の接続を終了するまでの、FTP サーバ側での無通信時の最大待機時間を設定します:

```
> sudo yast ftp-server idle-time set_idle_time=15
```

logging
: ログファイルへのログメッセージの保存可否を設定します:

```
> sudo yast ftp-server logging enable
> sudo yast ftp-server logging disable
```

max_clients
: 最大同時接続クライアント数を設定します:

```
> sudo yast ftp-server max_clients set_max_clients=1500
```

max_clients_ip
: IP アドレスごとの最大同時接続クライアント数を設定します:

```
> sudo yast ftp-server max_clients_ip set_max_clients=20
```

max_rate_anon
: 匿名ユーザに対して許可する最大伝送速度を KB/s 単位で設定します:

```
> sudo yast ftp-server max_rate_anon set_max_rate=10000
```

max_rate_authen
: ローカルでの認証済みユーザに対して許可する最大伝送速度を KB/s 単位で設定します:

```
> sudo yast ftp-server max_rate_authen set_max_rate=10000
```

port_range
: パッシブ接続への応答に使用するポート範囲を設定します:

```
> sudo yast ftp-server port_range set_min_port=20000 set_max_port=30000
```

全てのオプションの一覧を取得するには、`yast ftp-server port_range help` と入力して実行します。

show
: FTP サーバの設定を表示します。

startup
: FTP の起動方法を設定します:

```
> sudo yast ftp-server startup atboot
```

全てのオプションの一覧を取得するには、`yast ftp-server startup help` と入力して実行します。

umask
: **認証済みユーザ:匿名ユーザ** の形式で、それぞれに適用する umask を設定します:

```
> sudo yast ftp-server umask set_umask=177:077
```

welcome_message
: FTP サーバへの接続時に表示するテキストメッセージを設定します:

```
> sudo yast ftp-server welcome_message set_message="hello everybody"
```

1.4.3.8 yast http-server

HTTP サーバ (apache2) の設定を行います。`yast http-server` では、下記のコマンドに対応しています:

configure
: HTTP サーバのホスト側設定を行います:

```
> sudo yast http-server configure host=main servername=www.example.com \
 serveradmin=admin@example.com
```

全てのオプションの一覧を取得するには、`yast http-server configure help` と入力して実行します。

hosts

仮想ホストを設定します:

```
> sudo yast http-server hosts create servername=www.example.com \
 serveradmin=admin@example.com documentroot=/var/www
```

全てのオプションの一覧を取得するには、`yast http-server hosts help` と入力して実行します。

listen

HTTP サーバが待ち受けるべきポートとネットワークアドレスを設定します:

```
> sudo yast http-server listen add=81
> sudo yast http-server listen list
待ち受け設定:
==================
:80
:81
> sudo yast http-server delete=80
```

全てのオプションの一覧を取得するには、`yast http-server listen help` と入力して実行します。

mode

ウイザードモードの有効／無効を設定します:

```
> sudo yast http-server mode wizard=on
```

modules

Apache サーバモジュールの設定を行います:

```
> sudo yast http-server modules enable=php5,rewrite
> sudo yast http-server modules disable=ssl
> sudo http-server modules list
[...]
有効 rewrite
無効 ssl
有効 php5
[...]
```

1.4.3.9 yast kdump

`kdump` の設定を行います。`kdump` について、詳しくは『システム分析／チューニングガイド』、第18章「Kexec と Kdump」、18.7項「基本的な Kexec の設定」をお読みください。`yast kdump` では、下記のコマンドに対応しています:

copykernel
: カーネルをダンプディレクトリにコピーします。

customkernel
: 独自のカーネル名のうち、`カーネル文字列` 部分を設定します。カーネルのファイル名は `/boot/vmlinu[zx]-カーネル文字列[.gz]` のようになります。

```
> sudo yast kdump customkernel kernel=kdump
```

全てのオプションの一覧を取得するには、`yast kdump customkernel help` と入力して実行します。

dumpformat
: ダンプカーネルイメージの (圧縮) 形式を設定します。「none」(なし),「ELF」,「compressed」(圧縮),「lzo」(LZO圧縮) のいずれかを指定します:

```
> sudo yast kdump dumpformat dump_format=ELF
```

dumplevel
: ダンプレベルを 0 から 31 までの間で設定します:

```
> sudo yast kdump dumplevel dump_level=24
```

dumptarget
: ダンプイメージの保存先を設定します:

```
> sudo kdump dumptarget target=ssh server=name_server port=22 \
 dir=/var/log/dump user=user_name
```

全てのオプションの一覧を取得するには、`yast kdump dumptarget help` と入力して実行します。

immediatereboot
: Kdump カーネル内のコアを保存したあと、即時に再起動を行うかどうかを設定します:

```
> sudo yast kdump immediatereboot enable
> sudo yast kdump immediatereboot disable
```

keepolddumps
: 古いダンプイメージをどこまで保存するのかを設定します。0 を指定すると、全てを保存します:

```
> sudo yast kdump keepolddumps no=5
```

kernelcommandline
: Kdump カーネルに渡す必要のあるコマンドラインを設定します:

```
> sudo yast kdump kernelcommandline command="ro root=LABEL=/"
```

kernelcommandlineappend
: 既定のコマンドライン文字列に 追記 する必要のあるコマンドラインを設定します:

```
> sudo yast kdump kernelcommandlineappend command="ro root=LABEL=/"
```

notificationcc
: 通知メッセージのコピーの送信先電子メールアドレスを設定します:

```
> sudo yast kdump notificationcc email="user1@example.com user2@example.com"
```

notificationto
: 通知メッセージの送信先電子メールアドレスを設定します:

```
> sudo yast kdump notificationto email="user1@example.com user2@example.com"
```

show
: `kdump` の設定を表示します:

```
> sudo yast kdump show
kdumpは無効です
ダンプレベル: 31
ダンプ形式: compressed
ダンプ先設定
ターゲット: file
ファイルディレクトリ: /var/crash
Kdump後の即時再起動: 有効
保存する古いダンプ数: 5
```

smtppass
: 通知メッセージを送信する際に使用する、SMTP のパスワードを含むプレーンテキストファイルを設定します:

```
> sudo yast kdump smtppass pass=/path/to/file
```

smtpserver
: 通知メッセージを送信する際に使用する SMTP サーバを設定します:

```
> sudo yast kdump smtpserver server=smtp.server.com
```

smtpuser
: 通知メッセージを送信する際に使用する、SMTP サーバのユーザ名を設定します:

```
> sudo yast kdump smtpuser user=smtp_user
```

startup
: スタートアップオプションの有効／無効を設定します:

```
> sudo yast kdump startup enable alloc_mem=128,256
> sudo yast kdump startup disable
```

1.4.3.10 yast keyboard

仮想コンソール向けのシステムキーボードの設定を行います。ただし、GNOME や KDE などのデスクトップ環境に対しては効果がありません。`yast keyboard` では、下記のコマンドに対応しています:

list
: 利用可能な全てのキーボード配列を一覧表示します。

set
: 新しいキーボード配列を設定します:

```
> sudo yast keyboard set layout=czech
```

summary
: 現在のキーボード設定を表示します。

1.4.3.11 yast lan

ネットワークカードの設定を行います。`yast lan` では、下記のコマンドに対応しています:

add
: 新しいネットワークカードの設定を行います:

```
> sudo yast lan add name=vlan50 ethdevice=eth0 bootproto=dhcp
```

全てのオプションの一覧を取得するには、`yast lan add help` と入力して実行します。

delete
: 既存のネットワークカードの削除を行います:

```
> sudo yast lan delete id=0
```

edit
: 既存のネットワークカードの設定を変更します:

```
> sudo yast lan edit id=0 bootproto=dhcp
```

list
: ネットワークカードの設定概要を表示します:

```
> sudo yast lan list
id name,           bootproto
0 Ethernet Card 0, NONE
1 Network Bridge,  DHCP
```

1.4.3.12 yast language

システム言語の設定を行います。`yast language` では、下記のコマンドに対応しています:

list
: 利用可能な全ての言語を一覧表示します。

set
: 第一言語と第二言語をそれぞれ設定します:

```
> sudo yast language set lang=cs_CZ languages=en_US,es_ES no_packages
```

1.4.3.13 yast mail

メールシステムの設定を表示します:

```
> sudo yast mail summary
```

1.4.3.14 yast nfs

NFS クライアントの設定を行います。`yast nfs` では、下記のコマンドに対応しています:

add
: 新しい NFS マウントを追加します:

```
> sudo yast nfs add spec=remote_host:/path/to/nfs/share file=/local/mount/point
```

全てのオプションの一覧を取得するには、`yast nfs add help` と入力して実行します。

delete
: 既存の NFS マウントを削除します:

```
> sudo yast nfs delete spec=remote_host:/path/to/nfs/share file=/local/mount/point
```

全てのオプションの一覧を取得するには、`yast nfs delete help` と入力して実行します。

edit
: 既存の NFS マウントを変更します:

```
> sudo yast nfs edit spec=remote_host:/path/to/nfs/share \
 file=/local/mount/point type=nfs4
```

全てのオプションの一覧を取得するには、`yast nfs edit help` と入力して実行します。

list
: 既存の NFS マウントを一覧表示します:

```
> sudo yast nfs list
サーバ             リモートディレクトリ   マウントポイント オプション
-------------------------------------------------------------------
nfs.example.com   /mnt                  /nfs/mnt       nfs
nfs.example.com   /home/tux/nfs_share   /nfs/tux       nfs
```

1.4.3.15 yast nfs-server

NFS サーバの設定を行います。`yast nfs-server` では、下記のコマンドに対応しています:

add
: 公開すべきディレクトリを追加します:

```
> sudo yast nfs-server add mountpoint=/nfs/export hosts=*.allowed_hosts.com
```

全てのオプションの一覧を取得するには、`yast nfs-server add help` と入力して実行します。

delete

NFS で公開していたディレクトリを削除します:

```
 > sudo yast nfs-server delete mountpoint=/nfs/export
```

set

NFS サーバに対する追加のパラメータを設定します:

```
> sudo yast nfs-server set enablev4=yes security=yes
```

全てのオプションの一覧を取得するには、`yast nfs-server set help` と入力して実行します。

start

NFS サーバサービスを起動します:

```
> sudo yast nfs-server start
```

stop

NFS サーバサービスを停止します:

```
> sudo yast nfs-server stop
```

summary

NFS サーバの設定概要を表示します:

```
> sudo yast nfs-server summary
NFS サーバは有効です
NFS エクスポート
* /mnt
* /home

NFSv4 support is enabled.
idmap 用の NFSv4 ドメインは localdomain です。
NFS Security using GSS is enabled.
```

1.4.3.16 yast nis

NIS クライアントの設定を行います。`yast nis` では、下記のコマンドに対応しています:

configure

NIS クライアントのグローバル設定を行います:

```
> sudo yast nis configure server=nis.example.com broadcast=yes
```

全てのオプションの一覧を取得するには、`yast nis configure help` と入力して実行します。

disable
: NIS クライアントを無効化します:

```
> sudo yast nis disable
```

enable
: お使いのマシンを NIS クライアントとして有効化します:

```
> sudo yast nis enable server=nis.example.com broadcast=yes automounter=yes
```

全てのオプションの一覧を取得するには、`yast nis enable help` と入力して実行します。

find
: 指定したドメインに対して利用可能な NIS サーバを表示します:

```
> sudo yast nis find domain=nisdomain.com
```

summary
: NIS クライアントの設定概要を表示します。

1.4.3.17 yast nis-server

NIS サーバの設定を行います。`yast nis-server` では、下記のコマンドに対応しています:

master
: NIS マスターサーバの設定を行います:

```
> sudo yast nis-server master domain=nisdomain.com yppasswd=yes
```

全てのオプションの一覧を取得するには、`yast nis-server master help` と入力して実行します。

slave
: NIS ワーカーサーバの設定を行います:

```
> sudo yast nis-server slave domain=nisdomain.com master_ip=10.100.51.65
```

全てのオプションの一覧を取得するには、`yast nis-server slave help` と入力して実行します。

stop
: NIS サーバを停止します:

```
> sudo yast nis-server stop
```

summary
: NIS サーバの設定概要を表示します:

```
> sudo yast nis-server summary
```

1.4.3.18 yast proxy

プロキシの設定を行います。`yast proxy` では、下記のコマンドに対応しています:

authentication
: プロキシに対する認証オプションを設定します:

```
> sudo yast proxy authentication username=tux password=secret
```

全てのオプションの一覧を取得するには、`yast proxy authentication help` と入力して実行します。

enable, disable
: プロキシ設定を有効または無効にします。

set
: 現在のプロキシ設定を変更します:

```
> sudo yast proxy set https=proxy.example.com
```

全てのオプションの一覧を取得するには、`yast proxy set help` と入力して実行します。

summary
: プロキシ設定を表示します。

1.4.3.19 yast rdp

リモートデスクトップの設定を行います。`yast rdp` では、下記のコマンドに対応しています:

allow
: サーバのデスクトップへのリモートアクセスを許可します:

```
> sudo yast rdp allow set=yes
```

list
: リモートデスクトップの設定概要を表示します。

1.4.3.20 yast samba-client

Samba クライアントの設定を行います。`yast samba-client` では、下記のコマンドに対応しています:

configure
: Samba のグローバル設定を変更します:

```
> sudo yast samba-client configure workgroup=FAMILY
```

isdomainmember
: マシンがドメインのメンバーであるかどうかを確認します:

```
> sudo yast samba-client isdomainmember domain=SMB_DOMAIN
```

joindomain
: マシンをドメインのメンバーにします:

```
> sudo yast samba-client joindomain domain=SMB_DOMAIN user=username password=pwd
```

winbind
: Winbind サービス (`winbindd` デーモン) を有効または無効にします:

```
> sudo yast samba-client winbind enable
> sudo yast samba-client winbind disable
```

1.4.3.21 yast samba-server

Samba サーバの設定を行います。`yast samba-server` では、下記のコマンドに対応しています:

backend
: ユーザ情報を保存するためのバックエンドを設定します:

```
> sudo yast samba-server backend smbpasswd
```

全てのオプションの一覧を取得するには、`yast samba-server backend help` と入力して実行します。

configure

Samba サーバのグローバル設定を行います:

```
> sudo yast samba-server configure workgroup=FAMILY description='Home server'
```

全てのオプションの一覧を取得するには、`yast samba-server configure help` と入力して実行します。

list

利用可能な共有の一覧を表示します:

```
> sudo yast samba-server list
状態       種類         名前
============================
無効       ディスク     profiles
有効       ディスク     print$
有効       ディスク     homes
無効       ディスク     groups
有効       ディスク     movies
有効       プリンタ     printers
```

role

Samba サーバの役割を設定します:

```
> sudo yast samba-server role standalone
```

全てのオプションの一覧を取得するには、`yast samba-server role help` と入力して実行します。

service

Samba サービス (`smb` および `nmb`) を有効または無効にします:

```
> sudo yast samba-server service enable
> sudo yast samba-server service disable
```

share

単一の Samba 共有を操作します:

```
> sudo yast samba-server share name=movies browseable=yes guest_ok=yes
```

全てのオプションの一覧を取得するには、`yast samba-server share help` と入力して実行します。

1.4.3.22 yast security

ホストのセキュリティレベルの設定を行います。`yast security` では、下記のコマンドに対応しています:

level
: ホストのセキュリティレベルを設定します:

```
> sudo yast security level server
```

全てのオプションの一覧を取得するには、`yast security level help` と入力して実行します。

set
: 特定のオプションの値を設定します:

```
> sudo yast security set passwd=sha512 crack=yes
```

全てのオプションの一覧を取得するには、`yast security set help` と入力して実行します。

summary
: 現在のセキュリティ設定の概要を表示します:

```
sudoyast security summary
```

1.4.3.23 yast sound

サウンドカードの設定を行います。`yast sound` では、下記のコマンドに対応しています:

add
: 新しいサウンドカードの設定を行います。何もパラメータを指定しないと、最初に検出したサウンドカードを追加します。

```
> sudo yast sound add card=0 volume=75
```

全てのオプションの一覧を取得するには、`yast sound add help` と入力して実行します。

channels
: 利用可能なサウンドカードの音量チャンネルを一覧表示します:

```
> sudo yast sound channels card=0
Master 75
PCM 100
```

modules

利用可能な全てのサウンドカーネルモジュールを一覧表示します:

```
> sudo yast sound modules
snd-atiixp ATI IXP AC97 controller (snd-atiixp)
snd-atiixp-modem ATI IXP MC97 controller (snd-atiixp-modem)
snd-virtuoso Asus Virtuoso driver (snd-virtuoso)
[...]
```

playtest

サウンドカードでテストサウンドを再生します:

```
> sudo yast sound playtest card=0
```

remove

設定済みのサウンドカードを削除します:

```
> sudo yast sound remove card=0
> sudo yast sound remove all
```

set

サウンドカードに対して新しい値を設定します:

```
> sudo yast sound set card=0 volume=80
```

show

サウンドカードに関する詳細情報を表示します:

```
> sudo yast sound show card=0
カード 'ThinkPad X240' のパラメータ (モジュール snd-hda-intel を使用):

align_buffer_size
 Force buffer and period sizes to be multiple of 128 bytes.
bdl_pos_adj
 BDL position adjustment offset.
beep_mode
 Select HDA Beep registration mode (0=off, 1=on) (default=1).
 既定値: 0
enable_msi
 Enable Message Signaled Interrupt (MSI)
[...]
```

summary

システム内にある全てのサウンドカードの設定概要を表示します:

```
> sudo yast sound summary
```

volume

サウンドカードの音量レベルを設定します:

```
sudoyast sound volume card=0 play
```

1.4.3.24 yast sysconfig

`/etc/sysconfig` 以下にあるファイルの変数を制御します。`yast sysconfig` では、下記のコマンドに対応しています:

clear

変数に空の値を設定します:

```
> sudo yast sysconfig clear=POSTFIX_LISTEN
```

ヒント: 複数のファイル内にある変数の場合

複数のファイル内に変数が存在する場合、`変数名 $ ファイル名` の文法を使用してください:

```
> sudo yast sysconfig clear=CONFIG_TYPE$/etc/sysconfig/mail
```

details

変数の対する詳細な情報を表示します:

```
> sudo yast sysconfig details variable=POSTFIX_LISTEN
Description:
値:
ファイル: /etc/sysconfig/postfix
入力可能な値: Any value
既定値:
設定スクリプト: postfix
説明:
 Comma separated list of IP's
 NOTE: If not set, LISTEN on all interfaces
```

list

修正済みの変数の概要を表示します。`all` を指定すると、全ての変数とその値が表示されます:

```
> sudo yast sysconfig list all
AOU_AUTO_AGREE_WITH_LICENSES="false"
AOU_ENABLE_CRONJOB="true"
```

```
AOU_INCLUDE_RECOMMENDS="false"
[...]
```

set

変数に値を設定します:

```
> sudo yast sysconfig set DISPLAYMANAGER=gdm
```

ヒント: 複数のファイル内にある変数の場合

複数のファイル内に変数が存在する場合、`変数名 $ ファイル名` の文法を使用してください:

```
> sudo yast sysconfig set CONFIG_TYPE$/etc/sysconfig/mail=advanced
```

1.4.3.25 yast tftp-server

TFTP サーバの設定を行います。`yast tftp-server` では、下記のコマンドに対応しています:

directory

TFTP サーバのディレクトリを設定します:

```
> sudo yast tftp-server directory path=/srv/tftp
> sudo yast tftp-server directory list
ディレクトリパス: /srv/tftp
```

status

TFTP サーバサービスの状態を制御します:

```
> sudo yast tftp-server status disable
> sudo yast tftp-server status show
サービスの状態: false
> sudo yast tftp-server status enable
```

1.4.3.26 yast timezone

タイムゾーンの設定を行います。`yast timezone` では、下記のコマンドに対応しています:

list

利用可能な全てのタイムゾーンを、地域別に一覧表示します:

```
> sudo yast timezone list
```

```
Region: アフリカ
Africa/Abidjan (アビジャン)
Africa/Accra (アクラ)
Africa/Addis_Ababa (アディスアベバ)
[...]
```

set
: タイムゾーンの設定に対する新しい値を設定します:

```
> sudo yast timezone set timezone=Europe/Prague hwclock=local
```

summary
: タイムゾーンの設定概要を表示します:

```
> sudo yast timezone summary
現在のタイムゾーン: Europe/Prague
ハードウエアの時刻設定: Local time
現在の日付と時刻: Mon 12. March 2018, 11:36:21 CET
```

1.4.3.27 yast users

ユーザアカウントの設定を行います。`yast users` では、下記のコマンドに対応しています:

add
: 新しいユーザを追加します:

```
> sudo yast users add username=user1 password=secret home=/home/user1
```

全てのオプションの一覧を取得するには、`yast users add help` と入力して実行します。

delete
: 既存のユーザアカウントを削除します:

```
> sudo yast users delete username=user1 delete_home
```

全てのオプションの一覧を取得するには、`yast users delete help` と入力して実行します。

edit
: 既存のユーザアカウントを変更します:

```
> sudo yast users edit username=user1 password=new_secret
```

全てのオプションの一覧を取得するには、`yast users edit help` と入力して実行します。

list

指定したユーザ種類のユーザを一覧表示します:

```
> sudo yast users list system
```

全てのオプションの一覧を取得するには、`yast users list help` と入力して実行します。

show

ユーザに関する詳細を表示します:

```
> sudo yast users show username=wwwrun
フルネーム: WWW daemon apache
グループの一覧: www
既定のグループ: wwwrun
ホームディレクトリ: /var/lib/wwwrun
ログインシェル: /sbin/nologin
ログイン名: wwwrun
UID: 456
```

全てのオプションの一覧を取得するには、`yast users show help` と入力して実行します。

2 コマンドラインツールでのソフトウエア管理

改訂履歴

2022-02-11

本章では、いずれもソフトウエアを管理するためのコマンドラインツールである Zypper と RPM について説明しています。なお、本章内の各種用語 (たとえば リポジトリ や 修正, 更新 など) については、**『スタートアップ』、第9章「ソフトウエアのインストールと削除」、9.1 項「用語の定義」** をお読みください。

2.1 zypper の使用

zypper はパッケージのインストールや更新、削除などを行うことができる、コマンドライン型のパッケージマネージャです。特にリモートからのソフトウエア管理作業や、シェルスクリプトからのソフトウエア管理に便利な仕組みです。

2.1.1 一般的な使用方法

zypper の一般的な書式は、下記のとおりです:

```
zypper [--グローバルオプション] コマンド  [--コマンドオプション] [パラメータ]
```

[] で括られた箇所は任意指定のもので、必ずしも指定する必要があるものではありません。オプションとコマンドの一覧については、`zypper help` を実行してください。また、特定のコマンドに対してヘルプを表示したい場合は、`zypper help コマンド` と入力して実行してください。

zypper のコマンド

zypper で最も簡単な使用方法は、コマンドに続いて名前を入力する方法です。たとえば、システムに対して必要なすべての修正を適用するには、下記のように入力します:

```
> sudo zypper patch
```

グローバルオプション

また、コマンドの直前には 1 つ以上のグローバルオプションを指定することができます:

```
> sudo zypper --non-interactive patch
```

上記の例では `--non-interactive` というグローバルオプションを指定しています。これは、コマンドの実行時に一切の問い合わせを行わない (既定の回答がなされたものとして扱う) オプションです。

コマンド固有のオプション

特定のコマンドに固有のオプションを指定したい場合は、コマンドの直後に指定します:

```
> sudo zypper patch --auto-agree-with-licenses
```

上記の例では `--auto-agree-with-licenses` というオプションを指定しています。これは、システムに対して修正を適用する際、ライセンスの確認を行わない (ライセンスを自動的に受け入れる) オプションです。

パラメータ

コマンドによっては、1 つ以上のパラメータを受け付けるものがあります。たとえば `install` というコマンドでは、インストール したいパッケージを指定します:

```
> sudo zypper install mplayer
```

オプションによっては 1 つのパラメータを必要とするものがあります。たとえば下記のコマンドは、既知の全てのパターンを一覧表示します:

```
> zypper search -t pattern
```

これまでのオプションやパラメータは、組み合わせて指定することもできます。たとえば下記のコマンドは、`factory` という名前のリポジトリから `mc` と `vim` というパッケージをインストールし、インストール時に詳細な出力 (verbose) を行うようにしています:

```
> sudo zypper -v install --from factory mc vim
```

`--from` オプションは、依存関係の解決に際して全てのリポジトリを有効化した状態のまま、指定したリポジトリからパッケージをインストールするためのオプションです。`--repo` オプションは `--from` オプションと同じ意味で、どちらを使用してもかまいません。

また、ほとんどの zypper コマンドには `dry-run` オプションが用意されています。これは、指定したコマンドの実行をシミュレーションするためのもので、テスト用に使用することができます。

```
> sudo zypper remove --dry-run MozillaFirefox
```

このほか、zypper ではグローバルオプションとして `--userdata 文字列` というものも用意されています。このオプションは、指定した文字列を zypper およびプラグイン (たとえば btrfs プラグイン) のログファイル内に書き込み、後からログファイル内を検索しやすくするためのオプションです。

```
> sudo zypper --userdata 文字列 patch
```

2.1.2 zypper のサブコマンドの使用

zypper でのサブコマンドはそれ自身が実行ファイルの形で存在するものであり、`zypper_execdir` 設定オプションで指定したディレクトリ内にあるはずのものとして処理されます。既定ではこの値は `/usr/lib/zypper/commands` に設定されています。サブコマンドが zypper_execdir で指定したディレクトリ内に存在しない場合、zypper は $PATH 環境変数内の残りの箇所を自動的に検索します。これにより、独自の拡張を作成してユーザスペース内に配置することができるようになっています。

ただし、zypper シェル内でサブコマンドを実行したり、zypper のグローバルオプションを使用したりすることはできません。

利用可能なサブコマンドは下記のように実行することで表示することができます:

```
> zypper help subcommand
[...]
'/usr/lib/zypper/commands' 内で利用可能な zypper のサブコマンド

  appstream-cache
  lifecycle
  migration
  search-packages

お使いの $PATH 内の他の場所で利用可能な zypper サブコマンド

  log                    Zypper logfile reader
                             (/usr/sbin/zypper-log)
```

サブコマンドに対するヘルプ画面を表示するには、下記のように実行します:

```
> zypper help appstream-cache
```

2.1.3 zypper を利用したソフトウエアのインストールと削除

パッケージをインストールしたり削除したりしたい場合は、下記のコマンドをお使いください:

```
> sudo zypper install パッケージ名
> sudo zypper remove パッケージ名
```

警告: 必須パッケージの削除について

`glibc`, `zypper` , `kernel` など、システムにとって必須のパッケージを削除しないでください。これらを削除してしまうと、システムが不安定になったり、全く動作しなくなったりしてしまいます。

2.1.3.1 インストールまたは削除するパッケージの選択

`zypper install` や `zypper remove` のコマンドでは、様々な方法でパッケージを指定することができます。

パッケージ名のみ

```
> sudo zypper install MozillaFirefox
```

パッケージ名とバージョン番号

```
> sudo zypper install MozillaFirefox-52.2
```

リポジトリの別名とパッケージ名

```
> sudo zypper install mozilla:MozillaFirefox
```

ここで、`mozilla` はインストール元となるリポジトリの別名を意味します。

ワイルドカードを使用したパッケージ名

特定の文字列で始まる、もしくは特定の文字列で終わる全てのパッケージをインストールすることができます。なお、特にパッケージを削除する際には、よく注意して使用してください。たとえば下記のコマンドを実行すると、「Moz」で始まる全てのパッケージをインストールします:

```
> sudo zypper install 'Moz*'
```

ヒント: 全ての -debuginfo パッケージの削除

何らかの問題を調査するにあたっては、実行中のプロセスから様々な情報を取得する目的で、多数の `-debuginfo` パッケージを一時的にインストールしなければならない場合があります。デバッグの作業が終わると、これらは不要なパッケージとなりますので、環境を元に戻すために下記のようにコマンドを実行することができます:

```
> sudo zypper remove '*-debuginfo'
```

能力による指定

パッケージ名が分からない場合は、能力による指定を行うこともできます。たとえば下記のコマンドを実行すると、`MozillaFirefox` パッケージをインストールします:

```
> sudo zypper install firefox
```

能力やハードウエアアーキテクチャ、バージョンによる指定

能力とハードウエアアーキテクチャ、バージョンをそれぞれ組み合わせて指定することもできます:

- 能力の後ろにピリオド (.) を置き、続いて必要なハードウエアアーキテクチャを指定してください。たとえば AMD64/Intel 64 アーキテクチャ (zypper では `x86_64` と命名されています) を指定するには、下記のようにします:

```
> sudo zypper install 'firefox.x86_64'
```

- バージョン番号は文字列の末尾に指定するほか、合わせて下記のいずれかの演算子を指定します: `<` (より小さい), `<=` (より小さいか等しい), `=` (等しい), `>=` (より大きいか等しい), `>` (より大きい)

```
> sudo zypper install 'firefox>=74.2'
```

- ハードウエアアーキテクチャとバージョン番号の両方を指定することもできます:

```
> sudo zypper install 'firefox.x86_64>=74.2'
```

RPM ファイルのパスによる指定

パッケージに対するローカルのパス、もしくはリモートのパスを指定することもできます:

```
> sudo zypper install /tmp/install/MozillaFirefox.rpm
> sudo zypper install http://download.example.com/MozillaFirefox.rpm
```

2.1.3.2 パッケージのインストールと削除の組み合わせ

パッケージのインストールと削除を同時に実施したい場合は、`+/-` の修飾子を指定します。たとえば `emacs` パッケージをインストールし、同時に `vim` パッケージを削除したい場合は、下記のようにします:

```
> sudo zypper install emacs -vim
```

`emacs` パッケージを削除し、同時に `vim` パッケージをインストールしたい場合は、下記のようにします:

```
> sudo zypper remove emacs +vim
```

`-` でパッケージの削除を指定する場合は、これがコマンドオプションと取り違えられたりしないようにするため、`-` を 2 つめ以降に指定するようにしてください。2 つめ以降に書くことが難しい場合は、`--` でオプションの終了を表してもかまいません:

```
> sudo zypper install -emacs +vim       # 誤り
```

```
> sudo zypper install vim -emacs        # 正しい
> sudo zypper install -- -emacs +vim    # 正しい
> sudo zypper remove emacs +vim         # 正しい
```

2.1.3.3 削除したパッケージが必要としていた依存関係の清掃

特定のパッケージ名の指定と組み合わせることで、削除後に不要となるパッケージを自動的に削除させることもできます。これを行いたい場合は、`--clean-deps` オプションを指定してください:

```
> sudo zypper rm --clean-deps パッケージ名
```

2.1.3.4 スクリプト内での zypper の使用

既定では、zypper は選択したパッケージをインストール／削除する前、および何らかの問題が発生した場合に、確認メッセージを表示します。確認メッセージの表示は、`--non-interactive` オプションを指定することで省略することができます。ただし、このオプションは実際のコマンド (たとえば `install`, `remove`, `patch` など) よりも前に指定しなければなりません。具体的には、下記のようになります:

```
> sudo zypper --non-interactive install パッケージ名
```

このオプションを指定することで、zypper をスクリプト内や cron のジョブ内で利用できるようになります。

2.1.3.5 ソースパッケージのインストールとダウンロード

該当するソースパッケージをインストールしたい場合は、下記のように実行します:

```
> zypper source-install パッケージ名
```

上記のコマンドを `root` で実行すると、ソースパッケージは既定で `/usr/src/packages/` 以下にインストールされます。それ以外のユーザで実行すると、`~/rpmbuild` 以下にインストールされます。これらの設定は、`rpm` の設定で変更することができます。

このコマンドは、指定したパッケージが構築時に依存するパッケージについても、インストールを行います。これを回避したい場合は、`-D` スイッチを指定してください:

```
> sudo zypper source-install -D パッケージ名
```

構築時の依存関係のみをインストールしたい場合は、`-d` を指定してください。

```
> sudo zypper source-install -d パッケージ名
```

もちろん、これらの処理はソースパッケージのあるリポジトリが有効化されている場合にのみ動作するものです (ソースパッケージのあるリポジトリは、既定で追加されていますが、有効化はされていません) 。リポジトリの管理について、詳しくは 2.1.6項「zypper を利用したリポジトリの管理」をお読みください。

お使いのリポジトリ内にある全てのソースパッケージを一覧表示するには、下記のコマンドを実行してください:

```
> zypper search -t srcpackage
```

また、インストール済みの全てのパッケージに対して、対応するソースパッケージをダウンロードすることもできます。ソースパッケージをダウンロードしたい場合は、下記のコマンドをお使いください:

```
> zypper source-download
```

既定のダウンロード先ディレクトリは `/var/cache/zypper/source-download` に設定されていますが、`--directory` オプションを指定することで、自由に変更することができます。また、ダウンロードや削除を行うことなく、必要もしくは不要なパッケージを確認したい場合は、`--status` オプションを指定してください。不要なソースパッケージを削除したい場合は、`--delete` オプションを指定してください。削除を行いたくない場合は、`--no-delete` を指定してください。

2.1.3.6 無効化されているリポジトリからのパッケージのインストール

通常は、パッケージをインストールしたり更新したりする場合、有効化されたリポジトリのパッケージのみを使用することができます。ですが、`--plus-content タグ` オプションを指定すると、その zypper の実行時に限定して一時的にリポジトリを有効化し、インストールや交信を行うことができるようになります。zypper のコマンドが終了すると元に戻されます (無効化されます) 。

たとえば `-debuginfo` や `-debugsource` などのパッケージを提供する追加リポジトリを有効化したい場合は、`--plus-content debug` オプションを指定してください。なお、`--plus-content` は必要に応じて複数回指定することができます。

このような 'debug' (デバッグ) リポジトリを有効化して、特定の `-debuginfo` パッケージをインストールするには、下記のように実行します:

```
> sudo zypper --plus-content debug \
   install "debuginfo(build-id)=eb844a5c20c70a59fc693cd1061f851fb7d046f4"
```

ここで、`build-id` 以下の文字列は、`gdb` コマンドで表示される ID を指定します。

注記: 無効化されているインストールメディアについて

openSUSE Leap のインストールメディアを表すリポジトリが設定されますが、無効化された状態になります。オンラインのリポジトリではなく、インストールメディアからインストールを行いたい場合は、`--plus-content` オプションをお使いください。なお、`zypper` コマンドを使用する際には、あらかじめドライブに DVD を挿入しておくなど、メディア側を用意してから実行してください。

2.1.3.7 ユーティリティ

依存関係が全て満たされていることを確認し、満たされていない場合は修復作業を実施したい場合は、下記のコマンドを実行します:

```
> zypper verify
```

パッケージには、満たすべき依存関係だけでなく、「推奨」という設定も用意されています。これら推奨パッケージは、利用可能でインストールできる場合にのみインストールされます。特定のパッケージをインストールした後や、ハードウエアを追加したことによって、新しく推奨されるパッケージをインストールするには、下記のコマンドを実行します:

```
> sudo zypper install-new-recommends
```

このコマンドは、たとえば Web カメラや無線 LAN デバイスを接続したような場合に有用です。これにより、デバイスを利用するためのドライバと、関連するソフトウエアがそれぞれ存在すれば、インストールされるようになります。なお、ドライバと関連ソフトウエアは、ハードウエア要件が合致した場合にのみインストールされます。

2.1.4 zypper を利用したソフトウエアの更新

zypper では 3 種類の方法でソフトウエアを更新することができます。1 つめの方法は修正 (パッチ) をインストールする方法で、2 つめは同じパッケージの新しいバージョンをインストールする方法、そして最後はディストリビューション全体を更新する方法です。ディストリビューション全体を更新する場合は `zypper dist-upgrade` を使用しますが、こちらに関しては『スタートアップ』、第12章「システムのアップグレードとシステムの変更」をお読みください。

2.1.4.1 必要な全パッケージのインストール

openSUSE Leap に 修正を適用する 作業は、インストール済みのパッケージを新しいバージョンに更新する場合の最も信頼できる方法です。必要な全てのパッケージが適切なバージョンになるよう保証し、競合する (矛盾もしくは衝突の発生する) バージョンのパッケージは除外されるようになります。

お使いのシステムに対して、提供されている全ての修正をインストールするには、下記のコマンドを実行します:

```
> sudo zypper patch
```

お使いのコンピュータに設定されているリポジトリを利用して、提供されている全ての修正を現在のインストール状態と比較して、インストール作業を行います。これらの修正が必要なものであれば (`optional` (任意指定) または `feature` (機能追加) でなければ) 、それらを即時にインストールします。`zypper patch` コマンドが正常に終了すると、敢えてインストールしないように選択したものを除いて、全てのパッケージから脆弱性が取り除かれることになります。

なお、インストールしようとしている修正がシステムの再起動を必要とするような変更を伴う場合は、その旨が事前に警告されます。

また、`zypper patch` コマンドは、サードパーティ製のリポジトリが提供する修正については、適用を行いません。サードパーティ製のリポジトリについても更新を行いたい場合は、`with-update` というコマンドオプションを追加してください。

```
> sudo zypper patch --with-update
```

任意指定の修正もあわせてインストールしたい場合は、下記のように実行します:

```
> sudo zypper patch --with-optional
```

特定の Bugzilla バグ番号に関連する全ての修正をインストールしたい場合は、下記のように実行します:

```
> sudo zypper patch --bugzilla=番号
```

特定の CVE データベースエントリに対応する全ての修正をインストールしたい場合は、下記のように実行します:

```
> sudo zypper patch --cve=番号
```

たとえば、CVE 番号 `CVE-2010-2713` に対応するセキュリティ修正をインストールするには、下記のように実行します:

```
> sudo zypper patch --cve=CVE-2010-2713
```

また、zypper やパッケージ管理システム自身に対する修正のみをインストールするには、下記のように実行します:

```
> sudo zypper patch --updatestack-only
```

なお、`updatestack-only` オプションを指定すると、他のリポジトリが提供する更新をあわせてインストールするようなコマンドオプションは、無効化されることに注意してください。

2.1.4.2 修正の一覧表示

どのような修正が提供されているのかを確認するにあたって、zypper は下記のような情報を提供します:

必要な修正の数
: 必要な修正の数 (お使いのシステムに適用可能で、まだインストールされていないもの) を表示するには、`patch-check` コマンドを使用します:

```
> zypper patch-check
リポジトリのデータを読み込んでいます...
インストール済みのパッケージを読み込んでいます...

5 個のパッチが必要です（1 個のセキュリティパッチ）
```

このコマンドでは、`--updatestack-only` オプションをあわせて指定することができます。これを指定すると、zypper とパッケージ管理システム自身に関わる修正のみを表示します。

必要な修正の一覧表示
: 必要な全ての修正 (お使いのシステムに適用可能で、まだ適用されていないもの) を一覧表示するには、`list-patches` コマンドを使用します:

全ての修正の一覧表示
: 既に適用済みのものを含め、openSUSE Leap に対して提供されている全ての修正を一覧表示するには、`zypper patches` コマンドを使用します。

なお、特定の問題に限定して修正を一覧表示したり、インストールしたりすることもできます。特定の修正のみを一覧表示するには、`zypper list-patches` に続いて下記のオプションを設定してください:

Bugzilla のバグ番号を指定する方法
: 特定の Bugzilla バグ番号に関連する全ての修正を一覧表示するには、`--bugzilla` オプションを指定します。
特定のバグに対する修正を一覧表示するには、`--bugzilla=番号` のようにしてバグ番号を指定します。複数の Bugzilla バグに関連する修正を検索するには、バグ番号をカンマで区切って指定してください。たとえば下記のようになります:

```
> zypper list-patches --bugzilla=972197,956917
```

CVE 番号を指定する方法

CVE (Common Vulnerabilities and Exposures; 汎用脆弱性公開) データベースの項目番号に関連する全ての修正を一覧表示するには、`--cve` オプションを使用します。
特定の CVE データベースエントリに対応する修正を一覧表示するには、`--cve=NUMBER` のようにして CVE 番号を指定します。複数の CVE データベースエントリに関連する修正を検索するには、CVE 番号をカンマで区切って指定してください。たとえば下記のようになります:

```
> zypper list-patches --cve=CVE-2016-2315,CVE-2016-2324
```

競合するパッケージのある修正

```
パッチ openSUSE-SLE-15.3-2022-333 に関する情報:
-------------------------------------------------
リポジトリ    : Update repository with updates from SUSE Linux Enterprise 15
名前          : openSUSE-SLE-15.3-2022-333
バージョン    : 1
アーキテクチャ : noarch
ベンダ        : maint-coord@suse.de
状態          : 必要
カテゴリ      : security
重要度        : important
作成日        : Fri Feb  4 09:30:32 2022
対話性        : reboot
概要          : Security update for xen
説明          :
    This update for xen fixes the following issues:

    - CVE-2022-23033: Fixed guest_physmap_remove_page not removing the p2m mappings.
 (XSA-393) (bsc#1194576)
    - CVE-2022-23034: Fixed possible DoS by a PV guest Xen while unmapping a grant.
 (XSA-394) (bsc#1194581)
    - CVE-2022-23035: Fixed insufficient cleanup of passed-through device IRQs.
 (XSA-395) (bsc#1194588)
提供          : patch:openSUSE-SLE-15.3-2022-333 = 1
競合          : [22]
    xen.src < 4.14.3_06-150300.3.18.2
    xen.noarch < 4.14.3_06-150300.3.18.2
    xen.x86_64 < 4.14.3_06-150300.3.18.2
    xen-devel.x86_64 < 4.14.3_06-150300.3.18.2
    xen-devel.noarch < 4.14.3_06-150300.3.18.2
[...]
```

上記の修正には 22 個のパッケージの特定バージョンに対して競合が設定されていますが、これらにはいずれも脆弱性が存在していたり、その脆弱性によって影響を受けたりするバージョンが指定されています。列挙されている競合のうち、いずれか 1 つでも該当するものがあると、この修正は 必要 として認識されます。`zypper patch` を実行すると、必要とされる全ての修正をインストールしようとします。なお、何らかの問題が発生した場合はその旨を報告し、必要なすべて

の修正がインストールされていない旨を報告します。上記に示されているとおり、競合はパッケージのバージョンを上げるか、もしくはパッケージそのものを削除することで解決できます。SUSE の更新リポジトリには、上記の競合を解決できる修正済みバージョンが提供されていることから、そのまま更新するのが一般的な方法となるわけです。なお、依存関係の問題やパッケージロック (施錠) 等によってパッケージを更新できない場合は、ユーザの承認を得てこれらを取り除こうとします。

不要な修正であるかどうかに関わらず修正を一覧表示したい場合は、`--all` オプションをお使いください。たとえば CVE 番号が割り当てられている全ての修正を一覧表示するには、下記のように実行します:

```
> zypper list-patches --all --cve
リポジトリのデータを読み込んでいます...
インストール済みのパッケージを読み込んでいます...

発信番号   | 番号           | パッチ              | カテゴリ      | 重要度     | 状態
---------+---------------+-------------------+-------------+-----------
+-------------------------------------------------
cve      | CVE-2019-0287 | SUSE-SLE-Module.. | recommended | moderate   | 必要
cve      | CVE-2019-3566 | SUSE-SLE-SERVER.. | recommended | moderate   | 不要
[...]
```

2.1.4.3 新しいバージョンのパッケージのインストール

リポジトリ内に新しいバージョンのパッケージだけが用意されていて、修正としては提供していない場合、`zypper patch` は何も行いません。利用可能な新しいバージョンに更新したい場合は、下記のように実行します:

```
> sudo zypper update
```

重要

`zypper update` では問題のあるパッケージを無視するように動作します。たとえばパッケージがロック (施錠) されている場合は、より新しいバージョンが存在していても `zypper update` では何も行わなくなります。逆に `zypper patch` では、パッケージに脆弱性が含まれていると判断される場合、競合は報告されます。

個別のパッケージを更新したい場合は、update もしくは install のコマンドでパッケージ名を指定してください:

```
> sudo zypper update パッケージ名
> sudo zypper install パッケージ名
```

インストール可能な全ての新しいパッケージを一覧表示するには、下記のように実行します:

```
> zypper list-updates
```

ただし、下記の条件に該当する新しいバージョンだけが表示されることに注意してください:

- 既にインストールされているパッケージと同じベンダ (製造元) であること。
- 既にインストールされているパッケージのリポジトリと同じか、もしくは高い優先順位が設定されていること。
- インストール可能であること (全ての依存関係が満たされていること) 。

インストール可能であるかどうかにかかわらず、利用可能な 全ての 新しいパッケージを一覧表示するには、下記のように実行します:

```
> sudo zypper list-updates --all
```

なぜインストール可能であると判断されないのかについては、上述の `zypper install` または `zypper update` コマンドでパッケージ名を指定するとわかります。

2.1.4.4 孤立したパッケージの検出

zypper を利用してリポジトリを削除したり、システムをアップグレードしたりしたあとは、いくつかのパッケージが「孤立した」状態になることがあります。これらの 孤立した パッケージは、設定されている有効なリポジトリ内には存在していないパッケージを意味します。これらは下記のコマンドを実行することで、一覧表示することができます:

```
> sudo zypper packages --orphaned
```

出力された一覧を元に、それらを現状のまま維持しておくか、不要なパッケージであるものとして安全に削除しておくことができます。

2.1.5 削除したファイルを使用しているプロセスやサービスの検出

修正のインストールやパッケージの更新／削除などを行った場合、更新や削除によって置き換えられたり削除されたりしたファイルを、実行中のプロセスが使用したままになっていることがあります。このような状況を検出するには、`zypper ps` コマンドをお使いください。これにより、削除されたファイルを使用しているプロセスを一覧で表示することができます。なお、プロセスが既知のサービスに属している場合は、サービス側から再起動することができるようにするため、サービス名もあわせて表示されます。既定では `zypper ps` は表形式で出力します:

```
> zypper ps
```

```
PID   | PPID | UID | ユーザ | コマンド      | サービス     | ファイル
------+------+-----+--------+--------------+--------------+-------------------
814   | 1    | 481 | avahi  | avahi-daemon | avahi-daemon | /lib64/ld-2.19.s->
      |      |     |        |              |              | /lib64/libdl-2.1->
      |      |     |        |              |              | /lib64/libpthrea->
      |      |     |        |              |              | /lib64/libc-2.19->
[...]
```

PID: そのプロセスの ID
PPID: 親プロセスの ID
UID: プロセスを実行しているユーザの ID
ユーザ: プロセスを実行しているユーザのログイン名
コマンド: プロセスで使用しているコマンド
サービス: サービス名 (コマンドがシステムサービスに属している場合のみ)
ファイル: 削除されたファイルの一覧

`zypper ps` の出力形式は、下記のように指定することで変更することができます:

`zypper ps -s`

削除されたファイルの一覧を表示せず、短い表形式にします。

```
> zypper ps -s
PID   | PPID | UID  | ユーザ   | コマンド     | サービス
------+------+------+---------+--------------+--------------
814   | 1    | 481  | avahi   | avahi-daemon | avahi-daemon
817   | 1    | 0    | root    | irqbalance   | irqbalance
1567  | 1    | 0    | root    | sshd         | sshd
1761  | 1    | 0    | root    | master       | postfix
1764  | 1761 | 51   | postfix | pickup       | postfix
1765  | 1761 | 51   | postfix | qmgr         | postfix
2031  | 2027 | 1000 | tux     | bash         |
```

`zypper ps -ss`

システムサービスに属しているプロセスのみを表示します。

```
PID   | PPID | UID  | ユーザ   | コマンド      | サービス
------+------+------+---------+--------------+--------------
814   | 1    | 481  | avahi   | avahi-daemon | avahi-daemon
817   | 1    | 0    | root    | irqbalance   | irqbalance
1567  | 1    | 0    | root    | sshd         | sshd
1761  | 1    | 0    | root    | master       | postfix
1764  | 1761 | 51   | postfix | pickup       | postfix
1765  | 1761 | 51   | postfix | qmgr         | postfix
```

`zypper ps -sss`

削除されたファイルを使用しているシステムサービスのみを表示します。

```
avahi-daemon
```

```
irqbalance
postfix
sshd
```

`zypper ps --print "systemctl status %s"`

再起動を必要としているサービスに対して、状態を取得するためのコマンドを表示します。

```
systemctl status avahi-daemon
systemctl status irqbalance
systemctl status postfix
systemctl status sshd
```

サービスの取り扱いについて、詳しくは 第10章「systemd デーモン」をお読みください。

2.1.6 zypper を利用したリポジトリの管理

zypper におけるインストールや修正のインストールは、いずれもリポジトリの設定に依存しています。システムに設定されているリポジトリを一覧表示するには、下記のコマンドを実行します:

```
> zypper repos
```

出力は、たとえば下記のようになります:

例 2.1: ZYPPER - 既知のリポジトリの一覧表示

```
> zypper repos
#  | 別名                    | 名前                | 有効   | GPGチェック | 更新
---+-------------------------+---------------------+--------+-------------+-------
 1 | Leap-15.1-Main          | Main (OSS)          | はい   | (r ) はい   | はい
 2 | Leap-15.1-Update        | Update (OSS)        | はい   | (r ) はい   | はい
 3 | Leap-15.1-NOSS          | Main (NON-OSS)      | はい   | (r ) はい   | はい
 4 | Leap-15.1-Update-NOSS | Update (NON-OSS)  | はい   | (r ) はい   | はい
[...]
```

様々なコマンドでリポジトリを指定する際、`zypper repos` コマンドの出力で表示される別名や URI のほか、番号も指定することができます。リポジトリの別名とは、リポジトリに対して設定する短い名前で、コマンドでの指定を簡略化するための仕組みです。なお、リポジトリに対して何らかの編集作業を行うと、番号が変わることがありますので、ご注意ください。別名については、明示的に変更しない限り、変わることはありません。

既定では URI やリポジトリの優先順位は表示されません。全ての詳細情報をあわせて表示したい場合は、下記のコマンドを実行してください:

```
> zypper repos -d
```

2.1.6.1 リポジトリの追加

リポジトリを追加するには、下記のように実行します:

```
> sudo zypper addrepo URI 別名
```

`URI` はインターネットリポジトリの場合は URL を、ネットワークリソースやディレクトリ、CD や DVD の場合は、それぞれの形式で指定します (詳しくは https://ja.opensuse.org/openSUSE:Libzypp_URI をお読みください) 。`別名` はリポジトリに対して設定する、短い独自の名前を指定します。別名は自由に設定することができますが、他のリポジトリの別名とは異なるものでなければなりません。既に使用されている別名を設定しようとすると、zypper は警告を表示します。

2.1.6.2 リポジトリの更新

`zypper` は設定済みのリポジトリから、変更されたパッケージの情報を取得することができます。変更された情報を取得するには、下記のコマンドを実行します:

```
> sudo zypper refresh
```

注記: `zypper` の既定の動作

既定では、いくつかのコマンドで `refresh` を自動的に実行します。そのため、このコマンドを明示的に実行する必要はありません。

また、`refresh` コマンドでは、無効化されたリポジトリの変更も取得することができます。これを実行するには、`--plus-content` オプションを指定します:

```
> sudo zypper --plus-content refresh
```

このオプションは、リポジトリ内の変更を取得しますが、無効化されたリポジトリは同じ状態のまま (つまり、無効化されたまま) になります。

2.1.6.3 リポジトリの削除

一覧からリポジトリを削除するには、`zypper removerepo` で削除したいリポジトリの別名または番号を指定します。たとえば 例2.1「zypper - 既知のリポジトリの一覧表示」の状態から `Non-OSS Repository` リポジトリを削除するには、下記のいずれかのコマンドを実行します:

```
> sudo zypper removerepo 4
> sudo zypper removerepo "Non-OSS Repository"
```

2.1.6.4 リポジトリの修正

`zypper modifyrepo` コマンドを使用することで、リポジトリを有効化したり無効化したりすることができます。このほか、このコマンドではリポジトリの設定 (自動更新や名前、優先順位など) も変更することができます。たとえば下記のコマンドを実行すると、`updates` という名前のリポジトリの自動更新を有効化し、優先順位を 20 にします:

```
> sudo zypper modifyrepo -er -p 20 'updates'
```

リポジトリの編集は、リポジトリ 1 つに対して行うことができるだけではありません。複数のリポジトリに対して一括で変更することもできます:

`-a` : 全てのリポジトリ
`-l` : ローカルのリポジトリ
`-t` : リモートのリポジトリ
`-m 種類` : 特定の種類のリポジトリ (`種類` には下記のいずれかを指定することができます: `http`, `https`, `ftp`, `cd`, `dvd`, `dir`, `file`, `cifs`, `smb`, `nfs`, `hd`, `iso`)

リポジトリの別名を変更するには、`renamerepo` コマンドを使用します。下記の例では、`Mozilla Firefox` という別名のリポジトリを `firefox` という別名に変更しています:

```
> sudo zypper renamerepo 'Mozilla Firefox' firefox
```

2.1.7 zypper を利用したリポジトリやパッケージの問い合わせ

zypper では、様々な方法でリポジトリやパッケージの問い合わせを行うことができます。利用可能な全ての製品 (products) やパターン (patterns) 、パッケージ (packages) や修正 (patches) などを一覧で表示するには、下記のコマンドをお使いください:

```
> zypper products
> zypper patterns
> zypper packages
> zypper patches
```

特定のパッケージに対して全てのリポジトリを検索するには、`search` を使用します。また、特定のパッケージに関する情報を取得するには、`info` コマンドを使用します。

2.1.7.1 ソフトウエアの検索

`zypper search` コマンドはパッケージ名だけでなく、必要であればパッケージの概要説明や詳細説明を対象として検索することができます。また、`/` で括られた文字列は正規表現として扱われます。なお、既定では大文字と小文字を区別せずに検索します。

パッケージ名に `fire` という文字列を含むものの単純な検索

```
> zypper search "fire"
```

`MozillaFirefox` という名前に一致するパッケージの検索

```
> zypper search --match-exact "MozillaFirefox"
```

パッケージの概要説明と詳細説明の両方を対象にする検索

```
> zypper search -d fire
```

インストールされていないパッケージのみを表示する検索

```
> zypper search -u fire
```

`fir` という文字列を含むが、その後ろに `e` を含まないようなパッケージの検索

```
> zypper se "/fir[^e]/"
```

2.1.7.2 特定の能力を指定する検索

特定の能力設定を指定してパッケージを検索したい場合は、`what-provides` コマンドを使用します。たとえば Perl モジュールの `SVN::Core` を提供するパッケージを検索したい場合は、下記を実行します:

```
> zypper what-provides 'perl(SVN::Core)'
```

`what-provides パッケージ名` は `rpm -q --whatprovides パッケージ名` に似た仕組みですが、RPM コマンドは RPM データベース (インストールされている全てのパッケージに対するデータベース) 内のみを検索するのに対して、zypper はインストールされていないものを含めて、任意のリポジトリ内にあるパッケージを検索します。

2.1.7.3 パッケージの情報表示

単一のパッケージの情報を表示したい場合は、`info` コマンドに続いて正確なパッケージ名を指定します。これにより、指定したパッケージに対する詳細な情報が表示されます。指定したパッケージ名が、どのリポジトリ内にも見つからない場合は、パッケージ名以外を対象として検索を行い、詳細情報を表示します。また、種類を指定 (`-t` オプションを設定) して情報を要求した場合で、その種類が見つからない場合、詳細な情報を表示せずに該当するもののみを表示します。

また、ソースパッケージを指定すると、このコマンドはソースパッケージから構築されたバイナリパッケージを表示します。バイナリパッケージを指定した場合、このコマンドはパッケージを構築する際に使用するソースパッケージを表示します。

このパッケージが必要とするパッケージや、推奨するパッケージについても表示を行いたい場合は、それぞれ `--requires` (必要とするもの) または `--recommends` (推奨するもの) オプションを指定してください:

```
> zypper info --requires MozillaFirefox
```

2.1.8 zypper の設定

zypper では、設定ファイルを利用することで、システム全体もしくはユーザ単位で、動作を恒久的に変更することができます。システム全体に対する変更を行いたい場合は、`/etc/zypp/zypper.conf` ファイルを編集してください。また、ユーザ単位で設定を変更したい場合は、`~/.zypper.conf` ファイルを編集してください。`~/.zypper.conf` ファイルが存在しない場合は、`/etc/zypp/zypper.conf` ファイルを雛形としてお使いください。具体的には、左記のファイルを `~/.zypper.conf` にコピーしてから、必要な変更を行ってください。設定可能な項目について、詳しくはファイル内のコメントをお読みください。

2.1.9 トラブルシューティング

設定済みのリポジトリが提供するパッケージにアクセスできない場合 (たとえば、いずれかのリポジトリ内に存在するにもかかわらず、パッケージを見つけることができない場合) は、リポジトリを更新することで解決する場合があります:

```
> sudo zypper refresh
```

それでもうまく動かない場合は、下記をお試しください:

```
> sudo zypper refresh -fdb
```

これにより、生のメタデータを強制的にダウンロードして、完全な更新とデータベースの再構築を行います。

2.1.10 btrfs ファイルシステムにおける zypper のロールバック機能

ルートパーティションに btrfs ファイルシステムを使用していて、かつ `snapper` がインストールされている場合、zypper はファイルシステムに対して変更点をコミットする際、自動的に `snapper` を呼び出して、ファイルシステムのスナップショットを採取します。これらのスナップショットは、zypper によってなされた変更点を元に戻したいような場合に利用することができます。詳しくは 第3章「Snapper によるシステムの復元とスナップショット管理」をお読みください。

2.1.11 さらなる情報

コマンドラインからのソフトウエア管理について詳しく調べたい場合は、`zypper help` または `zypper help` コマンド を実行するか、`zypper(8)` のマニュアルページをお読みください。完全で詳細なコマンドリファレンスを読みたい場合や、主要なコマンドが書かれた **チートシート** を見たい場合、もしくは zypper をスクリプトやアプリケーション内で使用する方法について調べたい場合は、https://ja.opensuse.org/SDB:Zypper_%E3%81%AE%E4%BD%BF%E3%81%84%E6%96%B9 をお読みください。最新の openSUSE Leap でのソフトウエアの更新情報について知りたい場合は、https://ja.opensuse.org/openSUSE:Zypper_%E3%83%90%E3%83%BC%E3%82%B8%E3%83%A7%E3%83%B3 をお読みください。

2.2 RPM パッケージマネージャ

RPM (RPM パッケージマネージャ) は、ソフトウエアパッケージを管理するために使用するツールです。主なコマンドは `rpm` と `rpmbuild` です。ユーザやシステム管理者、そしてパッケージの作成者からパワフルな RPM データベースに対して問い合わせを行い、インストール済みのソフトウエアに関する詳細な情報を引き出すことができるようになっています。

`rpm` には 5 種類のモードが存在しています。それぞれインストールと削除 (もしくは更新) 、RPM データベースの再構築と RPM データベースや個別の RPM 書庫に対する問い合わせ、そしてパッケージの一貫性チェックおよびパッケージへの署名の各機能です。一方の `rpmbuild` は、純粋なソースコードからインストール可能なパッケージを構築する機能を提供します。

インストール可能な RPM 書庫は、特殊なバイナリ形式になっています。書庫にはインストールすべきプログラムファイルのほか、インストール時に `rpm` が読み込んで、ソフトウエアパッケージの設定に使用するメタデータや、RPM データベース内に保存する各種の情報などが含まれています。また、RPM 書庫は通常、`.rpm` という拡張子が設定されます。

ヒント: ソフトウエア開発パッケージ

パッケージによっては、ソフトウエア開発に必要なコンポーネント (ライブラリ、ヘッダファイルなど) が別途のパッケージに分割されていることがあります。これらの開発用パッケージは、ソフトウエア自身 (たとえば最新の GNOME パッケージなど) をコンパイルする場合にのみ必要なものです。開発用パッケージは `-devel` という名前で識別できるようになっていて、たとえば `alsa-devel` や `gimp-devel` のような名前で提供されています。

2.2.1 パッケージの発行者の検証

RPM パッケージには GPG の署名が付与されています。RPM パッケージの署名を検証するには、`rpm --checksig` パッケージ名-1.2.3.rpm のように実行します。これで SUSE から提供されているパッケージなのか、それとも他の信頼できる提供元からのパッケージなのかを判別することができます。また、この仕組みはインターネット経由でパッケージを更新する場合には、特にお勧めの機能です。

2.2.2 パッケージの管理: インストール／更新／削除

通常は、RPM パッケージのインストールは至ってシンプルに、`rpm -i` パッケージ.rpm を実行するだけで済みます。このコマンドは、依存関係が満たされていて、かつ他のパッケージとの間で矛盾が発生しない場合に限って成功します。何らかの問題が発生した場合、`rpm` は依存関係を満たすために必要なパッケージを一覧で表示します。裏側では、RPM データベースは 1 つのファイルが複数のパッケージに属してしまったりする矛盾が発生しないように処理を行います。なお、`rpm` にオプションを指定することで、これらの既定値を無視して強制的にインストールすることができますが、これは知識のあるユーザ向けの機能です。これによってシステムの一貫性が損なわれたり、場合によってはシステムの更新機能を阻害してしまったりすることがあります。

また、`-U` (もしくは `--upgrade`) と `-F` (もしくは `--freshen`) は、パッケージを更新する際に使用するオプションです (たとえば `rpm -F` パッケージ のように実行します) 。このコマンドは古いバージョンのファイルを削除して、即時に新しいバージョンのファイルをインストールする動作を行います。ただし、

`-U` を指定した場合、その時点でインストールされていないパッケージであればインストールを行うのに対し、`-F` ではインストールされていないパッケージの場合は何も行いません。また、更新を行う際、`rpm` は設定ファイルを下記の方式で注意深く書き換えます:

- システム管理者側で設定ファイルを書き換えていない場合、`rpm` は単純に新しいバージョンの設定ファイルを書き込みます。システム管理者側での作業は何も発生しません。
- システム管理者側で設定ファイルを書き換えている場合、`rpm` は変更したファイルを `.rpmorig` もしくは `.rpmsave` (バックアップファイル) という拡張子を付けて保存し、新しいバージョンの設定ファイルを書き込みます。これは通常、インストールされている設定ファイルと新しいバージョンの設定ファイルが異なる場合にのみ行われます。この場合は、バックアップファイル (`.rpmorig` もしくは `.rpmsave`) と新しくインストールされた設定ファイルとを比較して、新しい設定ファイルに必要な変更を加える作業を行うことになります。作業が終わったら、`.rpmorig` や `.rpmsave` のファイルは削除しておいてください。今後の更新の際に問題を引き起こす可能性があるためです。
- なお、対応する `.spec` ファイル内に `noreplace` という指定があった場合、設定ファイルが書き換えられていると `.rpmnew` というファイルが作成されます。

更新作業後、設定ファイルの比較が終わったら `.rpmsave` と `.rpmnew` ファイルは削除しておいてください。これは、今後の更新の際に問題を引き起こす可能性があるためです。なお、`.rpmorig` は、対象のファイルが RPM データベース内で認識されていなかった場合に設定されます。

それ以外の場合は `.rpmsave` が生成されます。言い換えると、`.rpmorig` は他の形式から RPM に更新した場合にのみ生成されるもので、`.rpmsave` は古い RPM から新しい RPM に更新した場合に生成されます。また、`.rpmnew` にはシステム管理者が変更した設定ファイルの内容は含まれていません。これらのファイルの一覧は、`/var/adm/rpmconfigcheck` 内に用意されています。また、設定ファイルによっては、継続的な稼働を目的として、上書きを行わないようにしているものもあります (例: `/etc/httpd/httpd.conf`) 。

ちなみに、`-U` スイッチでの更新は、`-e` オプションでいったんパッケージを削除してから、`-i` でインストールし直す処理と等価では ありません 。できる限り `-U` で更新することをお勧めします。

パッケージを削除するには、`rpm -e パッケージ名` を実行します。このコマンドは、パッケージの依存関係に未解決のものがない場合にのみ削除を行います。言い換えると、たとえば他のアプリケーションが Tcl/Tk を必要としている限り、理論上は Tcl/Tk を削除することができないことを意味します。この場合でも、RPM はデータベースの支援を必要とします。もしも (依存関係に問題が内にもかかわらず) 何らかの理由で削除ができなくなってしまった場合は、`--rebuilddb` オプションを使用して RPM データベースを再構築して解決を試みてください。

2.2.3 差分 (デルタ) RPM パッケージ

差分 (デルタ) RPM パッケージは、古いバージョンと新しいバージョンの RPM パッケージのうち、差異のある部分だけを保持するパッケージです。古いバージョンの RPM パッケージに対して差分 RPM を適用することで、新しいバージョンの RPM を生成することができます。差分 RPM はインストール済みの RPM に対しても適用することができますので、古いバージョンの RPM を保持しておく必要はありません。差分 RPM パッケージは修正 RPM よりもずっと小さいサイズにすることができますので、インターネットを介して更新パッケージを転送するような場合に便利な仕組みです。ただし、差分 RPM での更新処理は、純粋な RPM や修正 RPM に比べて CPU サイクルを多く消費するため、その点ではデメリットとなります。

差分 RPM スイート (`deltarpm` パッケージ) には、`makedeltarpm` と `applydelta` の各コマンドが用意されていて、これらを利用することで差分 RPM パッケージを作成したり適用したりすることができるようになっています。下記のコマンドは差分 RPM を作成するためのコマンドで、`new.delta.rpm` という差分 RPM を、`old.rpm` と `new.rpm` の各 RPM から作成します:

```
> sudo makedeltarpm old.rpm new.rpm new.delta.rpm
```

`applydeltarpm` を使用することで、インストール済みの古い RPM (`old.rpm`) から新しい RPM を再構築することができます:

```
> sudo applydeltarpm new.delta.rpm new.rpm
```

ファイルシステム内の古い RPM にはアクセスせずに差分 RPM を適用するには、`-r` オプションを使用します:

```
> sudo applydeltarpm -r old.rpm new.delta.rpm new.rpm
```

技術的な詳細について、詳しくは `/usr/share/doc/packages/deltarpm/README` をお読みください。

2.2.4 RPM データベースへの問い合わせ

`-q` オプションを使用することで、`rpm` は RPM 書庫 (`-p`) やインストール済みパッケージの RPM データベースに対して、問い合わせを行う機能を提供します。なお、必要な情報を取得するためのスイッチが各種用意されています。詳しくは 表2.1「RPM 検証オプション」 をご覧ください。

表 2.1: RPM 検証オプション

`-i`	パッケージ情報
`-l`	ファイル一覧

`-f FILE`	`FILE` で指定したファイルを含むパッケージ (`FILE` ではフルパスを指定する必要があります)
`-s`	ファイル一覧と状態に関する情報 (`-l` の機能を含みます)
`-d`	ドキュメンテーションファイルのみを一覧表示 (`-l` の機能を含みます)
`-c`	設定ファイルのみを一覧表示 (`-l` の機能を含みます)
`--dump`	ファイル一覧と情報 (`-l` , `-c` , `-d` とあわせて指定する必要があります)
`--provides`	他のパッケージから `--requires` で要求することのできる、パッケージの機能一覧
`--requires` , `-R`	パッケージが必要とする機能
`--scripts`	インストールスクリプト (preinstall (インストール前), postinstall (インストール後), uninstall (削除時))

たとえば `rpm -q -i wget` のように実行すると、例2.2「`rpm -q -i wget`」のような情報が表示されます。

例 2.2: `rpm -q -i wget`

```
Name        : wget
Name        : wget
Version     : 1.19.5
Release     : lp151.4.1
Architecture: x86_64
Install Date: 2019年07月30日 18時26分21秒Tue 30 Jul 2019 02:26:21 PM PDT
Group       : Productivity/Networking/Web/Utilities
Size        : 2881903
License     : GPL-3.0+
Signature   : RSA/SHA256, 2019年04月11日 18時23分42秒, Key ID b88b2fd43dbdc284
Source RPM  : wget-1.19.5-lp151.4.1.src.rpm
Build Date  : 2019年04月11日 18時23分27秒
Build Host  : cloud114
Relocations : (not relocatable)
Packager    : https://bugs.opensuse.org
Vendor      : openSUSE
URL         : https://www.gnu.org/software/wget/
```

```
Summary     : A Tool for Mirroring FTP and HTTP Servers
Description :
Wget enables you to retrieve WWW documents or FTP files from a server.
This can be done in script files or via the command line.
Distribution: openSUSE Leap 15.1
```

`-f` オプションで指定するファイル名は、フルパスを指定した場合にのみ動作します。また、ファイル名は必要な分だけ記述することができます。たとえば下記のようになります:

```
> rpm -q -f /bin/rpm /usr/bin/wget
rpm-4.14.1-lp151.13.10.x86_64
wget-1.19.5-lp151.4.1.x86_64
```

ファイル名の一部のみが判明している場合は、例2.3「パッケージを検索するためのスクリプト」で示しているようなスクリプトをお使いください。スクリプトのパラメータとしてファイル名の一部を指定して実行してください。

例 2.3: パッケージを検索するためのスクリプト

```
#! /bin/sh
for i in $(rpm -q -a -l | grep $1); do
    echo "\"$i\" is in package:"
    rpm -q -f $i
    echo ""
done
```

また、 `rpm -q --changelog` パッケージ名 のように実行すると、特定のパッケージに対する更新情報を、詳細な一覧形式で日付順に取得することができます。

インストール済み RPM データベースに対しては、検証作業を行うことができます。これを行うには、`-V` または `--verify` を指定してください。このオプションを指定すると、`rpm` はパッケージ内のファイルのうち、インストール時点から変更されているものを全て表示します。`rpm` では、8 文字の記号を表示して、どのような点に差異があったのかを表示します:

表 2.2: RPM 検証オプション

`5`	MD5 チェックサム
`S`	ファイルサイズ
`L`	シンボリックリンク
`T`	最終更新日時
`D`	メジャー／マイナーデバイス番号
`U`	所有者

G	グループ
M	モード (パーミッションとファイルの種類)

設定ファイルの場合は c という文字が出力されます。たとえば /etc/wgetrc ファイル (wget パッケージ) が書き換えられている場合、下記のような出力になります:

```
> rpm -V wget
S.5....T c /etc/wgetrc
```

RPM データベースのファイルは /var/lib/rpm 内にあります。/usr のあるパーティションが 1 GB 程度のサイズである場合、特に完全更新を行った場合は顕著ですが、データベースはおおよそ 30 MB 程度のサイズを占有します。データベースが期待するサイズよりもずっと大きい場合は、--rebuilddb オプションを使用してデータベースを再構築するとよいでしょう。ただし、再構築を行うにあたっては、事前に古いデータベースをバックアップしておいてください。また、cron スクリプトの cron.daily では、データベースのバックアップを採取して gzip で圧縮し、/var/adm/backup/rpmdb 内に保存する処理を行っています。保存するバックアップの最大数は /etc/sysconfig/backup ファイル内の MAX_RPMDB_BACKUPS で設定することができます (既定値: 5) 。なお、/usr が 1 GB 程度である場合、バックアップ 1 つあたり 1 MB 程度になります。

2.2.5 ソースパッケージのインストールとコンパイル

全てのソースパッケージには .src.rpm という拡張子 (ソース RPM) が設定されます。

注記: インストール済みのソースパッケージ

ソースパッケージは、YaST を利用することでインストールメディアからハードディスクにコピーすることができます。ただし、インストールを行ってもパッケージマネージャの表示が ([i]) になることはありません。これは、ソースパッケージが RPM データベース内で管理されないことによるものです。RPM データベースには、インストール済みの オペレーティングシステムに対するソフトウエアの情報のみが書かれています。ソースパッケージについては、「インストール」を行ってもシステム内にソースコードが保存されるだけです。

rpm と rpmbuild で利用する目的で、/usr/src/packages 以下に下記のディレクトリが作成されます (ただし、/etc/rpmrc のようなファイルで独自に設定した場合を除きます):

SOURCES
: オリジナルのソースコード (.tar.bz2 や .tar.gz など) と、ディストリビューション固有の調整 (多くは .diff や .patch など) が含まれます。

`SPECS`

`.spec` ファイルが配置されるディレクトリです。これは「メタ Makefile」と表現しうる仕組みで、構築 処理を制御するためのファイルです。

`BUILD`

ソースコードの展開や修正の適用、コンパイルなどをこのディレクトリで行います

`RPMS`

作成されたバイナリパッケージが配置されるディレクトリです。

`SRPMS`

ソース RPM が配置されるディレクトリです。

YaST からソースパッケージをインストールすると、`/usr/src/packages` 以下に必要な全てのコンポーネントが配置されます。具体的には、ソースコードと修正類が `SOURCES` に、対応する `.spec` ファイルが `SPECS` にそれぞれ配置されます。

警告: システムの安定性への危険について

システムコンポーネント (例: `glibc` , `rpm` など) に対しては、下記の手順を試してはなりません。場合によってはシステムの安定性を壊してしまう危険があるためです。

下記の例は、`wget.src.rpm` パッケージをインストールした場合の例です。ソースパッケージをインストールすると、下記のようなファイルが作成されます:

```
/usr/src/packages/SOURCES/wget-1.19.5.tar.bz2
/usr/src/packages/SOURCES/wgetrc.patch
/usr/src/packages/SPECS/wget.spec
```

`rpmbuild -bX /usr/src/packages/SPECS/wget.spec` のように実行すると、コンパイル処理が始まります。なお、`X` には構築プロセスのどの段階までを実行するのかを指定します (詳しくは `--help` で表示されるヘルプか、RPM のドキュメンテーションをお読みください) 。下記には概要説明を記載します:

`-bp`

`/usr/src/packages/BUILD` 以下にソースコードを展開し、必要であれば修正を適用します。

`-bc`

`-bp` までの処理に加え、追加でコンパイルまで実行します。

`-bi`

`-bp` までの処理に加え、構築済みのソフトウエアをインストールする処理まで実行します。注意: パッケージが BuildRoot 機能に対応していない場合、これを実行すると設定ファイルを上書きしてしまいます。

`-bb`

`-bi` までの処理に加え、バイナリパッケージの作成まで実行します。コンパイルが成功すれば、`/usr/src/packages/RPMS` ディレクトリ内に配置されます。

`-ba`

`-bb` までの処理に加え、ソース RPM の作成まで実行します。コンパイルが成功すれば、`/usr/src/packages/SRPMS` ディレクトリ内に配置されます。

`--short-circuit`

いくつかの手順を飛ばして実行します。

作成されたバイナリ RPM は、`rpm -i` または `rpm -U` でインストールすることができます。`rpm` でインストールを行うと、RPM データベース内に現れるようになります。

なお、spec ファイル内の `BuildRoot` ディレクティブは廃止予定としてマークされていることに注意してください。どうしてもこの機能が必要な場合は、回避策として `--buildroot` オプションを指定してください。

2.2.6 build を利用した RPM パッケージのコンパイル

構築処理を行うにあたっては、多くのパッケージをインストールする必要がありますが、これらはシステムを動作させるにあたっては不要なものです。不要なパッケージのインストールを回避するには、`build` をお使いください。これはパッケージを構築するための環境を、現在使用中のものとは別に作成できる仕組みで、chroot の仕組みを利用して環境の作成と構築を行います。構築環境はハードディスクのほか、NFS や DVD を利用することができます。環境の作成元となる場所を指定するには、`build --rpms ディレクトリ` のように指定してください。なお、`rpm` とは異なり、`build` コマンドではソースディレクトリ内に `.spec` を配置します。たとえば `wget` パッケージを `/media/dvd` にマウントされた DVD を利用して構築したい場合は、下記のコマンドを `root` で実行します:

```
# cd /usr/src/packages/SOURCES/
# mv ../SPECS/wget.spec .
# build --rpms /media/dvd/suse/ wget.spec
```

すると、最小構成の環境が `/var/tmp/build-root` 内に作成されます。パッケージはこの環境を利用して構築されます。構築が完了すると、作成されたパッケージは `/var/tmp/build-root/usr/src/packages/RPMS` に配置されます。

`build` スクリプトにはいくつかの追加オプションがあります。たとえばスクリプトに対して独自の RPM を使用させたり、構築環境の初期化を省略したり、`rpm` コマンドに対して上述の途中段階までで停止させたりすることができます。詳しくは `build --help` で表示されるヘルプか、もしくは `build` のマニュアルページをお読みください。

2.2.7 RPM 書庫および RPM データベース向けツール

Midnight Commander (`mc`) では、RPM 書庫の内容を表示することができるほか、一部をコピーしたりすることができます。Midnight Commander では仮想的なファイルシステムとして表示されるため、通常のメニューオプションを全て利用することができます。また、 `F3` を押すと `HEADER` を表示することができるほか、カーソルキーで書庫の内容を選択して `Enter` を押すと、内容を表示することができます。また、一部をコピーしたい場合は、 `F5` を押します。

YaST モジュールには、完全機能のパッケージマネージャが用意されています。詳しくは『**スタートアップ**』、第9章「**ソフトウエアのインストールと削除**」をお読みください。

3 Snapper によるシステムの復元とスナップショット管理

改訂履歴

2024-07-22

snapper はファイルシステムのスナップショットを作成したり、管理したりするためのツールです。ファイルシステムのスナップショットには、あの特定の時点におけるファイルシステムの状態に対するコピーが含まれていて、通常はシステムに対して行った変更を元に戻すために使用します。このほか、ユーザデータのディスク内バックアップ用途にも使用することができます。snapper の機能は btrfs ファイルシステムのほか、xfs や ext4 ファイルシステムを使用したシン・プロビジョン型の LVM ボリュームに対して使用することができます。

snapper にはコマンドラインインターフェイスのほか、YaST インターフェイスも提供されています。これにより、下記のような種類のファイルシステムに対して、ファイルシステムのスナップショットを作成したり管理したりすることができます:

- btrfs: コピーオンライト型の Linux ファイルシステムで、ファイルシステム内にサブボリュームの機能とスナップショットの機能が用意されています (サブボリュームとは、それぞれ別々にマウントすることのできる、ファイルシステム内パーティションです) 。
  このほか、`btrfs` スナップショットからの起動を行うこともできます。詳しくは 3.3項「スナップショットからの起動によるシステムのロールバック」をお読みください。
- xfs もしくは ext4 でフォーマットされたシン・プロビジョン型の LVM ボリューム。

Snapper を利用することで、下記のようなことができるようになります:

- `zypper` や YaST が実行したシステム変更の巻き戻し。詳しくは 3.2項「Snapper による変更点の巻き戻し」をお読みください。
- 以前のスナップショット時点で存在していたファイルの復元。詳しくは 3.2.2項「ファイルを復元するための Snapper の使用」をお読みください。
- スナップショットからの起動によるシステム全体の巻き戻し。詳しくは 3.3項「スナップショットからの起動によるシステムのロールバック」をお読みください。
- システムが動作している状態での手作業によるスナップショットの作成と管理。詳しくは 3.6項「スナップショットの手動作成と管理」をお読みください。

3.1 既定の設定

openSUSE Leap での Snapper は、システムの変更に対する巻き戻しと復元のためのツールとして動作するように設定されています。既定では、ルートパーティション (`/`) が `btrfs` でフォーマットされます。また、ルートパーティション `/` が十分に大きい (おおよそ 16 GB 以上) 場合、スナップショットの採取が自動的に有効化されます。`/` 以外のパーティションに対するスナップショットは、既定では無効化されます。

ヒント: インストール済みシステムにおける Snapper の有効化

インストール時に Snapper を無効化している場合は、任意の時点であとから有効化することもできます。有効化するには、下記を実行して既定の Snapper 設定を作成してください:

```
> sudo snapper -c root create-config /
```

あとは 3.1.4.1項「スナップショットの有効化／無効化」で説明している手順に従って、スナップショットの種類を有効化します。

なお、btrfs を設定したルートファイルシステムを使用する場合、スナップショットを採取するには、インストーラで提案したサブボリューム設定でルートファイルシステムを設定し、パーティションのサイズが少なくとも 16 GB 以上である必要があります。

スナップショットを作成すると、スナップショットと元のブロックはファイルシステム内で同じブロックを指し示します。そのため、スナップショットを採取した時点では、追加のディスク領域を必要とすることはありません。ただし、スナップショットを作成した時点からファイルが変更されると、変更した部分のデータブロックが新しく確保され、元の (スナップショットを採取した時点の) データブロックがそのまま保持されるようになります。そのためスナップショットは、修正された分だけのデータに対応するブロックを占有することになります。また、時間が経過するごとにスナップショットが占有する領域が定期的に増えていくことにも繋がります。そのため、`Btrfs` ファイルシステムでファイルを削除しても、ディスクの空き領域を増やすことには繋がらない場合があり得ます。

注記: スナップショットの場所

スナップショットは、採取したファイルシステムと同じパーティションやサブボリューム内に存在します。異なるパーティションやサブボリュームに保存することはできません。

そのため、スナップショットを含むパーティションは、通常のパーティションよりもサイズを大きくする必要があります。正確な容量は、保持すべきスナップショットの数とデータ変更の容量によって異なりますが、一般的には通常必要とするサイズの 2 倍程度を確保しておくことをお勧めします。なお、ディスクの空き容量が枯渇してしまうことがないよう、古いスナップショットは自動的にクリーンアップしておくとよいでしょう。詳しくは 3.1.4.4項「スナップショットのアーカイブの制御」をお読みください。

3.1.1 既定の設定

16 GB よりも大きいディスクサイズである場合

- 設定ファイル: `/etc/snapper/configs/root`
- `USE_SNAPPER=yes`
- `TIMELINE_CREATE=no`

16 GB よりも小さいディスクサイズである場合

- 設定ファイル: 作成されません
- `USE_SNAPPER=no`
- `TIMELINE_CREATE=yes`

3.1.2 スナップショットの種類

スナップショットそれ自身は技術的な意味で何も違いがありませんが、それを発生させたイベントごとに、スナップショットの意味が異なります:

タイムラインスナップショット
: 単独のスナップショットを 1 時間おきに作成します。YaST を利用して OS をインストールした場合 (既定) 、ルートファイルシステム以外に対してタイムラインスナップショットが有効化されます。タイムラインスナップショットは異なる間隔設定 (1 時間, 1 日, 1 週, 1 月, 1 年) 単位で採取することもできます。古いスナップショットは自動的に削除されます。既定では、直近の 10 日間, 10 ヶ月, 10 年の間に採取された最初のスナップショットを保持します。

インストールスナップショット
: YaST や zypper で 1 つまたは複数のパッケージをインストールすると、3 つのインストールスナップショットを作成します。また、インストールしたパッケージがカーネルなどの主要なパッケージであった場合、そのスナップショットは重要なものであるとマークが付けられます。古いスナップショットは自動で削除されます。インストールスナップショットは既定で有効化されます。

管理スナップショット

Zypper や YaST を利用してシステムの変更を行うと、スナップショットの対を作成します。1 つ目はシステム変更直前 (「Pre」(事前)) 、2 つ目はシステム変更後 (「Post」(事後)) にそれぞれ採取されます。古いスナップショットは自動的に削除されます。既定では、直近の重要な 10 回分のスナップショットと、直近の 10 回分の「定期」スナップショット (インストールスナップショットを含む) が保持されます。管理スナップショットは、既定で有効化されます。

3.1.3 スナップショットから除外されるディレクトリ

様々な理由から、いくつかのディレクトリについては除外を設定する必要があります。下記の一覧には、除外される全てのディレクトリを示しています:

`/boot/grub2/i386-pc` , `/boot/grub2/x86_64-efi` , `/boot/grub2/powerpc-ieee1275` , `/boot/grub2/s390x-emu`

ブートローダ設定の巻き戻しには対応していません。また、上記のディレクトリはいずれもアーキテクチャ固有のものであり、前半の 2 つは AMD64/Intel 64 マシンで使用されるディレクトリ、後半の 2 つは IBM POWER や IBM Z で使用されるディレクトリです。

`/home`

`/home` が個別のディレクトリ内に存在していない場合、ロールバックによってデータが失われてしまうことを防ぐため、除外を設定しています。

`/opt`

サードパーティ製の製品は `/opt` 以下にインストールされるのが一般的です。そのため、ロールバックによってこれらのアプリケーションが削除されてしまうことのないよう、除外を設定しています。

`/srv`

Web サーバや FTP サーバのデータを含むディレクトリです。ロールバックによってそれらのデータが失われてしまうことの無いよう、除外を設定しています。

`/tmp`

いずれも一時的な (テンポラリ) ファイルとキャッシュが保存されるディレクトリであるため、除外を設定しています。

`/usr/local`

このディレクトリは手作業でソフトウエアをインストールした場合に使用するディレクトリです。ロールバックによって、これらのソフトウエアが消えてしまったりしないようにするため、除外を設定しています。

`/var`

ログファイルや一時的なキャッシュが含まれるほか、サードパーティ製の製品が `/var/opt` 以下にインストールされることがあります。また、仮想マシンのイメージやデータベースを配置する既定のディレクトリでもあります。そのため、これらのデータをスナップショットから除外するためにサブボリュームを作成し、かつコピーオンライト機能を無効化しています。

3.1.4 設定のカスタマイズ

openSUSE Leap ではほとんどの用途に対応できる既定値が設定されています。ですが、必要であれば、自動スナップショットやスナップショットの保持設定について、全ての項目を設定することができます。

3.1.4.1 スナップショットの有効化／無効化

3 種類のスナップショット種類 (タイムライン, インストール, 管理) を個別に有効化／無効化することができます。

タイムラインスナップショットの有効化／無効化

有効化: `snapper -c root set-config "TIMELINE_CREATE=yes"`

無効化: `snapper -c root set-config "TIMELINE_CREATE=no"`
YaST で OS をインストールした既定の場合、タイムライン型のスナップショットはルートファイルシステム以外に対して有効化されます。

インストールスナップショットの有効化／無効化

有効化: `snapper-zypp-plugin` パッケージをインストールします。

無効化: `snapper-zypp-plugin` パッケージを削除します。
インストールスナップショットは既定で有効化されます。

管理スナップショットの有効化／無効化

有効化: `/etc/sysconfig/yast2` 内の `USE_SNAPPER` に `yes` を設定します。

無効化: `/etc/sysconfig/yast2` 内の `USE_SNAPPER` に `no` を設定します。
管理スナップショットは既定で有効化されます。

3.1.4.2 インストールスナップショットの制御

YaST や zypper でパッケージをインストールした際、スナップショットは `snapper-zypp-plugin` が処理します。このプラグインには XML の設定ファイル `/etc/snapper/zypp-plugin.conf` があり、これでスナップショットを採取するタイミングを設定することができます。既定では、設定ファイルは下記のようになっています:

```
 1 <?xml version="1.0" encoding="utf-8"?>
 2 <snapper-zypp-plugin-conf>
 3  <solvables>
 4   <solvable match="w"❶ important="true"❷>kernel-*❸</solvable>
 5   <solvable match="w" important="true">dracut</solvable>
 6   <solvable match="w" important="true">glibc</solvable>
 7   <solvable match="w" important="true">systemd*</solvable>
 8   <solvable match="w" important="true">udev</solvable>
 9   <solvable match="w">*</solvable>❹
10  </solvables>
11 </snapper-zypp-plugin-conf>
```

❶ match 属性はパターンの定義方法を決めるもので、Unix スタイルのワイルドカード (`w`) か、Python の正規表現 (`re`) のいずれかを指定することができます。

❷ 指定したパターンに該当し、関連するパッケージが重要であるとしてマークされている (たとえば kernel パッケージなど) と、スナップショットについても重要であるものとしてマークが付けられます。

❸ パッケージ名と比較処理される内容です。`match` 属性の設定に依存しますが、特殊文字はシェルのワイルドカードか、正規表現として扱われます。たとえばこのパターンは、`kernel-` で始まる全てのパッケージが該当する指定になっています。

❹ この行は、全てのパッケージが該当する箇所です。

このスナップショットの設定では、パッケージがインストールされると必ずスナップショット対を作成します (9 行目の設定による) 。また、重要であるとマークされた kernel, dracut, glibc, systemd, udev の各パッケージがインストールされると、スナップショットの対にも重要 (important) のマークが付けられます。

ルールを無効化するには、削除を行うか XML のコメント機能を利用してください。たとえば各パッケージのインストールでスナップショット対を作成したくない場合は、下記のようにして 9 行目をコメントアウトします:

```
1 <?xml version="1.0" encoding="utf-8"?>
2 <snapper-zypp-plugin-conf>
3  <solvables>
4   <solvable match="w" important="true">kernel-*</solvable>
5   <solvable match="w" important="true">dracut</solvable>
6   <solvable match="w" important="true">glibc</solvable>
```

```
 7   <solvable match="w" important="true">systemd*</solvable>
 8   <solvable match="w" important="true">udev</solvable>
 9   <!-- <solvable match="w">*</solvable> -->
10  </solvables>
11 </snapper-zypp-plugin-conf>
```

3.1.4.3 新しいサブボリュームの作成とマウント

`/` の階層構造以下に新しいサブボリュームを作成し、恒久的にマウントすることができます。新しく作成したサブボリュームは、スナップショットから除外されます。ただし、既存のスナップショット内には、新しいサブボリュームを作成してはなりません。なぜなら、ロールバックを行った後にスナップショットを削除できなくなってしまうためです。

openSUSE Leap では、`/opt` 、`/srv` 、`/home` などのディレクトリに対して、`/@/` で始まる個別のサブボリュームが設定されます。また、新しく作成し恒久的にマウントするサブボリュームは、この初期ルートファイルシステム内に作成しなければなりません。

これを行うには、下記のコマンドを実行します。下記の例では、新しいサブボリューム `/usr/important` を、`/dev/sda2` 内に作成します。

```
> sudo mount /dev/sda2 -o subvol=@ /mnt
> sudo btrfs subvolume create /mnt/usr/important
> sudo umount /mnt
```

`/etc/fstab` 内にサブボリュームのマウントを記述する場合は、下記のようになります:

```
/dev/sda2 /usr/important btrfs subvol=@/usr/important 0 0
```

ヒント: コピーオンライト (cow) 機能の無効化

サブボリュームには、仮想マシンのイメージやデータベースファイル、ログファイルなど、絶えず変更されるファイルが含まれることがあります。このような場合は、そのサブボリュームに対してコピーオンライト (Copy On Write (cow)) 機能を無効化し、ディスクブロックの重複を避けるよう設定しておくことをお勧めします。これを行うには、`/etc/fstab` 内に `nodatacow` マウントオプションを追加します:

```
/dev/sda2 /usr/important btrfs nodatacow,subvol=@/usr/important 0 0
```

また、単一のファイルやディレクトリに対して cow 機能を無効化することもできます。その場合は、`chattr +C パス` を実行してください。

3.1.4.4 スナップショットのアーカイブの制御

スナップショットはディスク領域を占有します。ディスクの容量が枯渇してシステムが止まってしまったりしないようにするため、古いスナップショットについては自動的に削除する必要があります。既定では最大で 10 回分までの重要なインストールスナップショットと管理スナップショットが保持されるほか、最大で 10 回分までの定期的なインストールスナップショットおよび管理スナップショットが保持されます。なお、これらのスナップショットが、ルートファイルシステムのサイズの 50% 以上を占有している場合は、さらに多くのスナップショットを削除します。ただし、最小でも 4 回分の重要なスナップショットと、2 回分の定期スナップショットは必ず保持されます。

これらの値の変更方法について、詳しくは 3.5.1項「既存の設定の管理」をお読みください。

3.1.4.5 シン・プロビジョン型の LVM ボリュームにおける Snapper の使用

`Btrfs` ファイルシステムに対するスナップショット機能とは別に、Snapper では xfs/ext4/ext3 でフォーマットされたシン・プロビジョン型の LVM ボリュームに対するスナップショットにも対応しています (通常型の LVM ボリュームに対するスナップショットには対応していません) 。LVM ボリュームに関する詳細やセットアップ手順については、5.3項「LVM 設定」をお読みください。

シン・プロビジョン型の LVM ボリュームに対して Snapper を使用するには、まず Snapper の設定を作成する必要があります。LVM の場合、ファイルシステムの種類を `--fstype=lvm(ファイルシステムの種類名)` で指定する必要があります。`ファイルシステムの種類名` には、`ext3` , `ext4` , `xfs` のいずれかを指定します。たとえば下記のようになります:

```
> sudo snapper -c lvm create-config --fstype="lvm(xfs)" /thin_lvm
```

ここで書き込んだ設定は、必要に応じて変更することができます。詳しくは 3.5.1項「既存の設定の管理」をお読みください。

3.2 Snapper による変更点の巻き戻し

openSUSE Leap における Snapper は、`zypper` や YaST が行った変更点を巻き戻すためのツールとして動作するよう設定されています。この目的を実現するため、Snapper では `zypper` や YaST を実行するごとに、事前と事後のスナップショット対を作成するよう設定しています。また、Snapper では誤って削除してしまったり、書き換えてしまったりしたシステムファイルを取り戻すためにも利用することができます。この用途で使用する場合は、ルートパーティションに対するタイムラインスナップショットを有効化する必要があります。詳しくは 3.1.4.1項「スナップショットの有効化／無効化」をお読みください。

既定では、上述の自動スナップショットがルートパーティションとサブボリュームに対して設定されます。たとえば `/home` などの別のパーティションに対して、スナップショットを作成するには、独自の設定を作成する必要があります。

重要: 変更点の巻き戻しとロールバックについて

データを復元する目的でスナップショットを利用する場合、Snapper が処理できるシナリオとしては、下記の 2 つのものがあることを知っておくのが重要です:

変更点の巻き戻し

: 下記に示す手順で変更点を巻き戻す場合、2 つのスナップショットを比較して、それらの間での変更点を巻き戻すことができます。この方法を利用することで、復元すべきファイルを明示的に選択することができるようになります。

ロールバック

: 3.3項「スナップショットからの起動によるシステムのロールバック」に示されている手順でロールバックを行うと、システムはスナップショットを採取した時点の状態にリセットされます。

変更点を元に戻す際、現在のシステムとスナップショットとを比較することができます。この比較で 全て のファイルを戻す場合は、ロールバックを行ったのと同じ結果になります。ただし、全てのファイルを戻す場合は、3.3項「スナップショットからの起動によるシステムのロールバック」で説明している手順に従ってロールバックを行うことをお勧めします。これは、ロールバックのほうがより高速であるだけでなく、ロールバックを行う前にシステムを確認できるという利点もあります。

警告: データの一貫性について

スナップショットを作成する際、データの一貫性に関するチェックを行う仕組みは用意されていません。たとえばデータベースファイルなどが顕著ですが、ファイルがスナップショットを採取したタイミングで同時に書き込まれていた場合、データベースファイルが壊れてしまうか、部分的に書き込まれた状態になります。このような場合、このファイルの巻き戻しで問題が発生することになります。これらに加えて、`/etc/mtab` のようなファイルについても、復元してはなりません。そのため、変更されたファイルの確認では、必ず 内容をよく確認するようにしてください。また、ファイル単位での復元では、巻き戻したいファイルだけを戻すようにしてください。

3.2.1 YaST や zypper が変更した内容の巻き戻し

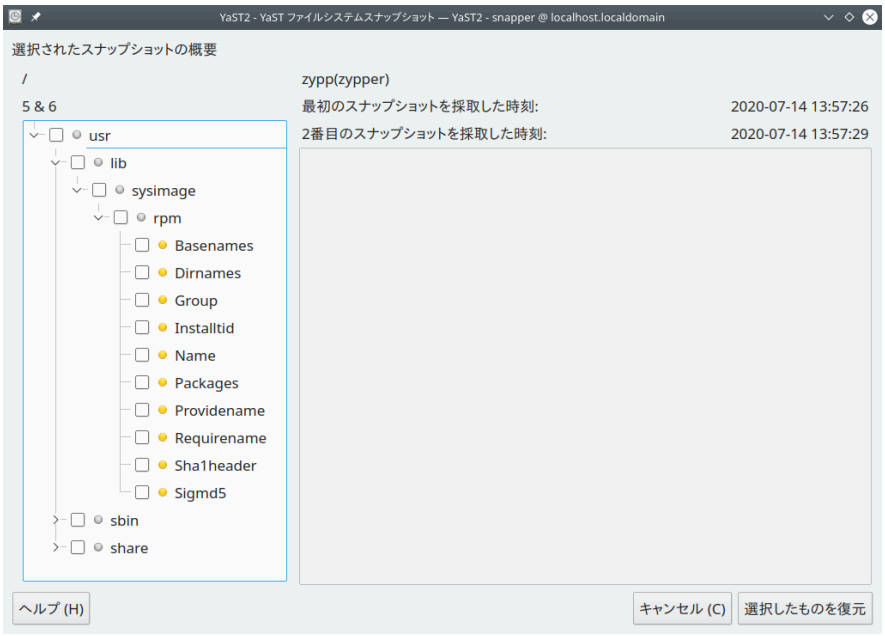
インストール時にルートパーティションを `btrfs` で設定すると、YaST や zypper の変更点を自動的に巻き戻すことができる設定を施した状態で、Snapper が自動的にインストールされます。YaST モジュールや zypper のトランザクションを開始すると、2 種類のスナップショットが必ず作成されるようになっています。1 つ目は「事前スナップショット」と呼ばれ、モジュールを開始する際にファイルシステムの状態を採取するスナップショットです。2 つ目は「事後スナップショット」と呼ばれ、モジュールが完了した際に採取するスナップショットです。

YaST の Snapper モジュールや `snapper` コマンドラインツールを使用すると、YaST や zypper が行った変更を、「事前スナップショット」からのファイルで戻すことができるようになります。2 つのスナップショットを比較する際、ツールではどのファイルが書き換えられたのかを表示することができるほか、変更されたファイルの違い (diff) も表示することができます。

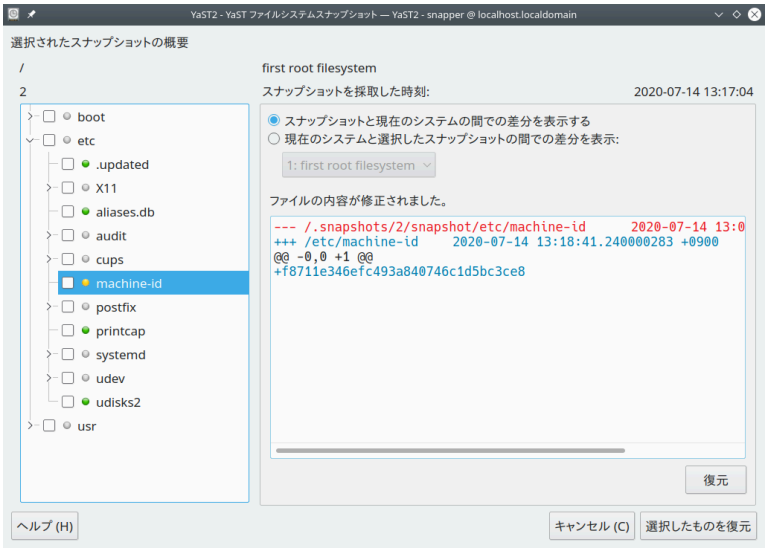
手順 3.1: YAST を使用した変更点の巻き戻し; [SNAPPER] モジュール

1. YaST を起動して [その他] にある [Snapper] を選択するか、もしくは `yast2 snapper` と入力して実行します。
2. [現在の設定] が [root] になっていることを確認してください。Snapper の設定を手動で追加していない限り、必ず確認すべき項目になります。
3. 一覧から事前および事後のスナップショットの対を選択します。YaST や zypper で採取されたスナップショット対には、[事前および事後] という種類になっているはずです。また、YaST でのスナップショットには [説明] 列に `zypp(y2base)` が書かれています。同様に、zypper でのスナップショットには、`zypp(zypper)` が書かれています。

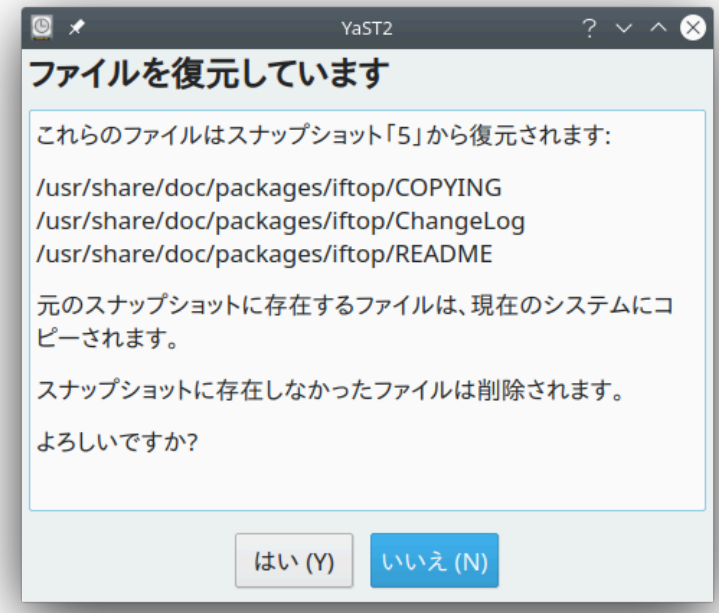
4. 2 つのスナップショットの間で変更されたファイルの一覧を表示するには、[変更点の表示] を押します。



5. ファイルの一覧が表示されます。事前と事後のスナップショットでの「差分」を表示するには、一覧からファイルを選択します。



6. 1 つまたは複数のファイルを復元するには、ファイルやディレクトリのチェックボックスにチェックを入れてから、[選択したものを復元] を押します。その後、確認メッセージが表示されますので、[はい] を押して確認します。



単一のファイルを復元する場合は、その名前欄を押して差分を表示します。あとは［最初から復元する］を押して確認メッセージを表示させ、［はい］を押して確認します。

手順 3.2: `snapper` コマンドを使用した変更点の巻き戻し

1. YaST や zypper で作成したスナップショットの一覧を表示するには、`snapper list -t pre-post` と入力して実行します。YaST でのスナップショットの場合、［説明］欄に `yast モジュール名` が表示されます。zypper でのスナップショットの場合は、`zypp(zypper)` が表示されます。

```
> sudo snapper list -t pre-post
前  # | 後   # | 前日付                   | 後日付                   | 説明
------+--------+-------------------------+-------------------------+--------------
311   | 312    | 2018年05月06日 14:05:46 | 2018年05月06日 14:05:52 | zypp(y2base)
340   | 341    | 2018年05月07日 16:15:10 | 2018年05月07日 16:15:16 | zypp(zypper)
342   | 343    | 2018年05月07日 16:20:38 | 2018年05月07日 16:20:42 | zypp(y2base)
344   | 345    | 2018年05月07日 16:21:23 | 2018年05月07日 16:21:24 | zypp(zypper)
346   | 347    | 2018年05月07日 16:41:06 | 2018年05月07日 16:41:10 | zypp(y2base)
348   | 349    | 2018年05月07日 16:44:50 | 2018年05月07日 16:44:53 | zypp(y2base)
350   | 351    | 2018年05月07日 16:46:27 | 2018年05月07日 16:46:38 | zypp(y2base)
```

2. `snapper status 前 # .. 後 #` と入力すると、スナップショット対の間で変更されたファイルの一覧を取得することができます。内容が変更されたファイルには［c］が表示されるほか、新規に作成されたファイルには［+］が、削除されたファイルには［-］がそれぞれ表示されます。

```
> sudo snapper status 350..351
+..... /usr/share/doc/packages/mikachan-fonts
+..... /usr/share/doc/packages/mikachan-fonts/COPYING
+..... /usr/share/doc/packages/mikachan-fonts/dl.html
c..... /usr/share/fonts/truetype/fonts.dir
c..... /usr/share/fonts/truetype/fonts.scale
+..... /usr/share/fonts/truetype/みかちゃん-p.ttf
+..... /usr/share/fonts/truetype/みかちゃん-pb.ttf
+..... /usr/share/fonts/truetype/みかちゃん-ps.ttf
+..... /usr/share/fonts/truetype/みかちゃん.ttf
c..... /var/cache/fontconfig/7ef2298fde41cc6eeb7af42e48b7d293-x86_64.cache-4
c..... /var/lib/rpm/Basenames
c..... /var/lib/rpm/Dirnames
c..... /var/lib/rpm/Group
c..... /var/lib/rpm/Installtid
c..... /var/lib/rpm/Name
c..... /var/lib/rpm/Packages
c..... /var/lib/rpm/Providename
c..... /var/lib/rpm/Requirename
c..... /var/lib/rpm/Sha1header
c..... /var/lib/rpm/Sigmd5
```

3. 特定のファイルに対する差分を表示するには、`snapper diff 前 # .. 後 # ファイル名` と入力して実行します。ファイル名を指定しない場合は、全てのファイルに対する差分が表示されます。

```
> sudo snapper diff 350..351 /usr/share/fonts/truetype/fonts.scale
--- /.snapshots/350/snapshot/usr/share/fonts/truetype/fonts.scale       2014-04-23
 15:58:57.000000000 +0200
+++ /.snapshots/351/snapshot/usr/share/fonts/truetype/fonts.scale       2014-05-07
 16:46:31.000000000 +0200
@@ -1,4 +1,4 @@
-1174
+1486
 ds=y:ai=0.2:luximr.ttf -b&h-luxi mono-bold-i-normal--0-0-0-0-c-0-iso10646-1
 ds=y:ai=0.2:luximr.ttf -b&h-luxi mono-bold-i-normal--0-0-0-0-c-0-iso8859-1
[...]
```

4. 1 つまたは複数のファイルを復元するには、`snapper -v undochange Pre の ID .. Post の ID ファイル名（複数可）` のように入力して実行します。ファイル名を何も指定しない場合は、変更された全てのファイルが復元されます。

```
> sudo snapper -v undochange 350..351
     create:0 modify:13 delete:7
     undoing change...
     deleting /usr/share/doc/packages/mikachan-fonts
     deleting /usr/share/doc/packages/mikachan-fonts/COPYING
     deleting /usr/share/doc/packages/mikachan-fonts/dl.html
```

```
deleting /usr/share/fonts/truetype/みかちゃん-p.ttf
deleting /usr/share/fonts/truetype/みかちゃん-pb.ttf
deleting /usr/share/fonts/truetype/みかちゃん-ps.ttf
deleting /usr/share/fonts/truetype/みかちゃん.ttf
modifying /usr/share/fonts/truetype/fonts.dir
modifying /usr/share/fonts/truetype/fonts.scale
modifying /var/cache/fontconfig/7ef2298fde41cc6eeb7af42e48b7d293-x86_64.cache-4
modifying /var/lib/rpm/Basenames
modifying /var/lib/rpm/Dirnames
modifying /var/lib/rpm/Group
modifying /var/lib/rpm/Installtid
modifying /var/lib/rpm/Name
modifying /var/lib/rpm/Packages
modifying /var/lib/rpm/Providename
modifying /var/lib/rpm/Requirename
modifying /var/lib/rpm/Sha1header
modifying /var/lib/rpm/Sigmd5
undoing change done
```

警告: ユーザ作成の巻き戻しについて

Snapper の巻き戻し機能を利用して、ユーザの作成を巻き戻すのはお勧めできません。特定のディレクトリがスナップショットから除外されるため、これらのユーザに属するファイルがファイルシステム内に残る可能性があるためです。巻き戻し機能でユーザの作成を取り消した後、別のユーザを作成してしまうと、作成を巻き戻したユーザのユーザ ID と、新たに作成したユーザのユーザ ID が同じになってしまうため、残ったファイルをそのユーザがアクセスできてしまいます。ユーザを削除する際は、YaST の［ユーザとグループの管理］を利用して削除することを強くお勧めします。

3.2.2 ファイルを復元するための Snapper の使用

インストールスナップショットや管理スナップショットとは別に、Snapper にはタイムラインスナップショットという機能があります。これは誤って削除してしまったり、誤って書き換えてしまったりしたファイルを復元するためのバックアップ用スナップショットです。Snapper の差分表示機能を利用することで、その時点のファイルからどのように書き換えてしまったのかについても、確認を行うことができるようになっています。

ファイルを復元できるという機能は、特に各種データ類に対して有用な仕組みですが、これらは、既定ではスナップショットの採取されないサブボリュームやパーティション内に置かれている場合があります。たとえばホームディレクトリなどでファイルを復元できるようにするには、`/home` に対して個別の Snapper 設定を作成し、自動的なタイムラインスナップショットを有効化する必要があります。詳しい手順については、3.5項「Snapper の設定の作成と修正」をお読みください。

警告: ファイルの復元と巻き戻しの違いについて

ルートファイルシステムに対して採取したスナップショット (Snapper のルートディレクトリ設定で設定されます) は、システムのロールバックとして使用することができます。システムのロールバックを実施するにあたっては、対応するスナップショットから起動を行って、その後にロールバックを行うのがお勧めです。詳しくは 3.3項「スナップショットからの起動によるシステムのロールバック」をお読みください。

ロールバックの方法としては、下記で説明するとおり、特定のルートファイルシステムのスナップショットからファイルを復元する方法もありますが、こちらはお勧めできません。`/etc` ディレクトリ内にある設定ファイルなど、個別のファイルを取り戻す方法としては有用ですが、全てのファイルを巻き戻す際にはお勧めできません。

なお、この制限はルートファイルシステムに対して採取されたスナップショットのみに適用すべきものです。

手順 3.3: YAST［SNAPPER］モジュールを利用したファイルの復元

1. YaST を起動して［その他］にある［Snapper］を選択するか、もしくは `yast2 snapper` と入力して実行します。
2. スナップショットを選択するため、［現在の設定］を選択します。
3. ファイルを復元したいタイムラインスナップショットを選択して、［変更点の表示］を押します。タイムラインスナップショットは、種類が［単一］になっているほか、説明欄に［timeline］と書かれています。
4. ファイル名を押して、テキストボックスからファイルを選択します。すると、スナップショットで保存された版と現在の版の差分が表示されるようになります。あとは復元したいファイルやディレクトリのチェックボックスに、チェックを入れます。
5. ［選択したものを復元］を押します。その後、確認メッセージが表示されますので、［はい］を押して確認します。

手順 3.4: `snapper` コマンドを利用したファイルの復元

1. 特定の設定に対するタイムラインスナップショットの一覧を表示するには、下記のコマンドを実行します:

```
> sudo snapper -c 設定 list -t single | grep timeline
```

`設定` には既存の Snapper 設定の値を指定します。`snapper list-configs` と入力して実行すると、設定の一覧を表示することができます。

2. 特定のスナップショットで変更されたファイルの一覧を表示するには、下記のコマンドを実行します:

```
> sudo snapper -c 設定 status スナップショット_ID..0
```

スナップショット_ID には、ファイルを復元したいスナップショットの ID を指定します。

3. 必要であれば、現在のファイルとスナップショット内のファイルの差分を表示することもできます:

```
> sudo snapper -c 設定 diff スナップショット_ID..0 ファイル名
```

ファイル名 を指定せずに実行すると、全てのファイルに対する差分が表示されます。

4. 1 つまたは複数のファイルを復元するには、下記のように実行します:

```
> sudo snapper -c 設定 -v undochange スナップショット_ID..0 ファイル名_1 ファイル名_2
```

ファイル名を指定しない場合は、変更された全てのファイルが復元されます。

3.3 スナップショットからの起動によるシステムのロールバック

openSUSE Leap に含まれている GRUB 2 では、btrfs のスナップショットからの起動を行うことができます。これは Snapper のロールバック機能と共に働くもので、これによって誤って設定してしまったシステムを、元に戻すことができるようになります。ただし、既定の Snapper 設定 (`root`) で作成したスナップショットのみを起動することができます。

重要: サポートされる設定

openSUSE Leap 15.7 では、ルートパーティションに対する既定のサブボリューム設定を変更していない場合にのみ、システムロールバックをサポートします。

スナップショットから起動を行う場合、スナップショット内に含まれるファイルシステムは読み込み専用でマウントされます。また、その他のファイルシステムやスナップショットから除外されているサブボリュームなどについては、読み書きが可能で修正することができるようになります。

重要: 変更点の巻き戻しとロールバックについて

データを復元する目的でスナップショットを利用する場合、Snapper が処理できるシナリオとしては、下記の 2 つのものがあることを知っておくのが重要です:

変更点の巻き戻し

3.2項「Snapper による変更点の巻き戻し」で説明しているとおりに変更点を巻き戻す場合、2 つのスナップショット間を比較して、これらの間での違いを元に戻すことになります。この方法を利用することで、選択したファイルが明示的に巻き戻されないようにすることができます。

ロールバック

下記に示す方法でロールバックを行うと、システムはスナップショットを採取した時点の状態にリセットされます。

起動可能なスナップショットからロールバックを行うには、下記の要件を満たさなければなりません。既定の状態でインストールを行った場合は、下記の要件を満たすように設定されます。

起動可能なスナップショットからのロールバックに対する必要条件

- ルートファイルシステムは btrfs でなければなりません。LVM ボリュームのスナップショットからの起動には、対応していません。
- ルートファイルシステムは単一のデバイスで構成されている必要があります。これを確認するには、`sudo /sbin/btrfs filesystem show` を実行してください。`Total devices 1` と表示されれば問題ありません。`1` より多くのデバイスが表示された場合、お使いの環境ではご利用いただけません。

注記: スナップショットから除外されるディレクトリによる影響

`/srv` など、スナップショットから除外されるディレクトリ (詳しくは 3.1.3項「スナップショットから除外されるディレクトリ」をお読みください) については、別途のパーティション内に存在していてもかまいません。

- システムはインストールされたブートローダから起動できる必要があります。
- `/` のサブボリューム内に含まれる内容のみをロールバックすることができます。それ以外のサブボリュームを含めることはできません。

起動可能なスナップショットからロールバックを行うには、下記のように行います:

1. まずはシステムを起動します。起動メニューでは［Bootable snapshots］(起動可能なスナップショット) を選択して、起動したいスナップショットを選択します。スナップショットは日付順に並べられていて、最も新しいものが最初に表示されます。

2. システムにログインします。全ての処理が期待通りに動作することを確認してください。また、スナップショットの一部になっているディレクトリに対しては、一切書き込むことができないことに注意してください。逆に、それ以外のディレクトリへの書き込みであれば、次の手順でどちらを選択した場合であっても、それらは失われることが ありません 。

3. ロールバックを行いたいかどうかによって、いずれかの手順を実行します:

   a. システムが期待通りに動作しない場合は、システムを再起動して現在のシステムに戻してください。再度別のスナップショットを選択してもかまいませんし、レスキューシステムをお使いいただいてもかまいません。

   b. ロールバックを実施するには、下記のように実行します:

   ```
   > sudo snapper rollback
   ```

   上記を実行した後、システムを再起動します。起動画面では既定の起動項目を選択して、復帰したシステムを起動してください。なお、ロールバックを行う前のファイルシステムに対するスナップショットが作成されます。ルートに対する既定のサブボリュームは、新しく読み書き可能なスナップショットに置き換えられます。詳しくは 3.3.1項「ロールバック後のスナップショット」をお読みください。
   なお、`-d` オプションを指定して、スナップショットに説明を設定しておくと便利です。たとえば下記のように指定します:

   ```
   日付 時刻 のロールバック以降の新しいルートファイルシステム
   ```

ヒント: 特定のインストール時点へのロールバック

インストール時にスナップショットを無効化していなかった場合、初期の起動可能なスナップショットがシステムインストールの最後に採取されます。このスナップショットから起動を行うことで、いつでも最初の時点に戻れるようになります。このスナップショットには、`after installation` という説明が設定されます。

起動可能なスナップショットは、サービスパックへのシステムアップグレードを開始した場合や、新しいメジャーリリースへのシステムアップグレードでも作成されます (ただし、スナップショットを無効化していない場合に限ります) 。

3.3.1 ロールバック後のスナップショット

ロールバックを実施する前に、現在実行中のファイルシステムに対するスナップショットが作成されます。説明には、ロールバックに復元した際のスナップショットの ID が書かれます。

ロールバックで作成されたスナップショットには、`Cleanup` 属性に `number` という値が設定されます。そのため、ロールバックスナップショットは設定されたスナップショット数に達したとき、自動的に削除されます。詳しくは 3.7項「自動的なスナップショットのクリーンアップ」をお読みください。スナップショットに重要なデータが含まれる場合は、スナップショットが削除される前に必要なデータを抽出するようにしてください。

3.3.1.1 ロールバックスナップショットの例

たとえば新規インストールを行った場合、システムには下記のスナップショットが作成されます:

```
# snapper --iso list
種類   | # |     | クリーンアップ | 説明                   | ユーザデータ
-------+---+ ... +---------------+------------------------+--------------
single | 0 |     |               | current                |
single | 1 |     |               | first root filesystem  |
single | 2 |     | number        | after installation     | important=yes
```

ここで `sudo snapper rollback` を実行すると、スナップショット `3` が作成され、ロールバック前のシステムの状態が保存されます。スナップショット `4` が新しい btrfs のサブボリュームとなり、システムの再起動後に使用されるものとなります。

```
# snapper --iso list
種類   | # |     | クリーンアップ | 説明                   | ユーザデータ
-------+---+ ... +---------------+------------------------+--------------
single | 0 |     |               | current                |
single | 1 |     | number        | first root filesystem  |
single | 2 |     | number        | after installation     | important=yes
single | 3 |     | number        | rollback backup of #1  | important=yes
single | 4 |     |               |                        |
```

3.3.2 スナップショットの起動項目へのアクセスと識別

スナップショットから起動を行うには、お使いのマシンを再起動して［Start Bootloader from a read-only snapshot］を選択します。すると、起動可能な全てのスナップショットが一覧表示されます。最も新しいスナップショットが最初に、最も古いスナップショットが最後に表示されます。`↓` および `↑` を利用して選択して、`Enter` を押してください。これで選択したスナップショットを有効化することができます。なお、起動メニューでスナップショットを選択しても、マシンはそのまま起動することはありません。選択したスナップショット内のブートローダを表示します。

図 3.1: ブートローダ: スナップショット

警告: UEFI を利用している場合に btrfs スナップショットからの Xen 起動が失敗する問題について

詳しくは https://www.suse.com/support/kb/doc/?id=000020602 をお読みください。

ブートローダ内でのスナップショット項目には、それらを容易に識別できるようにするための名前付け方式が設定されています:

```
[*]❶OS❷ (カーネル❸,日付❹T時刻❺,説明❻)
```

❶ スナップショットに `important` のマークが付けられている場合、項目には `*` というマークが付けられます。

❷ オペレーティングシステムのラベルです。

❹ `YYYY-MM-DD` 形式での日付です。

❺ `HH:MM` 形式での時刻です。

❻ この項目には、スナップショットの説明が書かれます。手作業で作成したスナップショットの場合、ここには作成時に `--description` で指定した文字列か、独自の文字列 (詳しくは ヒント: ブートローダのスナップショット項目に対する説明文の設定 をお読みください) が設定され

ます。自動作成されたスナップショットの場合は、呼び出し側のツール名が記録されます (例: `zypp(zypper)` または `yast_sw_single` など) 。説明が長い場合は自動的に後ろ側が切られますが、これは起動画面のサイズに依存して決まります。

ヒント: ブートローダのスナップショット項目に対する説明文の設定

スナップショットの説明文は、既定の文字列から変更することができます。これはたとえば、自動的に採取されたスナップショットの説明文では不十分だったり、ユーザが指定した説明文が長すぎたりするような場合に有用です。スナップショット `番号` に対して `文字列` を設定するには、下記のようなコマンドを実行します:

```
> sudo snapper modify --userdata "bootloader=文字列" 番号
```

なお、この説明は 25 文字以上の長さにはしないでください。このサイズを超過した場合、起動画面に表示しきれなくなることがあります。

3.3.3 制限事項

完全な システムロールバックには対応していません。たとえば、スナップショットを採取した時点と全く等価な状態にシステムを戻すことはできません。

3.3.3.1 スナップショットから除外されるディレクトリによる影響

ルートファイルシステムのスナップショットには、全てのディレクトリが含まれているというわけではありません。詳細と理由については、3.1.3項「スナップショットから除外されるディレクトリ」をお読みください。一般的には、これらのディレクトリのデータは復元されず、下記のような制限事項が生まれます。

ロールバック後のアドオンおよびサードパーティ製ソフトウエアの使用について

`/opt` など、スナップショットから除外されるサブボリューム内にデータをインストールするアプリケーションやアドオンについては、スナップショットに含まれるサブボリュームにデータをインストールするような場合は、ロールバックを行うことで動作しなくなってしまう場合があります。この場合は、アプリケーションやアドオンを再インストールして問題を解決してください。

ファイルアクセスの問題について

スナップショットと現在動作中のシステムとの間で、アプリケーションがファイルのアクセス権や所有者を変更している場合、アプリケーションがそれらのファイルにアクセスできなくなってしまうことがあります。この場合は、ロールバック後にアクセス権や所有者をリセットして対応してください。

非互換のデータ形式について

スナップショットと現在動作中のシステムでサービスやアプリケーションの使用するデータ形式が変更されているような場合、ロールバックを行うとアプリケーションがデータを読めなくなってしまうことがあります。

コードとデータが混在するサブボリュームの場合について

`/srv` のようなサブボリュームの場合、コードとデータが混在していることがあります。この場合ロールバックを行うと、動作しないコードになってしまう場合があります。これは PHP のダウングレードが発生する場合があるためで、これによって Web サーバ内で PHP スクリプトがうく動作しなくなってしまいます。

ユーザデータについて

ロールバックによってシステムからユーザが削除される場合、これらのユーザが所有していたデータは、スナップショットから除外されているディレクトリにあるものについては、削除が行われません。ロールバック後、同じユーザ ID でユーザを作成してしまうと、そのユーザが古いユーザのファイル権限を引き継ぐことになってしまいます。そのため、`find` のようなツールを利用して、孤立したファイルを検索して削除することをお勧めします。

3.3.3.2 ブートローダデータのロールバックについて

ブートローダをロールバックすることはできません。なぜなら、ブートローダにおける全ての「ステージ」同士が密接に関係し合っているためです。そのため、`/boot` のロールバックを行った場合には、正しく動作する保証ができません。

3.4 ユーザのホームディレクトリに対する Snapper の有効化

ユーザの `/home` ディレクトリに対してスナップショットを有効化することもできます。この場合、下記のような用途が考えられます:

- ユーザごとにスナップショットとロールバックを管理することができます。
- たとえばデータベースやシステム、ネットワークの管理者などのシステムユーザが、設定ファイルやドキュメンテーションなどのコピーを保持しておくことができるようになります。
- ホームディレクトリと btrfs バックエンドの組み合わせを利用した Samba 共有を作成することができます。

それぞれのユーザのホームディレクトリは `/home` の btrfs サブボリュームとなります。このような構成は手作業で設定することができます (詳しくは 3.4.3項「手作業でのホームディレクトリに対するスナップショットの有効化」をお読みください) 。ただし、`pam_snapper` を使用するより便利な方法もあります。`pam_snapper` パッケージは `pam_snapper.so` モジュールのほか、ユーザの作成や snapper の設定を作成するヘルパースクリプトをインストールします。

`pam_snapper` は `useradd` コマンドと Pluggable Authentication Modules (PAM) 、そして snapper との間を統合する機能を提供します。既定ではユーザのログイン時とログアウト時にスナップショットを作成するほか、ユーザがしばらくログインしたままの状態にあると、時間ベースのスナップショットも作成します。これらの既定値は通常の snapper コマンドと設定ファイルで変更することができます。

3.4.1 pam_snapper のインストールとユーザの作成

最も始めやすい方法は、btrfs でフォーマットした新しい `/home` ディレクトリを用意し、全くユーザが存在しない状況から行うことです。まずは `pam_snapper` をインストールします:

```
# zypper in pam_snapper
```

`/etc/pam.d/common-session` に下記の行を追加します:

```
session optional pam_snapper.so
```

ユーザとホームディレクトリを作成する場合は、`/usr/lib/pam_snapper/pam_snapper_useradd.sh` を使用します。既定では、スクリプトは表示のみを行って実行は行いませんので、スクリプトを編集して `DRYRUN=1` を `DRYRUN=0` に変更します。あとは下記のようにして新しいユーザを作成します:

```
# /usr/lib/pam_snapper/pam_snapper_useradd.sh \
ユーザ名 グループ passwd=パスワード
Create subvolume '/home/username'
useradd: 警告: ホームディレクトリが既に存在します。
skel ディレクトリからのコピーは行いません。
```

ユーザ側では、初回ログイン時にホームディレクトリ内に `/etc/skel` の内容がコピーされます。あとは下記のように実行して、ユーザの設定が作成されていることを確認します:

```
# snapper list --all
Config: home_username, subvolume: /home/username
種類   | # | 前 #  | 日付 | ユーザ | クリーンアップ | 説明     | ユーザデータ
-------+---+-------+------+--------+----------------+----------+--------------
single | 0 |       |      | root   |                | current |
```

時間が経過するごとに上記の出力は増えていきますが、ユーザ側では標準的な snapper コマンドで管理を行うことができます。

3.4.2 ユーザの削除

ユーザの削除についても `/usr/lib/pam_snapper/pam_snapper_userdel.sh` で行います。既定では、スクリプトは表示のみを行って実行は行いませんので、スクリプトを編集して `DRYRUN=1` を `DRYRUN=0` に変更します。あとは下記のようにしてユーザとユーザのサブボリューム、snapper の設定と全てのスナップショットを削除します:

```
# /usr/lib/pam_snapper/pam_snapper_userdel.sh username
```

3.4.3 手作業でのホームディレクトリに対するスナップショットの有効化

下記は、snapper を利用してユーザのホームディレクトリを手作業で設定するための手順を示しています。なお、`/home` は btrfs でフォーマットされていなければならず、実行する時点でユーザは存在していないものとします。

```
# btrfs subvol create /home/ユーザ名
# snapper -c home_ユーザ名 create-config /home/ユーザ名
# sed -i -e "s/ALLOW_USERS=\"\"/ALLOW_USERS=\"ユーザ名\"/g" \
/etc/snapper/configs/home_ユーザ名
# yast users add username=ユーザ名 home=/home/ユーザ名 password=パスワード
# chown ユーザ名.group /home/ユーザ名
# chmod 755 /home/ユーザ名/.snapshots
```

3.5 Snapper の設定の作成と修正

Snapper の動作は、各パーティションや `btrfs` サブボリューム別に設定ファイルで設定することができます。設定ファイルは `/etc/snapper/configs/` 内に存在しています。

ルートファイルシステムが十分に大きい (おおよそ 12 GB 以上) の場合、ルートファイルシステム `/` に対するスナップショットは、インストール時に自動的に有効化されます。この時点で作成される既定の設定は `root` という名前になっています。これにより、YaST や zypper のスナップショットを作成して管理しています。既定値の一覧について、詳しくは 3.5.1.1項「設定データ」をお読みください。

注記: スナップショットを有効化するための最小ルートファイルシステムサイズ

3.1項「既定の設定」で説明しているとおり、スナップショットの有効化を行うには、ルートファイルシステム内に追加の空き領域が必要となります。必要な容量はインストール済みのパッケージ数に依存するほか、スナップショットに含まれるボリュームで行われた変更量に依存して決まります。また、スナップショットの頻度やアーカイブされたスナップショットの数も影響します。

インストール時に自動的なスナップショットを有効化するのに必要な、最小のルートファイルシステムのサイズが決められています。現時点では、これは 12 GB に設定されています。この値は、アーキテクチャや基本システムのサイズによって、将来的に変更される可能性があります。また、この値はインストールメディア内の `/control.xml` ファイル内にある下記のタグで設定されています:

```
<root_base_size>
<btrfs_increase_percentage>
```

上記の値を利用して、下記の数式で計算することができます: `ROOT_BASE_SIZE の値` * (1 + `BTRFS_INCREASE_PERCENTAGE の値` /100)

ただし、ここでの値は最小値であることに注意してください。ルートファイルシステムに対しては、それ以上のサイズを設定することをお勧めします。一般的には、スナップショットを使用しない場合のサイズ想定の 2 倍程度を設定してください。

`btrfs` でフォーマットされた他のパーティションや、`btrfs` の既存のサブボリュームに対して、独自の設定を作成することもできます。下記の例では、`/srv/www` にマウントされた `btrfs` パーティション内の Web サーバのデータに対して、これらをバックアップするための設定を作成しています。

設定を作成したら、`snapper` を利用するか、もしくは YaST の［Snapper］モジュールを利用して、これらのスナップショットから復元することができるようになります。なお、YaST では［現在の設定］で作成した設定を選択する必要があります。`snapper` の場合は、グローバルスイッチ `-c` で設定を指定してください (例: `snapper -c myconfig list`) 。

Snapper の設定を作成するには、`snapper create-config` を実行します:

```
> sudo snapper -c www-data❶ create-config /srv/www❷
```

❶ 設定ファイルの名前

❷ スナップショットを採取したいパーティションのマウントポイント、もしくは `btrfs` サブボリューム名。

このコマンドは新しい設定ファイル `/etc/snapper/configs/www-data` を作成し、適切な既定値を設定します (設定値は `/etc/snapper/config-templates/default` にあるものを使用します) 。これらの既定値を調整する方法について、詳しくは 3.5.1項「既存の設定の管理」をお読みください。

ヒント: 設定の既定値

新しい設定に対する既定値は、`/etc/snapper/config-templates/default` にあるものを使用します。独自の既定値を設定したい場合は、同じディレクトリ内にファイルのコピーを作成して、必要な値を設定して保存してください。あとは設定を作成する際、create-config コマンドに `-t` オプションを追加して設定します:

```
> sudo snapper -c www-data create-config -t MY_DEFAULTS /srv/www
```

3.5.1 既存の設定の管理

`snapper` コマンドには、既存の設定を管理するためのいくつかのサブコマンドが用意されています。一覧表示のほか、表示と削除、修正にそれぞれ対応しています:

設定の一覧表示

既存の全ての設定を一覧表示するには、`snapper list-configs` コマンドを使用します:

```
> sudo snapper list-configs
設定    | サブボリューム
-------+----------
root   | /
usr    | /usr
local  | /local
```

設定の表示

指定した設定を表示するには、`snapper -c 設定 get-config` のサブコマンドを使用します。`設定` の箇所には `snapper list-configs` で表示される設定の名前を指定します。設定オプションについて、詳しくは 3.5.1.1項「設定データ」をお読みください。
既定の設定値を表示したい場合は、下記のように実行します:

```
> sudo snapper -c root get-config
```

設定の修正

指定した設定のオプションを変更するには、`snapper -c 設定 set-config オプション=値` のサブコマンドを使用します。`設定` の箇所には `snapper list-configs` で表示される設定の名前を指定します。`オプション` と `値` で設定可能な値は、3.5.1.1項「設定データ」に一覧表示しています。

設定の削除

指定した設定を削除するには、`snapper -c 設定 delete-config` のサブコマンドを使用します。`設定` の箇所には `snapper list-configs` で表示される設定の名前を指定します。

3.5.1.1 設定データ

各設定には、コマンドラインから変更することのできるオプションの一覧が含まれています。下記の一覧には、それぞれのオプションに対する詳細が記されています。値を変更するには、`snapper -c 設定 set-config "キー=値"` の形式で入力して実行します。

`ALLOW_GROUPS` , `ALLOW_USERS`

一般ユーザに対してスナップショットの使用を許可するためのオプションです。詳しくは 3.5.1.2項「一般ユーザからの Snapper の使用」をお読みください。

既定値は `""` です。

`BACKGROUND_COMPARISON`

事前と事後のスナップショットを作成後、裏でそれらを比較しておくべきかどうかを指定します。

既定値は `"yes"` (はい) です。

`EMPTY_*`

事前と事後のスナップショット対で、変更がない場合のクリーンアップアルゴリズムを設定します。詳しくは 3.7.3項「違いのないスナップショット対のクリーンアップ」をお読みください。

`FSTYPE`

パーティションのファイルシステムの種類を指定します。変更してはなりません。

既定値は `"btrfs"` です。

`NUMBER_*`

インストールスナップショットと管理スナップショットに対するクリーンアップアルゴリズムを指定します。詳しくは 3.7.1項「番号スナップショットのクリーンアップ」をお読みください。

`QGROUP` / `SPACE_LIMIT`

クリーンアップアルゴリズムにクォータへの対応を追加します。詳しくは 3.7.5項「ディスククォータサポートの追加」をお読みください。

`SUBVOLUME`

スナップショットを採取するパーティションのマウントポイントまたはサブボリュームを指定します。変更してはなりません。
既定値は `"/"` です。

`SYNC_ACL`

Snapper を一般ユーザから使用している場合 (詳しくは 3.5.1.2項「一般ユーザからの Snapper の使用」をお読みください) 、対象のユーザは `.snapshot` ディレクトリへのアクセス許可と、そのディレクトリ内のファイルの読み込み許可が必要となります。SYNC_ACL を `yes` に設定していると、Snapper は ALLOW_USERS と ALLOW_GROUPS の設定にあわせて、自動的にユーザとグループへのアクセス ACL を設定します。
既定値は `"no"` (いいえ) です。

`TIMELINE_CREATE`

`yes` (はい) に設定していると、1 時間ごとのスナップショットを作成します。設定可能な値は下記のとおりです: `yes` (はい), `no` (いいえ)
既定値は `"no"` (いいえ) です。

`TIMELINE_CLEANUP` / `TIMELINE_LIMIT_*`

タイムラインスナップショットに対するクリーンアップアルゴリズムを設定します。詳しくは 3.7.2項「タイムラインスナップショットのクリーンアップ」をお読みください。

3.5.1.2 一般ユーザからの Snapper の使用

既定では、Snapper は `root` だけが使用できます。しかし、場合によっては特定のグループやユーザに対してアクセスを許可し、スナップショットの作成やスナップショットからの巻き戻しを実施したい場合があります:

- `/srv/www` 以下のスナップショットを採取しておきたい Web サイト管理者
- 自分自身のホームディレクトリのスナップショットを採取しておきたい一般ユーザ

このような目的で、Snapper には一般ユーザやグループへのアクセス許可を設定することができます。なお、対応する `.snapshots` ディレクトリには、それらのユーザからアクセスできるよう権限を設定する必要があります。権限の設定を最も簡単に行うには、SYNC_ACL オプションを `yes` に設定するのがお勧めです。

手順 3.5: 一般ユーザに対する SNAPPER の使用の許可

> ここで示す手順は、いずれも `root` で実施すべきものです。

1. Snapper の設定をまだ作成していない場合は、設定を作成します。詳しい手順は 3.5項「Snapper の設定の作成と修正」をお読みください。たとえば下記のように実行します:

```
> sudo snapper --config web_data create /srv/www
```

2. 設定ファイルは `/etc/snapper/configs/設定` に作成されます (ここで、`設定` には `-c/--config` で指定した値が入ります。たとえば上記の例では、`/etc/snapper/configs/web_data` になります) 。必要に応じて設定を変更することもできます。詳しくは 3.5.1項「既存の設定の管理」をお読みください。

3. アクセスを許可したいユーザやグループを、`ALLOW_USERS` および `ALLOW_GROUPS` でそれぞれ指定します。複数のユーザやグループを指定したい場合は、`Space` で区切って指定してください。なお、`www_admin` ユーザに対して許可を与えたい場合は、下記のように実行します:

```
> sudo snapper -c web_data set-config "ALLOW_USERS=www_admin" SYNC_ACL="yes"
```

4. これで、指定した Snapper の設定を指定したユーザやグループから使用できるようになります。たとえば `list` のようなコマンドで、利用できることを確認することができます:

```
www_admin:~ > snapper -c web_data list
```

3.6 スナップショットの手動作成と管理

Snapper は設定で自動的にスナップショットを採取したり管理したりするだけでなく、コマンドラインや YaST モジュールを利用して、スナップショット対 (「事前」および「事後」) や単一のスナップショットを手動で作成することもできます。

全ての Snapper の操作は、既存の設定に対して行われます (詳しくは 3.5項「Snapper の設定の作成と修正」をお読みください) 。また、スナップショットは設定の存在するパーティションやボリュームに対してのみ採取することができます。既定では、システム設定 (`root`) を使用します。独自の設定でスナップショットを作成したり管理したりしたい場合は、設定を明示的に指定する必要があります。YaST の場合は［現在の設定］のドロップダウンボックスを、コマンドラインの場合は `-c` をそれぞれ指定してください (例: `snapper -c 設定 COMMAND`) 。

3.6.1 スナップショットのメタデータ

それぞれのスナップショットにはメタデータが含まれています。スナップショットを作成する際、メタデータを指定する必要があります。また、スナップショットの内容については変更ができないことから、スナップショットの修正はメタデータの変更を意味することにもなります。`snapper list` を実行すると、既存のスナップショットとそのメタデータを表示することができます:

`snapper --config home list`
: 上記を実行すると、`home` という設定のスナップショットを一覧表示します。既定の設定 (root) に対するスナップショットを一覧表示するには、`snapper -c root list` もしくは単に `snapper list` と実行します。

`snapper list -a`
: 全ての既存の設定に対するスナップショットを一覧表示します。

`snapper list -t pre-post`
: 既定の設定 (`root`) に対する全ての事前／事後スナップショット対を一覧表示します。

`snapper list -t single`
: 既定の設定 (`root`) に対する全ての `単一` スナップショットを一覧表示します。

各スナップショットに対して、下記のメタデータが用意されています:

- Type : スナップショットの種類。詳しくは 3.6.1.1項「スナップショットの種類」をお読みください。このデータは変更できません。
- Number : スナップショットに対する唯一の番号。このデータは変更できません。
- Pre Number : 対応する事前スナップショットの番号。Type=post のスナップショットにのみ設定されます。このデータは変更できません。
- Description : スナップショットの説明。
- Userdata : カンマ区切りで key=value 形式の独自データを設定することのできるメタデータです。たとえば下記のように設定します: `reason=testing, project=foo` 。この項目は重要性を表す際に使用される (`important=yes`) ほか、一般ユーザが作成したスナップショットに対しても設定されます (user=tux) 。
- Cleanup-Algorithm : スナップショットに対するクリーンアップアルゴリズム。詳しくは 3.7項「自動的なスナップショットのクリーンアップ」をお読みください。

3.6.1.1 スナップショットの種類

Snapper で採取できるスナップショットには、事前／事後／単一の種類があります。物理的には違いがありませんが、Snapper 側での処理に違いがあります。

`pre` (事前)
: 変更を行う 前 のファイルシステムのスナップショットです。それぞれの `pre` スナップショットには、対応する `post` スナップショットが存在します。たとえば YaST や zypper が自動的に採取するスナップショットがこれにあたります。

`post` (事後)
: 変更を行った 後 のファイルシステムのスナップショットです。それぞれの `post` スナップショットには、対応する `pre` スナップショットが存在します。たとえば YaST や zypper が自動的に採取するスナップショットがこれにあたります。

`single` (単一)
: 単独で採取されるスナップショットです。自動的な定時スナップショットなどで使用します。スナップショットを作成する際、既定の種類はこれになります。

3.6.1.2 クリーンアップアルゴリズム

Snapper には、古いスナップショットをクリーンアップ (清掃) するためのアルゴリズムが 3 種類存在しています。アルゴリズムは日々の `cron` ジョブで実行されます。また snapper の設定内に、複数の種類のスナップショットを設定することもできます (詳しくは 3.5.1項「既存の設定の管理」をお読みください) 。

number
: 特定のスナップショット数に達すると、古いスナップショットを削除します。

timeline
: 指定した時間が経過すると、古いスナップショットを削除します。ただし、いくつかの定期スナップショット (1 時間ごと, 1 日ごと, 1 ヶ月ごと, 1 年ごと) は保持されます。

empty-pre-post
: 差分を全く含んでいない事前／事後スナップショット対を削除します。

3.6.2 スナップショットの採取

スナップショットの採取は、`snapper create` を実行するか、もしくは YaST モジュールの［Snapper］で［作成］を押すことで作成することができます。下記の例では、コマンドラインからスナップショットを採取する方法を示しています。YaST インターフェイスであれば、より簡単に採取することができます。

ヒント: スナップショットの説明

スナップショットの説明にはわかりやすい説明を記述しておいて、後から何のために採取したものなのかを判別できるようにしておくことをお勧めします。`--userdata` オプションを使用すると、さらに詳しい情報を保存することができます。

`snapper create --from 17 --description "with package2"`

上記のように入力して実行すると、既存のスナップショットからの単一スナップショットを作成することができます。スナップショットの一覧は、`snapper list` コマンドを実行して確認します (これは Snapper バージョン 0.8.4 もしくはそれ以降のバージョンで利用できます) 。

`snapper create --description "Snapshot for week 2 2014"`

既定の設定 (`root`) に対して単一の (single) スナップショットを作成し、説明文を指定した場合の例です。クリーンアップアルゴリズムを指定していないため、このスナップショットは自動では削除されません。

`snapper --config home create --description "Cleanup in ~tux"`

`home` という設定に対して単一の (single) スナップショットを作成し、説明文を指定した場合の例です。クリーンアップアルゴリズムを指定していないため、このスナップショットは自動では削除されません。

`snapper --config home create --description "Daily data backup" --cleanup-algorithm timeline` >

`home` という独自の設定に対して単一の (single) スナップショットを作成し、説明文を指定した場合の例です。クリーンアップアルゴリズム (cleanup-algorithm) に timeline を指定しているため、その条件を満たすと自動的に削除されるようにもなっています。

`snapper create --type pre --print-number --description "Before the Apache config cleanup" --userdata "important=yes"`

種類に `pre` (事前) を指定してスナップショットを作成し、そのスナップショット番号を表示する場合の例です。何を行う「前」と「後」の状態を保存しておく際、「前」を採取する場合に使用するコマンドです。このスナップショットには important (重要) のマークが付けられます。

`snapper create --type post --pre-number 30 --description "After the Apache config cleanup" --userdata "important=yes"`

種類に `post` (事後) を指定してスナップショットを作成し、`pre` (事前) 側のスナップショット番号を `30` として採取する場合の例です。何を行う「前」と「後」の状態を保存しておく際、「後」を採取する場合に使用するコマンドです。このスナップショットには important (重要) のマークが付けられます。

`snapper create --command コマンド --description "Before and after COMMAND"`

`コマンド` で指定されるコマンドを実行する前後に、スナップショット対を自動的に作成するコマンドです。このオプションは、snapper をコマンドラインから実行する場合にのみ利用できます。

3.6.3 スナップショットのメタデータの変更

Snapper では、スナップショットに対する説明やクリーンアップアルゴリズム、ユーザデータをそれぞれ変更することができます。その他の全てのメタデータは変更できません。下記の例では、コマンドラインからスナップショットを変更する方法を示しています。YaST インターフェイスであれば、より簡単に変更することができます。

コマンドラインからスナップショットの変更を行うには、まずスナップショットの番号を知っておく必要があります。スナップショットの一覧と番号を表示するには、`snapper list` を実行してください。

YaST [Snapper] モジュールの場合は、開いた時点で全てのスナップショットが表示されます。一覧からいずれかを選んで [修正] を押してください。

`snapper modify --cleanup-algorithm "timeline"` 10

既定の設定 (`root`) のスナップショット 10 に対して、メタデータの修正を行います。この場合は、クリーンアップアルゴリズムを `timeline` に設定します。

`snapper --config home modify --description "daily backup" -cleanup-algorithm "timeline" 120`

`home` という名前の設定のスナップショット 120 に対して、メタデータの修正を行います。新しい説明を設定し、クリーンアップアルゴリズムを `timeline` に設定します。

3.6.4 スナップショットの削除

YaST [Snapper] モジュールでスナップショットを削除するには、一覧からスナップショットを選んで [削除] を押します。

コマンドラインツールからスナップショットを削除するには、まずスナップショットの番号を知っておく必要があります。スナップショットの一覧と番号を表示するには、`snapper list` を実行してください。その後、`snapper delete` 番号 と入力して実行します。

現在の既定のサブボリュームスナップショットを削除することはできません。

Snapper でスナップショットを削除する場合は、裏で動作する btrfs のプロセスが空き領域を開放します。そのため、空き領域の表示も遅れて反映されることになるほか、空き領域を利用する場合もしばらく待ってから行う必要があります。空き領域を即時に反映させたい場合は、delete コマンドに `--sync` オプションを指定してください。

ヒント: スナップショット対の削除

`pre` (事前) スナップショットを削除する際は、必ず対応する `post` (事後) スナップショットもあわせて削除してください (逆も同様です) 。

`snapper delete 65`

既定の設定 (`root`) のスナップショット 65 を削除します。

`snapper -c home delete 89 90`

`home` という名前の設定のスナップショット 89 と 90 を削除します。

`snapper delete --sync 23`

既定の設定 (`root`) のスナップショット 23 を削除し、空き容量を即時に反映させるようにします。

ヒント: 参照されていないスナップショットの削除

場合によっては、btrfs のスナップショットが存在しているものの、メタデータを含む XML ファイルが失われている場合があります。この場合、スナップショットは Snapper で表示することができません。また、手作業で削除する必要があります:

```
btrfs subvolume delete /.snapshots/SNAPSHOTNUMBER/snapshot
rm -rf /.snapshots/SNAPSHOTNUMBER
```

ヒント: 古いスナップショットの占有領域について

ハードディスクの空き容量を増やすためにスナップショットを削除する場合、古いスナップショットから順に削除することをお勧めします。これは、古いスナップショットほどより多くのディスク領域を占有するためです。

スナップショットは毎日の cron ジョブでも自動的に削除されます。詳しくは 3.6.1.2項「クリーンアップアルゴリズム」をお読みください。

3.7 自動的なスナップショットのクリーンアップ

スナップショットはディスク領域を占有するだけでなく、時間が経過するごとに占有されるサイズが大きくなっていく、という特徴があります。ディスクの空き容量が枯渇しないようにするため、Snapper では古いスナップショットを自動的に削除するアルゴリズムを提供しています。これらのアルゴリズムでは、タイムラインスナップショットと番号スナップショット (管理スナップショットとインストールスナップショット) とを区別して扱います。また、それぞれの種類に対して、維持すべきスナップショット数を設定することもできます。

これに加えて必要であれば、スナップショットが占有するディスク領域の最大サイズとして、ディスククォータを設定することもできます。また、pre (事前) と post (事後) のスナップショットのうち、差異が無いものを自動削除することもできます。

クリーンアップのアルゴリズムは、常に 1 つの Snapper 設定に結びつけられています。そのため、それぞれの設定に対してアルゴリズムを設定する必要があります。また、特定のスナップショットを自動削除されないように設定することもできます。詳しくは 問: をお読みください。

既定の設定 (`root`) では、番号スナップショットと pre (事前) と post (事後) で差異のないものをクリーンアップするよう設定しています。また、クォータサポートも有効化されていて、ルートパーティションのディスク領域に対して、スナップショットが 50% 以上を占有しないように設定されています。また、タイムラインスナップショットは無効化されているため、タイムラインスナップショットのクリーンアップアルゴリズムも無効化されています。

3.7.1 番号スナップショットのクリーンアップ

番号スナップショット (管理スナップショットとインストールスナップショット) のクリーンアップは、Snapper 設定のうち下記のパラメータで制御することができます。

`NUMBER_CLEANUP`

インストールスナップショットと管理スナップショットの対に対して、クリーンアップを有効にするか無効にするかを設定することができます。スナップショットの対は、`NUMBER_LIMIT` で指定した数を超過するか、もしくは `NUMBER_LIMIT_IMPORTANT` と `NUMBER_MIN_AGE` の両方 を超過した場合に削除されます。設定可能な値は `yes` (有効), `no` (無効) のいずれかです。
既定値は `"yes"` (はい) です。

変更や設定の例:

```
> sudo snapper -c 設定 set-config "NUMBER_CLEANUP=no"
```

`NUMBER_LIMIT` / `NUMBER_LIMIT_IMPORTANT`

通常の番号スナップショットと重要なインストール／管理スナップショットに対して、保持すべきスナップショット対の数を指定します。なお、`NUMBER_CLEANUP` か `"no"` の場合は無視されます。既定値はそれぞれ `NUMBER_LIMIT` が `"2-10"` 、`NUMBER_LIMIT_IMPORTANT` が `"4-10"` に設定されています。クリーンアップ処理では、それぞれ指定した最大値以上のスナップショットが削除され、不要なスナップショットとその領域が残らないようにします。また、クリーンアップ処理では、スナップショットやファイルシステムに対する制限に到達するまで、上記で指定した最小値を超えるスナップショットも削除します。
変更や設定の例:

```
> sudo snapper -c 設定 set-config "NUMBER_LIMIT=10"
```

重要: 範囲指定と固定値について

クォータサポートを有効にしている場合 (3.7.5項「ディスククォータサポートの追加」をお読みください) は、`2-10` のように最小-最大の形式で制限を範囲指定する必要があります。クォータサポートが無効の場合は、`10` のように固定値を指定する必要があります。それ以外の場合は、エラーで失敗します。

`NUMBER_MIN_AGE`

スナップショットを自動削除するまでの最小経過秒数を指定します。ここで指定した値よりも新しいスナップショットは、どれだけ存在していても削除されません。
既定値は `"1800"` です。
変更や設定の例:

```
> sudo snapper -c 設定 set-config "NUMBER_MIN_AGE=864000"
```

注記: 制限と世代

`NUMBER_LIMIT` , `NUMBER_LIMIT_IMPORTANT` , `NUMBER_MIN_AGE` は常に自動削除の判断基準となります。スナップショットは、全て の条件に合致した場合にのみ削除されます。

新しいか古いかに関係なく、常に `NUMBER_LIMIT*` で設定した数だけスナップショットを保持させたい場合は、`NUMBER_MIN_AGE` を `0` に設定してください。

下記の例は、直近の 10 個の重要なスナップショットと通常のスナップショットを、その古さに関係なく保持する設定例です:

```
NUMBER_CLEANUP=yes
NUMBER_LIMIT_IMPORTANT=10
NUMBER_LIMIT=10
NUMBER_MIN_AGE=0
```

特定の時間が経過したら常にスナップショットを削除してかまわない場合は、`NUMBER_LIMIT*` に `0` を設定し、`NUMBER_MIN_AGE` で時間を設定してください。

下記の例は、常に直近の 10 日分のスナップショットのみを保持する設定です:

```
NUMBER_CLEANUP=yes
NUMBER_LIMIT_IMPORTANT=0
NUMBER_LIMIT=0
NUMBER_MIN_AGE=864000
```

3.7.2 タイムラインスナップショットのクリーンアップ

タイムラインスナップショットのクリーンアップは、Snapper 設定のうち下記のパラメータで制御することができます。

`TIMELINE_CLEANUP`

タイムラインスナップショットに対して、クリーンアップを有効にするか無効にするかを設定することができます。スナップショットは `TIMELINE_LIMIT_*` と `TIMELINE_MIN_AGE` の 両方 を超過した場合に削除されます。設定可能な値は `yes` (有効), `no` (無効) のいずれかです。

既定値は `"yes"` (はい) です。

変更や設定の例:

```
> sudo snapper -c 設定 set-config "TIMELINE_CLEANUP=yes"
```

`TIMELINE_LIMIT_DAILY` , `TIMELINE_LIMIT_HOURLY` , `TIMELINE_LIMIT_MONTHLY` , `TIMELINE_LIMIT_WEEKLY` , `TIMELINE_LIMIT_YEARLY`

時間単位／日単位／週単位／年単位に保持すべきスナップショット数を指定します。

各項目の既定値はそれぞれ `"10"` になっています。ただし、`TIMELINE_LIMIT_WEEKLY` については、例外的に `"0"` が設定されます。

`TIMELINE_MIN_AGE`

スナップショットを自動削除するまでの最小経過秒数を指定します。

既定値は "1800" です。

例 3.1: タイムライン設定の例

```
TIMELINE_CLEANUP="yes"
TIMELINE_CREATE="yes"
TIMELINE_LIMIT_DAILY="7"
TIMELINE_LIMIT_HOURLY="24"
TIMELINE_LIMIT_MONTHLY="12"
TIMELINE_LIMIT_WEEKLY="4"
TIMELINE_LIMIT_YEARLY="2"
TIMELINE_MIN_AGE="1800"
```

この設定例は 1 時間おきのスナップショットを自動的にクリーンアップする例です。TIMELINE_MIN_AGE と TIMELINE_LIMIT_* の両方が常に自動削除の判断基準となります。この例では、スナップショットを削除するまでの最小経過秒数が 30 分 (1800 秒) になっています。1 時間おきのスナップショットを設定している場合は、常に最新のスナップショットのみを保持することになります。また、TIMELINE_LIMIT_DAILY を 0 以外にしていると、1 日の最初のスナップショットも保持されることになります。

保持されるスナップショット

- 1 時間単位: 直近の 24 個のスナップショットを保持します。
- 1 日単位: 直近の 7 日分に対して、1 日の最初のスナップショットを保持します。
- 1 ヶ月単位: 直近の 12 ヶ月分に対して、月の最後の日に作成された最初のスナップショットを保持します。
- 1 週単位: 直近の 4 週間に対して、週の最後の日に作成された最初のスナップショットを保持します。
- 1 年単位: 直近の 2 年に対して、年の最後の日に作成された最初のスナップショットを保持します。

3.7.3 違いのないスナップショット対のクリーンアップ

3.1.2項「スナップショットの種類」で説明しているとおり、YaST モジュールや zypper を実行すると、その起動時に pre (事前) スナップショットが作成され、終了時に post (事後) スナップショットが作成されます。ただ、何も変更せずにそれらを終了したような場合など、pre と post の間で何も変更が無い場合があります。このような「空の」スナップショットは、Snapper の設定内の下記パラメータを利用することで、自動削除を行うことができます:

EMPTY_PRE_POST_CLEANUP

yes (はい) を指定すると、何も差異のない pre と post のスナップショット対が削除されます。

既定値は `"yes"` (はい) です。

`EMPTY_PRE_POST_MIN_AGE`
: 何も差異のない pre/post スナップショット対のうち、自動削除するまでの最小経過秒数を指定します。
既定値は `"1800"` です。

3.7.4 手作業で作成したスナップショットのクリーンアップ

Snapper には、手作業で採取したスナップショットに対する独自のクリーンアップアルゴリズムが用意されていません。しかしながら、番号やタイムラインのクリーンアップアルゴリズムを、手作業で作成したスナップショットに適用することができます。これを行うと、スナップショットがアルゴリズムの「クリーンアップキュー」内に追加されるようになります。また、クリーンアップアルゴリズムは、スナップショットの作成時に指定することができるほか、既存のスナップショットに対して設定することもできます:

`snapper create --description "Test" --cleanup-algorithm number`
: 既定の設定 (root) に対して単一の (single) スナップショットを作成し、`number` (番号) クリーンアップアルゴリズムを適用した場合の例です。

`snapper modify --cleanup-algorithm "timeline" 25`
: スナップショット番号 25 に対して、クリーンアップアルゴリズム `timeline` (タイムライン) を適用した場合の例です。

3.7.5 ディスククォータサポートの追加

上述の番号／タイムラインのクリーンアップアルゴリズムに加えて、Snapper ではクォータ (容量制限) を設定することができます。容量制限は利用可能な空き容量に対する比率で指定します。この比率での指定は、それぞれの Snapper 設定内で指定された btrfs サブボリュームに対して常に適用されます。

btrfs のクォータは、ユーザではなくサブボリュームに対して適用されます。また、btrfs のクォータは、ユーザやグループに対するクォータ (`quota` コマンドを使用するクォータ) と併用することができます。

インストール時に Snapper を有効化していると、クォータサポートも自動的に有効化されます。Snapper を後から有効化した場合は、`snapper setup-quota` を実行することでクォータサポートを有効化することができます。また、この動作には正しい設定が必要です (詳しくは 3.5項「Snapper の設定の作成と修正」をお読みください) 。

クォータサポートは、Snapper 設定のうち下記のパラメータで制御することができます。

`QGROUP`

Snapper で使用されるクォータグループを指定します。何も指定されていない場合は、`snapper setup-quota` で設定することができます。既に設定されている場合は、`man 8 btrfs-qgroup` をよくお読みのうえ変更してください。また、この値は `snapper setup-quota` で設定すべきもので、通常は変更すべきではありません。

`SPACE_LIMIT`

1 を 100% とする小数で、スナップショットに対して許可される容量を指定します。指定可能な値は 0 から 1 まで (0.1 = 10%, 0.2 = 20%, ...) です。

なお、下記の制限事項とガイドラインをお読みください:

- クォータは既存の番号／タイムラインに対するクリーンアップアルゴリズムに 加えて のみ有効化されるべきものです。何もクリーンアップアルゴリズムを指定していないと、クォータによる制限も適用されません。
- クォータサポートが有効化されていると、Snapper は必要に応じて 2 回のクリーンアップ処理を実施します。1 回目の実行では番号もしくはタイムラインのスナップショットに対するルールを適用し、その時点でクォータを超過している場合、2 回目の実行としてクォータ固有のルールが適用されます。
- クォータサポートが有効化されていても、Snapper は `NUMBER_LIMIT*` と `TIMELINE_LIMIT*` の各値で指定された数のスナップショットを、クォータを超過している場合でも常に保持します。そのため、`NUMBER_LIMIT*` と `TIMELINE_LIMIT*` では値の範囲を指定 (`MIN-MAX`) して、クォータによる削除が適用されるようにしてください。
  たとえば `NUMBER_LIMIT=5-20` という設定の場合、Snapper は最初のクリーンアップ実行で定期的に採取されたスナップショットを 20 個まで減らします。20 個まで減らしてもクォータを超過している場合は、Snapper はクォータを超過しなくなるまで古いスナップショットを削除します。ただし、最新の 5 個分については、領域の問題が解決していなくても、必ず保持する動作になります。

3.8 スナップショットが占有しているディスク容量の表示

スナップショットではストレージ領域を効率的に使用する目的で、データの共有を行っています。そのため、`du` や `df` 等のコマンドを使用しても、残りのディスク領域を正確に測定することができません。btrfs でクォータを有効化している場合、ディスク領域を空けるには、まずそれぞれのスナップショットが占有している (共有していない) ディスク容量を正確に把握する必要があります。snapper 0.6 もしくはそれ以降のバージョンでは、**使用済み領域** の列内に占有している容量が表示されます:

```
# snapper --iso list
```

```
  # | 種類    | 前 #  | 日付                           | ユーザ | 使用済み領域 | クリーンアップ| 説明
                  | ユーザデータ
----+--------+-------+------------------------------+--------+--------------
+---------------+----------------------+-------------
 0  | single |       |                              |  root  |              |
 | current               |
 1* | single |       | 2019年07月22日 13時08分38秒 |  root  |   16.00 KiB  |               |
 first root filesystem |
 2  | single |       | 2019年07月22日 14時21分05秒 |  root  |   14.23 MiB  |    number     |
 after installation    | important=yes
 3  | pre    |       | 2019年07月22日 14時26分03秒 |  root  |  144.00 KiB  |    number     |
 zypp(zypper)          | important=no
 4  | post   |     3 | 2019年07月22日 14時26分04秒 |  root  |  112.00 KiB  |    number     |
                       | important=no
 5  | pre    |       | 2019年07月23日 08時19分36秒 |  root  |  128.00 KiB  |    number     |
 zypp(zypper)          | important=no
 6  | post   |     5 | 2019年07月23日 08時19分43秒 |  root  |   80.00 KiB  |    number     |
                       | important=no
 7  | pre    |       | 2019年07月23日 08時20分50秒 |  root  |  256.00 KiB  |    number     |
 yast sw_single        |
 8  | pre    |       | 2019年07月23日 08時23分22秒 |  root  |  112.00 KiB  |    number     |
 zypp(ruby.ruby2.5)    | important=no
 9  | post   |     8 | 2019年07月23日 08時23分35秒 |  root  |   64.00 KiB  |    number     |
                       | important=no
10  | post   |     7 | 2019年07月23日 08時24分05秒 |  root  |   16.00 KiB  |    number     |
                       |
```

`btrfs` コマンドを使用しても、それぞれのスナップショットが占有するディスク容量を表示することができます:

```
# btrfs qgroup show -p /
qgroupid         rfer         excl parent
--------         ----         ---- ------
0/5          16.00KiB     16.00KiB ---
[...]
0/272         3.09GiB     14.23MiB 1/0
0/273         3.11GiB    144.00KiB 1/0
0/274         3.11GiB    112.00KiB 1/0
0/275         3.11GiB    128.00KiB 1/0
0/276         3.11GiB     80.00KiB 1/0
0/277         3.11GiB    256.00KiB 1/0
0/278         3.11GiB    112.00KiB 1/0
0/279         3.12GiB     64.00KiB 1/0
0/280         3.12GiB     16.00KiB 1/0
1/0           3.33GiB    222.95MiB ---
```

`qgroupid` 列には、それぞれのサブボリュームの識別番号と qgroup レベル/ID の組み合わせが表示されます。

`rfer` 列には、サブボリューム内で参照されているデータの容量が表示されます。

`excl` 列には、それぞれのサブボリューム内に存在する占有データ容量が表示されます。

`parent` 列には、サブボリュームの親 qgroup が表示されます。

末尾にある `1/0` には、親 qgroup に対する合計容量が表示されています。上記の例では、全てのサブボリュームを削除すると、222.95 MiB が解放されることになります。下記のコマンドを実行すると、スナップショットとサブボリュームの関係性を表示することができます:

```
# btrfs subvolume list -st /
ID      gen     top level       path
--      ---     ---------       ----
267     298     266             @/.snapshots/1/snapshot
272     159     266             @/.snapshots/2/snapshot
273     170     266             @/.snapshots/3/snapshot
274     171     266             @/.snapshots/4/snapshot
275     287     266             @/.snapshots/5/snapshot
276     288     266             @/.snapshots/6/snapshot
277     292     266             @/.snapshots/7/snapshot
278     296     266             @/.snapshots/8/snapshot
279     297     266             @/.snapshots/9/snapshot
280     298     266             @/.snapshots/10/snapshot
```

また、一方のサービスパックから他方のサービスパックにアップグレードを行ったりすると、パッケージの更新により多数のデータが変更されるため、スナップショットがシステムサブボリューム内の多くの領域を占有するようになります。これらは、不要になった段階で、手作業による削除を行うことをお勧めします。詳しくは 3.6.4項「スナップショットの削除」をお読みください。

3.9 よくある質問とその回答

問: Snapper が `/var/log` や `/tmp` などのディレクトリ内の変更を表示しないのはなぜ?

答: ディレクトリによってはスナップショットから除外されているものがあります。詳しい一覧とその理由については、3.1.3項「スナップショットから除外されるディレクトリ」をご覧ください。スナップショットからの除外にあたっては、サブボリュームを作成して対応しています。

問: ブートローダから特定のスナップショットを起動するには?

答: 詳しくは 3.3項「スナップショットからの起動によるシステムのロールバック」をお読みください。

問: スナップショットを削除されたりしないように保護することはできますか?

答: 現時点では、スナップショットを手作業による削除から保護する方法はありません。しかしながら、クリーンアップアルゴリズムで自動削除されないように設定することは可能です。手作業で採取したスナップショットの場合 (詳しくは 3.6.2項「スナップショットの採取」をお読みください) 、`--cleanup-algorithm` でクリーンアップアルゴリズムを指定しない限り、何もクリーンアップア

ルゴリズムが割り当てられません。また、自動的に採取されたスナップショットには、`number` または `timeline` のいずれかのアルゴリズムが割り当てられます。1 つまたは複数のスナップショットに対して、この割り当てを削除したい場合は、下記のようにして行います:

1. まずは利用可能な全てのスナップショットを一覧表示します:

   ```
   > sudo snapper list -a
   ```

2. 削除したくないスナップショットの番号 (複数可) を覚えておきます。

3. 下記のコマンドを実行します。このとき、上記で覚えておいた番号を `#1` `#2` `#n` の部分に指定します:

   ```
   > sudo snapper modify --cleanup-algorithm "" #1 #2 #n
   ```

4. 最後にもう一度 `snapper list -a` を実行して、結果を確認します。変更したスナップショットの `Cleanup` 欄が空白になっていれば成功です。

問: Snapper に関してさらに詳しい情報を知るには?

答: Snapper の Web ページ http://snapper.io/ をご覧ください。

4 VNC によるリモートグラフィカルセッション

改訂履歴

2024-05-13

VNC は Virtual Network Computing (仮想ネットワークコンピューティング) の略で、ネットワーク上離れたコンピュータを (シェルではなく) グラフィカルなデスクトップ経由でアクセスする機能を提供します。VNC はプラットフォームに依存しない構造で、任意のオペレーティングシステムからアクセスする機能を提供します。本章では、vncviewer や Remmina のようなデスクトップクライアントを利用して VNC サーバに接続する方法と、VNC サーバの操作方法について説明しています。

openSUSE Leap では 2 種類の VNC セッションに対応しています。1 つはワンタイムセッションと呼ばれ、クライアントからの接続が切れるまで「有効」となるセッションです。もう 1 つは永続セッションと呼ばれ、明示的に終了させるまでずっと「有効」であり続けるセッションです。

それぞれ異なるポートであれば、両方の種類のセッションを同じマシン内で同時に提供することもできます。ただし、既に開いているセッションの種類を変更することはできません。

4.1 `vncviewer` クライアント

重要: 対応するディスプレイマネージャ

VNC 接続を受け付けるには、XDMCP プロトコルに対応したディスプレイマネージャを使用する必要があります。`gdm`，`lxdm`，`lightdm` は XDMCP に対応していますが、KDE 5 の既定のディスプレイマネージャである `sddm` の場合は、XDMCP に対応していません。ディスプレイマネージャを変更した場合は、現在の X セッションをログアウトしたあと、下記のコマンドで再起動を行ってください:

```
> sudo systemctl restart xdm.service
```

サーバ側で提供されている VNC サービスに接続するには、クライアントソフトウエアが必要です。openSUSE Leap での既定は `vncviewer` で、`tigervnc` パッケージ内に含まれています。

4.1.1 vncviewer コマンドラインを利用した接続

VNC ビューアを起動してサーバ内のセッションに接続するには、下記のコマンドを使用します:

```
> vncviewer jupiter.example.com:1
```

ディスプレイ番号を指定する代わりに、コロンを 2 つ繋げてポート番号を指定することもできます:

```
> vncviewer jupiter.example.com::5901
```

注記: ディスプレイ番号とポート番号について

VNC クライアント側で指定するディスプレイ番号やポート番号は、接続先のマシンで VNC サーバを設定する際に指定したディスプレイ番号やポート番号と同じ値を指定します。詳しくは 4.4項「永続 VNC サーバセッションの設定」をお読みください。

4.1.2 vncviewer GUI を利用した接続

`--listen` を指定せずに `vncviewer` を実行するか、もしくは接続先のホストを指定して実行すると、詳細な接続設定を行うためのウインドウが表示されます。[VNC server] の欄には、4.1.1項「vncviewer コマンドラインを利用した接続」で指定しているような接続先を指定して、[Connect] を押します。

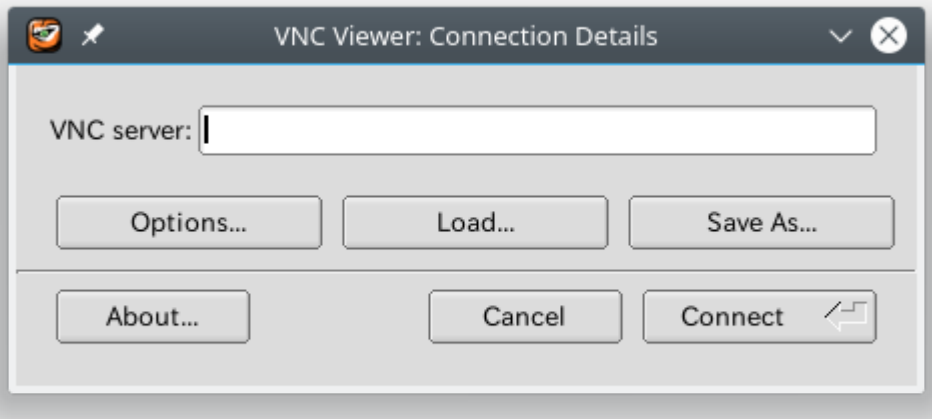


図 4.1: VNCVIEWER

4.1.3 暗号化されていない接続に対する通知

VNC プロトコルでは、様々な種類の暗号化接続 (注: パスワード認証とは全く異なり、通信そのものを暗号化して外部から内容を読み取られないようにする仕組みです) に対応しています。接続時に TLS を使用していないと、VNC ビューアのウインドウタイトルに「(Connection not encrypted!)」(通信が暗号化されていません) というメッセージが表示されるようになっています。

4.2 Remmina: リモートデスクトップクライアント

Remmina はモダンで機能の豊富なリモートデスクトップクライアントです。VNC, SSH, RDP, Spice などの各プロトコルに対応しています。

4.2.1 インストール

Remmina を使用するには、まず `remmina` パッケージをインストールする必要があります。インストールされていない場合は、下記のように実行して Remmina の VNC プラグインと共にインストールを行ってください:

```
# zypper in remmina remmina-plugin-vnc
```

4.2.2 メインウインドウ

Remmina を起動するには、`remmina` と入力して実行します。

図 4.2: REMMINA のメインウインドウ

アプリケーションのメインウインドウには、保存されているリモートセッションの一覧が表示されます。ここに新しいリモートセッションを登録して保存しておくことができるほか、保存を行わずに素早く新しいセッションを開始したり、保存したセッションを開始したりすることもできます。このほか、Remmina のグローバル設定を変更することもできます。

4.2.3 リモートセッションの追加

新しいリモートセッションを追加して保存するには、メインウインドウの左上にある を押します。すると、[リモートデスクトップの設定] というウインドウが表示されます。

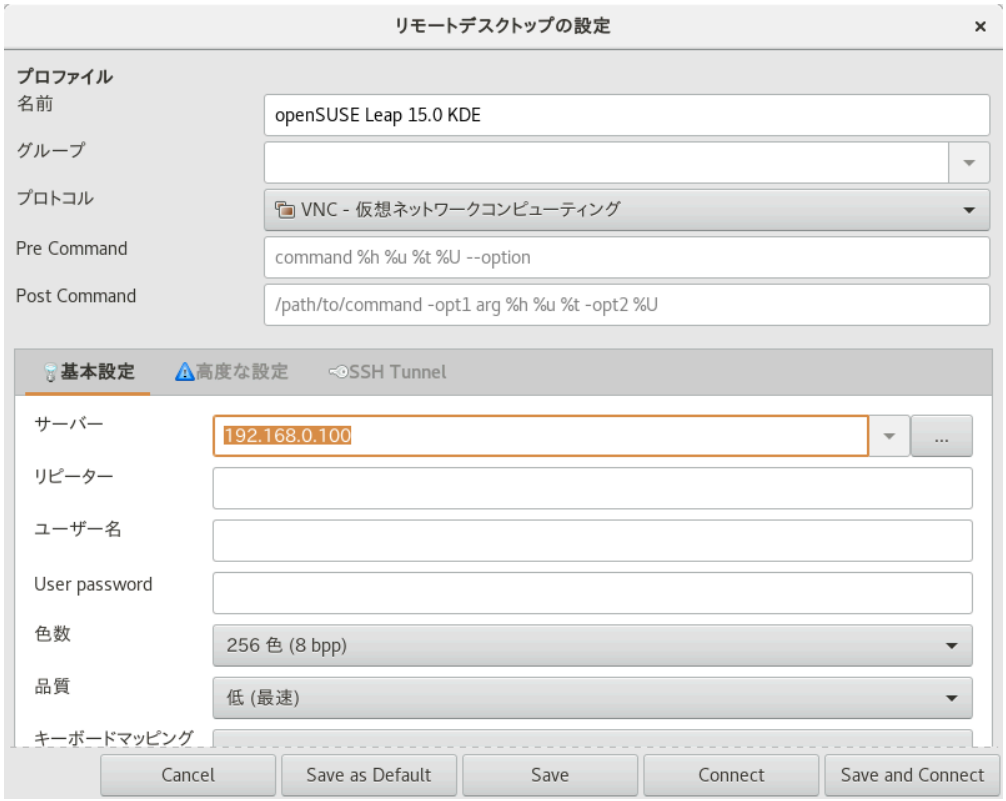


図 4.3: リモートデスクトップの設定

新しく追加するセッションプロファイルに対して、各種の情報を設定します。主なものは下記のとおりです:

名前
: プロファイルの名前を指定します。名前はメインウインドウ内の一覧に表示されます。

プロトコル
: リモートのセッションに接続する際に使用するプロトコル (例: VNC) を指定します。

サーバー
: リモートサーバの IP アドレスまたはホスト名と、ディスプレイ番号を指定します。

ユーザー名, パスワード
: リモートサーバに接続する際の認証情報を設定します。認証を行わない場合は、何も入力しないでください。

色数, 品質
: お使いの環境の接続速度と品質にあわせて、最適な設定を選択してください。

［高度な設定］タブを選択すると、より詳しいオプションを設定することができます。

ヒント: 暗号化の無効化

クライアントとリモートのサーバの間で暗号化を行わない場合は、［暗号化を無効にする］を選択してください。選択を行わないと、接続が失敗します。

SSH でトンネル接続するような場合や、認証オプションを設定する必要がある場合は、［SSH］タブを選択してください。

設定が終わったら［Save］を押します。新しいプロファイルがメインウインドウ内に表示されるようになります。

4.2.4 リモートセッションの開始

以前に保存したセッションから起動することができるほか、接続の詳細を保存せずに素早く接続することもできます。

4.2.4.1 リモートセッションを素早く開始する方法

様々な詳細設定を追加したり保存したりせずにリモートセッションに素早く接続するには、メインウインドウの上部にあるドロップダウンボックスとテキストボックスをお使いください。

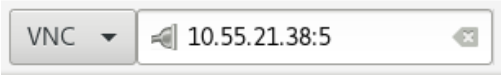


図 4.4: クイックスタート

まずはドロップダウンリストで通信プロトコルを選択 (例:「VNC」) して、その後 VNC サーバのホスト名または IP アドレスを入力してください。ディスプレイ番号を指定する場合は、ホスト名または IP アドレスに続いてコロンを入力し、番号を入力してください。入力が終わったら `Enter` で接続します。

4.2.4.2 保存済みのリモートセッションへの接続

特定のリモートセッションを開くには、セッションの一覧から項目を選んでダブルクリックします。

4.2.4.3 リモートセッションのウインドウ

リモートのセッションはウインドウ内のタブとして表示されます。それぞれのタブが 1 つのセッションに対応しています。また、ウインドウの左側にあるツールバーを利用することで、ウインドウやセッションを管理することができます。ここではフルスクリーンモードやセッションのディスプレイサイズに合わせたウインドウのサイズ変更、そしてセッションに対する特定キー入力の送信や画像品質の設定などを行うことができます。

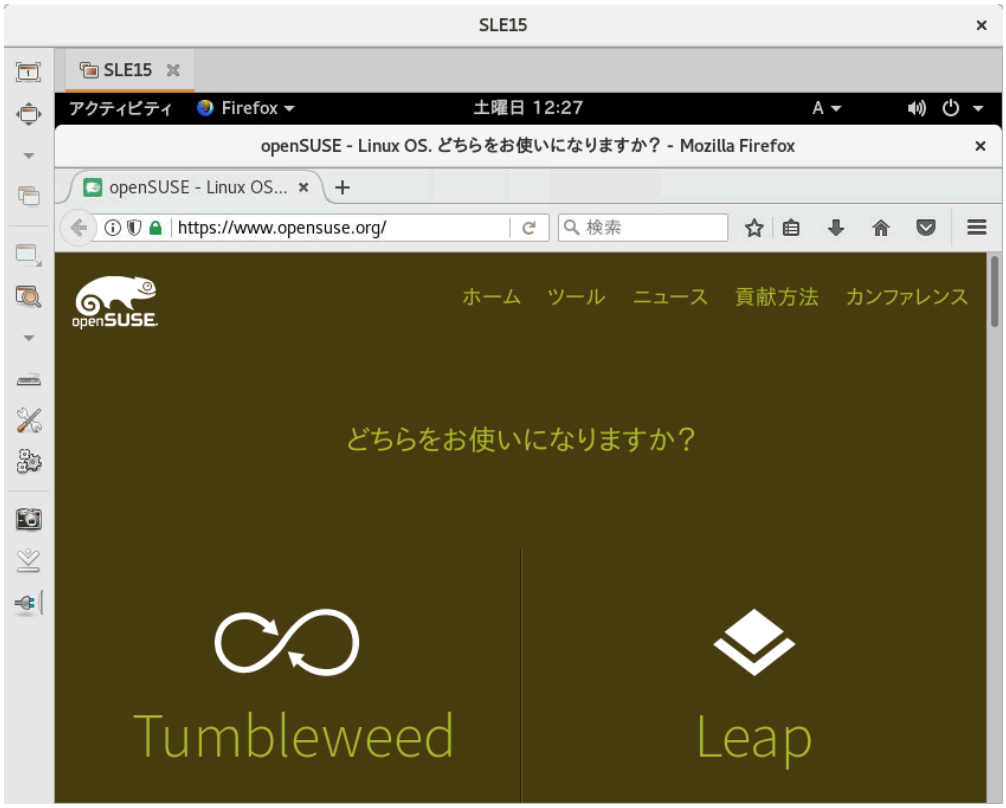


図 4.5: REMMINA によるリモートセッション接続

4.2.5 保存済みセッションの編集／コピー／削除

保存済みのリモートセッションを 編集 するには、Remmina のメインウインドウ内で名前を右クリックし、[Edit] を選択します。それぞれの項目に関する説明は、4.2.3項「リモートセッションの追加」をお読みください。

保存済みのリモートセッションを コピー するには、Remmina のメインウインドウ内で名前を右クリックし、［コピー］を選択します。［リモートデスクトップの設定］ウインドウが表示されたら、プロファイルの名前を変更してください。それ以外にも、必要に応じてオプションを変更してもかまいません。変更が終わったら［Save］を押してください。

保存済みのリモートセッションを 削除 するには、Remmina のメインウインドウ内で名前を右クリックし、［Delete］を選択します。削除確認のメッセージが表示されますので、［はい］を押して削除を行ってください。

4.2.6 コマンドラインからのリモートセッションの実行

リモートセッションをコマンドラインから開きたい場合や、バッチファイルから開く際に、メインのアプリケーションウインドウ無しで表示させたい場合は、下記のような書式で実行します:

```
> remmina -c プロファイル名.remmina
```

Remmina の設定ファイルは、ホームディレクトリ内の `.local/share/remmina/` ディレクトリ内に保存されます。設定ファイルは名前などとは無関係に付与されます。プロファイルからファイル名を調べたい場合は、Remmina を起動してからメインウインドウ内のプロファイルを選択し、ウインドウ下部のステータス行に表示されるパスからファイル名を判断してください。

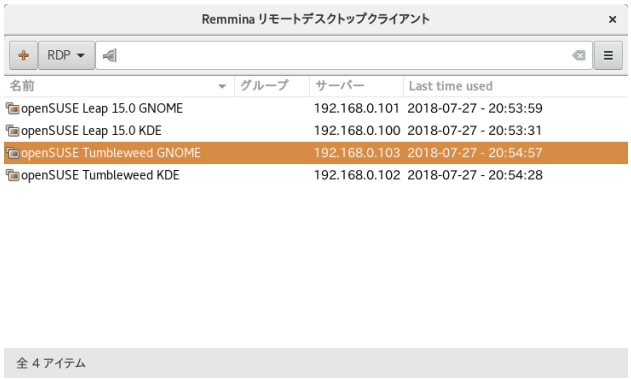


図 4.6: プロファイルのファイルパスの表示

Remmina を起動していない状態であれば、`sle15.remmina` のように、設定ファイルをよりわかりやすいファイル名に変更することができます。もちろん設定ファイルをどこか別のディレクトリにコピーして、`remmina -c` で指定して実行することもできます。

4.3 VNC サーバ側でのワンタイムセッションの設定

ワンタイムセッションはリモートのクライアントからの接続で起動する仕組みです。クライアントから接続を行うと、サーバ内のグラフィカルなログイン画面を表示し、セッションを開始することができます。また、ログインマネージャ側で対応していれば、デスクトップ環境を選択することもできます。ワンタイムセッションの接続を閉じると、その中で動作していたアプリケーションも終了します。ワンタイム VNC セッションは共有することができませんが、単一のホスト内で複数のセッションを扱うことができます。

手順 4.1: ワンタイム VNC セッションの有効化

1. [YaST] › [ネットワークサービス] › [リモート管理 (VNC)] を選択して起動します。
2. [セッション管理機能無しのリモート管理を許可する] を選択します。
3. VNC セッションを Web ブラウザからアクセスできるようにするには、[Web ブラウザを利用したアクセスを有効にする] を選択します。
4. 必要であれば、[ファイアウオールでポートを開く] を選択することもできます (たとえばネットワークインターフェイスが外部ゾーンとして設定されている場合など) 。複数のネットワークインターフェイスが設定されている場合は、[ファイアウオールの詳細] ボタンでネットワークインターフェイスを選択し、そのインターフェイスでだけ開くこともできます。
5. 設定を確認して [次へ] を押します。
6. この時点で必要なパッケージがインストールされていない場合は、それらのパッケージをインストールするよう承認を求められます。

ヒント: ディスプレイマネージャの再起動について

YaST ではディスプレイマネージャの設定を変更します。設定を反映させるには、いったんお使いのグラフィカルセッションを終了し、ディスプレイマネージャを再起動する必要があります。

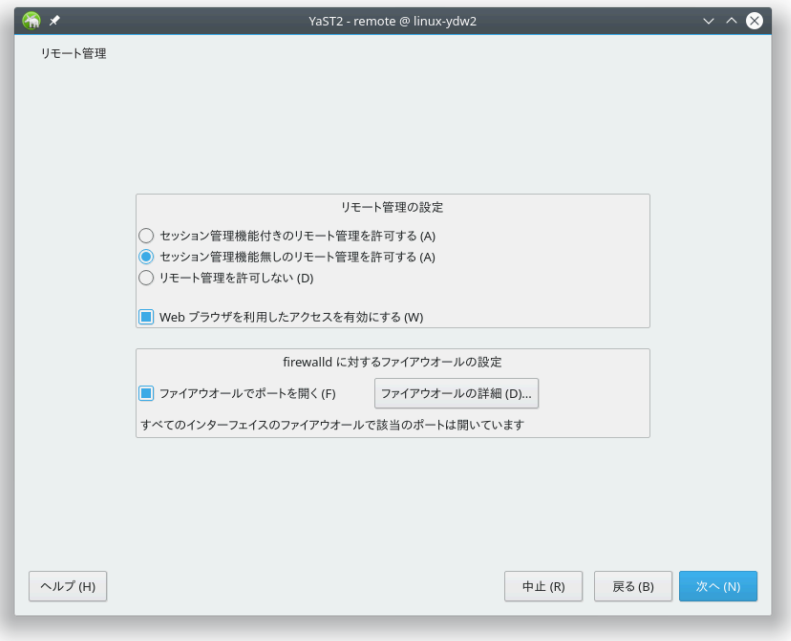


図 4.7: リモート管理

4.3.1 利用可能な設定

openSUSE Leap の既定の設定では、解像度が 1024x768 ピクセルで 16 ビットの色数になっています。ポートは「通常の」VNC ビューアからの接続が `5901` で、Web ブラウザからの接続が `5801` です。

異なるポートを使用するように設定することもできます。詳しくは 4.3.3項「ワンタイム VNC セッションの設定」をお読みください。

VNC でのディスプレイ番号と X のディスプレイ番号は、ワンタイムセッションではそれぞれ独立して管理されます。VNC のディスプレイ番号は、それぞれのサーバ設定で手作業による割り当てを行って使用します (上記の例では :1 がディスプレイ番号です) が、X のディスプレイ番号は、VNC セッションが開始された際、その時点で空いている X ディスプレイ番号を自動的に使用します。

既定では、VNC クライアントとサーバは、インストール時に生成した自己署名型の SSL 証明書を利用して、機密保持接続を確立しようとします。このままお使いいただくこともできますが、必要に応じて証明書を置き換えることもできます。ただし、自己署名型の証明書を利用する場合、VNC ビューア側や Web ブラウザ側で証明書の警告メッセージが表示されることになります。

ヒント

なお VNC クライアントによっては、既定で設定される自己署名型の証明書を利用した暗号化接続を拒否するようなものもあります。たとえば Vinagre では、GnuTLS のグローバルな証明書ストアを利用して証明書を検証しますので、自己署名型の証明書の場合には接続が失敗します。このような場合は、`x509` 以外の暗号化方式を使用するか、もしくは VNC サーバに対して何らかの証明機関から証明書を発行してもらい、その証明機関の証明書をクライアント側の証明書ストアに取り込んでください。

4.3.2 ワンタイム VNC セッションの開始

ワンタイム VNC セッションに接続するには、まず VNC ビューアをインストールしておかなければなりません。インストールに関する詳細は 4.1項「`vncviewer` クライアント」をお読みください。なお、JavaScript に対応した Web ブラウザを利用して、VNC のセッションに接続することもできます。この場合は、`http://jupiter.example.com:5801` のような URL を指定して接続してください。

4.3.3 ワンタイム VNC セッションの設定

既定値を変更する必要がなければ、この手順は実施しなくてもかまいません。

ワンタイム VNC セッションは、`systemd` のソケット `xvnc.socket` 経由で起動されます。既定では 6 つの設定ブロックが存在していて、そのうち 3 つが VNC ビューア向け (`vnc1` から `vnc3` まで) で、残りの 3 つが JavaScript クライアント向け (`vnchttpd1` から `vnchttpd3` まで) です。既定では `vnc1` と `vnchttpd1` だけが有効化されています。

VNC サーバソケットをシステムの起動時に有効化するには、下記のコマンドを実行します:

```
> sudo  systemctl enable xvnc.socket
```

即時にソケットを開始するには、下記のコマンドを実行します:

```
> sudo  systemctl start xvnc.socket
```

`Xvnc` サーバは、`server_args` オプションを介して設定することができます。オプションの一覧については `Xvnc --help` をお読みください。

独自の設定を加える場合、同じホストで動作している他の設定やサービス、既存の VNC セッションなどが使用していないポートを設定するよう注意してください。

設定の変更を反映させるには、下記のコマンドを実行します:

```
> sudo systemctl reload xvnc.socket
```

重要: ファイアウオールと VNC ポート

手順4.1「ワンタイム VNC セッションの有効化」で説明している手順でリモート管理を設定すると、ポート `5801` と `5901` を開くように設定されます。VNC セッションを提供するネットワークインターフェイスがファイアウオールで保護されている場合は、それぞれのポートを開くように設定する必要があります。詳しくは『セキュリティ強化ガイド』、第24章「マスカレードとファイアウオール」をお読みください。

4.4 永続 VNC サーバセッションの設定

永続セッションを使用すると、複数のクライアントから同時に接続できるようになります。たとえば 1 つのクライアントに対してだけ操作を許可しておき、それ以外の全てのクライアントは表示専用に設定することで、デモンストレーション目的で公開するようなこともできます。また、講師から参加者のデスクトップにアクセスできるように設定しておけば、トレーニングなどの際にも便利に使用することができます。

ヒント: 永続 VNC セッションへの接続

永続 VNC セッションに接続するには、まず VNC ビューアをインストールしておかなければなりません。インストールに関する詳細は 4.1項「`vncviewer` クライアント」をお読みください。なお、JavaScript に対応した Web ブラウザを利用して、VNC のセッションに接続することもできます。この場合は、`http://jupiter.example.com:5801` のような URL を指定して接続してください。

4.4.1 vncmanager を利用した VNC セッションの起動

手順 4.2: 永続 VNC セッションの有効化

1. [YaST] › [ネットワークサービス] › [リモート管理 (VNC)] を選択して起動します。
2. [セッション管理機能付きのリモート管理を許可する] を選択します。
3. VNC セッションを Web ブラウザからアクセスできるようにするには、[Web ブラウザを利用したアクセスを有効にする] を選択します。
4. 必要であれば、[ファイアウオールでポートを開く] を選択することもできます (たとえばネットワークインターフェイスが外部ゾーンとして設定されている場合など) 。複数のネットワークインターフェイスが設定されている場合は、[ファイアウオールの詳細] ボタンでネットワークインターフェイスを選択し、そのインターフェイスでだけ開くこともできます。

5. 設定を確認して［次へ］を押します。

6. この時点で必要なパッケージがインストールされていない場合は、それらのパッケージをインストールするよう承認を求められます。

ヒント: ディスプレイマネージャの再起動について

YaST ではディスプレイマネージャの設定を変更します。設定を反映させるには、いったんお使いのグラフィカルセッションを終了し、ディスプレイマネージャを再起動する必要があります。

4.4.1.1 永続 VNC セッションの設定

手順4.2「永続 VNC セッションの有効化」の手順に従って VNC セッションを有効化したあとは、`vncviewer` や Remmina などの VNC ビューアを利用して、リモートのセッションに接続できるようになります。ログイン後はデスクトップ環境のシステムトレイ内に、「VNC」というアイコンが表示されるようになります。アイコンを押すと［VNC セッション］ウインドウを表示することができます。お使いのデスクトップ環境がシステムトレイアイコンに対応していない場合は、`vncmanager-controller` コマンドをお使いのうえ、手作業で設定を行ってください。

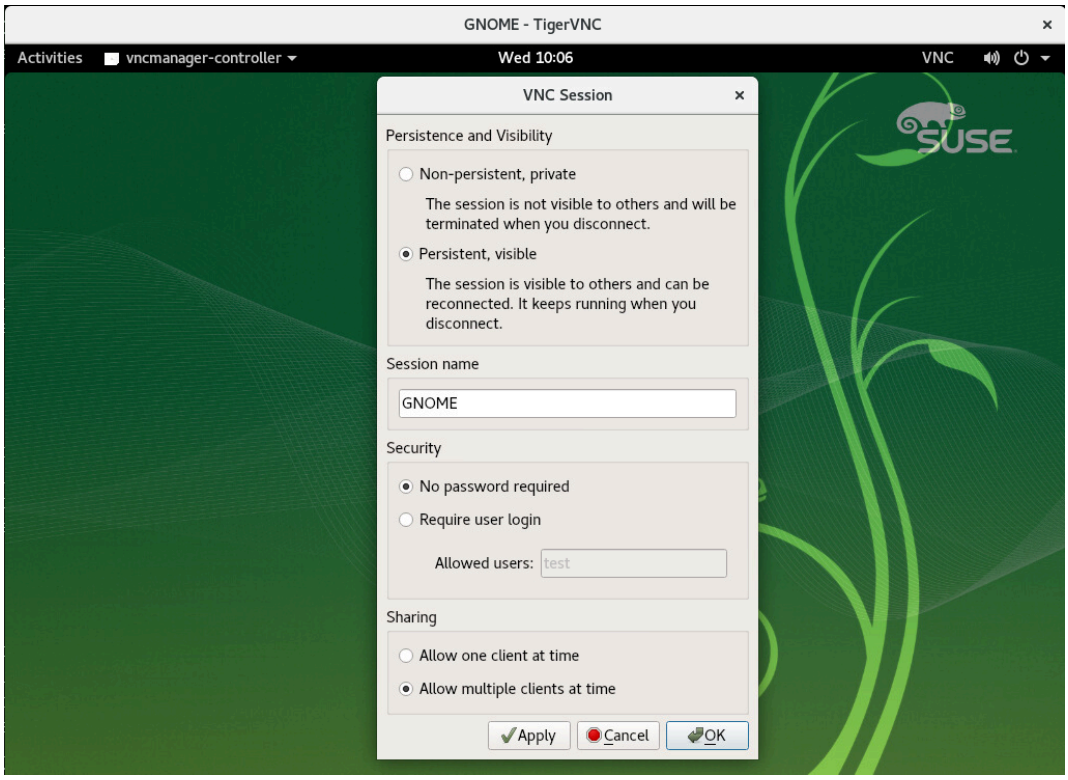


図 4.8: VNC セッションの設定

VNC セッションの動作を変えるには、それぞれ下記の設定を変更します:

［Non-persistent, private］
: これを選択すると、ワンタイムセッションと同じ動作になります。セッションは他のユーザからは見えなくなりますが、切断するとセッションは終了してしまいます。詳しくは 4.3項「VNC サーバ側でのワンタイムセッションの設定」をお読みください。

［Persistent, visible］
: これを選択すると、セッションは他のユーザから見えるようになり、切断してもセッションが終了しなくなります。

［Session name］
: ここには永続セッションの名前を指定します。これにより、再接続時の識別が容易になります。

［No password required］
: ユーザの認証情報の入力を行うことなく、セッションに対して自由にアクセスできるようにします。

［Require user login］
: セッションにアクセスするために、正しいユーザ名とパスワードでのログインを求めるようにします。許可するユーザの一覧は、［Allowed users］で指定します。

［Allow one client at a time］
: 複数のユーザが同時に接続することがないようにします。

［Allow multiple clients at a time］
: 永続セッションに対して、複数のユーザが同時に接続できるようにします。これはプレゼンテーションやトレーニングの際に便利な仕組みです。

［OK］を押して閉じます。

4.4.1.2 永続 VNC セッションへの参加

4.4.1.1項「永続 VNC セッションの設定」の手順で永続 VNC セッションの設定を行ったら、あとは VNC ビューアを利用して接続することで、セッションに参加できるようになります。VNC クライアントがサーバに接続すると、新しいセッションを作成するか、既存のセッションに参加するかを選択することができます:

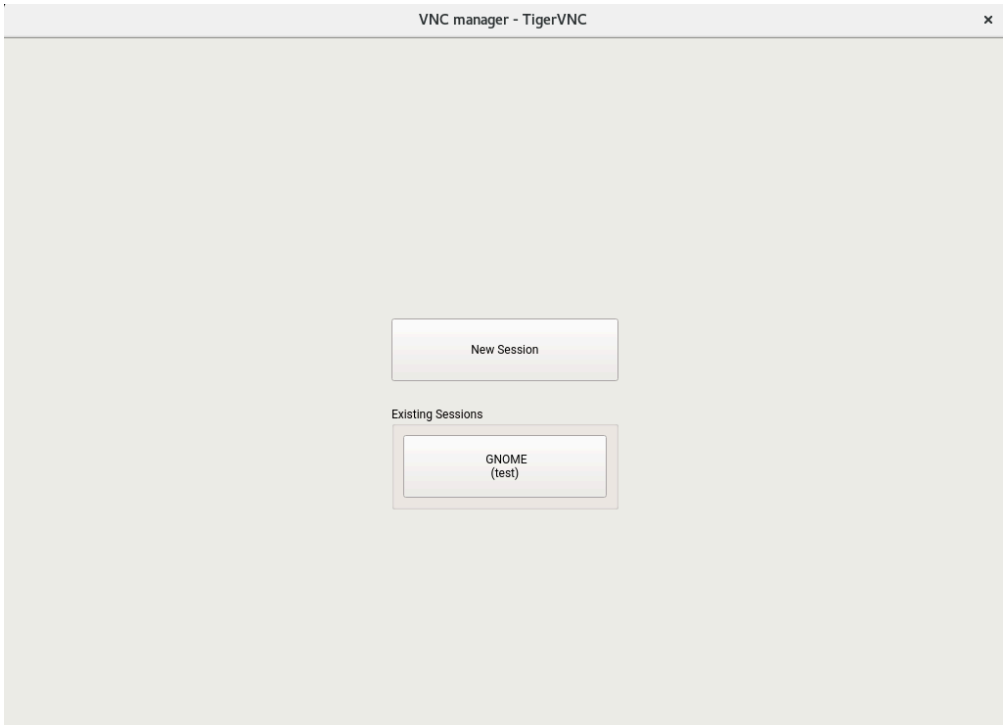


図 4.9: 永続 VNC セッションへの参加

既存のセッションの名前を選択すると、永続セッションの設定によってはログインを求められることがあります。

4.5 VNC サーバ側での暗号化の設定

VNC サーバを正しく設定すると、VNC サーバとクライアントとの間が暗号化されるようになります。認証はセッションの冒頭で行われ、実際のデータ転送はそれ以降に行われます。

ワンタイム VNC セッションや永続 VNC セッションでは、`server_args` 行で設定する `/usr/bin/Xvnc` コマンドの `-securitytypes` を指定することができます。`-securitytypes` オプションでは認証方法と暗号化の両方を選択します。下記のいずれかを設定することができます:

認証

None, TLSNone, X509None
: 認証を行いません。

VncAuth, TLSVnc, X509Vnc
: 独自のパスワードを利用して認証します。

Plain, TLSPlain, X509Plain
: ユーザのパスワードを確認するため、PAM を利用して認証します。

暗号化

None, VncAuth, Plain
: 暗号化を行いません。

TLSNone, TLSVnc, TLSPlain
: 匿名 TLS 暗号を使用します。全ての通信内容が暗号化されますが、リモートホスト側の検証は何も行いません。ネットワーク上は暗号化されるため、盗聴の危険からは保護されますが、中間者攻撃 (man-in-the-middle attack) には無力です。

X509None, X509Vnc, X509Plain
: 証明書付きの TLS 暗号を使用します。自己署名型の証明書を使用している場合、最初の接続でユーザの確認を求められます。それ以降の接続では、証明書が変更された場合にのみ警告されるようになります。これにより盗聴の危険だけでなく、初回の接続以降は中間者攻撃 (man-in-the-middle attack) にも耐えられるようになります (SSH における鍵の確認と同様の方法です) 。また、公的な証明機関が発行し、マシン名が一致した証明書を使用している場合は、完全なセキュリティを実現できるようになります (一般的な HTTPS 暗号化接続と同じです) 。

ヒント

なお VNC クライアントによっては、既定で設定される自己署名型の証明書を利用した暗号化接続を拒否するようなものもあります。たとえば Vinagre では、GnuTLS のグローバルな証明書ストアを利用して証明書を検証しますので、自己署名型の証明書の場合には接続が失敗します。このような場合は、`x509` 以外の暗号化方式を使用するか、もしくは VNC サーバに対して何らかの証明機関から証明書を発行してもらい、その証明機関の証明書をクライアント側の証明書ストアに取り込んでください。

ヒント: 証明書とその鍵へのパス

X509 ベースの暗号化を使用する際は、`-X509Cert` と `-X509Key` のオプションで証明書と鍵を指定する必要があります。

複数のセキュリティタイプを指定する場合は、カンマで区切ってください。サーバとクライアントとの間で、最初に対応し許可された仕組みを使用します。この方法により、サーバ側で暗号化の優先順位を付けることができるようになります。また、この方法では暗号化に対応していないクライアントを受け付けなければならないような場合にも、有用です。

クライアント側でも同様にセキュリティタイプを指定することができます。これにより、暗号化に対応しているはずのサーバに接続する際、誤って暗号化せずに接続するような問題を防ぐことができます (ただし、この設定を行わなくても、vncviewer であれば "Connection not encrypted!" (接続が暗号化されていません) と表示されるので、問題に気がつくことはできます) 。

4.6 Wayland との互換性について

VNC によるリモート管理機能は X11 に依存して作られているため、Wayland を有効化していると画面が表示されない問題が発生することがあります。そのため、リモート管理機能を使用する場合は、Wayland ではなく X11 を使用するように設定してください。たとえば `gdm` の場合、`/etc/gdm/custom.conf` ファイルにある `[daemon]` セクション内に `WaylandEnable=false` を追加してください。また、ログイン時には X11 互換のセッションを選択する必要があることにも注意してください。なお、GNOME で Wayland セッションを使用できないようにしたい場合は、`gnome-session-wayland` パッケージを削除してロック (施錠) してください。

5 ［熟練者向けパーティション設定］

改訂履歴

2026-02-11

システムの設定を洗練させていくには、ディスクのパーティション設定も洗練させる必要があります。パーティション関係の全ての設定はインストール時に行うことができます。

また、ブロックデバイスに対して永続的な名前を必要とする場合は、`/dev/disk/by-id` または `/dev/disk/by-uuid` 内のブロックデバイスをお使いください。

論理ボリューム管理 (LVM; Logical Volume Management) はディスクのパーティション方式のうちの 1 つで、一般的に使用していた物理パーティションよりもずっと柔軟な使用形態を提供することができます。このほか、スナップショットの機能はデータのバックアップにも使用することができますし、いわゆる RAID (Redundant Array of Independent Disks; 独立した複数のディスクによる冗長性アレイ) の仕組みを利用して、データの一貫性や性能を確保したり、耐障害性の仕組みを取り入れたりすることができます。また openSUSE Leap では、マルチパス I/O にも対応しています。このほか、iSCSI を利用してネットワーク経由のディスクを使用する構成にも対応しています 。

警告: ディスク領域の単位について

パーティション設定を行う際、ディスク領域は一般的な 10 進接頭辞 (1 (キロ) = 1000) ではなく、2 進接頭辞 (1 (キロ) = 1024) を使用することに注意してください。たとえばサイズの指定で `1GB` , `1GiB` , `1G` のいずれかを入力した場合、1 GB (ギガバイト) ではなく 1 GiB (ギビバイト) として扱われます。

2 進接頭辞
: 1 GiB = 1073741824 バイトを表します。

10 進接頭辞
: 1 GB = 1000000000 バイトを表します。

差
: 1 GiB ≈ 1.07 GB になります。

5.1 ［熟練者向けパーティション設定］の使用

［熟練者向けパーティション設定］(図5.1「YaST パーティション設定」) を使用することで、パーティションの追加や削除／変更のほか、ソフトウエア RAID や LVM などの設定を行うことができます。

警告: 動作中のシステムに対するパーティション設定の変更について

動作中でもパーティションの変更を行うことはできますが、誤った操作をしてしまうと、データを失ってしまうリスクが非常に高くなります。インストール済みのシステムに対するパーティション設定の変更はできる限り避けるものとし、どうしても必要な場合は、事前に完全なバックアップを採取してから実施してください。

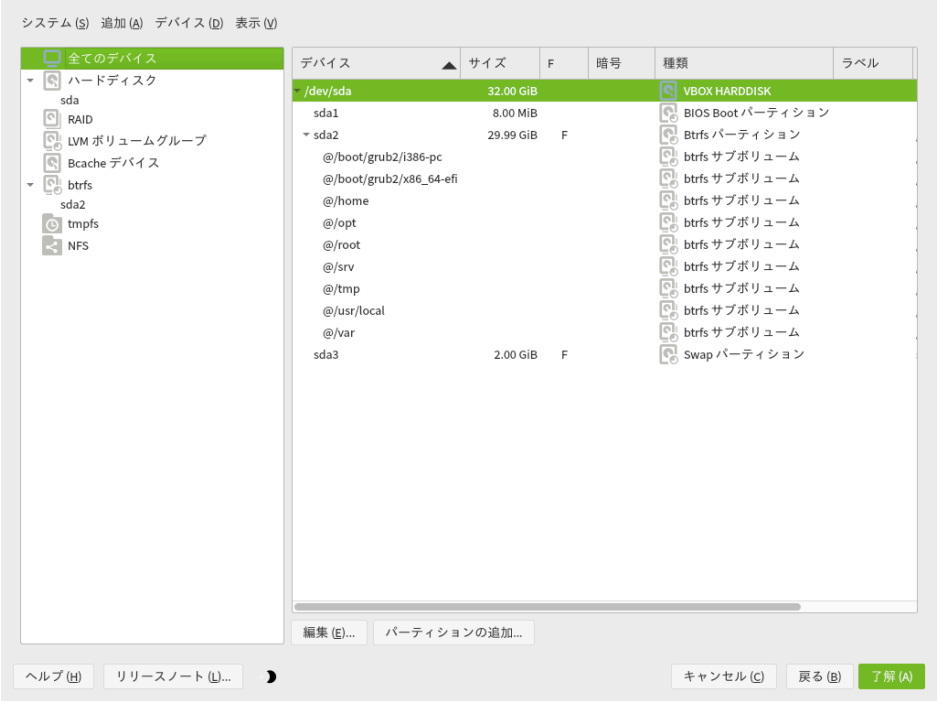


図 5.1: YAST パーティション設定

YaST の［熟練者向けパーティション設定］のダイアログ内の［利用可能なストレージ］には、接続されている全てのハードディスクに対して、既存のパーティションや提案するパーティション設定が表示されます。ハードディスク全体は `/dev/sda` のように番号無しで表示されていて、パーティションは `/dev/sda1` のように、それらのデバイス名に数字を足したもので表されています。それぞれのハードディスクやパーティションのサイズ、種類、暗号化状態、ファイルシステム、マウントポイントもそれぞれ表示されています。マウントポイントとは、Linux のファイルシステムツリー内のどこにパーティションを配置するのかを決める情報です。

左側の［システムビュー］には、いくつかの機能ビューが用意されています。これらのビューは既存のストレージ設定に関する情報を収集したり、各種の機能 (`RAID`, `ボリューム管理`, `暗号化ファイル` など) を設定したりするために使用することができます。また、btrfs や NFS, `TMPFS` などの特殊な機能を持つファイルシステムを表示することができます。

インストール時に熟練者向けパーティション設定を起動している場合は、空きのハードディスク領域が一覧表示され、それらが自動選択されます。openSUSE Leap に対して追加のディスク領域を割り当てるには、パーティションの一覧表示の下にあるボタンを利用して、割り当てを行ってください。

5.1.1 パーティションテーブル

openSUSE Leap では様々な パーティションテーブル を使用したり作成したりすることができます。パーティションテーブルは ディスクラベル と呼ばれることもあり、お使いのコンピュータを起動する際に重要な設定となります。新しく作成したパーティションテーブルから、お使いのマシンを起動するには、まずパーティションテーブルの形式がお使いのファームウエア側で対応しているかどうかをご確認ください。

パーティションテーブルを変更するには、[システムビュー] で対象のディスクを選択して、[熟練者向け機能] › [新しいパーティションテーブルの作成] を選択します。

5.1.1.1 マスターブートレコード

マスターブートレコード (MBR) とは IBM PC で使用されている古い形式のパーティションテーブルです。場合によっては MS-DOS パーティションテーブルと呼ぶ場合もあります。MBR には 4 個のプライマリパーティションだけを作成することができます。お使いのディスクに既に MBR が設定されている場合、openSUSE Leap では追加のパーティションを作成してインストール先として使用できるようにします。

この 4 個のパーティションの制限は、拡張パーティション という仕組みを利用することで解決することができます。拡張パーティションそれ自身はプライマリパーティションで、その中に 論理パーティション と呼ばれる複数のパーティションを設定して利用します。

なお、UEFI ファームウエアをお使いの場合でも、一般的にレガシーモードを利用すれば、MBR から起動することができます。

5.1.1.2 GPT パーティションテーブル

UEFI コンピュータの場合、GUID パーティションテーブル (GPT) を既定で使用します。openSUSE Leap では、まだ何もパーティションテーブルが作成されていない場合、GPT を作成しようとします。

なお、古い BIOS ファームウエアをお使いの場合、GPT パーティションから起動することはできません。

下記のいずれかの機能を使用する場合、GPT パーティションテーブルが必要となります:

- 4 個以上のプライマリパーティションを使用したい場合
- UEFI の Secure Boot を使用したい場合
- 2 TB 以上のディスクを使用する場合

注記: Parted 3.1 もしくはそれ以前におけるラベル設定の誤りについて

parted 3.1 もしくはそれ以前のバージョンで作成した GPT パーティションでは、Linux 固有の GPT GUID ではなく、Microsoft Basic Data パーティションのタイプが設定されてしまいます。新しいバージョンの parted では、これらのパーティションに対して、`msftdata` というフラグを設定してしまいます。そのため、他のパーティションツールを使用すると、これらのパーティションが Windows のパーティションであるものとして扱われてしまうことがあります。

フラグを削除したい場合は、下記のように実行します:

```
# parted デバイス名 set パーティション番号 msftdata off
```

5.1.2 パーティション

YaST パーティション設定では、いくつかのファイルシステムを作成してフォーマットすることができます。openSUSE Leap での既定のファイルシステムは `Btrfs` です。詳しくは 5.1.2.2項「btrfs パーティションの設定」をお読みください。

また、よく使用される他のファイルシステムにも対応しています: `Ext2`, `Ext3`, `Ext4`, `FAT`, `XFS`, `Swap`, `UDF`

5.1.2.1 パーティションの作成

パーティションを作成するには、［ハードディスク］を選んで空き領域のあるハードディスクを選択したあと、下記のようにして行います:

1. ［パーティションの追加］を押して新しいパーティションを作成します。MBR を使用している場合は、プライマリパーティションと拡張パーティションのどちらを作成するのかを尋ねられます。拡張パーティションテーブルを作成すると、その内部にさらにパーティションを作成できるようになります。詳しくは 5.1.1項「パーティションテーブル」をお読みください。

2. 新しいパーティションに対して設定するサイズを指定します。空き領域全てを割り当てることもできますし、サイズを指定することもできます。

3. 次に使用するファイルシステムとマウントポイントを設定します。YaST では各パーティションを作成する際、マウントポイントを自動的に判別して提案します。また、ラベルでのマウントなど、マウント方法を変更したい場合は、［fstab オプション］を押してください。

4. 必要であれば、追加のファイルシステムオプションを指定することもできます。たとえば永続性のあるデバイス名が必要な場合などが挙げられます。利用可能なオプションについて、詳しくは 5.1.3項「パーティションの編集」をお読みください。

5. 最後に［完了］を押すと、パーティション設定を適用してパーティション設定モジュールを終了することができます。
   インストール時にパーティションを作成している場合は、インストール概要の画面に戻ることができます。

5.1.2.2 btrfs パーティションの設定

ルートパーティションに対する既定のファイルシステムは btrfs になっています。詳しくは をお読みください。ルートファイルシステムは既定のサブボリュームになりますので、作成されるサブボリュームの一覧内には現れません。また、既定のサブボリュームであることから、通常のファイルシステムとしてマウントできるようになります。

重要: 暗号化されたルートパーティション内での btrfs について

既定のパーティション設定では、ルートパーティションを btrfs に設定し、`/boot` がディレクトリとなるように設定されます。ルートパーティションを暗号化する場合、既定の MSDOS パーティションテーブルではなく、GPT パーティションテーブルを使用することをご確認ください。そうでないと、GRUB2 ブートローダが第 2 ステージのブートローダを書き込む際、十分な領域を確保できなくなってしまう場合があります。

btrfs のサブボリュームに対しては、スナップショットを採取することができます。スナップショットはシステムイベントをベースにした自動採取のほか、必要に応じて手作業で採取することもできます。ファイルシステムに変更を加える例としては、`zypper` は `snapper` コマンドを呼び出して、変更の前後にスナップショットを採取する機能があります。このスナップショットは、`zypper` が変更した内容では何か問題があって、元の状態に戻したいような場合に便利な仕組みです。`zypper` から呼び出される

`snapper` では、既定では ルート ファイルシステムのスナップショットを採取しますが、特定のディレクトリをスナップショットから除外することもできます。このような仕組みを実現するため、YaST では下記のサブボリュームを作成します:

`/boot/grub2/i386-pc` , `/boot/grub2/x86_64-efi` , `/boot/grub2/powerpc-ieee1275` , `/boot/grub2/s390x-emu`

ブートローダ設定の巻き戻しには対応していません。また、上記のディレクトリはいずれもアーキテクチャ固有のものであり、前半の 2 つは AMD64/Intel 64 マシンで使用されるディレクトリ、後半の 2 つは IBM POWER や IBM Z で使用されるディレクトリです。

`/home`

`/home` が個別のディレクトリ内に存在していない場合、ロールバックによってデータが失われてしまうことを防ぐため、除外を設定しています。

`/opt`

サードパーティ製の製品は `/opt` 以下にインストールされるのが一般的です。そのため、ロールバックによってこれらのアプリケーションが削除されてしまうことのないよう、除外を設定しています。

`/srv`

Web サーバや FTP サーバのデータを含むディレクトリです。ロールバックによってそれらのデータが失われてしまうことの無いよう、除外を設定しています。

`/tmp`

いずれも一時的な (テンポラリ) ファイルとキャッシュが保存されるディレクトリであるため、除外を設定しています。

`/usr/local`

このディレクトリは手作業でソフトウエアをインストールした場合に使用するディレクトリです。ロールバックによって、これらのソフトウエアが消えてしまったりしないようにするため、除外を設定しています。

`/var`

ログファイルや一時的なキャッシュが含まれるほか、サードパーティ製の製品が `/var/opt` 以下にインストールされることがあります。また、仮想マシンのイメージやデータベースを配置する既定のディレクトリでもあります。そのため、これらのデータをスナップショットから除外するためにサブボリュームを作成し、かつコピーオンライト機能を無効化しています。

ヒント: btrfs のパーティションサイズについて

スナップショットの保存には追加のディスク容量が必要となります。そのため、btrfs を作成する際は、十分な容量を設定することをお勧めします。スナップショット機能を有効化して既定の形態でサブボリュームを作成する場合、ルートファイルシステムに対する btrfs の最小サイズは 16 GB ですが、SUSE では 32 GB 以上を推奨しています。`/home` を別途のパーティション内に配置しない場合は、それよりもさらに大きな容量を設定してください。

5.1.2.3 YaST を利用した btrfs のサブボリューム管理

btrfs パーティションのサブボリュームは、YaST［熟練者向けパーティション設定］で管理できるようになりました。ここではサブボリュームの追加や削除を行うことができます。

手順 5.1: YAST を利用した BTRFS サブボリュームの管理

1. 左側のペイン内で［btrfs］を選択します。
2. 管理したいサブボリュームのある btrfs パーティションを選びます。
3. サブボリュームに対して実施したい内容 (編集／追加／削除) によって、それぞれ下記を実施します:
    a. サブボリュームを編集するには、対象のサブボリュームを選んで［編集］を押します。ここではボリュームに対する `copy-on-write` 機能の有効化／無効化 (［noCoW］のチェックボックス) のほか、サイズ制限などを設定することができます。設定が完了したら［了解］を押してください。
    b. 新しいサブボリュームを追加するには、［サブボリュームの追加］を押してパスを入力します。必要であれば、ボリュームに対する `copy-on-write` 機能の有効化／無効化 (［noCoW］のチェックボックス) のほか、サイズ制限などを設定することができます。設定が完了したら［了解］を押してください。
    c. サブボリュームを削除するには、対象のサブボリュームを選んで［削除］を押します。確認メッセージが表示されたら、［はい］を押してください。
    d.

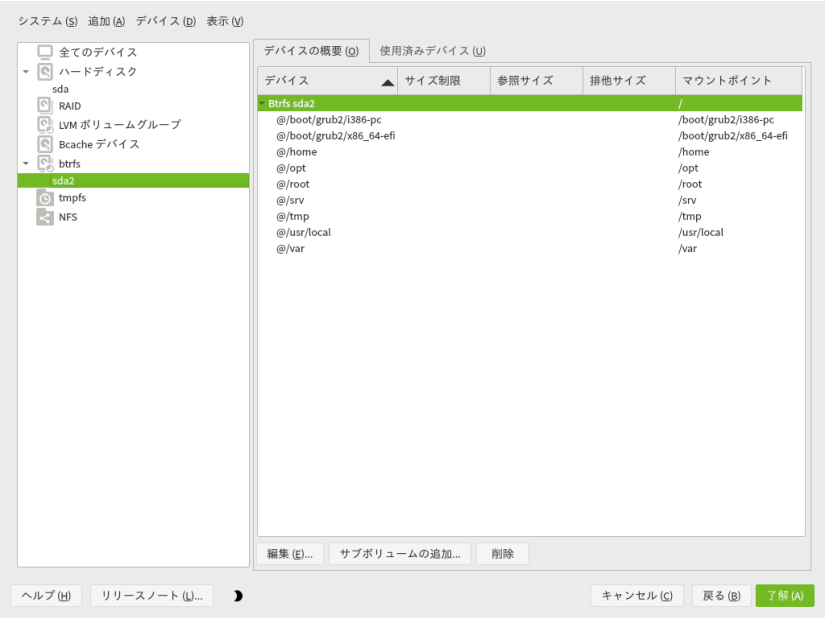


図 5.2: YAST パーティション設定における BTRFS サブボリューム

4. これで手順は完了です。

5.1.3 パーティションの編集

新しいパーティションを作成したり、既存のパーティションを編集したりする場合、様々なパラメータを設定することができます。新しいパーティションの場合、ほとんどは YaST が設定する既定のパラメータで十分であり、それ以上の変更は必要ありません。パーティションの設定を手作業で編集するには、下記の手順で行います:

1. パーティションを選択します。

2. ［編集］を押してパーティションの編集を開始し、必要なパラメータを設定します:

   ファイルシステム ID

   この時点ではパーティションをフォーマットしたくない場合、パーティションが正しく登録されるよう、ファイルシステムの ID を設定しておくことをお勧めします。一般的には［Linux］,［Linux swap］,［Linux LVM］, and［Linux RAID］のいずれかを指定します。

   ファイルシステム

   パーティションに対するファイルシステムを変更するには、［デバイスをフォーマットする］を選択して、［ファイルシステム］内の一覧からファイルシステムを選択します。

openSUSE Leap ではいくつかの種類のファイルシステムに対応しています。btrfs は他のファイルシステムに比べて高度な機能を持つことから、ルートパーティションに対してお勧めです。コピーオンライトの機能を持つほか、スナップショットやマルチデバイスへの分散、サブボリュームなど、さまざまな便利な技術が提供されています。また、XFS, ext3, ext4 はいずれもジャーナル機能のあるファイルシステムです。これらのファイルシステムは、書き込み時に操作の履歴 (ジャーナル) を持つことから、システムがクラッシュしても素早く回復できる特長を持ちます。ext2 はジャーナル機能のないファイルシステムですが、管理領域を小さくすることができるため、小さいパーティションにお勧めです。
ルートパーティションに対する既定のファイルシステムは btrfs です。また、その他のパーティションに対する規定のファイルシステムは xfs です。
UDF ファイルシステムは書き込み可能な光学メディアや読み込み専用の光学メディアのほか、USB メモリ やハードディスクでも使用することができます。また、複数のオペレーティングシステムに対応しています。
swap (スワップ) は仮想メモリとして使用することのできる特殊なファイルシステムです。スワップパーティションは、少なくとも 256 MB 以上で作成しておくことをお勧めします。ただし、スワップ領域が不足した場合は、スワップ領域の追加ではなくメモリの追加をお勧めします。

警告: ファイルシステムの変更時の注意

ファイルシステムを変更してパーティションを再フォーマットすると、そのパーティション内にあるデータは全て削除され、復元できなくなります。

暗号化デバイス

暗号化を有効化すると、ハードディスクに書き込まれる全てのデータが暗号化されるようになります。これにより機密データの安全性が高まりますが、暗号化の処理にはそれなりの時間を要するため、システムの速度は落ちることになります。ファイルシステムの暗号化について、詳しくは 5.2項「デバイスの暗号化」および『セキュリティ強化ガイド』、第13章「パーティションやファイルの暗号化」をお読みください。

マウントポイント

このパーティションを、ファイルシステムのツリー内のどの位置に配置するのかを指定します。YaST の提案する値の中から選択するか、もしくは任意のパスを指定してください。

fstab オプション

グローバルなファイルシステム管理ファイル (`/etc/fstab`) 内には、様々なパラメータが含まれています。ほとんどの場合において、既定値のまま使用すれば十分です。ただし、ファイルシステムの識別をデバイス名ではなくボリュームラベルで行うなど、いくつかの設定を変更することができます。なお、ボリュームラベルには `/` とスペースを除く、任意の文字を指定することができます。

永続性のあるデバイス名を使用するには、［デバイス ID］,［UUID］,［LABEL］のいずれかを選択してください。openSUSE Leap では、永続性のあるデバイス名を既定で使用します。

パーティションをラベルでマウントしたい場合は、［ボリュームラベル］のテキスト項目に名前を入力してください。たとえば `/home` 向けのパーティションであれば、`HOME` のようなボリュームラベルを指定します。

ファイルシステム内でクォータ (容量制限) を行いたい場合は、［クォータサポートを有効にする］を選択してください。この設定は、YaST の［ユーザ管理］モジュールでユーザ向けのクォータを設定する際、事前に設定しておかなければならない項目です。クォータの設定方法について、詳しくは**『スタートアップ』、第5章「YaST を利用したユーザ管理」、5.3.3項「クォータの管理」**をお読みください。

3. ［完了］を押すと設定を保存することができます。

注記: ファイルシステムのサイズ変更

既存のファイルシステムのサイズを変更するには、対象のパーティションを選択して［サイズ変更］を押します。ただし、パーティションのサイズ変更はマウントされている状態では行うことができません。サイズ変更を行うには、パーティション設定を実行する前にマウント解除を行ってください。

5.1.4 熟練者向けオプション

［システムビュー］ペイン内でハードディスク (例: ［sda］) を選択すると、［熟練者向けパーティション設定］ウインドウの右下に［熟練者向け機能］が表示されます。このボタンには、下記のコマンドが含まれています:

新しいパーティションテーブルの作成

このオプションを選択することで、選択したデバイスに新しいパーティションテーブルを作成することができます。

警告: 新しいパーティションテーブルの作成

新しいパーティションテーブルを作成すると、そのデバイス内にある全てのデータと全てのパーティションが削除され、復元できなくなります。

このディスクを複製する

このオプションを選択すると、デバイスのパーティションレイアウトを他のディスクデバイスに複製することができます (ただしデータは複製されません) 。

5.1.5 [tmpfs]

tmpfs は RAM ベースの擬似ファイルシステムで、非常に高速なファイルアクセスと、透過型の HugePage サポート (THP) に対応しています。なお、名前のとおり本ファイルシステムは一時的 (temporary) なものであり、保存されたファイルはハードディスク等に書き込まれることはなく、カーネル内部のキャッシュにのみ保存されます。そのため、このファイルシステム内に書き込まれたファイルは、マウント解除時に全て削除されます。

tmpfs ファイルシステムを追加するには［tmpfs の追加…］を押し、マウントポイントとその他のオプションを設定します。

5.1.6 [Bcache デバイス]

bcache は Linux カーネルのブロックレイヤに対するキャッシュ機構です。SSD のような高速なドライブを 1 台以上設定して、これをハードディスクに対するキャッシュとして動作させます。bcache はライトスルーとライトバックの両方に対応しているほか、使用しているファイルシステムとは独立して動作するため、任意のファイルシステムで動作させることができます。なお既定では、SSD が得意とするランダムリードとランダムライトに対してのみキャッシュ処理を行います。これはデスクトップやサーバだけでなく、ハイエンドのストレージシステムでも有用です。

bcache を設定するには、まず［バッキングデバイス］(遅いほうのドライブ、つまりハードディスク) と［キャッシュデバイス］(速いほうのドライブ、通常は SSD) をそれぞれ指定したあと、［キャッシュモード］を指定します。あとは［了解］を押して設定を保存するだけです。

5.1.7 高度なオプション

［システムビュー］ペイン内でコンピュータ名の書かれた箇所を選択すると、［熟練者向けパーティション設定］ウインドウ内の右下に［設定］というボタンが表示されます。このメニューには、下記のコマンドが含まれています:

iSCSI の設定

SCSI over IP のブロックデバイスにアクセスするには、まず iSCSI の設定を行う必要があります。これを実行することにより、パーティションの一覧内に新しいデバイスを表示することができるようになります。

マルチパスの設定

このオプションを選択することで、対応するマスストレージデバイスに対してマルチパス機能拡張を設定することができるようになります。

5.1.8 パーティション設定のヒント

下記の章には、お使いのシステムを設定する際に正しい判断を下すための、パーティション関連のヒントが記載されています。

5.1.8.1 シリンダ番号

パーティションツールによっては、シリンダ数が `0` からはじまるものと `1` からはじまるものがあります。シリンダ数を計算する場合は、始めと終わりのシリンダ番号を確認し、+1 するようにしてください。

5.1.8.2 スワップ の使用について

スワップは利用可能な物理メモリを拡張するための仕組みです。これにより、物理メモリよりも多くのメモリを使用することができるようになります。カーネルのバージョン 2.4.10 以前では、安全対策としてスワップを用意しておくのが通常で、一般的には物理メモリサイズの 2 倍程度を確保しておくものでした。ただし、現在はそのような制限はありません。

Linux では「Least Recently Used」(LRU) という仕組みを利用して、メモリからディスクに移動すべきページを判断します。そのため、実行中のアプリケーションに対してはより多くのメモリが割り当てられることになり、キャッシュもより効率的に動作するようになっています。

もしもアプリケーションが利用可能なメモリを目一杯使用しようとした場合、スワップまわりの問題が発生することがあります。この場合、設定に応じて下記のような動作になります:

スワップを設定していないシステムの場合
: アプリケーションは最大のメモリを確保します。全てのキャッシュメモリは解放されることになるため、他のアプリケーションの動作は遅くなります。しばらくすると、カーネルの Out-of-Memory Kill (メモリ枯渇解決) 機構が動作して、プロセスを強制終了 (kill) し始めます。

中程度のサイズ (128 MB から 512 MB 程度) のスワップがあるシステムの場合
: 初めのうちはシステムはスワップが無い場合と同様に遅くなります。物理メモリが全て割り当てられると、スワップ領域を使い始めます。この時点では、システムの動作は非常に遅くなり、リモートからコマンドを受け付けることもできないような状態になります。あとはスワップ領域を受け持っているハードディスクの速度によりますが、システムは Out-of-Memory Kill (メモリ枯渇解決) 機構が問題を解決するまで、10 から 15 分程度そのままの状態になります。なお、「ディスクへのサスペンド」 (休止状態) を使用するコンピュータの場合は、ある程度のスワップ領域が必要です。この場合は、物理メモリのデータをスワップ領域に待避するための、十分なサイズが必要です (512 MB から 1GB 程度必要になります) 。

大きなサイズ (1 GB 以上) のスワップがあるシステムの場合
: 制御ができず、スワップ領域を過剰に使用するようなアプリケーションは、使用しないのが最適です。ただし、そのようなアプリケーションを使用した場合、システムは回復するのに数時間程度を要してしまいます。プロセス内では、他のプロセスがタイムアウトや失敗になることも多く、それらのプロセスを終わらせたとしても、システムは不安定な状態になります。この場合は、物理的なリセットボタンでリセットを行い、回復を行うのが一般的です。大容量のスワップ領域は、この機能に依存するような特殊なアプリケーションを使用する場合に限って設定すべきものです。このようなアプリケーション (データベースやグラフィック編集プログラムなど) の場合は、独自のディスク待避機構を持つ場合がありますので、スワップ領域の拡大ではなく、それらの仕組みを使用するようにするのがお勧めです。

お使いのシステムが制御不可能になっていない場合で、スワップ領域を追加する必要がある場合は、スワップ領域をその場で追加することができます。スワップ領域用のパーティションを既に確保してある場合は、YaST で追加することができます。確保していない場合は、スワップ領域をファイルとして作成することもできます。スワップファイルは一般的にパーティションよりも遅いものですが、物理メモリに比べればどちらも同様に遅いので、ファイルであることを気にする必要はありません。

手順 5.2: スワップファイルの手動追加

動作中のシステムでスワップファイルを追加するには、下記の手順で行います:

1. まずはお使いのシステム内に空のファイルを作成します。たとえば `/var/lib/swap/swapfile` 内に 128 MB のスワップファイルを作成するには、下記のコマンドを実行します:

```
> sudo mkdir -p /var/lib/swap
> sudo dd if=/dev/zero of=/var/lib/swap/swapfile bs=1M count=128
```

2. 作成したスワップファイルをフォーマットします:

```
> sudo mkswap /var/lib/swap/swapfile
```

注記: `mkswap` でフォーマットした場合のスワップパーティションの UUID 更新について

`mkswap` でのフォーマットは、できる限り避けておくことをお勧めします。それは、`mkswap` でフォーマットし直してしまうと、スワップパーティションの UUID が変更されてしまうためです。その代わりに、YaST を利用してフォーマットし直す (YaST では、UUID が変更されても `/etc/fstab` を自動的に 更新します) か、`mkswap` コマンド実行後に手作業で `/etc/fstab` を更新してください。

3. あとは下記のコマンドでスワップを追加します:

```
> sudo swapon /var/lib/swap/swapfile
```

スワップを無効化するには、下記のコマンドを実行します:

```
> sudo swapoff /var/lib/swap/swapfile
```

4. 利用可能なスワップ領域を確認するには、下記のコマンドを実行します:

```
> cat /proc/swaps
```

ただし、上記の手順は一時的にスワップ領域を設定するための仕組みです。システムを再起動してしまうと、追加したスワップ領域は使われなくなります。

5. 作成したスワップファイルを恒久的に使用するには、下記のような行を `/etc/fstab` に追加します:

```
/var/lib/swap/swapfile swap swap defaults 0 0
```

5.1.9 パーティション設定と LVM

［熟練者向けパーティション設定］内には LVM の設定にアクセスできる機能が用意されています。LVM の設定にアクセスするには、［システムビュー］内の［ボリューム管理］を選択してください。ただし、お使いのシステム内に既に動作している LVM 設定が存在した場合、最初の LVM 設定に入る段階で自動的に LVM が有効化されます。この場合は、有効化されたボリュームグループに属するパーティションのディスクは、編集することができなくなります。Linux カーネルでは、ディスク内のいずれかのパーティションが使用中の場合、ディスクのパーティションテーブルを再読み込みできないことによるものです。つまり、お使いのシステム内に LVM 設定がある場合は、物理的なパーティション設定は行うことができないことになります。その代わり、論理ボリュームの設定を変更してください。

物理ボリュームを作成すると、その冒頭でボリュームに関する情報をパーティションに書き込みます。LVM 以外の用途でパーティションを再利用する場合は、このボリュームに書き込まれた情報を削除することをお勧めします。たとえば論理ボリューム `system` の物理ボリューム `/dev/sda2` がある場合、下記のようなコマンドを実行します:

```
dd if=/dev/zero of=/dev/sda2 bs=512 count=1
```

5.2 デバイスの暗号化

Linux Unified Key Setup (LUKS) は Linux におけるディスク暗号化の標準仕様です。ディスク内の形式も標準化されているため、ユーザからはデータを円滑に転送および移行することができるようになっています。

LUKS はブロックデバイスの暗号化を行う仕組みです。暗号化されたデバイス内には任意のデータを書き込むことができますので、ファイルシステムに対する制限がありません。もちろんスワップパーティションを暗号化することもできます。また、暗号鍵や暗号化の種類、鍵のサイズなど、全ての設定情報がパーティションのヘッダ内に書き込まれます。

暗号化は複数階層のアプローチで実施されます。まずブロックデバイスをマスターキーで暗号化し、このマスターキーをそれぞれの有効なユーザキーで暗号化します。ユーザキーはパスフレーズのほか、FIDO2 セキュリティキーや TPM 、スマートカードなどから生成します。このような複数階層型の仕組みにより、ディスクの暗号化をやり直すことなくパスフレーズを変更できるようになっています。

LUKS に関する詳細については、**『セキュリティ強化ガイド』、第14章「cryptctl を利用したアプリケーション向けのストレージ暗号化」**をお読みください。

5.2.1 暗号化方式

デバイスを暗号化するには、5.1.3項「パーティションの編集」の手順に従って作業を行ってください。

ヒント: YaST での LUKS2 サポートの有効化について

LUKS2 による暗号化は SUSE Linux Enterprise 15 SP4 およびそれ以降のバージョンでサポートしていますが、明示的に有効化する必要があります。下記のいずれかを実施してください:

1. システムの起動時に設定する場合は、カーネルのコマンドラインに対して `YAST_LUKS2_AVAILABLE` を追加します。起動時のパラメータについて、詳しくは『スタートアップ』、第2章「起動パラメータ」をお読みください。

2. YaST によるインストール時に設定する場合は、下記のいずれかを実施します:

   - グラフィカルなインターフェイスを使用している場合は、 `Ctrl` – `Alt` – `Shift` – `C` を押します。

   - テキストインターフェイスを使用している場合は、 `Ctrl` – `D` を押してから `Shift` – `C` を押します。

   あとは［実験中の LUKS2 暗号化サポートの有効化］にチェックを入れ、［OK］を押して設定画面を閉じます。

LUKS2 サポートを有効化しない場合、［暗号化方式］の選択は表示されません。暗号化パスワードの入力のみが必要となります。

［通常の LUKS1］
: この方式を選択すると、LUKS1 方式でデバイスを暗号化します。この場合、暗号化パスワードの入力が必要となります。また、後から `cryptsetup luksAddKey` を実行することで、追加のパスワードを最大 8 つまで追加することができます。

［通常の LUKS2］
: LUKS2 は新しいヘッダ形式のバージョンで、より壊れにくい形式になっているほか、最大で 32 個までのユーザキーおよびデバイスラベルを設定することができます。また暗号化パスワードのほか、パスワードベースの鍵導出関数 (PBKDF) を設定して、パスフレーズを保護することができるようになっています (詳しくは 5.2.2項「パスワードベースの鍵導出関数」をお読みください) 。

［Pervasive (全方位型) LUKS2］(IBM Z のみ)
: この方式を選択すると、CCA モードで設定した Crypto Express 暗号化コプロセッサによって処理されたセキュアマスターキーで、LUKS2 によるデバイス暗号化を実施します。なお、暗号化システム内に本ボリュームに関連づけられたセキュアキーが既に存在する場合は、その鍵を使用するようになります。存在しない場合は新しいセキュアキーが生成され、システムに登録されます。

なお、マスターキーを保護するための暗号化パスワードを設定する必要があります。これに加えて、システム内に複数の APQN が存在する場合は、どれを使用するのかを選択することもできます。

Pervasive encryption (全方位型暗号化) に関する詳細は、https://www.ibm.com/docs/ja/linux-on-systems?topic=security-pervasive-encryption をお読みください。

[揮発性の乱数鍵による暗号化] (スワップデバイスのみ)

この方式を選択すると、システムの起動時に乱数から生成された鍵を利用して、スワップデバイスを暗号化するようになります。この方式を選択した場合、ハードディスクへのハイバネーションはできなくなります。スワップデバイスはシステムの起動のたびに暗号化をやり直すため、それ以前の内容には全くアクセスできなくなります。また、データ損失を避けるため、お使いのシステムでハイバネーションを無効化し、代わりにシャットダウンするように設定してください。

なお、スワップデバイスの再暗号化の際、暗号鍵のほかにもデバイスラベルと UUID が変化することにも注意してください。いずれも乱数による暗号化を実施したスワップデバイスではサポートされないオプションです。そのため、`/etc/crypttab` 内では、スワップデバイスの参照に際して、起動のたびに変化しない名前を指定するようにしてください。また、パーティションのデバイス名を指定してしまうと、システムの起動時に名前が入れ替わることがありますので、udev デバイス ID やパスを使用するようにしてください。もしもデバイス名が入れ替わってしまうと、予期しないデバイスを暗号化して使用することになってしまいます!

YaST では常にデバイス名を使用するように設定した場合 (詳しくはパーティション設定内の [設定] 画面をご覧ください) を除き、`/etc/crypttab` 内では変化しない名前を選択して設定します。ですが、デバイスの接続方式によっては、どの方式でも名前が変化してしまう場合があります。そのため、揮発性の乱数鍵を利用した暗号化を使用する場合は、よく注意して設定してください。

[揮発性の保護鍵による暗号化] (スワップデバイスのみ)

この方式を選択すると、暗号化コプロセッサを使用せずに保護用の一時 AES 鍵を生成し、スワップデバイスを暗号化します。これは **揮発性の乱数鍵による暗号化** の改良版ですが、この方式に関する注意事項が引き続き適用されることに留意してください。

[揮発性の保安鍵による暗号化] (スワップデバイスのみ)

この方式を選択すると、暗号化コプロセッサを利用して保安用の一時 AES 鍵を生成し、スワップデバイスを暗号化します。これは **揮発性の乱数鍵による暗号化** の改良版ですが、この方式に関する注意事項が引き続き適用されることに留意してください。

5.2.2 パスワードベースの鍵導出関数

パスワードベースの鍵導出関数 (PBKDF) はハードウエアの性能のほか、その他のシステムコンポーネントとの互換性要求レベルに従って選択すべき項目です。

PBKDF2

`PBKDF2` は LUKS1 が使用する関数です。こちらは RFC 2898 (https://tools.ietf.org/html/rfc2898) で規定されています。

Argon2i

Argon2 はより安全性を高めるように設計された関数で、計算にあたっては多くのメモリを必要とします。こちらは RFC 9106 (https://tools.ietf.org/html/rfc9106) で規定されています。Argon2i は Argon2 の派生形で、パスワードには依存しない形式でメモリ配列に対するサイドチャネル攻撃への耐性を高めた形式となります。

Argon2id

Argon2id は Argon2 のハイブリッド版です。最初の半分のパスではメモリに対する攻撃について Argon2i の形式を採用しながら、残りのパスでは Argon2d (YaST ではサポートしていません) を利用して、GPU クラッキング攻撃を防ぐように設計されています。RFC 9106 では両者の違いがわからないような場合や、サイドチャネル攻撃が現実的に発生しうるような場合に、Argon2id を使用するように推奨しています。

なお、`Argon2` のほうがより安全ではありますが、`PBKDF2` を使用すべき場合もあります:

- Argon2 はセキュリティ機能の一部として、計算に多くのメモリを使用するように意図して設計されています。システムによってはこれが問題となる場合もあります。パスワードによる保護が十分な強度であれば PBKDF2 のほうがより安全で、メモリ消費量も少なくて済みます。
- また `grub2` では、LUKS2 で暗号化されたデバイスからの起動に対するサポートが制限されていて、PBKDF2 のみを使用することができます。これは `/boot` ディレクトリを含むファイルシステムに対して、Argon2 を選択できないことを意味しています。なお、PBKDF2 を使用した場合でも、LUKS2 で暗号化したデバイスから起動を行う場合は、いくつかの `grub2` 設定を手作業で実施する必要があります。

LUKS によるデバイス暗号化に対する詳細情報については、インストーラ内の `ヘルプ` ボタンを押すか、または『セキュリティ強化ガイド』、第14章「cryptctl を利用したアプリケーション向けのストレージ暗号化」をお読みください。

5.3 LVM 設定

この章では、LVM (Logical Volume Manager; 論理ボリュームマネージャ) を設定するための固有の手順について説明しています。

警告: データの事前バックアップについて

LVM を使用することで、場合によってはデータ損失を引き起こす可能性があります。データの損失は、アプリケーションのクラッシュや電源の問題、誤ったコマンドの入力によって発生する可能性もあります。そのため、LVM を設定したりボリュームを再設定したりする場合は、事前に必ずバックアップを採取して置いてください。バックアップ無しでの作業は危険です。

YaST の LVM 設定は、YaST の熟練者向けパーティション設定 (詳しくは 5.1項「[熟練者向けパーティション設定] の使用」をお読みください) の [システムビュー] ペイン内にある [ボリューム管理] の項目からアクセスすることができます。[熟練者向けパーティション設定] ではハードディスクやパーティションの管理のほか、RAID や LVM の設定を行うことができます。

5.3.1 物理ボリューム (PV) の作成

LVM を使用するにあたって最初の作業は、ボリュームグループに対して領域を提供する物理ボリュームの作成です:

1. [ハードディスク] からハードディスクを選択します。
2. [パーティション] タブに切り替えます。
3. [追加] を押して、このディスクにおける PV のサイズを入力します。
4. [デバイスをフォーマットしない] を選択したあと、[パーティション ID] で [0x8E Linux LVM] を選択します。[マウントのオプション] では [デバイスをマウントしない] を選択します。
5. 上記までの手順を繰り返して、必要な PV を作成していきます。

5.3.2 ボリュームグループ (VG) の作成

お使いのシステム内にボリュームグループが存在していない場合は、まずそれを追加しなければなりません (詳しくは 図5.3「ボリュームグループ (VG) の作成」をお読みください) 。また、[システムビュー] ペイン内の [ボリューム管理] を選択し、[追加] 内にある [ボリュームグループ] を選択することで、追加のボリュームグループを作成することもできます。通常はボリュームグループが 1 つだけ存在すれば十分です。

1. VG に対する名前を指定します。たとえば `system` のように指定します。
2. [物理エクステントサイズ] を選択します。この値は、ボリュームグループ内での物理的なブロックのサイズを設定するものです。ボリュームグループ内にある全てのディスク領域は、このサイズのブロック単位で処理が行われます。

3. デバイスを選択して［Add］を押し、VG 内に PV を追加していきます。デバイスを選択する際、`Ctrl` を押しながら選択すると、複数のデバイスを選択することができます。

4. ［完了］を押すと VG が作成され、後続の手順を実施できるようになります。

図 5.3: ボリュームグループ (VG) の作成

既にボリュームグループを作成済みで、物理ボリュームを追加もしくは削除したい場合は、まず［ボリューム管理］内で対象のボリュームグループを選択して、［サイズ変更］を選択します。あとは表示されているウインドウ内で、必要な物理ボリュームの追加や削除を行ってください。

5.3.3 論理ボリューム (LV) の設定

ボリュームグループに物理ボリュームを追加したら、あとはオペレーティングシステムが使用する論理ボリュームの作成を行います。まずは対象のボリュームグループを選択して、［論理ボリューム］タブに移動します。その後、［追加］，［編集］，［サイズ変更］，［削除］のボタンを利用して、ボリュームグループを使用していきます。ただし、1 つのボリュームグループに対して、1 つ以上の論理ボリュームを割り当てる必要があります。

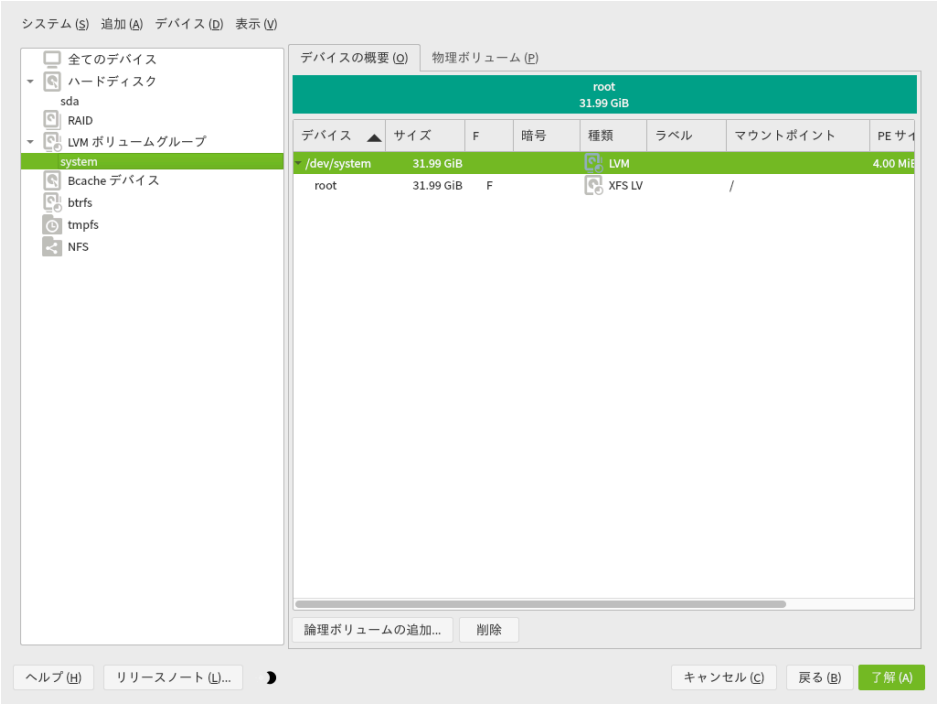


図 5.4: 論理ボリューム (LV) の管理

[追加] を押してウイザードを開きます。それぞれ下記のように設定します:

1. まずは LV の名前を入力します。たとえば `/home` にマウントする論理ボリュームであれば、`HOME` のような名前を指定します。

2. 次に LV の種類を指定します。[通常ボリューム], [Thin プール], [Thin ボリューム] のいずれかを選択することができます。なお、Thin ボリュームを作成する場合は、あらかじめ Thin プールを先に作成しておく必要があることに注意してください。Thin プロビジョニングを使用する最大の利点は、Thin プール内に保存する全ての Thin ボリュームの合計サイズは、プールそれ自身のサイズを超えて設定できる、という点です。

3. 論理ボリュームに対して設定するサイズと、ストライプ数を指定します。なお、物理ボリュームが 1 つしか無いボリュームグループの場合、複数のストライプを指定しても意味がありません。

4. LV 内で使用するファイルシステムと、マウントポイントを選択します。

ストライプを使用することで、データストリームを複数の物理ボリューム内に分散させて配置する (ストライピング) ことができます。ただし、ボリュームのストライピングは異なる物理ボリューム間でのみ行われるものであり、これによって分散を実現していることに注意してください。また、ストライプ数の最大値は物理ボリュームの個数で、"1" は "ストライピングしない" ことを意味します。さらに、ストライピングは異なるハードディスクにある複数の物理ボリュームでのみ効果を発揮するもので、同じハードディスク内で設定してしまうと、むしろ性能が下がってしまいます。

警告: ストライピング

この時点の YaST はストライピングに関するパラメータの正しさをチェックすることができません。ここでの設定が誤っていても、LVM がディスク内で設定されるまで、エラーにはなりません。

お使いのシステムに既に LVM を設定してある場合は、既存の論理ボリュームを利用して設定することもできます。ただし、続行する前に LV に対して適切なマウントポイントを設定してください。［完了］を押すと、YaST の［熟練者向けパーティション設定］に戻り、残りの作業を実施できるようになります。

5.4 ソフトウエア RAID

この章では、様々な種類の RAID を作成したり設定したりするために必要な手順を説明しています。

5.4.1 ソフトウエア RAID の設定

YaST の［RAID］設定は、YaST の［熟練者向けパーティション設定］(詳しくは 5.1項「［熟練者向けパーティション設定］の使用」をお読みください) の［システムビュー］ペイン内にある［RAID］の項目からアクセスすることができます。このパーティション設定では、ソフトウエア RAID で使用するためのパーティションの新規作成や編集、削除などを行うこともできます。

1. まずは［ハードディスク］でハードディスクを選択します。
2. ［パーティション］タブに切り替えます。
3. ［追加］を押して、このディスクにおける RAID パーティションのサイズを入力します。
4. ［デバイスをフォーマットしない］を選択したあと、［パーティション ID］で［0xFD Linux RAID］を選択します。［マウントのオプション］では［デバイスをマウントしない］を選択します。
5. 上記までの手順を繰り返して、必要な物理ボリュームを作成していきます。

RAID 0 と RAID 1 の場合、少なくとも 2 つ以上のパーティションが必要です。RAID 1 では通常、ちょうど 2 つのパーティションを使用し、それ以上は使用しません。また、RAID 5 の場合は少なくとも 3 つ以上のパーティションが、RAID 6 と RAID 10の場合は、少なくとも 4 つ以上のパーティションが必要です。ただし、いずれの場合も同じサイズのパーティションを使用することをお勧めします。RAID パーティションは様々な目的から、異なるハードディスク内にあるパーティションを使用します。RAID 1 , RAID 5 , RAID 6 の場合はディスク障害時のデータ損失を防ぐため、RAID 0 の場合は性能を最大限に引き出すため、このようにしています。RAID を設定するパーティションを作成したら、［RAID］› ［RAID の追加］を押して RAID 設定を開始します。

次のダイアログでは、RAID のレベル (0, 1, 5, 6, 10) を選択したあと、RAID システムに組み入れるべきパーティションを選択していきます。この場合、「Linux RAID」や「Linux native」になっているパーティションのみを選択することができます。スワップパーティションや DOS パーティションは表示されません。

図 5.5: RAID パーティション

上記の手順でパーティションを作成したら、あとは RAID ボリュームに追加していきます。パーティションを選択して［追加］を押してください。選択したパーティションが RAID 用に予約されるようになります。選択しなかったパーティションについては、何も行われません。全てのパーティションを割り当てたら、［次へ］を押して［RAID オプション］を選択します。

最後の手順では、RAID ボリュームで使用するファイルシステムと暗号化、およびマウントポイントを設定します。［完了］を押して設定を完了すると、`/dev/md0` 等が［熟練者向けパーティション設定］に表示されるようになります。

5.4.2 トラブルシューティング

まずは `/proc/mdstat` ファイルを確認して、RAID パーティションが壊れていないかどうかを調べてください。ハードディスクが壊れてしまったような場合は、Linux システムをシャットダウンしてディスクを交換し、新しいディスクを同じ手順でパーティション設定して追加してください。あとは `mdadm /dev/`

`mdX --add /dev/sdX` のように実行すると、RAID に組み入れることができるようになります。なお、'X' にはデバイスの識別子が入ります。これにより、ハードディスクが自動的に RAID システムに組み入れられ、再構築が開始されます。

なお、再構築中であってもデータにアクセスすることはできますが、再構築が完了するまでは RAID の性能が落ちることに注意してください。

5.4.3 さらなる情報

ソフトウエア RAID の設定手順や詳細な情報が、それぞれ下記に用意されています:

- https://raid.wiki.kernel.org

Linux RAID メーリングリストもご利用いただけます。詳しくは https://marc.info/?l=linux-raid (英語) をお読みください。

6 複数バージョンのカーネルのインストール

改訂履歴

2026-09-03

`multiversion.kernels` によって削除されるカーネルは、次回の再起動時に systemd サービスの `purge-kernels.service` によって削除される説明の追加。なお、このサービスは既定で有効化されているが、`zypper purge-kernels` を実行することで手作業でも実行できる。

2024-05-13

openSUSE Leap では、複数のバージョンのカーネルを同時にインストールすることができます。複数のバージョンのカーネルをインストールしていても、それぞれのバージョンに対する initrd と起動項目が作成されますので、手作業で何かを行う必要はありません。通常通り起動するだけで、いずれかのバージョンを選んで起動することができるようになっています。

このような機能を利用することで、カーネルの更新を行っても元のバージョンに戻ることができるようになります。新しいバージョンのカーネルで何か問題が発生しても、すぐに元の (問題の起こらない) バージョンのカーネルに戻すことができます。なお、この機能を利用する場合は、更新ツール (YaST オンライン更新や更新アプレット) を使用せず、本章で示している手順に従って作業を行ってください。

ヒント: ブートローダの設定確認について

複数のバージョンのカーネルをインストールした場合は、ブートローダの設定を確認し、どのバージョンを既定にするのかを選択してください。詳しくは 12.3項「YaST によるブートローダの設定」をお読みください。

6.1 マルチバージョン対応の有効化と設定

ソフトウエアパッケージに対して、複数のバージョンをインストールできるようにする機能 (マルチバージョン対応) は、openSUSE Leap およびそれ以降では既定で有効化されています。設定を確認するには、下記の手順で行います:

1. `root` でエディタを利用して `/etc/zypp/zypp.conf` を開きます。

2. `multiversion` で文字列検索を行います。この機能に対応する全てのカーネルパッケージに対して、マルチバージョン機能が有効化されている場合は、下記のような行がコメントアウトされていない形で書かれているはずです:

```
multiversion = provides:multiversion(kernel)
```

3. 特定のカーネルフレーバーに対してのみマルチバージョン対応を行う場合は、`/etc/zypp/zypp.conf` 内の `multiversion` オプション内に、下記のような形でパッケージ名をカンマ区切りで指定します:

```
multiversion = kernel-default,kernel-default-base,kernel-source
```

4. 設定を保存して終了します。

警告: カーネルモジュールパッケージ (KMP) について

更新されたカーネルをインストールする場合は、必要なカーネルモジュール (カーネルモジュールパッケージ) についても、更新されたバージョンに対応するものが製造元から提供されていることをご確認ください。上記の設定を行って複数のバージョンのカーネルがインストールできるようになると、古いカーネルにのみ対応するカーネルモジュールが存在していても、パッケージの依存関係としては問題が発生しませんので、新しいカーネルを起動しようとした際に、必要なカーネルモジュールを見つけることができずに問題が発生することがあります。

6.1.1 不要なカーネルの自動削除

複数バージョンのカーネルに対応するよう設定していて、頻繁に新しいカーネルをインストールしていると、起動メニューが非常に複雑化していってしまいます。一般的には、`/boot` パーティションにはそれほど大きなサイズが割り当てられていないため、`/boot` の容量不足として顕在化する場合もあります。このような場合は、YaST や zypper を利用して不要なバージョンのカーネルを削除できるだけでなく、`libzypp` に対してカーネルの自動削除を設定することができます。既定では何もカーネルを削除しない設定になっています。

1. `root` でエディタを利用して `/etc/zypp/zypp.conf` を開きます。

2. `multiversion.kernels` で文字列検索を行い、まずはその行のコメントを外します。このオプションでは、カンマ区切りで下記の値のいずれかを指定します:

   `5.3.18-53.3` : 指定したバージョンのカーネルを明示的に維持します

   `latest` : 最も新しいバージョン番号のカーネルを維持します

`latest-N`：N 番目に新しいバージョン番号のカーネルを維持します

`running`：現在使用中のカーネルを維持します

`oldest`：最も古いバージョン番号 (もしくは openSUSE Leap の出荷時に添付されている) のカーネルを維持します

`oldest+N`：N 番目に古いバージョン番号のカーネルを維持します
下記に例を示します:

`multiversion.kernels = latest,running`
: 最新のカーネルと、現在使用中のカーネルを維持します。これはマルチバージョン機能を有効化しない場合と似た動作になりますが、古いほうのカーネルの削除は、インストールの時点ではなく、次回の再起動時 にのみ行われる点が異なります。

`multiversion.kernels = latest,latest-1,running`
: 最新の 2 つのバージョンのカーネルと、現在使用中のカーネルを維持します。

`multiversion.kernels = latest,running,5.3.18-53.3`
: 最新のカーネルと現在使用中のカーネル、そしてバージョン指定で `5.3.18-53.3` を維持します。

ヒント: 現在使用中のカーネルの維持について

特別な設定をしていない限り、`running` 現在使用中のカーネルについては、削除せずに維持しておくことをお勧めします。

現在使用中のカーネルを維持するように設定していないと、カーネルを更新した際に使用中のカーネルパッケージが削除されてしまいます。パッケージにはカーネルモジュールが含まれますので、更新後は新しいカーネルモジュールを読み込めなくなってしまいます。

現在使用中のカーネルを維持しない場合は、カーネルの更新後すぐに再起動を行うようにして、モジュールによる問題を回避してください。

注記: 削除の実行タイミングについて

カーネルの削除は即時には実行されないことに注意してください。カーネルの削除は次回のシステム起動時に、既定で有効化されている `purge-kernels.service` systemd サービスによって行われます。再起動を行う前にカーネルの削除を行いたい場合は、`root` で `zypper`

`purge-kernels` コマンドを実行してください。なお、このコマンドに `--dry-run` オプションを指定すると削除予定のカーネルが表示され、実際の削除が行われなくなります。また `--details` オプションを指定すると、詳細なインストール概要が表示されるようになります。

6.1.2 設定例: 再起動後にのみ古いカーネルを削除する設定

新しいカーネルで正常に再起動ができた場合にのみ、再起動時に古いカーネルを削除する設定です。

`/etc/zypp/zypp.conf` 内を下記のように設定します:

```
multiversion.kernels = latest,running
```

上記のパラメータを指定すると、最新のカーネルと現在使用中のカーネル (最新のカーネルでなければ) を維持する動作になります。

6.1.3 設定例: 予備として古いカーネルを残す設定

1 つ以上のバージョンのカーネルを、「予備の」カーネルとして残しておく設定です。

この設定は、テスト目的でカーネルをインストールするような場合に便利です。マシンの起動ができないなど何らかの問題が発生した場合、予備として残しておいたカーネルに戻すことができます。

`/etc/zypp/zypp.conf` 内を下記のように設定します:

```
multiversion.kernels = latest,latest-1,latest-2,running
```

新しいバージョンのカーネルをインストールしてシステムを再起動すると、システムは 3 種類のバージョンのカーネル (現在のカーネル (`latest,running` と 2 種類の古いカーネル (`latest-1`, `latest-2`)) を残すようになります。

6.1.4 設定例: 特定バージョンのカーネルを維持する設定

通常のシステム更新を行い、必要に応じて新しいバージョンのカーネルをインストールします。ただし、独自のバージョンのカーネルをコンパイルしていて、そのバージョンについては、削除せずに残しておきたいものとします。

`/etc/zypp/zypp.conf` 内を下記のように設定します:

```
multiversion.kernels = latest,5.3.18-53.3,running
```

新しいバージョンのカーネルをインストールしてシステムを再起動すると、システムは 2 種類のバージョンのカーネル (現在のカーネル (`latest,running`) と独自にコンパイルしたカーネル (`5.3.18-53.3`)) を残すようになります。

6.2 YaST による複数バージョンのカーネルのインストールと削除

YaST を利用することで、複数のバージョンのカーネルをインストールしたり削除したりすることができます:

1. YaST を起動して［ソフトウエア］›［ソフトウエア管理］を選び、ソフトウエアマネージャを起動します。
2. ［表示］›［パッケージの分類］›［複数バージョンのパッケージ］を選択すると、マルチバージョンに対応したパッケージを一覧表示することができます。

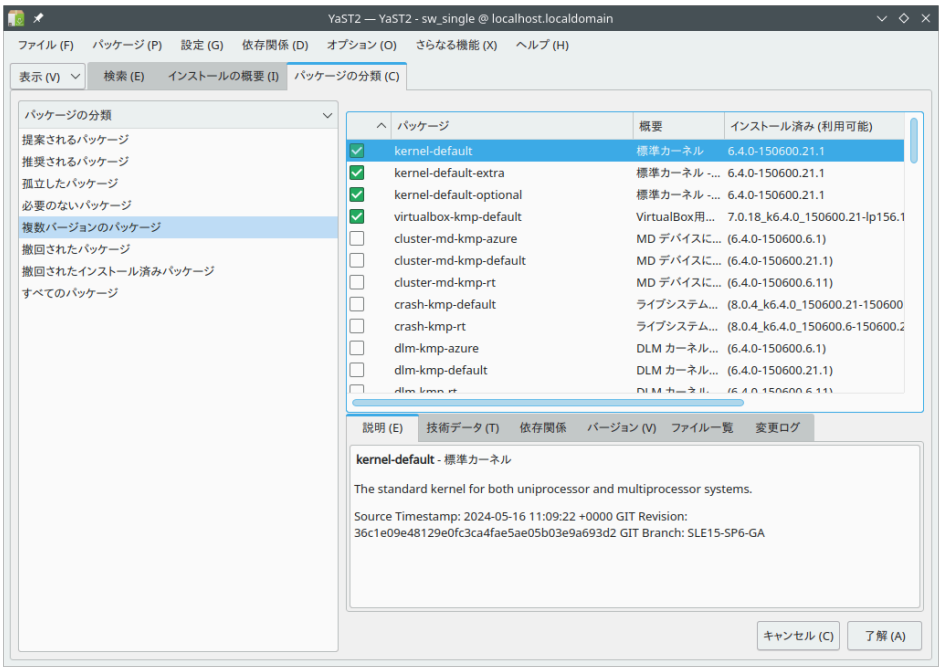


図 6.1: YAST ソフトウエアマネージャ: マルチバージョンのパッケージ

3. パッケージを選択したあと、右下にある［バージョン］タブを選択します。
4. パッケージをインストールするには、左側にあるチェックボックスにチェックを入れます。緑色のチェックマークが、インストールするよう選択されていることを表しています。
   インストール済みのパッケージ (白いチェックマークが付いているもの) を削除するには、左側にあるチェックボックスが赤い X 印になるようにチェックを入れます。
5. ［了解］を押してインストールを開始します。

6.3 zypper による複数バージョンのカーネルのインストールと削除

`zypper` を利用することでも、複数バージョンのカーネルをインストールしたり削除したりすることができます:

1. `zypper se -s 'kernel*'` と入力して実行すると、利用可能な全てのバージョンのカーネルパッケージを一覧表示することができます:

```
S  | Name                | Type    | Version           | Arch   | Repository
---+---------------------+---------+-------------------+--------
+-------------------------------------------------------
i+ | kernel-default                          | package | 6.4.0-150600.9.2              | x86_64 | SLE-
Module-Basesystem15-SP6-Pool
   | kernel-default-base                     | package | 6.4.0-150600.9.2.150600.10.40 | x86_64 | SLE-
Module-Basesystem15-SP6-Pool
   | kernel-default-devel                    | package | 6.4.0-150600.9.2              | x86_64 | SLE-
Module-Basesystem15-SP6-Pool
   | kernel-devel                            | package | 6.4.0-150600.9.2              | noarch | SLE-
Module-Basesystem15-SP6-Pool
i  | kernel-firmware-all                     | package | 20240201-150600.1.1           | noarch | SLE-
Module-Basesystem15-SP6-Pool
i  | kernel-firmware-amdgpu                  | package | 20240201-150600.1.1           | noarch | SLE-
Module-Basesystem15-SP6-Pool
i  | kernel-firmware-ath10k                  | package | 20240201-150600.1.1           | noarch | SLE-
Module-Basesystem15-SP6-Pool
i  | kernel-firmware-ath11k                  | package | 20240201-150600.1.1           | noarch | SLE-
Module-Basesystem15-SP6-Pool
i  | kernel-firmware-ath12k                  | package | 20240201-150600.1.1           | noarch | SLE-
Module-Basesystem15-SP6-Pool
i  | kernel-firmware-atheros                 | package | 20240201-150600.1.1           | noarch | SLE-
Module-Basesystem15-SP6-Pool
i  | kernel-firmware-bluetooth               | package | 20240201-150600.1.1           | noarch | SLE-
Module-Basesystem15-SP6-Pool
i  | kernel-firmware-bnx2                    | package | 20240201-150600.1.1           | noarch | SLE-
Module-Basesystem15-SP6-Pool
i  | kernel-firmware-brcm                    | package | 20240201-150600.1.1           | noarch | SLE-
Module-Basesystem15-SP6-Pool
i  | kernel-firmware-chelsio                 | package | 20240201-150600.1.1           | noarch | SLE-
Module-Basesystem15-SP6-Pool
i  | kernel-firmware-dpaa2                   | package | 20240201-150600.1.1           | noarch | SLE-
Module-Basesystem15-SP6-Pool
i  | kernel-firmware-i915                    | package | 20240201-150600.1.1           | noarch | SLE-
Module-Basesystem15-SP6-Pool
i  | kernel-firmware-intel                   | package | 20240201-150600.1.1           | noarch | SLE-
Module-Basesystem15-SP6-Pool
i  | kernel-firmware-iwlwifi                 | package | 20240201-150600.1.1           | noarch | SLE-
Module-Basesystem15-SP6-Pool
i  | kernel-firmware-liquidio                | package | 20240201-150600.1.1           | noarch | SLE-
Module-Basesystem15-SP6-Pool
i  | kernel-firmware-marvell                 | package | 20240201-150600.1.1           | noarch | SLE-
Module-Basesystem15-SP6-Pool
i  | kernel-firmware-media                   | package | 20240201-150600.1.1           | noarch | SLE-
Module-Basesystem15-SP6-Pool
i  | kernel-firmware-mediatek                | package | 20240201-150600.1.1           | noarch | SLE-
Module-Basesystem15-SP6-Pool
```

```
i  | kernel-firmware-mellanox               | package | 20240201-150600.1.1          | noarch | SLE-
Module-Basesystem15-SP6-Pool
i  | kernel-firmware-mwifiex                | package | 20240201-150600.1.1          | noarch | SLE-
Module-Basesystem15-SP6-Pool
i  | kernel-firmware-network                | package | 20240201-150600.1.1          | noarch | SLE-
Module-Basesystem15-SP6-Pool
i  | kernel-firmware-nfp                    | package | 20240201-150600.1.1          | noarch | SLE-
Module-Basesystem15-SP6-Pool
i  | kernel-firmware-nvidia                 | package | 20240201-150600.1.1          | noarch | SLE-
Module-Basesystem15-SP6-Pool
   | kernel-firmware-nvidia-gsp-G06         | package | 525.116.04-150500.1.1        | x86_64 | SLE-
Module-Basesystem15-SP6-Pool
   | kernel-firmware-nvidia-gspx-G06        | package | 550.54.14-150600.1.1         | x86_64 | SLE-
Module-Basesystem15-SP6-Pool
i  | kernel-firmware-platform               | package | 20240201-150600.1.1          | noarch | SLE-
Module-Basesystem15-SP6-Pool
i  | kernel-firmware-prestera               | package | 20240201-150600.1.1          | noarch | SLE-
Module-Basesystem15-SP6-Pool
i  | kernel-firmware-qcom                   | package | 20240201-150600.1.1          | noarch | SLE-
Module-Basesystem15-SP6-Pool
i  | kernel-firmware-qlogic                 | package | 20240201-150600.1.1          | noarch | SLE-
Module-Basesystem15-SP6-Pool
i  | kernel-firmware-radeon                 | package | 20240201-150600.1.1          | noarch | SLE-
Module-Basesystem15-SP6-Pool
i  | kernel-firmware-realtek                | package | 20240201-150600.1.1          | noarch | SLE-
Module-Basesystem15-SP6-Pool
i  | kernel-firmware-serial                 | package | 20240201-150600.1.1          | noarch | SLE-
Module-Basesystem15-SP6-Pool
i  | kernel-firmware-sound                  | package | 20240201-150600.1.1          | noarch | SLE-
Module-Basesystem15-SP6-Pool
i  | kernel-firmware-ti                     | package | 20240201-150600.1.1          | noarch | SLE-
Module-Basesystem15-SP6-Pool
i  | kernel-firmware-ueagle                 | package | 20240201-150600.1.1          | noarch | SLE-
Module-Basesystem15-SP6-Pool
i  | kernel-firmware-usb-network            | package | 20240201-150600.1.1          | noarch | SLE-
Module-Basesystem15-SP6-Pool
   | kernel-macros                          | package | 6.4.0-150600.9.2             | noarch | SLE-
Module-Basesystem15-SP6-Pool
```

2. 特定のバージョンをインストールしたい場合は、下記のように入力して実行します:

```
> sudo zypper in kernel-default-6.4.0-150600.9.2
```

3. カーネルを削除する場合は、まず `zypper se -si 'kernel*'` を実行してインストール済みのカーネルを確認して、`zypper rm` パッケージ名-バージョン のように入力して実行し、パッケージを削除します。

6.4 Kernel:HEAD リポジトリからの最新カーネルのインストール

1. Kernel:HEAD リポジトリを追加します (リポジトリには、kernel-repo という別名を付与します):

```
> sudo zypper ar \
http://download.opensuse.org/repositories/Kernel:/HEAD/standard/ \
kernel-repo
```

2. リポジトリを更新するには、下記のように実行します:

```
> sudo zypper ref
```

3. Kernel:HEAD が提供する最新バージョンのカーネルにアップグレードしたい場合は、下記のように実行します:

```
> sudo zypper dist-upgrade --allow-vendor-change --from kernel-repo
```

4. システムを再起動します。

警告: Kernel:HEAD 提供のカーネルに対する危険性について

カーネルに対する重要な修正は SUSE 側で既存のバージョンに移植 (バックポート) され、公式の更新として公開されるため、Kernel:HEAD が提供するカーネルをインストールする必要はありません。最新のバージョンのカーネルは、主にカーネルの開発者やテスターがインストールして使用するものです。また、Kernel:HEAD からインストールを行う場合は、お使いのシステムが壊れてしまうことがあり得ることにも注意してください。さらに、元のバージョンのカーネルについても、起動時の問題が発生した場合に向けて残しておいてください。

7 グラフィカルユーザインターフェイス

改訂履歴

2024-05-17

openSUSE Leap には X.org サーバと Wayland 、そして GNOME が含まれています。この章では、全てのユーザに対するグラフィカルユーザインターフェイスの設定方法を説明しています。

7.1 X Window System

X.org サーバは X11 プロトコルを実装した事実上の標準 (de facto standard) システムです。X はネットワークベースのプロトコルであり、任意の種類のネットワークを介して、一方のホストで動作させているアプリケーションの表示を、他のホストに転送することができます。

X Window System ではほとんどの場合において設定を行う必要はありません。ハードウエアは X の起動時に動的に検出されるようになっていますので、`xorg.conf` を使用する必要もなくなっています。ただし、どうしても X の動作を変更する必要があるような場合は、`/etc/X11/xorg.conf.d/` 内の設定ファイルを変更して対応してください。

openSUSE Leap 15.7 では、Wayland が X.org サーバの代替として提供されています。インストール時に選択することもできます。

X11 について、より深い情報をご希望の場合は、`xorg-docs` パッケージをインストールしてください。`man 5 xorg.conf` では、(必要であれば) 手作業での設定を行う際の書式に関する説明があります。X11 の開発について、詳しくはプロジェクトのページ (https://www.x.org) をお読みください。

ドライバ類は `xf86-video-*` パッケージ内 (たとえば `xf86-video-ati`) に含まれています。これらのパッケージ形式で配布されているドライバは、それぞれ対応するマニュアルページ内に説明があります。たとえば `ati` ドライバを使用する場合、このドライバに関する詳細は `man 4 ati` で表示することができます。

サードパーティ製のドライバに関する情報は、`/usr/share/doc/packages/<パッケージ名>` 内にあります。たとえば `x11-video-nvidiaG03` に関する説明は、`/usr/share/doc/packages/x11-video-nvidiaG04` 内にあります (ただし、パッケージをインストールしておく必要があります) 。

リモートデスクトッププロトコル (RDP) を利用してサーバに接続したい場合は、サーバ側に `xrdp` パッケージをインストールしてください。

7.2 フォントのインストールと設定

Linux でのフォントは、下記の 2 種類に分類することができます:

アウトラインフォント／ベクトルフォント

字体を描画するための描画命令として、数学的な記述を行っているフォントです。このような仕組みであることから、品質を損なうことなく任意のサイズに拡大や縮小をすることができます。ただしフォントを使用する際には、そのような数学的な記述をドットパターン (グリッド) に変形させる必要があります。この処理を一般的に、フォントのラスタライズ と呼びます。また、フォントによってはヒンティング (ビットマップへの調整のためのフォント内組み込み) 情報として、特定のサイズでの描画結果を保持させておき、描画結果を改善したり高速化したりすることもあります。ラスタライズやヒンティングの動作は、FreeType ライブラリが担っています。
Linux での一般的なフォント形式は、PostScript Type 1, Type 2, TrueType, OpenType です。

ビットマップフォント／ラスタフォント

特定のフォントサイズ向けにドットパターンの配列を持つフォントです。ビットマップフォントは非常に高速かつシンプルに描画することができます。ただし、ベクトルフォントとは異なり、品質を損なうことなく拡大したり縮小したりすることができません。このような構造であることから、一般的には複数のサイズのフォントが同梱されています。ビットマップフォントは、現在も Linux コンソールや場合によっては端末などで使用されています。
Linux での一般的なフォント形式は、Portable Compiled Format (PCF) と Glyph Bitmap Distribution Format (BDF) です。

フォントの外観は、主に 2 つの要素から構成されます:

- フォントの特徴を決めるファミリ
- 利用者にとって見やすい結果を生み出すためのフォント描画アルゴリズム

後者はベクトルフォントの場合にのみ意味がある項目です。また、上記はいずれも主観的な要素が強いものですが、いくつかの既定値を作成する必要があります。

Linux のフォント描画システムは、異なる関係性を持つ複数のライブラリから構成されています。基本的なフォント描画ライブラリは FreeType (https://www.freetype.org/) で、対応する形式のフォントの字体を最適化されたビットマップ (ドットパターン) に変換します。描画処理はアルゴリズムとパラメータによって制御されています (ここに特許に関する問題が存在します) 。

FreeType を使用するプログラムやライブラリに対しては、Fontconfig (https://www.fontconfig.org/) ライブラリの使用をお勧めします。このライブラリは、ユーザやシステムからのフォント設定を収集する仕組みを提供します。ユーザが Fontconfig の設定を修正すると、影響は Fontconfig に対応する全てのアプリケーションに及びます。

アラビア文字やパスパ文字 、日本語 など、より洗練された OpenType 描画やその他の高レベルなテキスト処理機能を必要とする場合は、Harfbuzz (https://harfbuzz.github.io/) や Pango (https://www.pango.org/) をお使いになることをお勧めします。

7.2.1 インストール済みのフォントの一覧表示

お使いのシステムにインストールされているフォントの概要を知るには、`rpm` や `fc-list` のコマンドを使用します。いずれのコマンドとも必要な情報を出力しますが、システムやユーザの設定によっては異なる一覧を返すことがあります:

`rpm`

`rpm` コマンドを利用することで、お使いのシステム内にどのようなフォントを含むソフトウエアパッケージがインストールされているのかを知ることができます:

```
> rpm -qa '*fonts*'
```

それぞれのフォントパッケージは上記のコマンドで表示されますが、`fonts-config` パッケージのように、フォントそのものでもなく、フォントを含むものでもないものが、一部表示されてしまいます。

`fc-list`

`fc-list` を実行することで、どのフォントファミリを利用できるのかを知ることができます。ただし、システム内にインストールされているものと、ユーザが独自にインストールしているものの両方が表示されます:

```
> fc-list ':' family
```

注記: `fc-list` コマンド

`fc-list` は Fontconfig ライブラリに対するラッパーです。Fontconfig を利用することで、さらに多くの興味深い、かつ正確な情報をキャッシュから問い合わせることができます。詳しくは `man 1 fc-list` をお読みください。

7.2.2 フォントの表示

インストールされているフォントの外観を確認したい場合は、`ftview` コマンド (`ft2demos` パッケージ) を使用するか、もしくは https://fontinfo.opensuse.org/ にアクセスしてください。たとえば FreeMono フォントを 14 ポイントで表示したい場合は、下記のようにして `ftview` コマンドを実行します:

```
> ftview 14 /usr/share/fonts/truetype/FreeMono.ttf
```

さらに詳しい情報を得るには、https://fontinfo.opensuse.org/ にアクセスしてください。ここにはスタイルに関する情報 (レギュラー, ボールド, イタリックなど) と対応する言語に関する情報が提供されています。

7.2.3 フォントの問い合わせ

特定のパターンに対して適用されるフォントを確認するには、`fc-match` コマンドを使用します。

たとえばパターン内に既にインストール済みのフォントが含まれている場合、`fc-match` はファイル名とフォントファミリ、スタイルに関する情報を表示します:

```
> fc-match 'Liberation Serif'
LiberationSerif-Regular.ttf: "Liberation Serif" "Regular"
```

もしも必要なフォントがお使いのシステム内に存在していない場合は、Fontconfig のマッチングルールを利用して、最も近いフォントを見つけようとします。言い換えると、要求内容が置き換えられるということになります:

```
> fc-match 'Foo Family'
DejaVuSans.ttf: "DejaVu Sans" "Book"
```

Fontconfig では、一方の名前を別のファミリ名に置き換えることのできる 別名 に対応しています。一般的には、「sans-serif」,「serif」,「monospace」などの一般的な名前を、実際のファミリ名やファミリ名の優先順に置き換える際に使用します:

```
> for font in serif sans mono; do fc-match "$font" ; done
DejaVuSerif.ttf: "DejaVu Serif" "Book"
DejaVuSans.ttf: "DejaVu Sans" "Book"
DejaVuSansMono.ttf: "DejaVu Sans Mono" "Book"
```

上記の結果は、システムにどのフォントがインストールされているかによって異なります。

注記: fontconfig に対しても同様のルールが適用される件について

Fontconfig は指定された要求に対して、常に 最も近い実際のファミリ名 (少なくとも 1 つがインストールされていれば) を返します 。「近さ」の判断基準は Fontconfig の内部仕様と、ユーザやシステム管理者が設定した Fontconfig の設定に従って決まります。

7.2.4 フォントのインストール

新しいフォントをインストールするにあたっては、いくつかの主要な方法が存在します:

1. `*.ttf` や `*.otf` のようなフォントファイルを、既知のフォントディレクトリにインストールする方法があります。システム全体に対して反映させる場合は、標準のディレクトリ `/usr/share/fonts` にインストールします。ユーザ個人でインストールしたい場合は、`~/.config/fonts` にインストールします。
   標準以外のディレクトリにインストールしたい場合は、Fontconfig 側を設定する必要があります。Fontconfig に対して `<dir>` 要素を追加して、ディレクトリを指定してください。詳しくは 7.2.5.2項「fontconfig XML の紹介」をお読みください。

2. `zypper` を利用してフォントをインストールする方法もあります。多数のフォントが SUSE ディストリビューションの一部としてパッケージの形態で提供されているほか、M17N:fonts (https://download.opensuse.org/repositories/M17N:/fonts/) リポジトリにも多数のフォントが用意されています。たとえば openSUSE Leap 15.7 で M17N:fonts リポジトリを追加する場合は、下記のようにして行います:

   ```
   > sudo zypper ar
       https://download.opensuse.org/repositories/M17N:/fonts/openSUSE_Leap_15.7/
   ```

   `フォントファミリ名` で検索するには、下記のコマンドを使用します:

   ```
   > zypper se 'フォントファミリ名*fonts'
   ```

7.2.5 フォントの外観の設定

レンダリングの状態やフォントサイズによっては、見た目があまりきれいに描画されない場合があります。たとえば現在の一般的なモニタは 100dpi 程度の解像度ですが、そのままではドットとしては大きすぎて、角張った表示になってしまいます。

アンチエイリアス (灰色を利用した平滑化) やヒンティング (ビットマップへの調整のためのフォント内組み込み) 、サブピクセルレンダリング (一方向の解像度を 3 倍化する技術) など、低解像度でもきれいに描画するためのアルゴリズムがいくつか用意されています。これらのアルゴリズムは、フォント形式によっても異なります。

Fontconfig では、描画アルゴリズムを各フォントに対して個別に設定することができるほか、フォントセットを指定して設定することもできます。

7.2.5.1 sysconfig を介したフォントの設定

openSUSE Leap には、Fontconfig に対する `sysconfig` と呼ばれるレイヤが用意されています。このファイルを利用すれば、最も簡単に設定を試すことができます。既定の設定を変更するには、`/etc/sysconfig/fonts-config` ファイルを編集してください (もしくは、YaST の sysconfig モジュールを利用してもかまいません) 。ファイルを編集したら、`fonts-config` を実行します:

```
> sudo /usr/sbin/fonts-config
```

アプリケーションを再起動すると、効果が適用されます。ただし、下記に注意してください:

- アプリケーションによっては、再起動をする必要がないものもあります。たとえば、Firefox では Fontconfig を自動的に再読み込みします。新しく作成したタブや、タブを再読み込みすることで、新しいフォント設定を適用することができます。
- `fonts-config` スクリプトは、フォントパッケージをインストールしたり削除したりするごとに、自動的に呼び出されます (もしもそうでなければ、フォントパッケージのバグです) 。
- それぞれの sysconfig の値は、`fonts-config` のコマンドラインオプションを使用することで、一時的に上書きすることができます。詳しくは `fonts-config --help` をお読みください。

sysconfig で変更可能な変数には、様々なものがあります。詳しくは `man 1 fonts-config` もしくは YaST sysconfig モジュールのヘルプページをお読みください。下記に主な変数を示します:

使用する描画アルゴリズムの設定
: `FORCE_HINTSTYLE` , `FORCE_AUTOHINT` , `FORCE_BW` , `FORCE_BW_MONOSPACE` , `USE_EMBEDDED_BITMAPS` , `EMBEDDED_BITMAP_LANGAGES`

一般的な別名に対する優先順位リスト
: `PREFER_SANS_FAMILIES` , `PREFER_SERIF_FAMILIES` , `PREFER_MONO_FAMILIES` , `SEARCH_METRIC_COMPATIBLE`

下記の一覧では、「最も読みやすい」(最も濃淡のある) フォントを先に、「最も美しい」(最もなめらかな) フォントを後ろに並べて設定例を示しています。

ビットマップフォント

`PREFER_*_FAMILIES` の変数を利用することで、ビットマップフォントを優先して使用するように設定することができます。これらの値について、詳しくはヘルプセクション内の例をお読みください。ただし、ビットマップフォントは白黒の 2 値のみで描画されるフォントであり、限られたサイズの文字しか表示することができないことに注意してください。なお、

```
SEARCH_METRIC_COMPATIBLE="no"
```

を指定して、メトリック基準のファミリ名置換を無効化しておくことをお勧めします。

白黒描画のアウトラインフォント

アウトラインフォントをアンチエイリアス処理無しで描画すると、アウトラインによる拡大／縮小の機能を生かしたまま、ビットマップフォントに似た出力を行うことができます。なお、Liberation ファミリのように適切なヒンティング情報のあるフォントをお使いください。ただし、ヒンティングのあるフォントは少数です。この方式を強制する場合は、下記の変数を設定してください:

```
FORCE_BW="yes"
```

白黒描画の等幅フォント

アンチエイリアス処理のみを無くして等幅のフォントを描画したい場合は、下記のように設定します:

```
FORCE_BW_MONOSPACE="yes"
```

既定の設定

全てのフォントに対してアンチエイリアス処理が行われます。適切にヒント情報が書かれたフォントであれば、バイトコードインタプリタ (Byte Code Interpreter; BCI) を利用して描画し、それ以外のフォントは自動ヒント (`hintstyle=hintslight`) を利用して描画します。この場合、sysconfig の設定は全て既定値に設定してください。

CFF フォント

CFF 形式のフォントを使用します。これらは FreeType2 での最新の改善版で提供される既定の TrueType フォントよりも、より読みやすく描画されます。下記にある `PREFER_*_FAMILIES` の例をご覧になり、これらのフォントをお試しください。なお、より暗く太く表示したい場合は、下記を設定します:

```
SEARCH_METRIC_COMPATIBLE="no"
```

これは既定値である `hintstyle=hintslight` と比較した場合のものです。また、下記もお試しになることをお勧めします:

```
SEARCH_METRIC_COMPATIBLE="no"
```

排他的な自動ヒント

たとえ十分なヒント情報のあるフォントであっても、FreeType2 の自動ヒント機能を使ってみる価値があります。この仕組みにより、低解像度の環境でもより太く滑らかに表示することができます。この機能を有効化するには、下記とおり変数を設定します:

```
FORCE_AUTOHINTER="yes"
```

ヒント処理のレベルを制御したい場合は、`FORCE_HINTSTYLE` をお使いください。

7.2.5.2 fontconfig XML の紹介

Fontconfig の設定形式は、eXtensible Markup Language (XML) と呼ばれる形式になっています。下記のいくつかの設定例は完全なリファレンスにはなっていませんが、簡潔な説明としては十分であるものと考えております。詳細やその他の考え方について、詳しくは `man 5 fonts-conf` や `/etc/fonts/conf.d/` をお読みください。

Fontconfig での中心的な設定ファイルは `/etc/fonts/fonts.conf` です。また、`/etc/fonts/conf.d/` 以下にある設定も自動的に取り込まれるよう設定されています。Fontconfig をカスタマイズするには、下記のいずれかの場所で設定を行ってください:

FONTCONFIG 設定ファイル

1. **システム全体の変更:** `/etc/fonts/local.conf` ファイルを編集してください (既定では、何も要素の入っていない `fontconfig` タグが書かれているだけのはずです) 。
2. **ユーザ固有の変更:** `~/.config/fontconfig/fonts.conf` ファイルを編集するか、独自にファイルを作成してある場合は `~/.config/fontconfig/conf.d/` ディレクトリ内に置いてもかまいません。

ユーザ固有の変更は、システム全体の変更を上書きして動作します。

注記: 廃止予定のユーザ設定ファイルについて

`~/.fonts.conf` ファイルは廃止予定とされているものであり、現在は使用すべきではありません。代わりに `~/.config/fontconfig/fonts.conf` をお使いください。

それぞれの設定ファイルには、`fontconfig` というタグ要素が存在する必要があります。このような構造から、最小限の設定ファイルは下記のようになります:

```
<?xml version="1.0"?>
   <!DOCTYPE fontconfig SYSTEM "fonts.dtd">
   <fontconfig>
   <!-- Insert your changes here -->
```

```
  </fontconfig>
```

既定のディレクトリでは不十分な場合、`dir` 要素を入れてディレクトリを指定します:

```
<dir>/usr/share/fonts2</dir>
```

Fontconfig では、再帰的に フォントを検索します。

フォントの描画アルゴリズムは、それぞれ 例7.1「レンダリングアルゴリズムの指定」のように指定することで選択することができます:

例 7.1: レンダリングアルゴリズムの指定

```
<match target="font">
 <test name="family">
  <string>ファミリ名</string>
 </test>
 <edit name="antialias" mode="assign">
  <bool>true</bool>
 </edit>
 <edit name="hinting" mode="assign">
  <bool>true</bool>
 </edit>
 <edit name="autohint" mode="assign">
  <bool>false</bool>
 </edit>
 <edit name="hintstyle" mode="assign">
  <const>hintfull</const>
 </edit>
</match>
```

フォントに対する様々なプロパティをテストすることができます。たとえば `<test>` 要素はフォントファミリ (上記の例のように) のほか、サイズ間隔やスペーシング、フォント形式など様々なものがあります。`<test>` で指定する以外にも、全てのフォントに対して適用する場合は `<edit>` 要素を使用することもできます。

例 7.2: 別名とファミリ名の置き換え

ルール 1

```
<alias>
 <family>Alegreya SC</family>
 <default>
  <family>serif</family>
 </default>
</alias>
```

ルール 2

```
<alias>
```

```
 <family>serif</family>
 <prefer>
  <family>Droid Serif</family>
 </prefer>
</alias>
```

ルール 3

```
<alias>
 <family>serif</family>
 <accept>
  <family>STIXGeneral</family>
 </accept>
</alias>
```

例7.2「別名とファミリ名の置き換え」にあるルールは、優先ファミリリスト (PFL; Prioritized Family List) を作成しています。それぞれ要素によって異なる処理を行っています:

ルール 1 の `<default>`
: このルールは、PFL の 末尾に 、`serif` のファミリ名ファミリ名を追加しています。

ルール 2 の `<prefer>`
: このルールは、PFL 内の最初にある `serif` の 直前に 、「Droid Serif」を追加しています。ただし、`Alegreya SC` が PFL 内に存在している必要があります。

ルール 3 の `<accept>`
: このルールは、PFL 内の `serif` ファミリ名の 直後に 、「STIXGeneral」を追加しています。

これらを組み合わせて、たとえば設定ファイルの中で ルール 1 - ルール 2 - ルール 3 の順に現れたとし、ユーザが「Alegreya SC」を要求すると、PFL は 表7.1「fontconfig ルールからの PFL の生成」のように構築されることになります。

表 7.1: FONTCONFIG ルールからの PFL の生成

順序	その時点での PFL
要求	`Alegreya SC`
ルール 1	`Alegreya SC` , `serif`
ルール 2	`Alegreya SC` , `Droid Serif` , `serif`
ルール 3	`Alegreya SC` , `Droid Serif` , `serif` , `STIXGeneral`

Fontconfig の方式では、他のパターンやスタイル、サイズなどと比べて、ファミリ名は最優先の順位が設定されます。Fontconfig では、システムにインストールされているフォントファミリを確認し、もしも「Alegreya SC」がインストールされている場合、Fontconfig はそのまま返します。インストールされていない場合は、「Droid Serif」などを探そうとします。

なお、Fontconfig の設定順序を変えてしまうと、Fontconfig が異なる動作をしてしまうことになります。たとえば 表7.2「順序を変更した fontconfig ルールから生成した PFL による結果」のようになります。

表 7.2: 順序を変更した FONTCONFIG ルールから生成した PFL による結果

順序	その時点での PFL	注意
要求	`Alegreya SC`	同じ処理を実施したものとします。
ルール 2	`Alegreya SC`	`serif` が PFL 内に存在しないので、何も起こりません。
ルール 3	`Alegreya SC`	`serif` が PFL 内に存在しないので、何も起こりません。
ルール 1	`Alegreya SC` , `serif`	`Alegreya SC` が PFL 内に存在するので、置き換えが起こります

注記: 暗黙設定について

`<default>` で指定した別名は、このグループに対する分類や包含 (インストールされていなければ) の意味があるものとお考えください。例に示されているとおり、`<default>` の設定は、そのグループに対する `<prefer>` と `<accept>` よりも常に優先して動作します。

`<default>` の分類は、serif, sans-serif, monospace に対する一般的な別名だけに限りません。より複雑な例については、`/usr/share/fontconfig/conf.avail/30-metric-aliases.conf` をお読みください。

下記 例7.3「別名とファミリ名の置き換え」の Fontconfig の設定例では、`serif` というグループを作成しています。このグループ内の各ファミリは、前のフォントがインストールされていない場合に、後ろ側のフォントで置き換える動作をします。

例 7.3: 別名とファミリ名の置き換え

```
<alias>
```

```
 <family>Alegreya SC</family>
 <default>
  <family>serif</family>
 </default>
</alias>
<alias>
 <family>Droid Serif</family>
 <default>
  <family>serif</family>
 </default>
</alias>
<alias>
 <family>STIXGeneral</family>
 <default>
  <family>serif</family>
 </default>
</alias>
<alias>
 <family>serif</family>
 <accept>
  <family>Droid Serif</family>
  <family>STIXGeneral</family>
  <family>Alegreya SC</family>
 </accept>
</alias>
```

優先順位は `<accept>` の順序で設定されています。なお、より強い `<prefer>` の別名を使用することもできます。

例7.4「別名とファミリ名の置き換え」は例7.2「別名とファミリ名の置き換え」をさらに拡張する設定です。

例 7.4: 別名とファミリ名の置き換え

ルール 4

```
<alias>
 <family>serif</family>
 <accept>
  <family>Liberation Serif</family>
 </accept>
</alias>
```

ルール 5

```
<alias>
 <family>serif</family>
 <prefer>
```

```
  <family>DejaVu Serif</family>
 </prefer>
</alias>
```

例7.4「別名とファミリ名の置き換え」によって拡張した設定は、下記のような PFL になります:

表 7.3: FONTCONFIG ルールから生成した PFL による結果

順序	その時点での PFL
要求	`Alegreya SC`
ルール 1	`Alegreya SC` , `serif`
ルール 2	`Alegreya SC` , `Droid Serif` , `serif`
ルール 3	`Alegreya SC` , `Droid Serif` , `serif` , `STIXGeneral`
ルール 4	`Alegreya SC` , `Droid Serif` , `serif` , `Liberation Serif` , `STIXGeneral`
ルール 5	`Alegreya SC` , `Droid Serif` , `DejaVu Serif` , `serif` , `Liberation Serif` , `STIXGeneral`

注記: 暗黙設定について

- 同じ汎用名に対して `<accept>` の定義が複数存在する場合、最後に処理された定義が「勝ち残ります」。また、可能であれば、システム全体の設定を作成する際に、ユーザ設定 (`/etc/fonts/conf.d/*-user.conf`) の 後ろで `<accept>` を使用しないでください。
- 同じ汎用名に対して `<prefer>` の定義が複数存在する場合、最後に処理された定義が「勝ち残ります」。また、可能であれば、システム全体の設定を作成する際に、ユーザ設定 (`/etc/fonts/conf.d/*-user.conf`) の 前に `<prefer>` を使用しないでください。
- それぞれの `<prefer>` 宣言は、同じ汎用名に対する `<accept>` 宣言を上書きします。もしも管理者がユーザに対して `<prefer>` だけでなく `<accept>` も使用させたい場合は、管理者はシステム全体の設定で `<prefer>` を使用しないでください。一方のユーザは `<prefer>` を多く使用しますが、こちらについては問題はありません。ただ、システム全体の設定で `<prefer>` が使用されているのをよく見かけます。

7.3 管理者向けの GNOME 設定

7.3.1 dconf システム

GNOME デスクトップでは、設定を `dconf` というシステムで管理します。これは階層構造化されたデータベースやレジストリと呼ばれるような仕組みで、ユーザは自分自身の設定を、システム管理者は全てのユーザに対する既定値や必須の設定などを行うことができる仕組みです。`dconf` は GNOME 2 で使用されていた `gconf` を置き換えるものでもあります。

グラフィカルユーザインターフェイスで `dconf` のオプションを表示するには、`dconf-editor` を使用します。コマンドラインで設定にアクセスしたり、変更したりしたい場合は、`dconf` コマンドを使用します。

GNOME では通常の GNOME 設定に加えて、さらに細かい設定を行うための使いやすい **調整** ツールを提供しています。このツールは GNOME のアプリケーションメニューから起動することができるほか、`gnome-tweak-tool` コマンドでも起動することができます。

7.3.2 システム全体の設定

グローバルな `dconf` 設定パラメータは、`/etc/dconf/db/` ディレクトリ内に設定します。ここには GDM に対する設定のほか、特定のオプションをユーザから変更させないような設定も含まれます。システム全体の設定を作成したい場合は、たとえば下記のような手順で行います:

1. `/etc/dconf/db/` 内に、`.d` で終わる新しいディレクトリを作成します。このディレクトリには、テキストファイル形式で任意の設定を行うことができます。この例では、`/etc/dconf/db/network.d/00-proxy` というファイルを作成して、下記の内容を入力します:

   ```
   # ここはコメントです
   [system/proxy/http]
   host='10.0.0.1'
   enabled=true
   ```

2. 新しく追加した内容を dconf データベース形式に取り込むには、下記のように実行します:

   ```
   > sudo dconf update
   ```

3. 新しく作成した `network` の設定データベースを既定のユーザプロファイルに追加します。`/etc/dconf/profile/user` ファイルを作成して、下記の内容を入力します:

   ```
   system-db:network
   ```

`/etc/dconf/profile/user` は GNOME の既定値を表すファイルです。その他のプロファイルについては、環境変数 `DCONF_PROFILE` で指定します。

4. 必要であれば、プロキシの設定をユーザに変更させないようにすることもできます。この場合は、`/etc/dconf/db/network/locks/proxy` ファイルを作成して、変更させたくないキーを一覧で入力します:

```
/system/proxy/http/host
/system/proxy/http/enabled
```

グラフィカルな `dconf-editor` を使用することで、1 人のユーザでプロファイルを作成することができます。その後 `dconf dump /` を実行することで、全ての設定オプションを一覧表示することができます。設定オプションは、グローバルプロファイル内に保存されます。

グローバル設定に関する詳細な説明は、https://wiki.gnome.org/Projects/dconf/SystemAdministrators にあります。

7.3.3 さらなる情報

さらに詳しく知るには、https://help.gnome.org/system-admin-guide/ をお読みください。

7.4 SUSE Prime を利用した Intel および NVIDIA Optimus GPU の切り替え

SUSE Prime は、Intel 社製のオンボードのグラフィック処理ユニット (GPU) に加えて NVIDIA 社製の GPU を搭載した製品のうち、Optimus と呼ばれる切り替え技術に対応したもの向けのツールです。Optimus は Intel 社の内蔵 GPU と NVIDIA 社の外付け GPU を簡単に切り替えることができるツールです。これはラップトップでの省電力と性能向上の両方を兼ね備えるための仕組みで、省電力が必要な場合は Intel 社製の内蔵 GPU を、3D アプリケーションなど性能が必要な場面では NVIDIA GPU をそれぞれ動作させる目的で使用します。

SUSE Prime は X11 の動作するシステムでのみ効果があり、Wayland には対応していません。お使いのシステムが Wayland で動作している場合は、まず Wayland を無効化して X11 に切り替えてください (詳しくは 7.4.1項「事前要件」をお読みください) 。

7.4.1 事前要件

まずは openSUSE コミュニティリポジトリが提供する NVIDIA プロプライエタリドライバを利用して、NVIDIA Optimus GPU (詳しくは 7.4.3項「NVIDIA ドライバのインストール」と Intel GPU をそれぞれ動作させてください。なお、NVIDIA Optimus 向けの古い切り替えツールである Bumblebee については、インストールしてはなりません。

また、`/etc/X11/xorg.conf` ファイルを使用してはなりません。さらに、`/etc/X11/xorg.conf.d` ディレクトリ内に、`ServerLayout`, `Device`, `Screen` のセクションを含む設定ファイルを含んでいてもいけません。

SUSE Prime は X11 でのみ動作します。`loginctl` を使用することで、お使いのシステムが X11 で動作しているのか、もしくは Wayland で動作しているのかを確認することができます:

```
> loginctl
   SESSION        UID USER             SEAT             TTY
         2       1000 tux              seat0
> loginctl show-session 2|grep Type
Type=x11
```

お使いのシステムが Wayland を使用している場合は、`/etc/gdm/custom.conf` ファイルを編集して `WaylandEnable=false` の行のコメントを外したあと、システムを再起動してください。

7.4.2 SUSE Prime のインストールと使用

下記の手順を実施する前に、NVIDIA グラフィックカード向けのドライバがインストールされ、動作していることをご確認ください。詳しくは 7.4.3項「NVIDIA ドライバのインストール」をお読みください。

まずは `suse-prime` パッケージをインストールします:

```
> sudo zypper install suse-prime
```

下記のコマンドのいずれかを実行することで、GPU を切り替えることができます。実行後はいったんログアウトして、ログインし直してください:

```
> sudo prime-select intel
> sudo prime-select intel2
> sudo prime-select nvidia
```

modesetting ドライバの場合は `intel` を、`xf86-video-intel` ドライバの場合は `intel2` を指定してください。また、`inxi` パッケージをインストールして実行することで、ドライバに関する情報を取得することもできます:

```
> inxi -G
```

```
Graphics: Device-1: Intel Xeon E3-1200 v3/4th Gen Core Processor Integrated Graphics
 Controller
          Display Server: x11(X.org 1.20.1 ) drivers: modesetting (unloaded: fbdev, vesa)
          Resolution: 1920x1080@60.00hz
          OpenGL: renderer: Mesa DRI Intel Haswell Desktop version: 4.5 Mesa 18.2.8
```

現在どちらの GPU を使用しているのかを確認するには、下記のように入力して実行します:

```
> sudo /usr/sbin/prime-select get-current
Driver configured: intel
```

7.4.3 NVIDIA ドライバのインストール

まずはお使いの NVIDIA カードの型番を識別して、使用すべきドライバを判別します。下記のようなコマンドを入力し実行してください:

```
>  /sbin/lspci | grep -i nvidia
```

あとは zypper を利用してドライバのインストールを行います。まずはお使いのディストリビューションに対応したコミュニティリポジトリを追加します。openSUSE Tumbleweed の場合は、下記のように入力して実行します:

```
> sudo zypper addrepo --refresh https://download.nvidia.com/opensuse/tumbleweed nvidia
```

openSUSE 15.7 の場合は、下記のように入力して実行します:

```
> sudo zypper addrepo --refresh https://download.nvidia.com/opensuse/leap/15.7 nvidia
```

次に利用可能なドライバパッケージの一覧を表示させます:

```
> sudo zypper se nvidia
```

最後にお使いの NVIDIA グラフィックカードに対応したドライバをインストールします:

```
> sudo zypper se パッケージ名
```

II システム

8 64 ビットシステム環境における 32 ビット／ 64 ビットアプリケーション 170

9 起動処理の紹介 172

10 systemd デーモン 180

11 journalctl：systemd ジャーナルへの問い合わせコマンド 210

12 ブートローダ GRUB 2 218

13 ネットワークの基礎 238

14 UEFI (Unified Extensible Firmware Interface) 313

15 特殊なシステム機能 324

16 udev による動的なカーネルデバイス管理 336

8 64 ビットシステム環境における 32 ビット／ 64 ビットアプリケーション

改訂履歴

2024-05-13

openSUSE® Leap は 64 ビットプラットフォームで動作します。ですが、開発者側で全てのアプリケーションを 64 ビットプラットフォームに移植しているわけではありません。本章では、64 ビットの openSUSE Leap プラットフォームで、どのようにして 32 ビットアプリケーションを動作させているのかについて、概要を説明しています。

openSUSE Leap は AMD64 や Intel 64 の各 64 ビットプラットフォームで動作しますが、この 64 ビット環境内で 32 ビットアプリケーションを標準で動作させることができます。このサポートにより、64 ビット環境に移植されるのを待つまでもなく、既存の 32 ビットアプリケーションをそのまま使い続けることができるということです。

注記: 32 ビットアプリケーションの構築に対するサポートについて

openSUSE Leap では、32 ビットアプリケーションのコンパイルには対応していません。32 ビットアプリケーションの実行のみをサポートしています。

8.1 ランタイムのサポート

重要: アプリケーションのバージョン間矛盾について

あるアプリケーションに対して 32 ビット版と 64 ビット版の両方が存在する場合、両方のバージョンをインストールしてしまうと、問題が発生する場合があります。このような場合は、いずれかのバージョンをインストールしてお使いください。

ただし、PAM (プラグイン型の認証モジュール; Pluggable Authentication Modules) については例外です。認証処理時の PAM は、ユーザとアプリケーションとの間での仲介として動作します。32 ビットアプリケーションを動作させる 64 ビットオペレーティングシステムの場合は、両方のバージョンの PAM モジュールをインストールする必要があります。

また、アプリケーションを正しく動作させるためには、様々なライブラリが必要となります。これらのライブラリの 32 ビット版と 64 ビット版は全く同じ名前になっていますので、名前以外の手段でこれらを互いに違うものとして扱わなければなりません。

32 ビット版のアプリケーションの互換性を維持するために、32 ビット版と 64 ビット版のライブラリは同じ場所に配置する必要があります。たとえば 32 ビット版の `libc.so.6` は、32 ビットオペレーティングシステムでも 64 ビットオペレーティングシステムでも、`/lib/libc.so.6` に配置します。

一方の 64 ビット版のライブラリとオブジェクトは、`lib64` 以下に配置されます。32 ビット版で `/lib` や `/usr/lib` に配置していたオブジェクトファイルは、それぞれ `/lib64` と `/usr/lib64` に配置します。このような仕組みにより、32 ビット版のライブラリは以前と同じ配置のまま、64 ビット版のライブラリを同時に配置することができるようになっています。

ただし、32 ビット版の `/lib` ディレクトリには、ビットサイズに関係のないデータも含まれていますが、これらについては移動されておらず、従来通りデータが配置されています。これは LSB (Linux Standards Base) と FHS (Filesystem Hierarchy Standard) の仕様に準拠しているためです。

8.2 カーネルの仕様

AMD64/Intel 64 の 64 ビットカーネルでは、64 ビットと 32 ビットの両方に対して ABI (アプリケーションバイナリインターフェイス) が提供されています。しかも、32 ビット版の ABI については、32 ビット版のカーネルと全く同じ仕様になっています。これにより、32 ビットアプリケーションであっても、32 ビットカーネルの場合と全く同じ方法で 64 ビットカーネルにアクセスできることになります。

64 ビットカーネルでの 32 ビットシステムコールは、システムプログラムが使用する全ての API まではサポートしていません。これはプラットフォームに依存して決まります。このような理由から、`lspci` のような少数のアプリケーションでは、同じビット数のライブラリを使用しなければなりません。

なお、64 ビット版のカーネルは 64 ビット版のカーネルモジュールだけを読み込むことができます。32 ビット版のカーネルモジュールを読み込むことはできません。

ヒント: カーネルモジュールについて

アプリケーションによってはカーネルモジュールを提供していて、それを読み込まなければ正しく動作しないものがあります。64 ビットシステム環境で、このような 32 ビット版アプリケーションを動作させたい場合は、このアプリケーションの製造元に連絡をとって、64 ビット版のカーネルモジュールを作成してもらい、32 ビット版のアプリケーションで動作するよう依頼する必要があります。

9 起動処理の紹介

改訂履歴

2024-06-27

Linux システムの起動に際しては、様々なコンポーネントや処理が介在しています。ファームウエアとハードウエアの初期化処理が完了した後は、マシンのアーキテクチャによって異なりますが、ブートローダ GRUB 2 などを利用してカーネルを起動します。この時点まで到達すると、ようやくオペレーティングシステムが制御できる状況になり、`systemd` が続きを処理するようになります。`systemd` は「ターゲット」と呼ばれるセットを提供し、日々の費用やメンテナンス、緊急時用など、それぞれに役割が設定されています。

9.1 用語説明

この章では、誤解を生みやすい用語に関して、詳しい説明を行っています:

`init`
: 一般的に「init」というと、下記の 2 つの意味があります:

  - ルートファイルシステムをマウントするための `initramfs` の処理
  - 実際のルートファイルシステムをマウントしたあとに実行される、様々なプログラムを起動するためのオペレーティングシステムの処理

  いずれの処理に対しても `systemd` がその役割を担っています。まず `systemd` はルートファイルシステムをマウントするため、`initramfs` 内に存在するものが最初に起動され、ルートファイルシステムのマウントが成功すると、最初のプロセスとしてルートファイルシステム内に存在するものが起動され、元のプロセスを置き換えて動作します。このような構造から、`systemd` のプロセスのうち前者を init on initramfs 、後者を systemd と呼びます。

`initrd` / `initramfs`
: `initrd` (INITial Ram Disk (初期 RAM ディスク) の略) は一時的なルートファイルシステムのイメージを含むファイルで、カーネルによって読み込まれ、`/dev/ram` からマウントされる仕組みです。このファイルシステムをマウントするには、ファイルシステムドライバが必要となります。

カーネルバージョン 2.6.13 より、initrd は `initramfs` (INITial RAM File System) と呼ばれるようになり、マウント時にファイルシステムのドライバを必要とはしなくなりました。openSUSE Leap では `initramfs` のみを使用しています。ですが、現在も `initramfs` は `/boot/initrd` というファイル名で保存されることから、今も「initrd」と呼ばれることが多くなっています。本章では `initramfs` の用語に統一して説明しています。

9.2 Linux の起動処理

Linux の起動処理は、複数のフェーズ (段階) に分けられます。それぞれは別々のコンポーネントが使われます:

1. 9.2.1項「初期化とブートローダの処理フェーズ」
2. 9.2.2項「カーネルの処理フェーズ」
3. 9.2.3項「initramfs での初期化フェーズ」
4. 9.2.4項「systemd の処理フェーズ」

9.2.1 初期化とブートローダの処理フェーズ

初期化フェーズでは、マシンのハードウエアの初期化を行い、デバイスの準備を実施します。この処理は、ハードウエアのアーキテクチャによって異なる処理になります。

openSUSE Leap では、全てのアーキテクチャで GRUB 2 を使用しています。アーキテクチャとファームウェアによりますが、GRUB 2 ブートローダの開始処理は複数の段階に分けることができます。ブートローダの目的は、カーネルと initramfs (初期 RAM ファイルシステム) を読み込むことです。GRUB 2 についての詳細は、第12章「ブートローダ GRUB 2」をお読みください。

9.2.1.1 AArch64 と AMD64/Intel 64 における初期化とブートローダの処理フェーズ

コンピュータの電源を投入すると、BIOS や UEFI が画面とキーボードを初期化し、メインメモリのテストを行います。この段階では、マシンはストレージメディアにアクセスすることは行いません。続いて現在の日付と時刻と、最も重要な周辺機器類に関する情報を CMOS から読み込みます。起動メディアとそのジオメトリ情報が認識できるようになると、システムは BIOS/UEFI からブートローダに処理を渡します。

従来型の BIOS を搭載したマシンの場合、起動ディスク内の最初の 512 バイト分のデータセクタ (マスターブートレコード、略して MBR) だけを読み込むことができます。MBR には最小限の機能だけを持つ GRUB 2 を書き込むことができます。この最小限の機能の GRUB 2 には、ファイルシステムのドライバを含む GRUB 2 のメインイメージを読み込むだけの機能が含まれています。メインイメージは MBR と最初のパーティションまでの間のギャップに含まれている (MBR パーティションテーブルの場合) か、もしくは BIOS の起動パーティション内に含まれています (GPT パーティションテーブルの場合) 。メインイメージにはファイルシステムドライバが含まれていますので、ルートファイルシステムにある `/boot` にアクセスすることができます。`/boot` には GRUB 2 のメインイメージに対応する追加のモジュールのほか、カーネルや initramfs のイメージが含まれていますので、あとはこれらを読み込めばカーネルに処理を渡すことができるようになります。

暗号化された `/boot` を含む暗号化ファイルシステムから BIOS のシステムを起動する場合、暗号化を解くためのパスワードは 2 回入力する必要があります。1 回目は GRUB 2 が `/boot` を解読するために、2 回目は `systemd` が暗号化されたボリュームをマウントするために必要となります。

UEFI のマシンの場合は、従来型の BIOS マシンより簡単です。ファームウェアは GPT パーティションテーブル内の FAT でフォーマットされたパーティションを読み込むことができます (これを EFI システムパーティションと呼び、起動後のシステムでは `/boot/efi` にマウントします) ので、このパーティション内に完全機能の GRUB 2 を配置すれば、それを直接読み込んで実行することができるようになります。

BIOS/UEFI のマシンがネットワーク経由での起動に対応している場合は、PXE と呼ばれる仕組みを利用して起動用のサーバからブートローダを取得することができます。この場合、BIOS/UEFI 自身がブートローダ機能の一部を持つことになります。あとは起動用のサーバからイメージを取得してシステムを起動しますので、ローカルのハードディスクを全く使用することなくシステムを立ち上げることができます。

9.2.2 カーネルの処理フェーズ

ブートローダの処理が終わると、起動処理は全てのアーキテクチャで同じになります。ブートローダはカーネルと `initramfs` (RAM ベースの初期ファイルシステム) をメモリに読み込んで、カーネルが動き始めます。

カーネルがメモリ管理機能を設定し、CPU の種類と機能を検出すると、ハードウエアの準備を行い、`initramfs` として読み込んでおいた、メモリ内にある一時的なルートファイルシステムをマウントします。

9.2.2.1 initramfs ファイル

`initramfs` (INITial RAM File System; 初期 RAM ファイルシステム) は小さな cpio アーカイブで、カーネルが RAM ディスク内に読み込むことができる仕組みです。これは `/boot/initrd` に配置されるもので、`dracut` というツールで作成することができます (詳しくは `man 8 dracut` をお読みください) 。

`initramfs` は最小限の Linux 環境を提供するもので、実際のルートファイルシステムをマウントするまでの間に、プログラムを実行する仕組みを提供します。この最小限の Linux 環境は BIOS や UEFI のルーチンからメモリ内に読み込まれ、メモリ以外には特定のハードウエア要件を持ちません。また、`initramfs` には `init` と呼ばれる実行ファイルが存在していなければなりません。このプログラムはルートファイルシステムの `systemd` デーモンを読み込んで、続きの起動処理を実行させる役割を持ちます。

ルートファイルシステムがマウントできるようになり、オペレーティングシステムが開始できるようになるには、カーネルはルートファイルシステムの存在するデバイスにアクセスするためのドライバを読み込む必要があります。これらのドライバは特殊なハードディスク向けのドライバであったり、ネットワーク経由でアクセスする場合はネットワークドライバであったりする場合もあります。ルートファイルシステムに対して必要なモジュールは、`initramfs` 内にある `init` が読み込みます。必要なモジュールが読み込まれると、`udev` が `initramfs` に対して必要なデバイスを提供するようになります。その後ルートファイルシステムの切り替えが行われ、再度デバイスの生成が行われます。これは `systemd` のユニットである `systemd-udev-trigger.service` が行います。

9.2.2.1.1 initramfs の再生成

`initramfs` にはドライバが含まれているため、新しいバージョンのドライバが提供されるようになった場合は、`initramfs` を更新する必要があります。なお、ドライバ更新を含むパッケージをインストールした場合には自動で実行されますし、YaST や zypper でそれらのパッケージをインストールした場合は、`initramfs` を生成したことを表す出力を表示します。しかしながら、場合によっては `initramfs` を手作業で生成しなければならなくなる場合があります:

ハードウエアを変更したことによってドライバを追加する必要がある場合

ハードウエア (例: ハードディスク) の変更を行った場合で、変更したハードウエアが起動時に必要であり、かつ従来とは異なるドライバを必要とする場合は、`initramfs` ファイルの更新が必要になります。

`/etc/dracut.conf.d/10-ドライバ.conf` ファイルを開くか新規に作成して、下記の行を追加します (最初の引用符の後にスペースが含まれていることに注意してください):

```
force_drivers+=" ドライバ_1 "
```

ドライバ_1 の箇所はドライバのモジュール名に置き換えてお使いください。なお、複数のドライバが必要な場合は、スペース区切りで指定します:

```
force_drivers+=" ドライバ_1 ドライバ_2 "
```

あとは 手順9.1「initramfs の生成」の手順に従って進めます。

システムディレクトリを RAID や LVM に移動した場合

スワップファイルや `/usr` などのシステムディレクトリを、RAID や論理ボリュームに移動した場合も、`initramfs` を再生成して、ソフトウエア RAID や LVM のドライバが含まれるようにする必要があります。

この場合は、`/etc/fstab` 内に対応する項目を作成して、その項目をマウント (例: `mount -a` または `swapon -a`) してください。

あとは 手順9.1「initramfs の生成」の手順に従って進めます。

ルートファイルシステムを含む LVM グループや btrfs RAID にディスクを追加した場合

ルートファイルシステムを含む論理ボリュームグループや btrfs の RAID に対して、ディスクを追加 (または削除) した場合も、サイズの変更したボリュームにアクセスすることができるようにするため、`initramfs` を再生成する必要があります。この場合は 手順9.1「initramfs の生成」にある手順に従ってください。

あとは 手順9.1「initramfs の生成」の手順に従って進めます。

カーネルの変数を変更した場合

`/etc/sysctl.conf` や `/etc/sysctl.d/*.conf` などのファイルを利用して、`sysctl` インターフェイスを介してカーネルの変数を変更した場合、この変更は次回の再起動で失われてしまいます。これは実行中に `sysctl --system` で変数を指定していても同様で、これらは `initramfs` ファイル内に保存が行われません。この場合も 手順9.1「initramfs の生成」に示されている手順を実行する必要があります。

スワップ出刃は椅子の追加や削除、再作成を行った場合

スワップデバイスの追加や削除のほか、異なる UUID で再作成を行った場合は、手順9.1「initramfs の生成」にある手順に従って initramfs を更新する必要があります。また、`/etc/default/grub` 内の `GRUB_CMDLINE_*` 変数についても、`resume=` 以下にスワップデバイスの UUID が書かれていますので、こちらについても修正を行い、12.2.1項「/boot/grub2/grub.cfg ファイル」に書かれた手順に従って `/boot/grub2/grub.cfg` を作り直してください。

手順 9.1: INITRAMFS の生成

重要

なお、下記の手順では、いずれも `root` による実行が必要となります。

1. まずは `/boot` ディレクトリに移動します:

```
# cd /boot
```

2. `dracut` コマンドを実行して、新しい `initramfs` ファイルを生成します。このとき、出力先ファイル名 には出力先のファイル名を指定します:

```
# dracut 出力先ファイル名
```

`dracut -f 出力先ファイル名` のように実行してもかまいません。この場合は、既存のファイルを上書きします。

3. (この手順は、以前の手順で `dracut -f` を実行した場合には不要です) 以前の手順で生成した `initramfs` ファイルから、`initrd` に対してシンボリックリンクを作成します:

```
#  ln -sf 出力先ファイル名 initrd
```

9.2.3 initramfs での初期化フェーズ

`systemd` (`initramfs` 内では `init` とも呼びます。詳しくは 9.1項「用語説明」をお読みください) を含む `initramfs` がカーネルによってマウントされ、一時的なルートファイルシステムになります。このプログラムは適切なルートファイルシステムをマウントするのに必要な、全ての処理を実施するほか、必要なファイルシステム向けの機能や、`udev` によるマスストレージコントローラ向けのデバイスドライバの読み込みなどを行います。

`initramfs` における `init` の主な目的は、実際のルートファイルシステムのマウントを準備し、アクセスすることにあります。システムの設定によっても異なりますが、`initramfs` における `init` は、下記を実施します:

カーネルモジュールの読み込み

ハードウエア設定にも依存しますが、お使いのコンピュータのハードウエアコンポーネント (ハードディスクなどの最も重要なもの) に対して、特別なドライバが必要となる場合があります。最終的なルートファイルシステムにアクセスするため、カーネルは適切なファイルシステムドライバも読み込む必要があります。

ブロックスペシャルファイルの提供

カーネルは読み込んだモジュールに従って、デバイスイベントを生成します。`udev` はそれらのイベントを処理して、RAM ファイルシステムである `/dev` 内に、必要なブロックスペシャルファイルを作成します。これらのスペシャルファイルが存在しないと、ファイルシステムやその他のデバイスにアクセスすることができなくなります。

RAID と LVM の設定管理

RAID や LVM の中にルートファイルシステムを保持するような設定を行っている場合、`initramfs` 内の `init` は、LVM や RAID を設定して後続の手順でルートファイルシステムにアクセスできるように構成します。

ネットワーク設定の管理

お使いのシステムのルートファイルシステムが、ネットワーク経由でマウントするもの (例: NFS) であった場合、`init` は必要なネットワークドライバを読み込んで、ルートファイルシステムにアクセスできる状態にする必要があります。

iSCSI や SAN のようなネットワークブロックデバイス内にファイルシステムが存在する場合も、`initramfs` 内の `init` がネットワークを設定します。openSUSE Leap では、プライマリ (最初) のターゲットにアクセスできない場合、セカンダリ (第 2) の iSCSI ターゲットから起動する機能に対応しています。

注記: マウント失敗時の処理

起動環境内でのルートファイルシステムのマウントが失敗した場合は、起動を続行するためにそれらを確認したり修復したりする必要があります。ext3 や ext4 のファイルシステムである場合は、ファイルシステムチェッカーが自動で起動します。XFS や btrfs のファイルシステムの場合は、修復処理が自動化されていませんので、指定可能なオプションを利用して手作業で修復作業を行います。修復が成功すると、起動環境はシステムに対してマウントのやり直しを指示します。これで問題なくマウントできれば、起動処理はそのまま続行されます。

9.2.3.1 インストール処理内での initramfs の処理

`initramfs` 内の `init` がインストール処理内の一部として起動時に呼び出された場合、上述とは異なる方式で処理が進みます。インストールシステムでは `initramfs` から `systemd` を開始することはせず、代わりに `linuxrc` がこれらの処理を行います。

インストールメディアの検出

インストール処理を開始すると、お使いのマシンはインストール用のカーネルと、YaST インストーラを含む特別な `init` を読み込みます。YaST インストーラは RAM 内のファイルシステムで動作するもので、オペレーティングシステムのインストールを行うためのインストールメディアの場所に関する情報を与える必要があります。

ハードウエア検出の実行と適切なカーネルモジュールの読み込み

9.2.2.1項「initramfs ファイル」でも説明しているとおり、起動処理はほとんどのハードウエア設定で使用できる最小セットのドライバで開始されます。AArch64, POWER, AMD64/Intel 64 の各マシンでは、`linuxrc` が初期のハードウエア検出処理を行い、お使いのハードウエア設定でどのようなドライバが必要なのかを判別します。IBM Z では、linuxrc や parmfile などで、ドライバとパラメータの一覧を指定する必要があります。

これらのドライバは、システムを起動するのに必要な独自の `initramfs` を生成するために使用されることになります。なお、起動時にはモジュールを必要としないコールドプラグ型のハードウエアの場合は、それらのモジュールは `systemd` 側で読み込みます。詳しくは 10.6.4項「カーネルモジュールの読み込み」をお読みください。

インストールシステムの読み込み

ハードウエアが正しく認識されると、必要なドライバが読み込まれます。`udev` プログラムは特殊なデバイスファイルを作成し、YaST インストーラを含む `linuxrc` を開始します。

YaST の起動

最後に `linuxrc` が YaST を起動します。これでパッケージのインストールやシステムの設定などを行うことができるようになります。

9.2.4 systemd の処理フェーズ

「実際の」ルートファイルシステムが見つかると、まずはエラーがないかどうかを確認して、マウントを行います。マウントが成功すれば、`initramfs` がメモリ内から削除され、ルートファイルシステム内の `systemd` デーモンを起動します。`systemd` は Linux システムとサービスのマネージャです。また、プロセス ID (PID) 1 で起動される親プロセスでもあり、ユーザスペースのサービスを立ち上げて管理するための init システムとしても動作します。詳しくは 第10章「systemd デーモン」をお読みください。

10 systemd デーモン

改訂履歴

2024-05-13

`systemd` はシステムの準備を担っているプロセスで、プロセス ID 1 のプロセスでもあります。`systemd` はカーネルから直接起動され、通常であればプロセスを強制終了させるシグナル 9 を受けても耐えられる仕組みを備えています。`systemd` 以外のプログラムは `systemd` から直接起動されるか、直接起動されたプロセスの子プロセスとして起動されます。また、`systemd` は System V init デーモンの代替として提供され、System V init との完全な互換性もあります (init スクリプトを動作させることができます) 。

`systemd` を使用する主なメリットとして、サービスの起動を同時並行で行うことができることによる起動時間の短縮があげられます。これに加えて、`systemd` では必要な場合にのみサービスを開始する仕組みも存在しています。これにより、システムの起動時には必要のないサービスの起動を省略することができるようになっています。また、`systemd` ではカーネルコントロールグループ (cgroups) にも対応しているほか、スナップショットやシステムの状態の保存と復元なども行うことができます。詳しくは https://www.freedesktop.org/wiki/Software/systemd/ をお読みください。

ヒント: WSL 環境下での systemd

Windows Subsystem for Linux (WSL) は Microsoft Windows オペレーティングシステムの中で Linux アプリケーションおよびディストリビューションを動作させることができる仕組みです。既定の WSL は `systemd` ではなく独自の init プログラムを使用しますので、WSL 内で動作している openSUSE Leap で `systemd` を使用したい場合は、まず `wsl_systemd` パターンをインストールします。これにより、必要な設定を自動的に行うことができます:

```
> sudo zypper in -t pattern wsl_systemd
```

手作業で行いたい場合は、`/etc/wsl.conf` ファイルを開いて下記のような行を記述します:

```
[boot]
systemd=true
```

なお、WSL 内での `systemd` へのサポートは部分的なものであることに注意してください。具体的には、`systemd` のユニットファイル側で必要なプロセス管理処理を行わなければなりません。

10.1 systemd の考え方

本章では、systemd の裏側にある考え方について説明しています。

systemd は Linux 向けのシステムとセッションのマネージャです。System V および LSB の init スクリプトとの互換性があります。主な機能は下記のとおりです:

- 並行動作機能の提供
- サービス開始のためのソケットや D-Bus 機能
- 必要に応じたデーモンの起動
- Linux cgroups を利用したプロセスの追跡
- システム状態のスナップショット採取と復元
- マウント／自動マウントのポイント管理
- 入念な依存関係ベースのサービス制御ロジックの実装

10.1.1 ユニットファイル

ユニットファイルには、サービスやソケット、デバイスやマウントポイント、自動マウントポイントやスワップファイル／パーティション、起動ターゲットや監視対象のファイルシステムパス、タイマー制御や systemd による監督制御、一時的なシステム状態のスナップショットやリソース管理スライス、外部生成されたプロセスのグループなど、様々なものに対する情報が含まれています:

systemd では「ユニットファイル」という用語が多く現れますが、下記のような意味があります:

- **サービス:** プロセス (たとえばデーモン) に関する情報が含まれています。ファイル名は .service で終わります。
- **ターゲット:** ユニットのグループ化や起動時の同期ポイントで使用されます。ファイル名は .target で終わります。
- **ソケット:** inetd のように、ソケットベースの有効化を行うための IPC やネットワークソケット、ファイルシステム FIFO などの情報が含まれています。ファイル名は .socket で終わります。
- **パス:** たとえばファイルが変更されたときにサービスを開始するなど、他のユニットのトリガーとして使用するものが含まれています。ファイル名は .path で終わります。
- **タイマー:** タイマー制御の情報やタイマーベースの有効化に関する情報が含まれています。ファイル名は .timer で終わります。

- **マウントポイント:** 通常は fstab 生成器で自動生成されるものです。ファイル名は `.mount` で終わります。
- **自動マウントポイント:** ファイルシステムの自動マウントポイントに関する情報が含まれています。ファイル名は `.automount` で終わります。
- **スワップ:** スワップデバイスやメモリページングのためのファイルに関する情報が含まれています。ファイル名は `.swap` で終わります。
- **デバイス:** sysfs/udev(7) のデバイスツリーで開示されるデバイスユニットに関する情報が含まれています。ファイル名は `.device` で終わります。
- **スコープ／スライス:** プロセスのグループを階層構造的にリソース管理するための考え方です。ファイル名は `.scope` または `.slice` で終わります。

`systemd` のユニットファイルについてさらに詳しく知るには、https://www.freedesktop.org/software/systemd/man/latest/systemd.unit.html をお読みください。

10.2 基本的な使い方

System V init システムには、サービスを管理するためのいくつかのコマンドが用意されていました。`insserv`, `telinit` などがそれにあたります。`systemd` ではサービスの管理を簡単にするために、これらの機能を `systemctl` という 1 つのコマンドにまとめて、覚えやすくしています。このコマンドは、`git` や `zypper` のように、「サブコマンド」を指定して各種の処理を行います。

```
systemctl 一般オプション サブコマンド サブコマンドのオプション
```

詳しい説明をご希望の場合は、`man 1 systemctl` をお読みください。

ヒント: 端末への出力と bash 補完

出力先が端末であり、パイプやファイルなどではない場合、`systemd` が長い出力を行う際には既定でページ制御機能を起動する仕組みを備えています。ページ制御機能を無効化するには、`--no-pager` オプションを指定します。

`systemd` では bash の補完機能も用意されています。これにより、サブコマンドの冒頭部分だけを入力して `<Tab>` を押すと、自動的に残りを補完することができます。この機能は `bash` シェル内でのみ利用可能な機能で、`bash-completion` パッケージをインストールする必要があります。

10.2.1 動作中のシステムにおけるサービスの管理

サービスを管理するためのサブコマンドは、System V init でサービスを管理するためのサブコマンドとほぼ同じです (`start` , `stop` , ...) 。サービス管理コマンドの書式は下記のとおりです:

`systemd`

```
systemctl reload|restart|start|status|stop|... サービス名_(複数可)
```

System V init

```
rcサービス名 reload|restart|start|status|stop|...
```

`systemd` では、複数のサービスを一括で管理することができます。System V init ではそれぞれの init スクリプトを別々に動作させる必要がありましたが、systemd では下記のように実行するだけです:

```
> sudo systemctl start 1_つめのサービス名 2_つめのサービス名
```

システムで利用可能な全てのサービスを一覧表示するには、下記のように実行します:

```
> sudo systemctl list-unit-files --type=service
```

下記の表では、`systemd` と System V init における主要なサービス管理コマンドについて、その比較を行っています:

表 10.1: サービス管理のコマンド

処理	`systemd` のコマンド	System V init のコマンド
開始:	`start`	`start`
停止:	`stop`	`stop`
再起動: サービスを一旦停止し、その後起動し直します。その時点でサービスが動作していない場合は、単純に起動します。	`restart`	`restart`
条件付き再起動: 現在動作中の場合にのみサービスを再起動します。その時点でサービスが動作していない場合は、何も行いません。	`try-restart`	`try-restart`

処理	`systemd` のコマンド	System V init のコマンド
再読み込み: サービスの運用を止めることなく、サービスに対して設定ファイルを読み込み直すように指示します。使用例: Apache に対して修正された `httpd.conf` 設定ファイルを読み込み直すなど。ただし全てのサービスが再読み込みに対応しているわけではありません。	`reload`	`reload`
再読み込みまたは再起動: 再読み込みに対応するサービスの場合は、再読み込みを行います。対応していない場合は再起動を行います。サービスが動作していない場合は、起動を行います。	`reload-or-restart`	無し
条件付き再読み込みまたは再起動: 再読み込みに対応するサービスの場合は、再読み込みを行います。対応していない場合は、サービスが動作している場合に限って再起動を行います。サービスが動作していない場合は、何も行いません。	`reload-or-try-restart`	無し
状態に関する詳細な情報の取得: サービスの状態に関する情報を一覧表示します。`systemd` では、説明や実行ファイル、状態や cgroup のほか、サービスが直近に発したメッセージ (詳しくは 10.6.9項「サービスのデバッグ」をお読みください) も表示します。System V init との詳細度の違いは、サービスによって異なります。	`status`	`status`
状態に関する簡潔な情報の取得: サービスが動作中かどうかを表示します。	`is-active`	`status`

10.2.2 サービスの恒久的な有効化／無効化

前のセクション内で説明しているサービス管理コマンドを利用することで、現在のセッションに対してサービスを制御することができます。`systemd` ではサービスを恒久的に有効化したり無効化したりすることで、必要に応じて起動したり、もしくは常に利用できないようにしたりすることができます。これは YaST で行うことができるほか、コマンドラインでも実施することができます。

10.2.2.1 コマンドラインからのサービスの有効化／無効化

下記の表には、それぞれ `systemd` と System V init で有効化したり無効化したりするためのコマンドを示しています。

重要: サービスの起動

コマンドラインからサービスを有効化しても、サービスは自動では起動しません。時間のシステムの起動時や、ランレベル／ターゲットの変更時に起動されるよう設定されるだけです。有効化したあとにサービスを起動するには、`systemctl start サービス名` や `rc サービス名 start` を明示的に実行する必要があります。

表 10.2: サービスの有効化と無効化のコマンド

処理	`systemd` のコマンド	System V init のコマンド
有効化:	`systemctl enable サービス名_(複数可)`	`insserv サービス名_(複数可)`, `chkconfig -a サービス名_(複数可)`
無効化:	`systemctl disable サービス名_(複数可).service`	`insserv -r サービス名_(複数可)`, `chkconfig -d サービス名_(複数可)`
確認: サービスを有効化しているかどうかを表示します。	`systemctl is-enabled サービス名`	`chkconfig サービス名`
再有効化: このコマンドは、サービスを再起動するのと同様に、いったんサービスを無効化して有効化し直します。サービスを既定値に戻したい場合に便利です。	`systemctl reenable サービス名`	(無し)
マスク: サービスを「無効化」したあとでも、手作業でサービスを開始することができます。マスクを実施すると、サービスを完全に無効化することがで	`systemctl mask サービス名`	(無し)

処理	`systemd` のコマンド	System V init のコマンド
きます。注意してお使いください。		
マスク解除： マスクを設定したサービスは、マスクを解除しないと再度使用することができなくなります。	`systemctl unmask サービス名`	(無し)

10.3 システムの起動とターゲットの管理

システムの起動やシャットダウンは、いずれも `systemd` が管理しています。この観点からすると、カーネルは裏で動作するバックグラウンド (背景) プロセスであり、CPU 時間と他のプログラムからのハードウエアアクセス要求を処理する存在であると言えます。

10.3.1 ターゲットとランレベルの違い

System V init では、システムは「ランレベル」と呼ばれる状態に対して起動を行います。ランレベルは、どのようにシステムを起動するのかと、どのサービスを起動するのかを定義するものです。ランレベルは番号で設定されていて、よく知られているものは `0` (システムのシャットダウン), `3` (ネットワーク付きのマルチユーザ環境), `5` (ネットワークとディスプレイマネージャ付きのマルチユーザ環境) の 3 種類です。

`systemd` では「ターゲットユニット」と呼ばれる新しい考え方が導入されています。しかしながら、従来のランレベルの考え方と完全な互換性をもっています。ターゲットユニットには、番号ではなく名前が設定され、特定の用途ごとに用意されています。たとえば `local-fs.target` や `swap.target` などでは、ローカルのファイルシステムやスワップ領域をマウントします。

また、ターゲット `graphical.target` はネットワークとディスプレイマネージャ機能付きのマルチユーザ環境を表すターゲットで、これがランレベルで言うところの 5 に相当します。`graphical.target` などは複雑なターゲットとして扱われていて、これは複数の他のターゲットを組み合わせた「メタ」ターゲットとして動作します。このように、既存の複数のターゲットを組み合わせることで、独自のターゲットを作成するだけの柔軟性を提供しています。

下記の一覧には、最も重要な `systemd` のターゲットユニットを一覧で示しています。完全な一覧は、`man 7 systemd.special` をご覧ください。

主な systemd ターゲットユニット

`default.target`
: 既定で起動されるターゲットです。「実際の」ターゲットではなく、`graphic.target` などの他のターゲットに対するシンボリックリンクになっています。これは YaST を利用して恒久的な変更を行うことができます (詳しくは 10.4項「YaST を利用したサービスの管理」をお読みください) 。一時的に変更したい場合は、起動プロンプトのカーネルのパラメータに `systemd.unit=ターゲット名.target` のように指定してください。

`emergency.target`
: コンソール上に最小限の機能のみが用意された緊急用の `root` シェルを起動するターゲットです。起動プロンプトで `systemd.unit=emergency.target` のように指定して使用します。

`graphical.target`
: ネットワーク機能付きでマルチユーザ対応、そしてディスプレイマネージャが有効化されるターゲットです。

`halt.target`
: システムをシャットダウンします。

`mail-transfer-agent.target`
: 電子メールを送受信するのに必要な全てのサービスを開始するターゲットです。

`multi-user.target`
: ネットワーク機能付きのマルチユーザ環境を開始するターゲットです。

`reboot.target`
: システムを再起動します。

`rescue.target`
: ネットワーク機能の無いシングルユーザモードの `root` セッションを開始します。システム管理用の基本的なツールのみが利用できます。`rescue` ターゲットは一般ユーザでのログインが失敗するような場合や、ディスプレイドライバに問題があるような場合の解決時に利用します。

System V init のランレベルシステムとの互換性を維持するため、`systemd` には `runlevelX.target` という名前の特殊なターゲットが用意されています。ここで、`X` には対応するランレベルの番号が入ります。

現在のターゲットを知りたい場合は、下記のコマンドを実行します: `systemctl get-default`

表 10.3: SYSTEM V ランレベルと systemd のターゲットユニット

System V ランレベル	systemd ターゲット	目的
0	runlevel0.target , halt.target , poweroff.target	システムのシャットダウン
1, S	runlevel1.target , rescue.target ,	シングルユーザモード
2	runlevel2.target , multi-user.target ,	リモートネットワーク無しでのローカルマルチユーザ環境
3	runlevel3.target , multi-user.target ,	ネットワーク機能付きの完全マルチユーザ環境
4	runlevel4.target	未使用もしくはユーザ定義
5	runlevel5.target , graphical.target ,	ネットワークとディスプレイマネージャを起動する完全マルチユーザ環境
6	runlevel6.target , reboot.target ,	システムの再起動

重要: systemd は /etc/inittab を無視する問題について

System V init システムでは、ランレベルを /etc/inittab で管理してきましたが、systemd ではこの設定ファイルを 使用していません 。独自の起動可能なターゲットを作成する方法について、詳しくは 10.5.5項「独自のターゲットの作成」をお読みください。

10.3.1.1 ターゲットを変更するためのコマンド

ターゲットユニットの運用には、下記のコマンドを使用します:

処理	`systemd` のコマンド	System V init のコマンド
現在のターゲット／ランレベルの変更	`systemctl isolate` ターゲット名.target	`telinit` X
既定のターゲット／ランレベルの変更	`systemctl default`	(無し)

現在のターゲット／ランレベルの取得	`systemctl list-units --type=target`
	`systemd` の場合、有効なターゲットが複数存在するのが一般的です。このコマンドは、現在有効なターゲットを全て表示します。
		`who -r`
		もしくは
		`runlevel`

既定のランレベルの恒久的な変更	サービスマネージャ を使用するか、下記のコマンドを実行します: `ln -sf /usr/lib/systemd/system/` ターゲット名.target /etc/systemd/system/default.target	サービスマネージャ を使用するか、`/etc/inittab` 内の下記の行を変更します: `id:` X :initdefault: "¥n ¥n"
現在の起動処理に対する既定のランレベルの変更	起動プロンプトで下記のオプションを指定します: `systemd.unit=` ターゲット名.target	起動プロンプトで設定したいランレベル番号を入力します。
ターゲット／ランレベルの依存関係の表示	`systemctl show -p "Requires"` ターゲット名.target `systemctl show -p "Wants"` ターゲット名.target 「Requires」ではハード依存関係 (解決しなければならないもの) を一覧表示します。「Wants」ではソフト依存関係 (可能であれば解決すべきもの) を一覧表示します。	(無し)

10.3.2 システム起動時のデバッグ

`systemd` では、システムの起動処理を分析する機能を提供しています。全てのサービスの一覧とその状態を (`/var/log/` などを処理することなく) 確認することができます。また systemd では、起動処理で時間のかかっている箇所を検出することもできます。

10.3.2.1 サービスの起動確認

システム起動以降に開始されたサービスの完全な一覧を表示するには、`systemctl` と入力して実行します。このコマンドを実行すると、下記のような出力が現れます (ただし、下記では多くの行を省略しています) 。特定のサービスに対して、より詳しい状態を取得したい場合は、`systemctl status サービス名` と入力して実行します。

例 10.1: 有効なサービスの一覧表示

```
# systemctl
UNIT                         LOAD   ACTIVE SUB       JOB DESCRIPTION
[...]
iscsi.service                loaded active exited    Login and scanning of iSC+
kmod-static-nodes.service    loaded active exited    Create list of required s+
libvirtd.service             loaded active running   Virtualization daemon
nscd.service                 loaded active running   Name Service Cache Daemon
chronyd.service              loaded active running   NTP Server Daemon
polkit.service               loaded active running   Authorization Manager
postfix.service              loaded active running   Postfix Mail Transport Ag+
rc-local.service             loaded active exited    /etc/init.d/boot.local Co+
rsyslog.service              loaded active running   System Logging Service
[...]
LOAD   = Reflects whether the unit definition was properly loaded.
ACTIVE = The high-level unit activation state, i.e. generalization of SUB.
SUB    = The low-level unit activation state, values depend on unit type.

161 loaded units listed. Pass --all to see loaded but inactive units, too.
To show all installed unit files use 'systemctl list-unit-files'.
```

開始に失敗したサービスのみを出力したい場合は、`--failed` オプションを追加します:

例 10.2: 失敗したサービスの一覧表示

```
# systemctl --failed
UNIT                    LOAD   ACTIVE SUB    JOB DESCRIPTION
apache2.service         loaded failed failed     apache
NetworkManager.service loaded failed failed     Network Manager
```

```
plymouth-start.service loaded failed failed     Show Plymouth Boot Screen

[...]
```

10.3.2.2 起動時間のデバッグ

システムの起動にかかった時間をデバッグするために、`systemd` では `systemd-analyze` というコマンドが用意されています。これは起動にかかった時間の合計のほか、起動にかかった時間順で並べたサービス一覧を表示することができます。また、必要であれば SVG の画像を生成して、起動にかかった時間とサービス同士の関係性を表示することもできます。

システムの起動時間の一覧表示

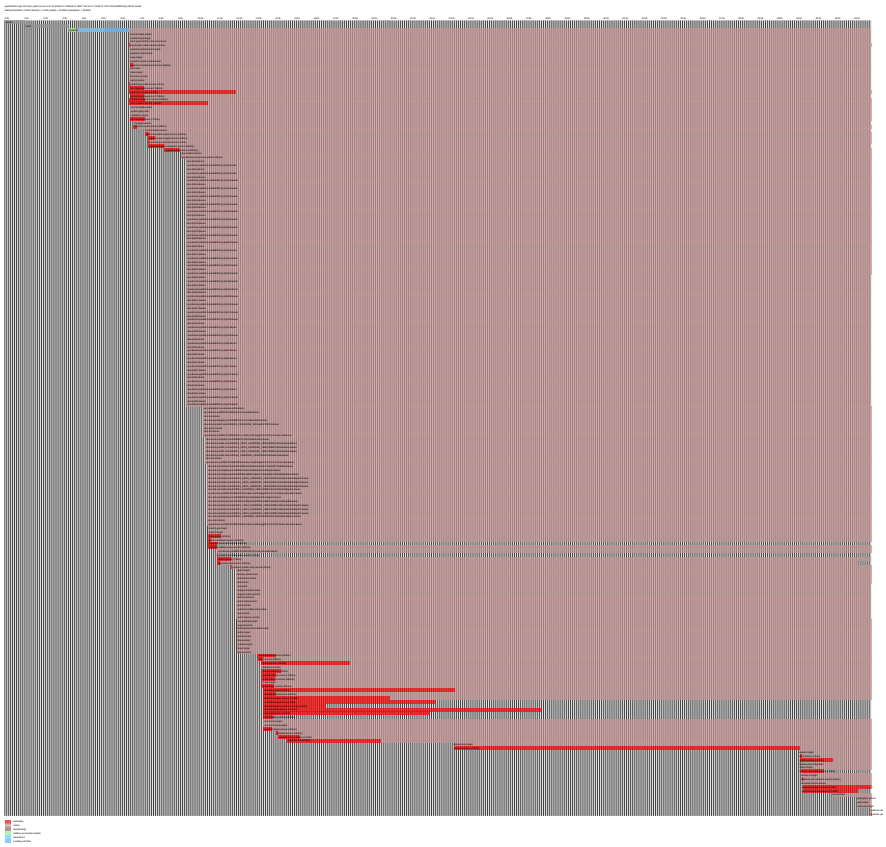
```
# systemd-analyze
Startup finished in 2666ms (kernel) + 21961ms (userspace) = 24628ms
```

サービスの起動時間の一覧表示

```
# systemd-analyze blame
    15.000s backup-rpmdb.service
    14.879s mandb.service
     7.646s backup-sysconfig.service
     4.940s postfix.service
     4.921s logrotate.service
     4.640s libvirtd.service
     4.519s display-manager.service
     3.921s btrfsmaintenance-refresh.service
     3.466s lvm2-monitor.service
     2.774s plymouth-quit-wait.service
     2.591s firewalld.service
     2.137s initrd-switch-root.service
     1.954s ModemManager.service
     1.528s rsyslog.service
     1.378s apparmor.service
    [...]
```

サービスの起動時間のグラフィック出力

```
# systemd-analyze plot > jupiter.example.com-startup.svg
```



10.3.2.3 起動処理の詳細な確認

上述のコマンドを実行することで、起動されたサービスとそれらにかかった時間を一覧表示することができます。より詳しい状況を知りたい場合は、起動プロンプトのパラメータに下記を追加することで、`systemd` に対して完全な起動処理のログを出力させることができます:

```
systemd.log_level=debug systemd.log_target=kmsg
```

上記のパラメータ指定により、`systemd` はカーネルのリングバッファにログメッセージを出力するようになります。リングバッファを表示するには、`dmesg` を使用します:

```
> dmesg -T | less
```

10.3.3 System V との互換性

`systemd` は System V との互換性があるため、既に System V init スクリプトがある場合は、それらを使用することができます。ただし、1 つだけ大きな問題があります。それは、System V init スクリプトをそのまま `systemd` で動作させようとしても、動作しない場合があるという問題です。init スクリプト内で `su` や `sudo` を利用してユーザを切り替えてサービスを起動している場合、スクリプトの実行が失敗して「Access denied」(アクセスが拒否されました) というエラーを発生させてしまいます。

`su` や `sudo` を利用してユーザを切り替える場合、PAM のセッションが起動されることになります。このセッションは init スクリプトが完了したときに終了されますので、init スクリプトから開始されたサービスについても、終了してしまいます。このエラーを回避するには、下記のようにしてください:

1. init スクリプトと同じ名前で、ファイル名の拡張子が `.service` になっているサービスラッパーファイルを作成します:

```
[Unit]
Description=説明
After=network.target

[Service]
User=ユーザ
Type=forking ❶
PIDFile=PID_ファイルのパス ❶
ExecStart=init_スクリプトのパス start
ExecStop=init_スクリプトのパス stop
ExecStopPost=/usr/bin/rm -f PID_ファイルのパス ❶

[Install]
WantedBy=multi-user.target ❷
```

   それぞれ `日本語` で書かれた箇所に必要な値を記入します。

   ❶ init スクリプトがデーモンを起動する場合にのみ設定します。

   ❷ `graphical.target` で起動する場合でも、`multi-user.target` を指定した init スクリプトを起動します。ディスプレイマネージャを起動する場合にのみ init スクリプトを起動したい場合は、ここを `graphical.target` にします。

2. あとは `systemctl start アプリケーション` でデーモンを開始するだけです。

10.4 YaST を利用したサービスの管理

基本的なサービス管理機能は、YaST サービスマネージャ モジュールでも実施することができます。サービスの開始や停止のほか、有効化や無効化も設定することができます。また、サービスの状態を表示したり、既定のターゲットを変更したりすることもできます。この YaST モジュールは、[YaST] › [システム] › [サービスマネージャ] で起動することができます。

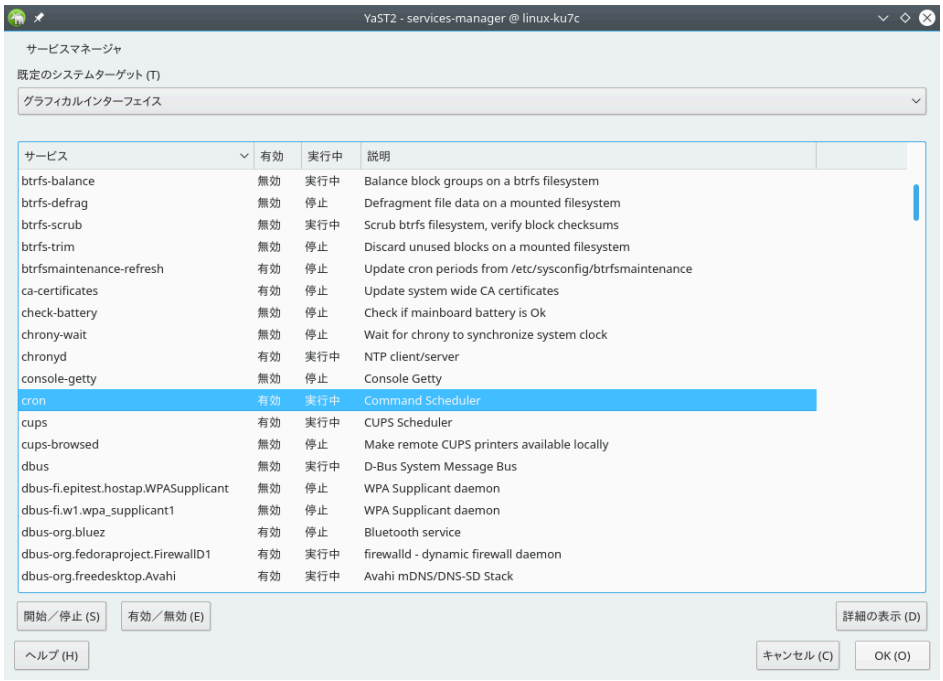


図 10.1: サービスマネージャ

[既定のシステムターゲット] の変更

システムを起動する際のターゲットを変更するには、[既定のシステムターゲット] のプロップダウンボックスで、ターゲットを選択します。最もよく使用するのは [グラフィカルインターフェイス] (グラフィカルなログイン画面を表示するターゲット) と [マルチユーザシステム] (コマンドラインモードでのシステムの開始) です。

サービスの開始と停止

一覧からサービスを選択します。このとき、[状態] の列には、現在動作中かどうかが表示されます ([実行中] であれば動作中、[停止] であれば停止していることを表します) 。開始または停止を行うには、サービスを選択したあと [開始] または [停止] ボタンを押します。

サービスの開始または停止は、現時点での変更のみに影響します。システムを再起動した際に開始または停止を行いたい場合は、有効化または無効化を実施してください。

サービスの起動動作の設定

サービスはシステムの起動時に開始させることができるほか、手動で開始させるように設定することもできます。一覧内の［開始］の列には、［手動］または［起動時］として起動動作が示されています。起動動作を変更するには、［開始モード］を押してください。

現時点での起動状態を変更したい場合は、上述の開始もしくは停止の手順を行ってください。

状態メッセージの表示

サービスの状態メッセージを表示するには、一覧からサービスを選んで［詳細の表示］を押します。表示される内容は、`systemctl -l status サービス名` と同じ内容になります。

10.5 systemd のカスタマイズ

本章では、`systemd` のユニットファイルに対するカスタマイズ方法を説明しています。

10.5.1 ユニットファイルの配置場所

SUSE 製品に同梱される `systemd` ユニットファイルは `/usr/lib/systemd/` に、カスタマイズしたユニットファイルやユニットファイルのドロップイン は `/etc/systemd/` にそれぞれ配置されます。

警告: カスタマイズを行う際の注意事項

`systemd` のカスタマイズは `/etc/systemd/` 内で行うものとし、`/usr/lib/systemd/` 内では行わないでください。`/usr/lib/systemd/` でカスタマイズを行ってしまうと、次回の `systemd` の更新でそれらが上書きされて消えてしまいます。

10.5.2 ドロップイン・ファイルによる上書き

ドロップイン・ファイル (または単にドロップイン と記述する場合もあります) はユニットファイルの一部分のみを記述したファイルで、元のユニットファイルの一部分のみを修正するためのファイルです。ドロップインは元のユニットファイルより優先して解釈されます。たとえば `systemctl edit サービス名` を実行すると、`/etc/systemd/system/サービス名.service.d/` ディレクトリを作成してその中に `override.conf` というファイルを作成し、既定のテキストエディタを起動してそのファイルを編集することができます。また、テキストエディタを終了すると、動作中の `systemd` プロセスに対して変更を通知します。

例として、MariaDB の起動時に待機する時間をカスタマイズしてみることにします。`sudo systemctl edit mariadb.service` コマンドを実行し、下記の内容を記述します:

```
# 起動または停止の際の待機時間
TimeoutSec=300
```

なお、`TimeoutSec` の値は適宜変更して保存してください。変更をシステムに適用するには、`sudo systemctl daemon-reload` コマンドを実行します。

詳しくは `man 1 systemctl` コマンドで表示されるマニュアルページを参照してください。

警告: ユニットファイル全体のコピーの作成について

`systemctl edit --full サービス名` のように `--full` オプションを指定すると、元のユニットファイルをコピーしてから編集を行うようになりますが、元のユニットファイルは SUSE 側の更新によって内容が変化する可能性があるため、このような方式でのカスタマイズは推奨しません。カスタマイズを行う場合は、`/etc/systemd/system/` ディレクトリにカスタマイズしたい箇所のみを記述するようにしてください。またこれに加えて SUSE 製品では、ディストリビューションが独自にドロップイン・ファイルを提供する場合もあることに注意してください。この場合も `--full` で作成したドロップイン・ファイルが最優先で処理されます。このような理由から、混乱を防ぐために `--full` の使用は避けてください。

10.5.3 手作業によるドロップイン・ファイルの作成

`systemctl edit` コマンドとは別に、ドロップイン・ファイルを独自に作成して優先順位を任意に調整することもできます。これらのドロップイン・ファイルは、元のユニットファイルを編集したり変更したりすることなく、ユニットやデーモンの設定を自由に変更することができるようになります。このような独自のユニットファイルは、それぞれ下記のディレクトリに配置します:

`/etc/systemd/*.conf.d/` , `/etc/systemd/system/*.service.d/`
: システム管理者が追加したりカスタマイズしたりする場合のドロップイン用ディレクトリです。

`/usr/lib/systemd/*.conf.d/` , `/usr/lib/systemd/system/*.service.d/`
: 提供元の設定を書き換えるためのカスタマイズパッケージが使用するドロップイン用ディレクトリです。たとえば SUSE では `systemd-default-settings` 等があります。

ヒント

ドロップインの検索パスの完全な一覧については、`man 5 systemd.unit` で表示されるマニュアルページをお読みください。

たとえば `systemd-journald` の既定で設定されている流量制限を無効化したい場合は、下記のような手順を実施します:

1. `/etc/systemd/journald.conf.d` という名前のディレクトリを作成します。

```
> sudo mkdir /etc/systemd/journald.conf.d
```

注記

ディレクトリ名はドロップイン・ファイルを作成したいサービス名の後ろに `.conf.d` を付けて指定します。

2. このディレクトリ内に `/etc/systemd/journald.conf.d/60-rate-limit.conf` というファイルを作成して、その中に書き換えたい内容を記述します。たとえば下記のようになります:

```
> cat /etc/systemd/journald.conf.d/60-rate-limit.conf
# 流量制限の無効化
RateLimitIntervalSec=0
```

3. あとは作成したファイルを保存して `systemd` サービスを再起動するだけです。

```
> sudo systemctl restart systemd-journald
```

注記: 名前の衝突回避について

ドロップイン・ファイルと SUSE が提供するファイルとの間で名前の衝突が起こらないように注意してください。一般的には、全てのドロップイン・ファイルのファイル名を 2 桁の数値とハイフンで始めてください。たとえば `80-override.conf` のようになります。

なお、下記の数値範囲が予約されています:

- `0-19`: `systemd` の提供元が使用します。
- `20-29`: `systemd` 向けに SUSE が独自に使用します。
- `30-39`: `systemd` 以外の SUSE パッケージが使用します。
- `40-49`: その他のサードパーティ製パッケージが使用します。
- `50`: `systemctl set-property` で作成されるドロップイン・ファイルが使用します。

上記に示した範囲よりも大きい 2 桁の数値を指定することで、SUSE や `systemd` の提供元が設定する値を上書きできるようになります。

ヒント

なお、ドロップイン・ファイルでの上書き結果を表示したい場合は、`systemctl cat $UNIT` コマンドを実行してください。

ヒント

なお、`systemd` のコンポーネントに対する設定はファイルシステム内の様々なディレクトリに分散して存在しているため、全体の概要を知るのは難しいのが通常です。`systemd` コンポーネントの設定を調査するには、下記のコマンドを使用してください:

- `systemctl cat ユニットパターン`: 1 つもしくは複数の `systemd` ユニットファイルの設定内容を出力します。たとえば下記のように実行します:

```
> systemctl cat atd.service
```

- `systemd-analyze cat-config デーモン名またはパス`: `systemd` デーモンに対する設定ファイルとドロップインの内容をコピーします。たとえば下記のように実行します:

```
> systemd-analyze cat-config systemd/journald.conf
```

10.5.4 xinetd から systemd へのサービス変換

openSUSE Leap 15 のリリース以降では、`xinetd` のインフラストラクチャが削除されています。この章では、独自の `xinetd` サービスファイルを `systemd` のソケットファイルに変換する方法を説明しています。

それぞれの `xinetd` サービスファイルに対して、2 つの `systemd` ユニットファイルが必要になります。1 つはソケットファイル (`*.socket`) で、もう 1 つはサービスファイル (`*.service`) になります。ソケットファイルでは `systemd` に対して、作成すべきソケットを指示し、サービスファイルでは起動すべきプログラムを指定しています。

たとえば下記のような `xinetd` サービスファイルがあるものとします:

```
# cat /etc/xinetd.d/example
service example
{
  socket_type = stream
  protocol = tcp
  port = 10085
  wait = no
```

```
  user = user
  group = users
  groups = yes
  server = /usr/libexec/example/exampled
  server_args = -auth=bsdtcp exampledump
  disable = no
}
```

`systemd` に変換するには、下記の 2 つのファイルを作成します:

```
# cat /usr/lib/systemd/system/example.socket
[Socket]
ListenStream=0.0.0.0:10085
Accept=false

[Install]
WantedBy=sockets.target
```

```
# cat /usr/lib/systemd/system/example.service
[Unit]
Description=example

[Service]
ExecStart=/usr/libexec/example/exampled -auth=bsdtcp exampledump
User=user
Group=users
StandardInput=socket
```

`systemd` の「ソケット」および「サービス」ファイルの全オプションを知りたい場合は、それぞれ systemd.socket と systemd.service のマニュアルページをお読みください (`man 5 systemd.socket` , `man 5 systemd.service`) 。

10.5.5 独自のターゲットの作成

System V init の SUSE システムでは、ランレベル 4 は未使用で、管理者が独自のランレベル設定を作成できるようになっていました。`systemd` では、必要に応じて任意の数のターゲットを作成することができます。作成するにあたっては、`graphical.target` などの既存のターゲットから始めることをお勧めします。

1. まずは設定ファイル `/usr/lib/systemd/system/graphical.target` を `/etc/systemd/system/ターゲット名.target` にコピーして、必要な編集を施します。
2. 上記の手順でコピーしたファイルには、ターゲットに対する必須の (「ハード」) 依存関係が既に設定されています。「ソフト」依存関係を設定したい場合は、まず `/etc/systemd/system/ターゲット名.target.wants` ディレクトリを作成してください。

3. ソフト依存関係を設定するには、`/usr/lib/systemd/system` 内から `/etc/systemd/system/MY_TARGET.target.wants` にシンボリックリンクを作成します。

4. ターゲットの設定が終わったら、`systemd` の設定を再読み込みし、新しいターゲットが利用できるようにします:

```
> sudo systemctl daemon-reload
```

10.6 高度な使い方

下記の章では、システム管理者向けの高度な使い方を説明しています。さらに高度な `systemd` のドキュメンテーションについては、Lennart Pöttering 氏による管理者向けドキュメンテーション https://0pointer.de/blog/projects/ (英語) をお読みください。

10.6.1 一時ディレクトリのクリーニング

`systemd` では、一時 (テンポラリ) ディレクトリを定期的にクリーニングする機能が用意されています。また、以前のバージョンのシステムに存在していた設定は、自動的に移行され有効化されます。`tmpfiles.d` という一時ディレクトリの管理用ファイル、具体的には `/etc/tmpfiles.d/*.conf` , `/run/tmpfiles.d/*.conf` , `/usr/lib/tmpfiles.d/*.conf` の各設定ファイルを利用しています。`/etc/tmpfiles.d/*.conf` 内にある設定は、残りの 2 つのディレクトリの設定を上書きするために用意されているディレクトリです (通常は `/usr/lib/tmpfiles.d/*.conf` 内にパッケージ側の設定ファイルが配置されます) 。

設定の書式は 1 行に 1 つずつアクションとパスを指定する形式で、必要であればモードや所有者、世代やアクションに応じたその他のパラメータなどを指定します。下記の例では、X11 のロック (施錠) ファイルを削除しています:

```
Type Path               Mode UID  GID  Age Argument
r    /tmp/.X[0-9]*-lock
```

tmpfile タイマーの状態を取得するには、下記のように実行します:

```
> sudo systemctl status systemd-tmpfiles-clean.timer
systemd-tmpfiles-clean.timer - Daily Cleanup of Temporary Directories
 Loaded: loaded (/usr/lib/systemd/system/systemd-tmpfiles-clean.timer; static)
 Active: active (waiting) since Tue 2018-04-09 15:30:36 CEST; 1 weeks 6 days ago
   Docs: man:tmpfiles.d(5)
         man:systemd-tmpfiles(8)
```

```
Apr 09 15:30:36 jupiter systemd[1]: Starting Daily Cleanup of Temporary Directories.
Apr 09 15:30:36 jupiter systemd[1]: Started Daily Cleanup of Temporary Directories.
```

一時ファイルの処理方法について、より詳しく知るには `man 5 tmpfiles.d` をお読みください。

10.6.2 システムログ

10.6.9項「サービスのデバッグ」では、特定のサービスに対してログメッセージを閲覧する方法を説明しています。しかしながら、ログメッセージはサービスに限ったものではありません。`systemd` 自身が生成した完全なログメッセージにアクセスしたり、それを検索したりすることもできます。`systemd` では、ログメッセージを総称して「ジャーナル」と読んでいます。`journalctl` コマンドを使用すると、最も古い記録から順に完全なログメッセージを表示することができます。フィルタの適用や出力形式の変更など、様々なオプションについては、`man 1 journalctl` をお読みください。

10.6.3 スナップショット

`systemd` では、現在の状態を名前付きのスナップショットとして保存し、後から `isolate` サブコマンドで復元できる機能が用意されています。これは、任意の時点の状態にいつでも戻れるという機能から、サービスや独自のターゲットをテストする際に便利な機能です。ただし、スナップショットは現在のセッションに対してのみ有効で、システムを再起動すると自動的に削除されます。また、スナップショットの名前は `.snapshot` で終わらなければなりません。

スナップショットの作成

```
> sudo systemctl snapshot スナップショット名.snapshot
```

スナップショットの削除

```
> sudo systemctl delete スナップショット名.snapshot
```

スナップショットの表示

```
> sudo systemctl show スナップショット名.snapshot
```

スナップショットの適用

```
> sudo systemctl isolate スナップショット名.snapshot
```

10.6.4 カーネルモジュールの読み込み

`systemd` では、`/etc/modules-load.d` 内に存在する設定ファイルに従って、システムの起動時に自動的にカーネルモジュールを読み込みます。ファイル名は `モジュール名`.conf という名前でなければならず、下記のような内容を記述します:

```
# モジュール を起動時に読み込む
モジュール名
```

特定のパッケージが、カーネルモジュールを読み込ませるために設定ファイルをインストールすると、そのファイルは `/usr/lib/modules-load.d` 内に保存されます。また、同じ名前の設定ファイルが `/etc/modules-load.d` 内にも存在すると、`/etc/modules-load.d` 内に存在するもののほうが優先的に処理されます。

詳しくは `modules-load.d(5)` のマニュアルページをお読みください。

10.6.5 サービス読み込み前の処理の実行

System V init では、サービスを読み込む際に何らかの処理を行う必要がある場合、`/etc/init.d/before.local` 内に記述を行っていました。`systemd` ではこのようなファイルには対応していませんが、サービスの開始前に何らかの処理をする必要がある場合は、下記のようにして行います:

カーネルモジュールの読み込み

`/etc/modules-load.d` ディレクトリ内にドロップ・インファイルを作成します (書式については `man modules-load.d` をお読みください) 。

ファイルやディレクトリの作成／ディレクトリのクリーンアップ／所有者の変更

`/etc/tmpfiles.d` ディレクトリ内にドロップイン・ファイルを作成します (書式については `man tmpfiles.d` をお読みください) 。

その他の処理

下記のようにしてシステムサービスファイルを作成します (たとえば `/etc/systemd/system/before.service` という名前で作成します):

```
[Unit]
Before=あらかじめ開始しておくべきサービスの名前
[Service]
Type=oneshot
RemainAfterExit=true
ExecStart=コマンド
# 注意: 上記のコマンドは、シェル経由ではなく直接実行されます。詳しい書式は
# systemd.service と systemd.unit の各マニュアルページをお読みください
[Install]
# サービスを開始するターゲット
```

```
WantedBy=multi-user.target
#WantedBy=graphical.target
```

サービスファイルを作成したら、後は下記のコマンドを実行します (`root` で実行します):

```
> sudo systemctl daemon-reload
> sudo systemctl enable before
```

なお、サービスファイルを変更したあとは、下記を実行してください:

```
> sudo systemctl daemon-reload
```

10.6.6 カーネルコントロールグループ (cgroups)

従来の System V init システムでは、サービスとして起動したプロセスを明示的には管理していませんでした。ただし、Apache のようなサービスでは、Apache 自身が必要な多数のプロセス (CGI や Java プロセスなど) を起動して管理していました。このような方法では、プロセスからサービスを探るのが難しくなってしまうか、場合によっては全く不可能になってしまっていました。これに加えて、サービスが正しく終了しなかった場合、子プロセスが残ったままになってしまう場合もありました。

`systemd` では、このような問題を cgroup で解決しています。cgroup はカーネルの機能で、プロセスをまとめ、生成された子プロセスを階層構造型のグループとして管理する仕組みです。`systemd` では、それぞれのサービスを別々の名前付き cgroup として管理しています。なお、非特権プロセスに対しては cgroup からの「脱退」は認められていませんので、サービスが起動した全てのプロセスを、効率的に名前で管理できるようになっています。

サービスに属する全てのプロセスを一覧表示するには、`systemd-cgls` コマンドを使用します。たとえば下記のようになります:

例 10.3: サービスに属する全プロセスの一覧表示

```
# systemd-cgls --no-pager
├─1 /usr/lib/systemd/systemd --switched-root --system --deserialize 20
├─user.slice
│ └─user-1000.slice
│   ├─session-102.scope
│   │ ├─12426 gdm-session-worker [pam/gdm-password]
│   │ ├─15831 gdm-session-worker [pam/gdm-password]
│   │ ├─15839 gdm-session-worker [pam/gdm-password]
│   │ ├─15858 /usr/lib/gnome-terminal-server

[...]

└─system.slice
  ├─systemd-hostnamed.service
  │ └─17616 /usr/lib/systemd/systemd-hostnamed
  ├─cron.service
```

```
  | └─1689 /usr/sbin/cron -n
  ├─postfix.service
  | ├─ 1676 /usr/lib/postfix/master -w
  | ├─ 1679 qmgr -l -t fifo -u
  | └─15590 pickup -l -t fifo -u
  ├─sshd.service
  | └─1436 /usr/sbin/sshd -D

[...]
```

cgroup についての詳細は、**『システム分析／チューニングガイド』、第10章「カーネルコントロールグループ」**をお読みください。

10.6.7 サービスの終了 (シグナルの送信)

10.6.6項「カーネルコントロールグループ (cgroups)」でも説明しているとおり、System V init のシステムでは、プロセスとサービスとの関係性の管理が確実にできるという保証がありませんでした。これにより、サービスを終了させて全ての子プロセスを確実に終了させるということも難しくなっていました。サービスが正しく終了できない場合、子プロセスがゾンビとして残ってしまうこともありました。

それぞれのサービスを cgroup に閉じこめる、という `systemd` の考え方により、サービスの全ての子プロセスを明確に管理できるようになるほか、これによってこれらのプロセスそれぞれに対して、シグナルを送信できる仕組みも提供できるようになっています。サービスに対してシグナルを送信するには、`systemctl kill` コマンドを使用します。利用可能なシグナルの一覧については、`man 7 signals` をお読みください。

サービスに対する `SIGTERM` の送信

`SIGTERM` は既定のシグナルです。

```
> sudo systemctl kill サービス名
```

サービスに対するその他の `シグナル` の送信

`-s` オプションを使用して、送信すべきシグナルを指定します。

```
> sudo systemctl kill -s シグナル名 サービス名
```

プロセスの選択

既定では、`kill` サブコマンドは指定した cgroup 内の「全ての」プロセスに対してシグナルを送信しますが、`control` (制御プロセスのみ) または `main` (メインプロセスのみ) のオプションを指定して、特定のプロセスのみに送信することもできます。`main` は特に、`SIGHUP` を送信して設定ファイルを再読み込みさせるような場合に便利です:

```
> sudo systemctl kill -s SIGHUP --kill-who=main サービス名
```

10.6.8 D-Bus サービスに対する重要な情報

D-Bus サービスは `systemd` クライアントと systemd マネージャ (プロセス ID=1) との間のメッセージバスとなるサービスです。`dbus` は単独のデーモンではありますが、初期化インフラストラクチャでは必須の構成部品です。

動作中のシステムで `dbus` を終了したり再起動したりする行為は、PID 1 の終了や再起動と同様であり、`systemd` のクライアント／サーバ間通信を破壊してしまい、`systemd` が正しく動作しなくなってしまいます。

そのため、`dbus` の終了や再起動は非推奨であるほか、サポートの対象外となっています。

`dbus` それ自身や `dbus` に関連するパッケージを更新した場合、システムの再起動が必要となります。システムの再起動が必要となっているかどうかを調べたい場合は、`sudo zypper ps -s` と入力して実行してください。表示されているサービスの中に `dbus` が存在する場合、システムの再起動が必要となります。

なお自動更新の設定で、システムの再起動を必要とするパッケージの更新を行わないようにしている場合でも、`dbus` は更新されてしまうことに注意してください。

10.6.9 サービスのデバッグ

既定では、`systemd` はそれほど多くの出力を行いません。サービスが問題なく起動できた場合は、全く出力を返しません。失敗した場合にのみ、短いエラーメッセージを表示します。しかしながら、`systemctl status` コマンドを使用すると、サービスの起動や動作に関して様々なデバッグメッセージを取得することができます。

`systemd` には独自のログ記録機構 (「ジャーナル」と呼びます) が備えられています。これにより、サービスのメッセージと状態に関するメッセージの両方を取得することができます。`status` サブコマンドは `tail` コマンドに似た仕組みであるほか、異なる形式でログメッセージを表示することができるなど、パワフルなデバッグツールとして便利な存在になっています。

サービスの起動失敗の表示

サービスの起動に失敗した場合は、`systemctl status サービス名` を実行することで、詳細なエラーメッセージを取得することができます:

```
# systemctl start apache2
Job failed. See system journal and 'systemctl status' for details.
# systemctl status apache2
   Loaded: loaded (/usr/lib/systemd/system/apache2.service; disabled)
   Active: failed (Result: exit-code) since Mon, 04 Apr 2018 16:52:26 +0200; 29s ago
   Process: 3088 ExecStart=/usr/sbin/start_apache2 -D SYSTEMD -k start (code=exited,
 status=1/FAILURE)
   CGroup: name=systemd:/system/apache2.service
```

```
Apr 04 16:52:26 g144 start_apache2[3088]: httpd2-prefork: Syntax error on line
205 of /etc/apache2/httpd.conf: Syntax error on li...alHost>
```

最新の `N` 件のサービスメッセージの表示

`status` サブコマンドは、既定では最新の 10 件のメッセージのみを表示します。表示する件数を変更したい場合は、`--lines=N` パラメータを指定します:

```
> sudo systemctl status chronyd
> sudo systemctl --lines=20 status chronyd
```

サービスメッセージの追記モードでの表示

サービスメッセージを「リアルタイムに」表示したい場合は、`--follow` オプションを指定します。これは `tail -f` に似た動作になります:

```
> sudo systemctl --follow status chronyd
```

メッセージの出力形式

`--output=モード` オプションを指定すると、サービスメッセージの出力形式を変更することができます。最もよく使用するモードは下記のとおりです:

`short`
: 既定の出力形式です。ログメッセージを読みやすいタイムスタンプ表示と共に出力します。

`verbose`
: 全ての項目を出力する形式です。

`cat`
: タイムスタンプ表示のない簡潔な出力形式です。

10.7 systemd タイマーユニット

`systemd` のタイマーユニットは cron と同等の機能を提供するもので、Linux でジョブのスケジュールを行うことができる仕組みです。`systemd` のタイマーユニットは cron と同じ目的で提供されるものですが、いくつかの追加機能があります。

- 別の `systemd` サービスに依存する形でジョブをスケジュールすることができます。
- タイマーユニットの取り扱いが `systemd` のサービスと同じであるため、`systemctl` を利用して管理することができます。

- タイマーはリアルタイムタイマーと単調タイマーのいずれか、もしくは両方を指定することができます。
- タイマーユニットは `systemd` のジャーナル内に記録されるため、監視やトラブルシューティングが容易です。

`systemd` のタイマーユニットのファイル名は、拡張子が `.timer` になっているため、簡単に識別することができます。

10.7.1 systemd タイマーの種類

タイマーユニットでは、間隔の指定だけでなく、具体的な日時を指定することもできます。

- cron のジョブと同様に、リアルタイムタイマーは特定の日時に到達することで実行するタイマーです。リアルタイムタイマーは、`OnCalendar` と呼ばれるオプションで指定します。
- 単調タイマーは、起動してからの経過時間を元に実行します。起動はシステムの起動のほか、ユニットの有効化が基準となります。単調タイマーの場合には複数のオプションが提供されていて、それぞれ `OnBootSec`，`OnUnitActiveSec`，`OnTypeSec` を使用します。単調タイマーはシステムの再起動が発生するとリセットされ、間隔が維持されなくなります。

10.7.2 systemd とサービスユニット

`systemd` のタイマーユニットにはそれぞれ対応するユニットファイルが存在しています。言い換えると、`.timer` ファイルは、対応する `.service` ファイルを有効化し、管理する役割を担っていることになります。タイマーとしてユニットファイルを使用する場合は、`.service` ファイル内に `[Install]` セクションを指定する必要はありません。これは、タイマー側で管理するためです。

10.7.3 具体的な例

`systemd` のタイマーユニットの基本を理解するため、例として `foo.sh` というシェルスクリプトをタイマーユニットで起動する場合を想定してみます。

最初に `systemd` のサービスユニットファイルを作成します。このファイルがシェルスクリプトの制御を行うことになります。テキストエディタを起動して、下記のような内容でサービスユニットファイルの設定を作成してください:

```
[Unit]
Description="Foo shell script"
```

```
[Service]
ExecStart=/usr/local/bin/foo.sh
```

このファイルを、`/etc/systemd/system/` ディレクトリ内の `foo.service` という名前で保存します。

次にタイマーユニットを作成します。テキストエディタで下記のような内容を記述します:

```
[Unit]
Description="Run foo shell script"

[Timer]
OnBootSec=5min
OnUnitActiveSec=24h
Unit=foo.service

[Install]
WantedBy=multi-user.target
```

上記の例では、`[Timer]` セクションで対応するサービスユニットファイルを指定 (`foo.service`) し、起動間隔を指定しています。この例では `OnBootSec` を指定していますので、まずはシステムの起動後を基準として 5 分間隔でシェルスクリプトを起動することになります。また、`OnUnitActiveSec` の指定も存在することから、ユニットの有効化から 24 時間が経過するタイミングでも起動することになります (つまり 1 日 1 回) 。また、`WantedBy` の指定では、システムがマルチユーザターゲットに到達している場合にタイマーを開始すべきである旨を示しています。

単調タイマーの代わりに、`OnCalendar` でリアルタイムタイマーを指定することもできます。たとえば下記のように指定すると、月曜日の 12:00 を開始時点として、週 1 回の間隔で対応するサービスを起動することができます。

```
[Timer]
OnCalendar=weekly
Persistent=true
```

ここでは `Persistent=true` を指定していますが、これはシステムが直前の起動時間を判断できない場合 (たとえばシステムの電源を落としていた場合) に、タイマーを即時に起動することを示しています。

このほか、`OnCalendar` では `曜日 年-月-日 時:分:秒` のような書式で指定することで、特定の日時に起動を行うこともできます。たとえば下記のように指定すると、毎日 AM5:00 にサービスを起動することになります:

```
OnCalendar=*-*-* 5:00:00
```

cron と同様にアスタリスク記号で任意の数値を、カンマ区切りで複数の数値を指定することもできます。また、2 つの値の間に .. を入れることで、連続した値を指定することもできます。たとえば下記の例は、毎月最初の金曜日の PM6:00 に起動する意味になります:

```
OnCalendar=Fri *-*-1..7 18:00:00
```

複数の日時を指定したい場合は、`OnCalendar` の項目を複数に分割してください:

```
OnCalendar=Mon..Fri 10:00
OnCalendar=Sat,Sun 22:00
```

上記の例では、月曜日から金曜日までの AM10:00 と、週末の PM10:00 にそれぞれ起動する意味になります。

タイマーユニットファイルを作成したら、`/etc/systemd/system/` ディレクトリ内に `foo.timer` というファイル名で保存してください。ユニットファイルが正しく設定できているかどうかを確認するには、下記のようなコマンドを実行します:

```
> sudo  systemd-analyze verify /etc/systemd/system/foo.*
```

上記のコマンドを実行しても何も出力されなければ、問題なくユニットファイルが設定できていることを示しています。

タイマーを開始するには `sudo systemctl start foo.timer` を、システムの起動時にタイマーを有効化するには `sudo systemctl enable foo.timer` をそれぞれ実行してください。

10.7.4 systemd タイマーの管理

タイマーは `systemd` のユニットとして扱われるため、管理も `systemctl` コマンドで行います。タイマーを起動するには `systemctl start` を、タイマーを有効化するには `systemctl enable` をそれぞれ実行してください。これに加えて、`systemctl list-timers` を実行すると、現在有効なタイマーの一覧を表示することができます。有効になっていないタイマーも含めたい場合は、`systemctl list-timers --all` を実行してください。

10.8 さらなる情報

`systemd` について、より詳しく知りたい場合は、下記のオンラインリソースをお読みください:

Web ページ

https://systemd.io/ ↗

`systemd` for administrators

`systemd` の作者のうちの 1 人である Lennart Pöttering 氏が投稿したブログ記事 (本章記述時点で 13 個, いずれも英語): https://0pointer.de/blog/projects/ ↗

11 journalctl：systemd ジャーナルへの問い合わせコマンド

改訂履歴

2024-05-13

systemd には独自のログ記録機能として、ジャーナル と呼ばれる機能が用意されています。これにより、syslog ベースのサービスを動作させる必要はなくなり、全てのイベントがジャーナルに書き込まれるようになっています。

ジャーナルそれ自身は systemd が管理するシステムサービスで、正確には systemd-journald.service という名称です。カーネルやユーザスペース、標準入力やシステムサービスのエラーなどから、様々なログ記録情報を収集し、構造化され順序番号の付いたジャーナルを管理して、データを記録し保存することができます。systemd-journald サービスは既定で有効化されています:

```
> sudo systemctl status systemd-journald
systemd-journald.service - Journal Service
   Loaded: loaded (/usr/lib/systemd/system/systemd-journald.service; static)
   Active: active (running) since Mon 2014-05-26 08:36:59 EDT; 3 days ago
     Docs: man:systemd-journald.service(8)
           man:journald.conf(5)
 Main PID: 413 (systemd-journal)
   Status: "Processing requests..."
   CGroup: /system.slice/systemd-journald.service
           └─413 /usr/lib/systemd/systemd-journald
[...]
```

11.1 ジャーナルの恒久化

ジャーナルは既定では /run/log/journal/ 内にデータを保存します。ただし、/run/ ディレクトリはメモリ内にのみ保持されるディレクトリであるため、システムを再起動するとログデータが失われてしまいます。ログデータを恒久的に保存するには、/var/log/journal/ を作成して適切な所有権とアクセス許可を設定し、systemd-journald に対してデータを保存するように設定する必要があります。具体的には下記のコマンドを実行します:

```
> sudo  mkdir /var/log/journal
> sudo  systemd-tmpfiles --create --prefix=/var/log/journal
> sudo  journalctl --flush
```

上記を実行することにより、/run/log/journal/ 内に保存される全てのログデータが /var/log/journal/ に書き込まれるようになります。

11.2 `journalctl` での便利なスイッチ

本章では、既定の `journalctl` の動作を拡張するにあたって、便利なオプション類をいくつか紹介しています。全てのスイッチについての説明をご希望の場合は、`journalctl` のマニュアルページ (`man 1 journalctl`) をご覧ください。

ヒント: 特定の実行ファイルに関連するメッセージ

特定の実行ファイルに関連する全てのジャーナルメッセージを表示するには、実行ファイルのフルパスを指定します:

```
> sudo journalctl /usr/lib/systemd/systemd
```

-f

直近のジャーナルメッセージのみを表示し、以後ジャーナルに更新があるたびに項目を随時表示します。

ジャーナルメッセージを表示して末尾に移動します。ページャを介して最新の項目から閲覧できるようになります。

-r

ジャーナルのメッセージを逆順で表示します。最新の項目が最初に表示されるようになります。

-k

カーネルメッセージのみを表示します。これは項目ベースのフィルタで、`_TRANSPORT=kernel` と同じ意味になります (詳しくは 11.3.3項「項目をベースにしたフィルタ」をお読みください) 。

-u

指定した `systemd` ユニットに対するメッセージのみを表示します。これは項目ベースのフィルタで、`_SYSTEMD_UNIT=ユニット名` と同じ意味になります (詳しくは 11.3.3項「項目をベースにしたフィルタ」をお読みください) 。

```
> sudo journalctl -u apache2
[...]
Jun 03 10:07:11 pinkiepie systemd[1]: Starting The Apache Webserver...
Jun 03 10:07:12 pinkiepie systemd[1]: Started The Apache Webserver.
```

11.3 ジャーナル出力のフィルタリング

何もスイッチを指定しないで `journalctl` を実行すると、ジャーナル内にある全ての情報を古いものから順に表示します。出力はスイッチと項目を指定することでフィルタすることができます。

11.3.1 起動番号をベースにしたフィルタ

`journalctl` では、特定のシステム起動時に限ってメッセージを表示することができます。全てのシステム起動を表示するには、下記のように実行します:

```
> sudo journalctl --list-boots
-1 097ed2cd99124a2391d2cffab1b566f0 Mon 2014-05-26 08:36:56 EDT—Fri 2014-05-30 05:33:44
 EDT
 0 156019a44a774a0bb0148a92df4af81b Fri 2014-05-30 05:34:09 EDT—Fri 2014-05-30 06:15:01
 EDT
```

一番左の列は起動オフセットと呼ばれる値で、`0` が現在の起動、`-1` が前回、`-2` が前々回のようになっています。2 つ目の列には起動 ID が書かれていて、その後ろにはそれぞれのタイムスタンプが書かれています。

現在の起動に限定して全てのメッセージを表示するには、下記のように実行します:

```
> sudo journalctl -b
```

以前の起動時のジャーナルメッセージを表示したい場合は、オフセット値を指定します。たとえば前回の起動時のメッセージを表示したい場合は、下記のように実行します:

```
> sudo journalctl -b -1
```

起動時を指定するもう 1 つの方法としては、起動 ID を指定する方法があります。この場合は _BOOT_ID フィールドでフィルタを設定します:

```
> sudo journalctl _BOOT_ID=156019a44a774a0bb0148a92df4af81b
```

11.3.2 時間範囲をベースにしたフィルタ

`journalctl` の出力は、開始と終了の日時を指定してフィルタすることもできます。日付を指定する場合は、`2014-06-30 9:17:16` のように指定します。また、時刻を省略した場合は `00:00:00` であるものと解釈され、時刻の秒を省略した場合は `:00` であるものと解釈されます。日付部分を省略した場合は、今日の日付であるものとして解釈されます。また、日付での表記の代替として、`yesterday` (昨日), `today` (今日), `tomorrow` (明日) のような表現を使用することもできます。これらを指定した場合、時刻は `00:00:00` と解釈されます。さらに、`now` と指定すると、現在の日時であるものとして解釈されます。このほか、`-` や `+` を頭に付けて相対日時を指定することもできます。

現在以降の新しいメッセージを表示し、以後更新があるたびに随時表示するには、下記のように実行します:

```
> sudo journalctl --since "now" -f
```

今日になってから午前 3:20 までの全てのメッセージを表示するには、下記のように実行します:

```
> sudo journalctl --since "today" --until "3:20"
```

11.3.3 項目をベースにしたフィルタ

ジャーナルのメッセージは、特定の項目でフィルタして出力することができます。項目を指定する場合は、`フィールド名=値` の形式 (例: `_SYSTEMD_UNIT=httpd.service`) で指定します。また、1 回の問い合わせで複数のフィルタを指定することもできます。既定で利用できる項目の一覧について、詳しくは `man 7 systemd.journal-fields` をお読みください。

特定のプロセス ID が生成したメッセージのみを表示するには、下記のように実行します:

```
> sudo journalctl _PID=1039
```

特定のユーザ ID に属するメッセージのみを表示するには、下記のように実行します:

```
# journalctl _UID=1000
```

カーネルのリングバッファからのメッセージのみを表示するには、下記のように実行します (`dmesg` コマンドが生成するものと同じ内容になります):

```
> sudo journalctl _TRANSPORT=kernel
```

サービスの標準出力や標準エラー出力のメッセージのみを表示するには、下記のように実行します:

```
> sudo journalctl _TRANSPORT=stdout
```

指定したサービスが生成したメッセージのみを表示するには、下記のように実行します:

```
> sudo journalctl _SYSTEMD_UNIT=avahi-daemon.service
```

別々の項目に対して複数の条件を指定した場合は、同時に両方の条件に合致したメッセージのみを表示します:

```
> sudo journalctl _SYSTEMD_UNIT=avahi-daemon.service _PID=1488
```

同じ項目に対して複数の条件を指定した場合は、いずれかの条件に合致したメッセージを表示します:

```
> sudo journalctl _SYSTEMD_UNIT=avahi-daemon.service _SYSTEMD_UNIT=dbus.service
```

なお、`+` のセパレータを指定することで、論理 `OR` を実現することができます。下記の例では、Avahi サービスプロセスのプロセス ID 1480 のメッセージと、D-Bus サービスのメッセージをあわせて表示します:

```
> sudo journalctl _SYSTEMD_UNIT=avahi-daemon.service _PID=1480 +
 _SYSTEMD_UNIT=dbus.service
```

11.4 systemd のエラー調査

本章では、`systemd` が `apache2` を起動する際に発生した問題を、調べて修正するまでの流れを説明しています。

1. apache2 を開始してみます:

   ```
   # systemctl start apache2
   Job for apache2.service failed. See 'systemctl status apache2' and 'journalctl -xn'
    for details.
   ```

2. サービスの状態はどのようになっているのかを調べてみます:

   ```
   > sudo systemctl status apache2
   apache2.service - The Apache Webserver
      Loaded: loaded (/usr/lib/systemd/system/apache2.service; disabled)
      Active: failed (Result: exit-code) since Tue 2014-06-03 11:08:13 CEST; 7min ago
     Process: 11026 ExecStop=/usr/sbin/start_apache2 -D SYSTEMD -DFOREGROUND \
              -k graceful-stop (code=exited, status=1/FAILURE)
   ```

   エラーを発生させているのはプロセス ID 11026 であることがわかります。

3. プロセス ID 11026 に関連するメッセージを詳細に表示してみます:

   ```
   > sudo journalctl -o verbose _PID=11026
   [...]
   MESSAGE=AH00526: Syntax error on line 6 of /etc/apache2/default-server.conf:
   [...]
   MESSAGE=Invalid command 'DocumenttRoot', perhaps misspelled or defined by a module
   [...]
   ```

4. `/etc/apache2/default-server.conf` ファイル内にスペルミスがあることがわかります。これを修正して apache2 サービスを開始しなおし、再度状態を確認してみます:

   ```
   > sudo systemctl start apache2 && systemctl status apache2
   apache2.service - The Apache Webserver
      Loaded: loaded (/usr/lib/systemd/system/apache2.service; disabled)
   ```

```
   Active: active (running) since Tue 2014-06-03 11:26:24 CEST; 4ms ago
  Process: 11026 ExecStop=/usr/sbin/start_apache2 -D SYSTEMD -DFOREGROUND
           -k graceful-stop (code=exited, status=1/FAILURE)
Main PID: 11263 (httpd2-prefork)
   Status: "Processing requests..."
   CGroup: /system.slice/apache2.service
           ├─11263 /usr/sbin/httpd2-prefork -f /etc/apache2/httpd.conf -D [...]
           ├─11280 /usr/sbin/httpd2-prefork -f /etc/apache2/httpd.conf -D [...]
           ├─11281 /usr/sbin/httpd2-prefork -f /etc/apache2/httpd.conf -D [...]
           ├─11282 /usr/sbin/httpd2-prefork -f /etc/apache2/httpd.conf -D [...]
           ├─11283 /usr/sbin/httpd2-prefork -f /etc/apache2/httpd.conf -D [...]
           └─11285 /usr/sbin/httpd2-prefork -f /etc/apache2/httpd.conf -D [...]
```

11.5 journald の設定

systemd-journald サービスの動作は、`/etc/systemd/journald.conf` を編集することで調整することができます。本章では基本的な設定のみを紹介しています。より詳しい設定ファイルの説明については、`man 5 journald.conf` をお読みください。また、設定ファイルを変更したあとは、下記のコマンドで systemd-journald を再起動する必要があります:

```
> sudo systemctl restart systemd-journald
```

11.5.1 ジャーナルのサイズ制限の変更

ジャーナルのログデータを恒久的な場所 (詳しくは 11.1項「ジャーナルの恒久化」をお読みください) に保存する場合、`/var/log/journal` が配置されているファイルシステムに対して、最大 10% までの領域を使用します。たとえば `/var/log/journal` が 30 GB の `/var` パーティション内に存在する場合、ジャーナルは最大で 3 GB までのディスク領域を使用します。この制限を変更したい場合は、`SystemMaxUse` オプションを変更し (およびコメントを外し) てください:

```
SystemMaxUse=50M
```

11.5.2 ジャーナルから /dev/ttyX への転送

ジャーナルは、`/dev/tty12` などの特定の端末デバイスに転送することもできます。下記の journald オプションを変更してください:

```
ForwardToConsole=yes
TTYPath=/dev/tty12
```

11.5.3 ジャーナルから syslog への転送

journald には、`rsyslog` をはじめとした従来の syslog 実装との後方互換性が用意されています。この機能を利用するにあたっては、あらかじめ下記をご確認ください:

- rsyslog がインストールされていること。

```
> sudo rpm -q rsyslog
rsyslog-7.4.8-2.16.x86_64
```

- rsyslog サービスが有効化されていること。

```
> sudo systemctl is-enabled rsyslog
enabled
```

- syslog への転送は `/etc/systemd/journald.conf` で設定します。

```
ForwardToSyslog=yes
```

11.6 YaST を利用した systemd ジャーナルのフィルタリング

systemd ジャーナルを journalctl の文法を使用せずに簡単にフィルタリングするには、YaST の journal モジュールを使用するとよいでしょう。`sudo zypper in yast2-journal` でインストールを行ったあと、YaST を起動して［その他］›［systemd ジャーナル］を選択します。それ以外にも、コマンドラインから `sudo yast2 journal` と入力して実行してもかまいません。

図 11.1: YAST SYSTEMD ジャーナル

このモジュールは、ログ項目を表形式で表示します。上部にある検索ボックスを使用すると、`grep` を使用したときのように、指定した文字列を含む項目のみを表示することができます。また、日時やユニット、ファイルや重大度でフィルタを行いたい場合は、［フィルタの変更］を押して必要な設定を行ってください。

11.7 GNOME でのログ表示

また、ジャーナルは GNOME Logs でも読むことができます。アプリケーションメニューから起動してお使いください。システムログメッセージを読むには root 権限が必要となりますので、`xdg-su gnome-logs` などのようにして起動してください。このコマンドは、`Alt`–`F2` を押して起動することもできます。

12 ブートローダ GRUB 2

改訂履歴

2025-05-07

本章では、openSUSE® Leap 内で使用されているブートローダである GRUB 2 について、その設定方法を説明しています。なお、YaST モジュールを利用すれば、ほとんどの重要な設定を行うことができます。起動処理そのものに関する説明は 第9章「起動処理の紹介」を、UEFI 環境における Secure Boot サポートについては 第14章「UEFI (Unified Extensible Firmware Interface)」をそれぞれお読みください。

12.1 GRUB Legacy と GRUB 2 との主な違い

- 設定ファイルのファイル名が変更されています。
- さらに多くのファイルシステム (例: btrfs など) に対応しています。
- LVM や RAID デバイスに保存されているファイルを直接読み込むことができます。
- ユーザインターフェイスを各国語に翻訳したり、テーマで置き換えたりすることができます。
- 追加のファイルシステムへの対応など、機能を追加するためのモジュールを読み込むことができるようになっています。
- 他のカーネルやオペレーティングシステム (例: Windows) を検索し、自動的にそれらの起動項目を作成する機能にも対応しています。
- 最小限の機能に絞った、bash に似たコンソールが用意されています。

12.2 設定ファイルの構造

GRUB 2 の設定は、それぞれ下記のファイルが基本となっています:

`/boot/grub2/grub.cfg`

このファイルには、GRUB 2 のメニュー項目に関する設定が記述されています。GRUB Legacy では `menu.lst` ファイルとして存在していたものですが、`grub.cfg` は `grub2-mkconfig -o /boot/grub2/grub.cfg` コマンドで自動生成されるため、編集すべきではありません。

`/boot/grub2/custom.cfg`

このファイルは存在していなくてもかまわないファイルで、システムの起動時に `grub.cfg` から参照され、起動メニューに独自の項目を追加するためのファイルです。openSUSE Leap 42.2 以降では、`grub2-once` を使用する際にもこれらの項目が処理されます。

`/etc/default/grub`

このファイルは、GRUB 2 のユーザ設定を制御するためのファイルで、通常は背景やテーマなどの追加の環境設定が含まれます。

`/etc/grub.d/` 以下のスクリプト

このディレクトリ内にあるスクリプトは、`grub2-mkconfig -o /boot/grub2/grub.cfg` を実行する際に読み込まれます。これらのスクリプトの実行結果は、メインの設定ファイルである `/boot/grub/grub.cfg` に組み込まれます。

`/etc/sysconfig/bootloader`

この設定ファイルには、ブートローダの種類や UEFI Secure Boot サポートの有効化可否など、基本的な設定が含まれています。

`/boot/grub2/x86_64-efi` ,

これらの設定ファイルには、アーキテクチャ固有のオプションが含まれています。

GRUB 2 は様々な方法で制御を行うことができます。既存のメニュー内にある起動項目の選択であれば、グラフィカルなメニュー (スプラッシュスクリーン) から行うことができますし、そのメニューを表示している設定ファイルは `/boot/grub2/grub.cfg` にあり、このファイルは他の設定ファイル類から生成される仕組みになっています (下記参照) 。なお、全ての GRUB 2 設定ファイルはシステムファイルですので、編集には `root` の権限が必要となります。

注記: 変更した設定の反映

GRUB 2 の設定ファイルを手作業で編集した場合は、それらを反映させるために `grub2-mkconfig -o /boot/grub2/grub.cfg` を実行する必要があります。ただし、YaST で設定を変更した場合は、YaST が `grub2-mkconfig` を呼び出しますので、後からこれを実行する必要はありません。

12.2.1 /boot/grub2/grub.cfg ファイル

起動メニュー付きのグラフィカルなスプラッシュスクリーンは、GRUB 2 の設定ファイル `/boot/grub2/grub.cfg` をベースにして作成されます。このファイルには、メニューから起動することのできる全てのパーティションやオペレーティングシステムに関する情報が含まれています。

システムを起動する場合、GRUB 2 はファイルシステムから設定ファイルを毎回読み込みます。このような構造から、GRUB 2 では設定ファイルを変更しても、再インストールを行う必要はありません。`grub.cfg` はカーネルを追加したり削除したりするたびに、自動的に再構築されます。

また、`grub.cfg` は `/etc/default/grub` ファイルと `/etc/grub.d/` ディレクトリ内にあるスクリプトを利用して、`grub2-mkconfig -o /boot/grub2/grub.cfg` コマンドで生成されます。そのため、上記のファイルを手作業で編集することはお勧めできません。その代わり、関連するソースファイルや YaST の［ブートローダ］モジュールを利用して、設定ファイルを変更することをお勧めします。YaST での設定方法について、詳しくは 12.3項「YaST によるブートローダの設定」をお読みください。

12.2.2 `/etc/default/grub` ファイル

このファイルには、GRUB 2 に関する一般的なオプションが用意されています。たとえばメニュー画面を表示する時間や起動する既定の OS などです。利用可能な全てのオプションを表示するには、下記のコマンドの出力をお使いください:

```
> grep "export GRUB_DEFAULT" -A50 /usr/sbin/grub2-mkconfig | grep GRUB_
```

なお、`/etc/grub.d` 内のスクリプトから参照するため、独自のユーザ変数を設定することもできます。`/etc/default/grub` を編集したあとは、`grub2-mkconfig -o /boot/grub2/grub.cfg` を実行してメインの設定ファイルを更新してください。

注記: 適用範囲について

このファイル内に書かれているオプションは、全ての起動項目に対して適用される汎用オプションです。Xen ハイパーバイザ向けの固有オプションを設定したい場合は、対応する `_XEN_` オプションのほうに記述してください。

重要: 引用符の処理について

スペース (空白) を含む複雑な文字列を指定したい場合は、引用符を利用して指定してください。また、引用符内はエスケープ処理を行う必要もあります。たとえば下記のようになります:

```
GRUB_CMDLINE_LINUX_XEN="debug loglevel=9 log_buf_len=5M \"ddebug_query=file drivers/
xen/xen-acpi-processor.c +p\""
```

`GRUB_DEFAULT`

既定で起動する起動メニューの項目を指定します。数値やメニュー項目の完全な名前で指定することができるほか、「saved」を指定することもできます。

たとえば `GRUB_DEFAULT=2` のように指定すると、3 番目の (数値の場合は 0 から数えます) 起動メニュー項目を起動します。
また、`GRUB_DEFAULT="2>0"` のように指定すると、メニュー項目のトップレベルの 3 番目にあるメニュー内の最初のサブメニュー項目を起動します。
そのほか、`GRUB_DEFAULT="Example boot menu entry"` のように指定すると、「Example boot menu entry」というタイトルの起動メニュー項目を起動します。
`GRUB_DEFAULT=saved` を指定すると、`grub2-once` や `grub2-set-default` で指定された項目を起動するようになります。`grub2-reboot` では次回の再起動に対してのみ既定の起動項目を設定するのに対して、`grub2-set-default` では明示的に変更するまでの間、既定の起動項目を維持し続けるようになります。なお、`grub2-editenv list` を実行すると、次回の起動項目を一覧表示することができます。

`GRUB_HIDDEN_TIMEOUT`

ユーザがキーを押すまで待機する時間を秒単位で指定します。この待機時間中は、何もメニューを表示せず、ユーザがキーを押して初めて表示する動作になります。指定した時間、何もキー入力が行われないと、制御は `GRUB_TIMEOUT` に渡ります。`GRUB_HIDDEN_TIMEOUT=0` を指定した場合は、まず `Shift` が押されているかどうかを確認し、押されていた場合はメニュー項目を表示し、押されていなかった場合は即時に既定のメニュー項目を起動するようになります。これは GRUB 2 で起動可能な OS が 1 つしか存在しない場合の、既定の動作でもあります。

`GRUB_HIDDEN_TIMEOUT_QUIET`

`false` を指定すると、`GRUB_HIDDEN_TIMEOUT` の動作中、画面にカウントダウンタイマーだけを表示します。

`GRUB_TIMEOUT`

既定の起動項目を自動的に起動するまでの、起動メニューを表示してからの時間を秒単位で指定します。何かキーが押されるとカウントダウンは停止され、GRUB 2 はいずれかの項目を選択するまで待ち続けるようになります。`GRUB_TIMEOUT=-1` を指定すると、いずれかの項目を選択するまで、ずっと待ち続けるようになります。

`GRUB_CMDLINE_LINUX`

この行に書かれた内容は、通常モードと修復モードの両起動項目の末尾に追加されます。起動項目にカーネルのパラメータを指定したい場合に使用します。

`GRUB_CMDLINE_LINUX_DEFAULT`

`GRUB_CMDLINE_LINUX` と同じような意味合いを持ちますが、こちらは通常モードの起動項目の末尾にのみ追加されます。

`GRUB_CMDLINE_LINUX_RECOVERY`

`GRUB_CMDLINE_LINUX` と同じような意味合いを持ちますが、こちらは修復モードの起動項目の末尾にのみ追加されます。

`GRUB_CMDLINE_LINUX_XEN_REPLACE`

この項目は `GRUB_CMDLINE_LINUX` を置き換えるための項目で、Xen の起動項目にのみ適用されます。

`GRUB_CMDLINE_LINUX_XEN_REPLACE_DEFAULT`

`GRUB_CMDLINE_LINUX_XEN_REPLACE` と同じような意味合いを持ちますが、こちらは `GRUB_CMDLINE_LINUX_DEFAULT` のパラメータを置き換えるためのものです。

`GRUB_CMDLINE_XEN`

この行に書かれた内容は、Xen ハイパーバイザの Xen メニュー項目のうち、通常モードと修復モードの両起動項目の末尾に追加されます。たとえば下記のようになります:

```
GRUB_CMDLINE_XEN="loglvl=all guest_loglvl=all"
```

ヒント: Xen ハイパーバイザのオプションについて

Xen ハイパーバイザに対して指定可能な全てのオプションを確認したい場合は、https://xenbits.xen.org/docs/unstable/misc/xen-command-line.html をお読みください。

`GRUB_CMDLINE_XEN_DEFAULT`

`GRUB_CMDLINE_XEN` と同じような意味合いを持ちますが、こちらは通常モードの起動項目の末尾にのみ追加されます。

`GRUB_TERMINAL`

入出力を行う端末デバイスを有効化し、指定するための項目です。`console` (PC BIOS および EFI コンソール), `serial` (シリアル端末), `ofconsole` (Open Firmware コンソール), `gfxterm` (グラフィックモード出力; 既定値) の中からいずれかを指定します。複数の端末を使用したい場合は、引用符で括って `GRUB_TERMINAL="console serial"` のように指定します。

`GRUB_GFXMODE`

`gfxterm` グラフィック端末で使用する解像度を指定します。ただし、指定可能な解像度はお使いのグラフィックカード (VBE) に依存します。既定値は 'auto' で、自動的に解像度を選択する動作を行います。なお、利用可能な解像度を一覧表示したい場合は、GRUB 2 のコマンドラインから `videoinfo` と入力して実行してください。コマンドラインは、GRUB 2 の起動画面が表示されている状態で、`c` を押すとアクセスすることができます。

解像度の設定に加えて、色数 (色深) も指定することができます。たとえば `GRUB_GFXMODE=1280x1024x24` のようになります。

`GRUB_BACKGROUND`

`gfxterm` グラフィカル端末で使用する背景画像を指定します。画像は GRUB 2 が起動時に読み込めるファイルでなければならず、`.png`、`.tga`、`.jpg`、`.jpeg` のいずれかで終わるファイル名でなければなりません。また、必要であれば画面全体に適合するように拡大または縮小されます。

`GRUB_DISABLE_OS_PROBER`

このオプションを `true` にすると、他のオペレーティングシステムの自動検出機能が無効化されます。`/boot/` 内にあるカーネルと、`/etc/grub.d/` 内にある独自のスクリプトからのオプションのみを検出します。

`SUSE_BTRFS_SNAPSHOT_BOOTING`

このオプションを `true` にすると、GRUB 2 が snapper のスナップショットから直接起動できるようになります。詳しくは 3.3項「スナップショットからの起動によるシステムのロールバック」をお読みください。

オプションの一覧については、GNU GRUB マニュアル (https://www.gnu.org/software/grub/manual/grub/grub.html#Simple-configuration) をお読みください。

12.2.3 `/etc/grub.d` 内にあるスクリプト

このディレクトリ内にあるスクリプトは、`grub2-mkconfig -o /boot/grub2/grub.cfg` コマンドを実行する際に読み込まれます。これらのスクリプトの実行結果は、メインの設定ファイルである `/boot/grub2/grub.cfg` に統合されます。`grub.cfg` でのメニュー項目内の順序は、このディレクトリ内の実行順になります。つまり、数字の小さいものが先に、大きいものが後に実行されることになります。たとえば `00_header` は、`10_linux` よりも前に実行されますし、`10_linux` は `40_custom` よりも前に実行されます。また、アルファベットで始まる名前のファイルが存在する場合は、数字で始まるファイルよりも後に実行されます。また、`grub2-mkconfig` の実行時には、実行可能なファイルのみが `grub.cfg` に出力を行います。既定では `/etc/grub.d` 内にある全てのファイルが実行可能ファイルになっています。

ヒント: grub.cfg 内のカスタマイズについて

`/boot/grub2/grub.cfg` は `grub2-mkconfig` を実行するたびに上書きされる仕組みであるため、独自のカスタマイズを行っても、`grub2-mkconfig` が実行されるタイミングで消えてしまいます。`/boot/grub2/grub.cfg` に直接書き込まなければならないカスタマイズがある場合は、そのカスタマイズ項目を下記の 2 つの行の間に記述してください:

```
### BEGIN /etc/grub.d/90_persistent ###
```

および

```
### END /etc/grub.d/90_persistent ###
```

これにより、`90_persistent` スクリプトが、その中のコンテンツを維持するようになります。

最も重要なスクリプトの一覧を下記に示します:

`00_header`

システムファイルの場所やディスプレイの設定、テーマや以前に保存されていた項目など、各種の環境変数を設定します。また、`/etc/default/grub` 内に書かれている設定も取り込みます。通常は、このファイルを編集する必要はありません。

`10_linux`

ルートデバイス内の Linux カーネルを判別して、必要なメニュー項目を生成します。これには修復モード用のメニューの生成なども含まれます。ただし、最新のカーネルのみがメインメニューに表示され、それ以外のカーネルはサブメニュー内に表示されるように設定されます。

`30_os-prober`

このスクリプトは `os-prober` を使用して、他の Linux や他のオペレーティングシステムを検出し、その結果を GRUB 2 メニューの形式で出力します。メニューの項目名には、それぞれ Windows や MacOS などの実際の名前が設定されます。

`40_custom`

このスクリプトは、`grub.cfg` 内に独自の起動項目を追加するためのシンプルな方法を提供します。ただし、スクリプトの冒頭にある `exec tail -n +3 $0` については、変更してはなりません。

処理順序は番号順で、小さい番号から大きい番号の順に行われます。同じ番号のスクリプトが存在した場合は、その後ろの文字のアルファベット順で実行されます。

ヒント: /boot/grub2/custom.cfg

`/boot/grub2/custom.cfg` ファイルを作成して独自の項目を設定すると、起動時にその内容が `/boot/grub2/grub.cfg` の `40_custom` の直後に取り込まれるようになります。

12.2.4 BIOS ドライブと Linux デバイスのマッピングについて

GRUB Legacy では、`device.map` という設定ファイルを利用して BIOS のドライブ番号から Linux デバイス名を紐づけていました。ただし、BIOS ドライブと Linux デバイス名のマッピングは、必ずしも正しいものであるとは言えませんでした。たとえば GRUB Legacy では、BIOS の設定内で IDE ドライブと SCSI ドライブの起動順序を入れ替えたりすると、順序を誤って設定してしまうようなことがありました。

GRUB 2 では、`grub.cfg` を生成する際、デバイス ID 文字列 (UUID) やファイルシステムラベルを利用することで、この問題を解決しています。GRUB 2 ユーティリティ側では、一時的なデバイスマップをその場で作成することで、特にディスク 1 台のみのシステムで正しく動作するようにしています。

しかしながら、GRUB 2 の自動的なデバイスマッピングの仕組みが期待通りにならない場合があります。そのような場合は、独自のマッピングファイルを `/boot/grub2/device.map` に作成してください。下記の例では、`ディスク 3` を起動ディスクとして使用しています。ただし、GRUB 2 のパーティション番号は `1` から始まることに注意してください (GRUB Legacy では `0` から始まっていました) 。

```
(hd0)  /dev/disk-by-id/ディスク 3 の ID
(hd1)  /dev/disk-by-id/ディスク 1 の ID
(hd2)  /dev/disk-by-id/ディスク 2 の ID
```

12.2.5 起動処理中でのメニュー項目の編集

誤った設定を書き込んでしまって起動ができなくなった場合や、システムの設定を一時的に試してみたいなどの場合には、メニュー項目を直接編集する機能が必要となる場合があります。これを行うには、下記の手順を実施してください。

1. グラフィカルな起動メニューが表示されたら、矢印キーを利用して編集したい項目を選択します。
2. `E` を押してテキストベースのエディタを開きます。
3. 矢印キーを利用して、編集したい行まで移動します。

図 12.1: GRUB 2 ブートエディタ

以下の 2 つの方法があります:

a. カーネルのパラメータを編集したい場合は、`linux` や `linuxefi` で始まる行の末尾に、スペース区切りで必要なパラメータを追加します。パラメータの完全な一覧については、https://ja.opensuse.org/Linuxrc をご覧ください。

b. カーネルのバージョンなどの情報を編集したい場合は、一般的なエディタのやり方で必要な項目を編集します。`<Tab>` を押すと、補完機能を利用することができます。

4. `F10` を押すと編集した内容でシステムを起動することができます。また、`Esc` を押すと編集内容を破棄し、GRUB 2 のメニューに戻ることができます。

この方法で変更した内容は、今回の起動処理に限って適用されるものであり、恒久的には保存されません。

重要: 起動処理中のキーボードレイアウトについて

起動時にはアメリカ英語のキーボードレイアウトのみを使用することができます。詳しくは『スタートアップ』、第4章「トラブルシューティング」、4.3項「インストールメディアからの起動が失敗する」、US キーボードレイアウト をご覧ください。

注記: インストールメディア内のブートローダについて

従来型の BIOS のシステムでは、インストールメディアのブートローダは GRUB Legacy になっています。起動パラメータを追加したい場合は、項目を選んでそのままパラメータを入力してください。ここで入力したパラメータは、インストール後のシステムでも利用できるよう、恒久的に保存されます。

12.2.6 起動パスワードの設定

オペレーティングシステムが起動する前であっても、GRUB 2 はファイルシステムにアクセスすることができます。root の権限のない利用者であっても、ブートローダにさえアクセスできてしまえば、システムの起動後はアクセスできないようなファイルにも、アクセスできてしまいます。このような種類のアクセスを防止したり、メニュー項目を自由に選択できる状況を阻止したりしたい場合は、起動パスワードを設定することをお勧めします。

重要: 起動時にパスワードを求める設定

この設定を行うと、起動時にパスワードの入力を求められるようになります。言い換えると、システムは自動では起動しなくなります。

下記の手順で起動パスワードを設定します。下記の手順以外にも、YaST (詳しくは［ブートローダをパスワードで保護する］をお読みください) でも設定することができます。

1. まずは `grub2-mkpasswd-pbkdf2` を利用して、パスワードを暗号化します:

```
> sudo grub2-mkpasswd-pbkdf2
パスワードを入力してください: ****
Reenter password: ****
PBKDF2 hash of your password is grub.pbkdf2.sha512.10000.9CA4611006FE96BC77A...
```

2. 出力された文字列を、`set superusers` コマンドと共に `/etc/grub.d/40_custom` 内に貼り付けます。

```
set superusers="root"
password_pbkdf2 root grub.pbkdf2.sha512.10000.9CA4611006FE96BC77A...
```

3. 内容をメインの設定ファイルに取り込むには、下記のように実行します:

```
> sudo grub2-mkconfig -o /boot/grub2/grub.cfg
```

システムを再起動すると、GRUB 2 は起動メニューの項目を選択する際にユーザ名とパスワードの入力を求めるようになります。それぞれユーザ名には `root` を、パスワードには `grub2-mkpasswd-pbkdf2` で入力したパスワードを入力してください。両方の項目が正しく入力されると、システムは選択した項目を起動するようになります。

詳しくは https://www.gnu.org/software/grub/manual/grub/grub.html#Security をお読みください。

12.2.7 起動メニュー項目に対するアクセス保護

GRUB 2 では、認可レベルを設定して起動メニュー項目へのアクセスを制限することができます。具体的には、複数のユーザアカウントとパスワードを設定して、それぞれ異なるメニュー項目へのアクセス許可を設定することができます。GRUB 2 でこのようなアクセス制限を設定するには、下記の手順を実施します:

1. まずは GRUB 2 内に設定したいユーザ名とパスワードを用意します。このとき、パスワードは 12.2.6項「起動パスワードの設定」で説明しているとおり、`grub2-mkpasswd-pbkdf2` を利用して暗号化しておきます。
2. `/etc/grub.d/10_linux` ファイルを削除します。これにより、既定の GRUB 2 メニュー項目が作成されないようにすることができます。
3. あとは `/boot/grub2/custom.cfg` ファイルを編集して、独自のメニュー項目を追加していきます。下記は例ですので、必要に応じて編集してお使いください:

```
set superusers=admin
password admin 管理者パスワード
password maintainer メンテナパスワード

menuentry 'Operational mode' {
  insmod ext2
  set root=hd0,1
  echo 'Loading Linux ...'
  linux /boot/vmlinuz root=/dev/vda1 $GRUB_CMDLINE_LINUX_DEFAULT $GRUB_CMDLINE_LINUX
 mode=operation
  echo 'Loading Initrd ...'
  initrd /boot/initrd
}

menuentry 'Maintenance mode' --users maintainer {
  insmod ext2
  set root=hd0,1
  echo 'Loading Linux ...'
  linux /boot/vmlinuz root=/dev/vda1 $GRUB_CMDLINE_LINUX_DEFAULT $GRUB_CMDLINE_LINUX
 mode=maintenance
```

```
  echo 'Loading Initrd ...'
  initrd /boot/initrd
}
```

4. 内容をメインの設定ファイルに取り込むには、下記のように実行します:

```
> sudo grub2-mkconfig -o /boot/grub2/grub.cfg
```

上記の例は下記のような構成です:

- GRUB 2 には 2 種類の項目［Operational mode］と［Maintenance mode］が存在しています。
- 何もユーザを指定しない場合、いずれのメニュー項目とも使用できますが、GRUB 2 のコマンドラインに入ることはできず、かつ既存のメニュー項目を編集することもできなくなります。
- `admin` ユーザを指定した場合、GRUB 2 のコマンドラインに入ることができるほか、既存のメニュー項目を編集することもできるようになります。
- `maintenance` ユーザを指定した場合は回復用のメニュー項目を選択できます。

12.3 YaST によるブートローダの設定

openSUSE Leap システムで、ブートローダに関する一般的な設定を行うにあたって、最も簡単なのは YaST を使用することです。YaST のブートローダモジュールを起動するには、［YaST コントロールセンター］から［システム］›［ブートローダ］を選択します。

［ブートコードのオプション］タブを利用して、種類や場所、高度なブートローダオプションの設定を行うことができます。また、GRUB 2 を標準モードで使用するか、EFI モードで使用するかを選択することもできます。

重要: EFI システムでは GRUB2-EFI が必要となる件について

EFI システムをお使いの場合は、GRUB2-EFI だけをご利用いただけます。それ以外を選択してしまうと、お使いのシステムを起動できなくなってしまいます。

重要: ブートローダの再インストール

ブートローダを再インストールするには、まず YaST 内でいったん設定を変更してから、元に戻します。たとえば GRUB2-EFI を再インストールしたい場合は、いったん［GRUB2］を選択してから、［GRUB2-EFI］に戻します。

上記の手順を実施しないと、ブートローダ全体が書き込まれません。

注記: 独自のブートローダ

一覧にあるもの以外のブートローダを使用したい場合は、[管理しない] を選択します。なお、この選択を行う前に、お使いのブートローダのドキュメンテーションをよくお読みください。

12.3.1 ブートローダの場所とブートコードのオプション

ブートローダの既定の配置場所はパーティションの設定に依存して決まり、マスターブートレコード (MBR) もしくは `/` パーティションのブートセクタのいずれかに配置します。ブートローダの配置場所を変更するには、下記の手順を実施します:

手順 12.1: ブートローダの配置場所の変更

1. [ブートコードのオプション] を選択し、[ブートローダの場所] で下記のいずれかを選択します:

   [マスターブートレコード (MBR) から起動]
   : これを選択すると、`/boot` を含むディスクの MBR 内にブートローダをインストールします。通常は `/` と同じディスクになることが多いはずですが、`/boot` と `/` とがを別々のディスクにインストールされている場合、`/boot` を含むディスクにインストールします。

   [ルートパーティションから起動]
   : これを選択すると、`/` パーティションのブートセクタ内にブートローダをインストールします。

   [独自の起動パーティション]
   : ブートローダの配置先を独自に指定したい場合は、こちらに設定を行います。

2. [OK] を押すと変更を適用することができます。

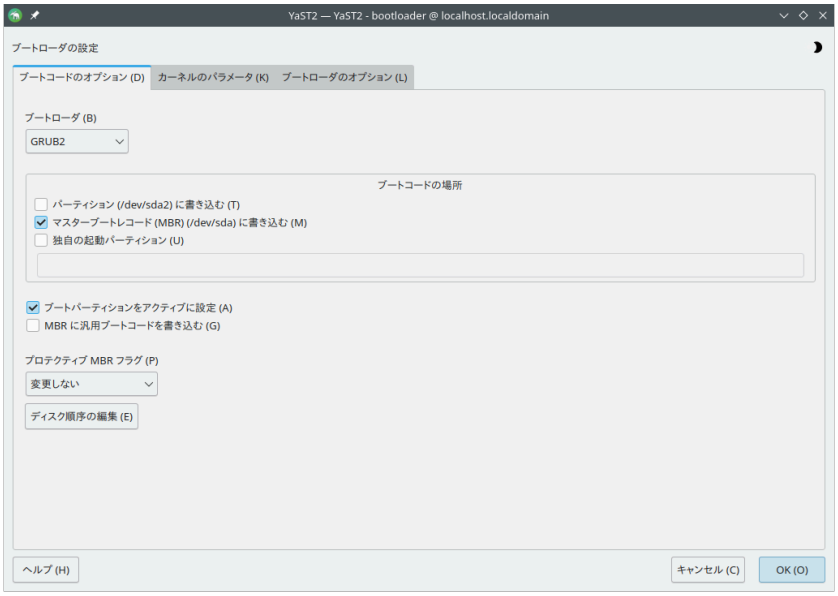


図 12.2: ブートコードのオプション

[ブートコードのオプション] タブには、下記の追加のオプションが用意されています:

[ブートパーティションをアクティブに設定]

`/boot` ディレクトリを含むパーティションをアクティブ化します。POWER システムの場合は、PReP パーティションをアクティブ化します。このオプションは、古い BIOS や古いオペレーティングシステムをお使いの場合で、アクティブに設定されたパーティションでないと、起動が失敗するような場合に設定します。ただし、このオプションは有効にしたままで問題ありません。

[MBR に汎用ブートコードを書き込む]

MBR 内に「GRUB 以外の」コードが含まれている場合は、このオプションを選択することで、オペレーティングシステムに依存しない汎用のコードに置き換えることができます。このオプションの選択を外してしまうと、システムが起動できなくなってしまう場合があります。

[Trusted Boot サポートを有効にする]

Trusted Computing 機能 (Trusted Platform Module (TPM)) に対応する TrustedGRUB2 を開始します。詳しくは https://github.com/Sirrix-AG/TrustedGRUB2 をお読みください。

[プロテクティブ MBR フラグ] セクションでは、下記のいずれかを選択することができます:

[設定]

従来の BIOS による起動の場合に適切な選択です。

［削除］

UEFI による起動の場合に適切な選択です。

［変更しない］

既に動作しているシステムの場合、こちらを選択するのが最適です。

ほとんどの場合において、YaST 側であらかじめ設定された選択が適切なものとなります。

12.3.2 ディスクの順序の調整

お使いのコンピュータに複数のディスクが接続されている場合、ここでディスクの起動順序を指定することができます。MBR から起動する場合、一覧内にある最初のディスクが GRUB 2 のインストール先となります。また、既定で openSUSE Leap がインストールされるディスクでもあります。一覧内の残りの項目は、GRUB 2 のデバイスマッパーに対するヒント情報になっています (詳しくは 12.2.4項「BIOS ドライブと Linux デバイスのマッピングについて」をお読みください) 。

警告: 起動不可能なシステムになってしまう問題について

ほほ全てのシステムにおいて、既定値のままご利用いただくのが通常です。ディスクの順序を誤って変更してしまうと、次回の起動時に起動ができなくなってしまいます。たとえば一覧内の最初のディスクが BIOS の起動順序内に含まれていないと、一覧内の他のディスクにはブートコードが書き込まれていないので、起動できなくなってしまいます。

手順 12.2: ディスク順序の変更

1. ［ブートコードのオプション］タブを選択します。
2. ［ディスク順序の編集］を押します。
3. 複数のディスクが一覧内にある場合は、ディスクを選択して［上へ］または［下へ］を押して、ディスクの順序を変更してください。
4. ［OK］を 2 回押して、変更を保存します。

12.3.3 高度なオプションの変更

高度なオプションを設定するには、［ブートローダのオプション］タブを選択します。

12.3.3.1 ［ブートローダのオプション］タブ

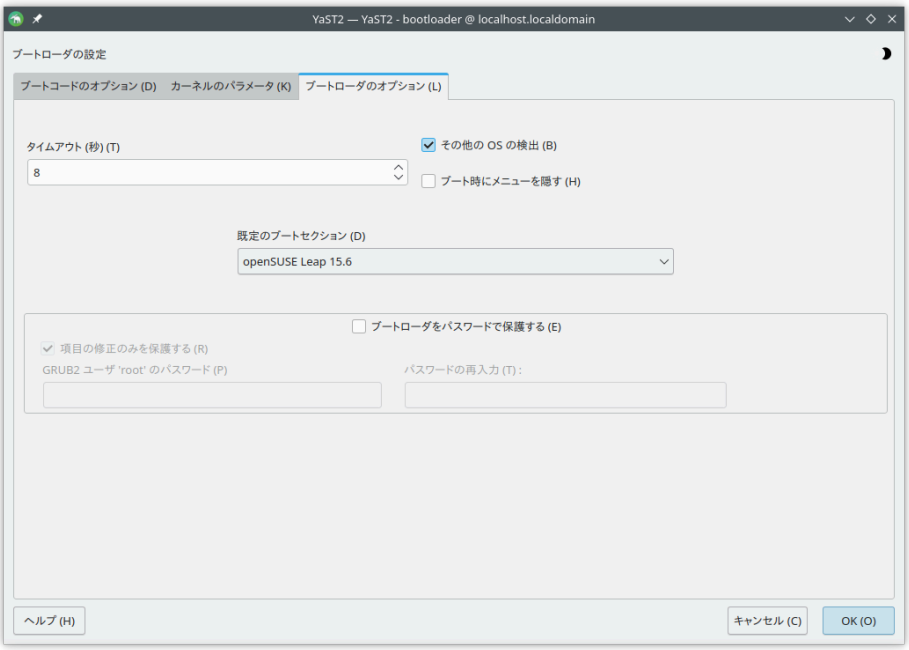


図 12.3: ブートローダのオプション

［タイムアウト］

［タイムアウト (秒)］の項目で設定したい値を入力するか、もしくは右にある上下のボタンを利用して、タイムアウトを変更します。

［その他の OS の検出］

これを選択すると、ブートローダは Windows や他の Linux ディストリビューションなどを検索するようになります。

［ブート時にメニューを隠す］

起動メニューを隠して、既定の項目を起動するようにします。

［既定のブートセクション］

「既定のブートセクション」の一覧内から必要な項目を選択します。なお、「>」マークは起動項目の区切り文字で、メニューとサブメニューの区切りを示しています。

［ブートローダをパスワードで保護する］

ブートローダを保護し、追加のパスワード設定を行うための仕組みです。手作業での設定方法については 12.2.6項「起動パスワードの設定」をお読みください。このオプションを選択した場合、起動する際に毎回パスワードの入力を求められることになります。言い換えると、システムの

起動を自動では行うことができなくなります。ただし、GRUB 1 をお使いの場合にのみ、［項目の修正のみを保護する］を選択することができます。こちらは既存の起動項目を修正する際にのみ、GRUB 2 の管理者パスワードの入力を求められることになります。

12.3.3.2 ［カーネルのパラメータ］タブ

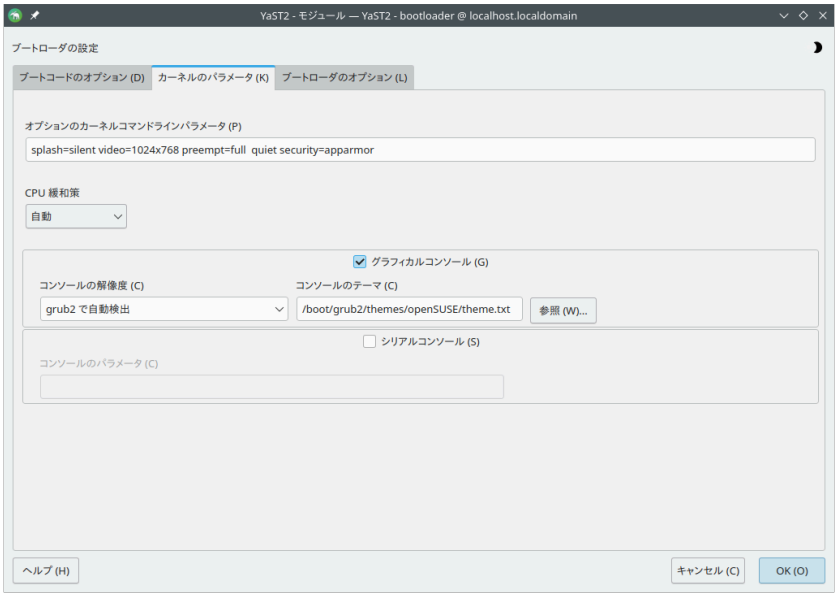


図 12.4: カーネルのパラメータ

［オプションのカーネルコマンドラインパラメータ］

システムの機能を有効化したり無効化したり、ドライバを追加したりする際に使用する、オプションのカーネルパラメータを指定します。

［CPU 緩和策］

SUSE では、CPU に対するサイドチャネル攻撃を防ぐため、お使いの CPU に適用することのできる全ての CPU 緩和策をまとめて設定する機能を提供しています。なお、選択した項目によっては、性能面の悪化を引き起こす可能性があります。下記の選択肢の中から、セキュリティと性能の要件に適合するものを選択してください:

［自動］: お使いの CPU モデルに応じた緩和策を全て有効化しますが、CPU を跨いだスレッド 攻撃については保護を行いません。この設定では、ご利用の負荷状況に応じて、性能面に幾分かの影響が発生します。

［自動 + SMT 無し］: 利用可能な全ての緩和策を全て適用します。お使いの CPU モデルに応じた全ての 緩和策を適用し、かつ同時マルチスレッディング Simultaneous Multithreading

(SMT) も無効化し、CPU スレッドを跨いだサイドチャネル攻撃を無効化します。ご利用の負荷状況にもよりますが、さらに性能面への影響が高くなります。

[無し]: 全ての緩和策を無効化します。お使いの CPU モデルに応じて、CPU に対するサイドチャネル攻撃が成立することになります。この設定では、性能面への影響はありません。

[手動]: 緩和策のレベルを指定しません。カーネルのコマンドラインオプションを利用して、手作業で緩和策を個別に指定することができるようになります。

[グラフィカルコンソールを使用する]

この項目にチェックを入れると、起動メニューがテキストモードではなく、グラフィカルなスプラッシュスクリーンとして表示されるようになります。この画面での解像度は既定では自動検出されますが、必要であれば [コンソールの解像度] で選択することもできます。また、テーマは [コンソールのテーマ] でファイル選択を行って設定します。テーマについては、独自のテーマファイルをお持ちの場合のみ変更してください。

[シリアルコンソールを使用する]

お使いのマシンがシリアルコンソールで制御するシステムである場合は、このオプションを選択して使用する COM ポートと速度を設定してください。詳しくは `info grub` または https://www.gnu.org/software/grub/manual/grub.html#Serial-terminal をお読みください。

12.4 便利な GRUB 2 コマンド

`grub2-mkconfig`

`/etc/default/grub` と `/etc/grub.d/` にあるスクリプトからの結果をもとにして、新しい `/boot/grub2/grub.cfg` を生成します。

例 12.1: GRUB2-MKCONFIG の使用方法

```
grub2-mkconfig -o /boot/grub2/grub.cfg
```

ヒント: 文法チェック

`grub2-mkconfig` を何もパラメータを付けずに実行すると、生成された設定が標準出力 (STDOUT) に出力されますので、ここから想定通りに出力されているかどうかを確認することができます。また、`/boot/grub2/grub.cfg` に出力した後でも、`grub2-script-check` を利用して文法をチェックすることができます。

重要: grub2-mkconfig では UEFI の Secure Boot テーブルを修復できない問題について

UEFI Secure Boot を使用していて、お使いのシステムが GRUB 2 にまで正しく到達できない場合、shim の再インストールと UEFI 起動テーブルの再生成が必要となります。具体的には、下記のように実行します:

```
# shim-install --config-file=/boot/grub2/grub.cfg
```

`grub2-mkrescue`

インストール済みの GRUB 2 設定に対して、起動可能なレスキューイメージを作成します。

例 12.2: GRUB2-MKRESCUE の使用方法

```
grub2-mkrescue -o save_path/name.iso iso
```

`grub2-script-check`

指定したファイルに文法上のエラーがないかどうかをチェックします。

例 12.3: GRUB2-SCRIPT-CHECK の使用方法

```
grub2-script-check /boot/grub2/grub.cfg
```

`grub2-once`

次回の起動時のみに適用する規定の起動項目を設定します。起動項目の一覧を取得したい場合は、`--list` オプションをお使いください。

例 12.4: GRUB2-ONCE の使用方法

```
grub2-once 番号
```

ヒント: grub2-once のヘルプ

何もオプションを指定しないでプログラムを実行すると、利用可能なオプションの一覧を表示することができます。

12.5 レスキューモード

レスキューモード とは `root` 専用のユーザセッションで、システムの起動が失敗してしまうような場合に、その調査と修復を行うモードです。このモードはシングルユーザモードで、ローカルのファイルシステムと一部のシステムサービスのみが起動され、ネットワークインターフェイスは有効化されません。レスキューモードを開始したい場合は、下記の手順を実施してください。

手順 12.3: レスキューモードへの突入

1. まずはシステムを再起動します。GRUB 2 の起動メニューが表示されるまで待ちます。
2. GRUB 2 の起動メニューが表示されたら `e` を押し、起動コマンドラインの編集モードに入ります。
3. カーネルパラメータを含む行に、下記のような内容を追記します:

   ```
   systemd.unit=rescue.target
   ```

4. あとは `Ctrl` + `X` を押して、システムの起動を行います。
5. あとは `root` のパスワードを入力します。
6. 必要な調査や修復作業を実施します。
7. 作業が終わったら `systemctl isolate multi-user.target` または `systemctl isolate graphical.target` を実行することで、通常の起動モードに戻すことができます。

12.6 さらなる情報

GRUB 2 に関する広い範囲の情報は、https://www.gnu.org/software/grub/ で提供されています。また、`grub` の info ページにも、様々な記載があります。

13 ネットワークの基礎

改訂履歴

2024-06-21

Linux には、全ての種類のネットワーク構成で利用することのできる、様々なネットワークツールやネットワーク機能が用意されています。ネットワークカードを利用したアクセス設定は YaST で行うことができます。もちろん手作業での設定も可能です。本章では、ごく基本的な構造と、関連するネットワーク設定ファイルについて説明しています。

Linux や他の Unix オペレーティングシステムでは、TCP/IP プロトコルを使用します。これは単一のネットワークプロトコルを指す用語ではなく、様々なサービスを提供するネットワークプロトコルファミリを指す用語です。TCP/IP プロトコルファミリで使用される主なプロトコル に示されているプロトコルの一覧は、TCP/IP を介して 2 台のマシン間でデータを交換するために提供されているものです。また、TCP/IP で構成されたネットワークを世界中に広げたものを、「インターネット」と呼びます。

RFC とは Request for Comments の略で、インターネットプロトコルの説明やオペレーティングシステム内での実装手順のほか、応用例などを説明している文書です。言い換えると、RFC の文書はインターネットプロトコルの構造と仕組みを説明しているものになります。RFC についての詳細は https://datatracker.ietf.org/ (英語) をお読みください。

TCP/IP プロトコルファミリで使用される主なプロトコル

TCP

伝送制御プロトコル (Transmission Control Protocol) と呼ばれるプロトコルで、接続指向のプロトコルです。アプリケーションからもたらされた送信データは、一連のデータとしてオペレーティングシステム側で書式変換が行われます。宛先のホストにその変換済みのデータが到着すると、逆変換を行って元のデータに戻して、受信側のアプリケーションに配信されます。TCP では途中のデータが喪失してしまったり、順序が入れ替わってしまったりした場合でも、それを修復して元通りの構成に戻す機能を備えています。つまり、TCP は順序が重要となるデータを送信する際に便利な仕組みです。

UDP

ユーザデータグラムプロトコル (User Datagram Protocol) と呼ばれるプロトコルで、接続制御のないプロトコルです。アプリケーションからもたらされた送信データは、書式変換のみを行って送信されます。受信側では逆変換を行いますが、受信側での到着順序については保証が無く、場合によってはデータを喪失してしまうこともあります。UDP はひとかたまり (パケット) のデータを送信する際に便利なプロトコルで、TCP よりも遅延を少なくすることができます。

ICMP

インターネット制御メッセージプロトコル (Internet Control Message Protocol) と呼ばれるプロトコルで、エンドユーザ側で使用するプロトコルではなく、ネットワーク内でのエラー報告を行ったり、TCP/IP データ伝送に参加している機器を制御したりするためのプロトコルです。これに加えて、ping というプログラムを利用してネットワークの通信が成立しているかどうかを調べるプロトコルも用意されています。

IGMP

インターネットグループ管理プロトコル (Internet Group Management Protocol) と呼ばれるプロトコルで、IP マルチキャストと呼ばれる仕組みを実装する際、それに参加するマシンの動作を制御するためのプロトコルです。

図13.1「TCP/IP での単純化したレイヤモデル」に示されているとおり、コンピュータ間でのデータ交換は階層 (レイヤ) 型の構造で処理されます。ネットワーク層は実際には IP (インターネットプロトコル) を介した通信で、IP の上にトランスポート層の TCP (伝送制御プロトコル) が乗り、順序保証やある程度のセキュリティが確保されたデータ転送を実現しています。また、IP 層はその下にあるハードウエア依存のプロトコル (例: イーサネット) 上で動作しています。

図 13.1: TCP/IP での単純化したレイヤモデル

上の図では、各レイヤに対して 1 つもしくは 2 つの例を示しています。レイヤは 抽象レベル とも呼べる仕組みで、低いレイヤほどハードウエアに近く、高いレイヤほどハードウエアからは離れた独立した仕組みになります。また、それぞれのレイヤには、名前からおおよそ推測のできる独自の機能が用意されています。なお、データリンク層と物理層は、イーサネットなどの物理的なネットワークを表しています。

ほぼ全てのハードウエアプロトコルはパケット指向の仕組みで動作します。通常は送信すべきデータを 1 回では送信できないため、パケット と呼ばれる単位で小分けして送信します。TCP/IP の場合、パケットの最大サイズはおおよそ 64 KB 程度になりますが、ネットワークハードウエア側ではそれより小さい単位でしか送信できないことから、実際にはもっと小さくなります。たとえばイーサネットの場合、これは 1500 バイト程度のもので、イーサネット経由で TCP/IP パケットを送信する場合も、この単位で送信することになります。

それぞれのレイヤがその機能を提供するため、データパケット内にはそれぞれのレイヤが必要とする情報を保存しなければなりません。この情報は、パケット内では プロトコルヘッダ と呼ばれ、この中に情報を保存します。プロトコルヘッダは、名前の通り実際のデータよりも前に置かれますので、たとえば

イーサネットケーブル上を通過する TCP/IP データパケットの場合は、図13.2「TCP/IP イーサネットパケット」で示す図のようになります。なお、チェックサム (checksum) のみが冒頭ではなく、末尾に付いています。このような仕組みにより、ネットワークハードウエアでの処理を容易にしています。

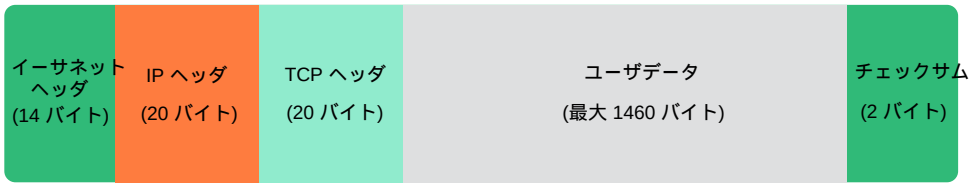


図 13.2: TCP/IP イーサネットパケット

アプリケーションがネットワークを介してデータを送信する際、データはそれぞれのレイヤを通過します。物理レイヤを除き、全てのレイヤが Linux カーネルで実装されています。それぞれのレイヤはデータを準備して次のレイヤに送信する仕組みになっていて、最も下のレイヤがデータ送信に対する最終的な責任を負う構造になっています。データを受信した場合は、この逆順で処理が行われます。タマネギの皮のように、受信したデータからそれぞれのプロトコルヘッダを削除していって、最終的にはトランスポート層がアプリケーションにデータを提供します。このような構造により、それぞれのレイヤは隣り合ったレイヤとだけやりとりを行えば十分な仕組みになっていますので、アプリケーション側では有線なのか無線なのかを気にせずに通信できるようになっています。同様に、パケットが正しい形式に変換されている限り、データ回線側もデータの種類を気にする必要はありません。

13.1 IP アドレスとルーティング

本章では、IPv4 ネットワークに限定して説明しています。IPv4 プロトコルの後継である IPv6 プロトコルについては、13.2項「IPv6—次世代インターネット」をお読みください。

13.1.1 IP アドレス

インターネットに接続されているコンピュータには、他のコンピュータと重複しない 32 ビットのアドレスが付与されています。この 32 ビット (4 バイト) は通常、例13.1「IP アドレスの書き方」の 2 行目のような書式で記述します。

例 13.1: IP アドレスの書き方

```
IP アドレス (バイナリ):  11000000 10101000 00000000 00010100
IP アドレス (10 進数) :      192.     168.       0.       20
```

10 進数の表示では、4 バイトそれぞれを 10 進数で表し、ピリオドでそれぞれを区切ります。IP アドレスはホストやネットワークインターフェイスに割り当てるもので、世界中を通して唯一のものでなければなりません。一部例外もありますが、本章の説明では省略します。

IP アドレスは階層構造型の仕組みで作られています。1990 年代までは、IP アドレスに「クラス」と呼ばれる分類が設定され、厳格に守られてきました。しかしながら、このような構造は柔軟性に欠けるため、現在は使用されていません。現在は クラスレスルーティング (CIDR, Classless InterDomain Routing) と呼ばれる仕組みで、クラスを使用しないようになっています。

13.1.2 ネットマスクとルーティング

ネットマスクとは、サブネットのアドレス範囲を定義するために使用するものです。2 つのアドレスが同じサブネット内にある場合、それらは直接通信することができます。同じサブネット内に無い場合は、そのサブネットに到達するまでの中継器 (ゲートウエイ) のアドレスを介して、通信を行う必要があります。2 つの IP アドレスが同じサブネット内にあるかどうかを確認するには、それぞれのアドレスをネットマスクで「論理積 (AND)」して、結果が同じであるかどうかを調べます。結果が異なる場合、双方が通信する際にはゲートウエイを介して通信する必要があることになります。

ネットマスクの仕組みについて知るには、まず 例13.2「IP アドレスとネットマスクのつながり」 をご覧ください。ネットマスクは 32 ビットの値で、IP アドレスのどの部分までがサブネットであるのかを示しています。サブネット側に属する部分が `1` に、それ以外の部分が `0` になっていますので、論理積 (AND) を取れば、ネットワーク部分だけを抽出できることになります。言い換えると、より小さなサブネットほど `1` の数が多いことになります。また、ネットマスクは `1` が連続して現れることから、`1` の長さでネットマスクを表すこともできます。たとえば 例13.2「IP アドレスとネットマスクのつながり」 では 24 個の `1` になっていますので、前者を `192.168.0.0/24` と記述することができます。

例 13.2: IP アドレスとネットマスクのつながり

```
IP アドレス (192.168.0.20) : 11000000 10101000 00000000 00010100
ネットマスク(255.255.255.0): 11111111 11111111 11111111 00000000
------------------------------------------------------------------
論理積の結果:                11000000 10101000 00000000 00000000
10 進数への変換結果:             192.      168.        0.        0

IP アドレス (213.95.15.200): 11010101 10111111 00001111 11001000
ネットマスク(255.255.255.0): 11111111 11111111 11111111 00000000
------------------------------------------------------------------
論理積の結果:                11010101 10111111 00001111 00000000
10 進数への変換結果:             213.       95.       15.        0
```

もう 1 つの例を挙げましょう。同じ LAN 内に接続されているコンピュータの場合、それら全ては同じサブネット内に存在していて、直接アクセスすることができます。スイッチやブリッジで物理的に区切られている場合でも、これらのホスト同士は直接アクセスすることができます。

逆に、異なるサブネットの IP アドレス同士は、ゲートウエイの介在無しにはアクセスすることができません。最もよくある環境では、ゲートウエイは 1 台だけで、そのゲートウエイが全てのサブネットに対するアクセスを中継します。環境によっては、複数のゲートウエイが存在していたりすることもあります。

このように、お使いのネットワーク内にゲートウエイが存在する場合は、異なるサブネット宛の通信は全てゲートウエイを介して接続します。ゲートウエイはホスト間で通信を行うのと同じ手順で、一方から他方にデータ (パケット) を中継します。ただし、各パケットには TTL (Time To Live) という値が設定されていて、ゲートウエイを通過するごとに 1 つずつ減らされ、0 になると廃棄されてしまいます。

様々なアドレス

ネットワークアドレス

このアドレスは、コンピュータの IP アドレスとネットマスクの論理積を取った結果で、例13.2「IP アドレスとネットマスクのつながり」の図では **論理積の結果** として示しているものです。このアドレスは、コンピュータに割り当てることができません。

ブロードキャストアドレス

このアドレスは、「そのサブネット内にある全てのホスト」を意味するアドレスです。このアドレスを生成するには、ネットマスクの `1` と `0` を入れ替えて、IP アドレスとの論理和 (OR) を取った値になります。上記の例では、192.168.0.255 となります。このアドレスも、コンピュータに割り当てることができません。

ローカルホスト

`127.0.0.1` というアドレスは特殊扱いのアドレスで、各ホストの 「ループバックデバイス」に割り当てます。このアドレスを指定することで、自分自身のコンピュータにアクセスすることができるようになっています。より正確に表現すると、このアドレスは `127.0.0.0/8` というネットワークアドレスで、IPv4 の規格として決められているものです。なお、IPv6 の場合は、ループバックアドレスは 1 つだけ (`::1`) になっています。

IP アドレスは全世界で唯一のものでなければならないことから、適当にアドレスを割り当てて良いというものではありません。ただし、内部で使用するためだけの「プライベート IP アドレス」と呼ばれるアドレスがあり、インターネットには直接接続しない環境下で使用することができるようになっています。これらのアドレス帯域は RFC 1597 で規定されているもので、表13.1「プライベート IP アドレスの範囲」に一覧を示しています。

表 13.1: プライベート IP アドレスの範囲

ネットワーク／ネットマスク	範囲
`10.0.0.0` / `255.0.0.0`	`10.x.x.x`
`172.16.0.0` / `255.240.0.0`	`172.16.x.x` – `172.31.x.x`

ネットワーク／ネットマスク	範囲
`192.168.0.0` / `255.255.0.0`	`192.168.x.x`

13.2 IPv6一次世代インターネット

World Wide Web (WWW) の普及に伴い、インターネットは爆発的に拡大しています。それに伴い、過去 15 年の間に TCP/IP を介して通信するコンピュータも、爆発的な数に膨れあがってきています。CERN (https://public.web.cern.ch) の Tim Berners-Lee 氏が 1990 年に WWW を発明して以降、数千台規模のインターネットホストが、今は 1 億台以上にもなってきています。

上述のとおり、IPv4 アドレスは 32 ビットしかありません。また、ネットワークの管理上の都合から、いくつかの IP アドレスが使用できなくなってしまっています。また上述のとおり、サブネットは 2 進数で表記するため、2 のべき乗 - 2 個 (ネットワークアドレスとブロードキャストアドレス) のアドレスしか利用できません。たとえば 128 台のコンピュータをインターネットに接続する場合、256 個のアドレスのサブネット (実際には 256 - 2 = 254 個のアドレスが利用可能) を利用する必要があります。

なお、現在の IPv4 プロトコルには、DHCP や NAT (ネットワークアドレス変換 (Network Address Translation)) の仕組みがあり、アドレスが枯渇する問題を回避しうる仕組みを備えています。また、上述のプライベートアドレスを併用して、それぞれを個別に管理することでも、確実に枯渇を防いでいます。なお、IPv4 のネットワーク内でホストを構築するには、そのホストに割り当てるアドレスとサブネットマスク、ゲートウエイのアドレスとネームサーバのアドレス (必要であれば) をそれぞれ設定することになります。これら全ての設定はそれぞれ異なるものであるため、あらかじめ知っておく必要があります。

IPv6 を利用することで、アドレスの枯渇と複雑化した設定を過去のものにすることができます。下記の章では、IPv6 で改善された内容や新しい内容を紹介しているほか、IPv4 からの移行方法についても説明しています。

13.2.1 利点

IPv6 プロトコルにおいて最も重要でかつ目立つ改善点として、アドレス領域の巨大化があります。IPv4 では 32 ビットしかなかったアドレスが、IPv6 では 128 ビットの巨大なアドレス領域になっています。これにより、数千兆個ものアドレスを割り当てることができるようになります。

しかしながら、IPv6 アドレスはアドレス領域を拡張しただけではありません。内部構造を新しくすることで、システムとネットワークに関する情報をより効率的に処理できるようになっています。詳しくは 13.2.2項「アドレスの種類と構造」をお読みください。

IPv6 プロトコルにおける利点の一覧は下記のとおりです:

自動設定

IPv6 ではネットワークを「プラグ&プレイ」対応に進化させています。これにより、新しいコンピュータを (ローカルの) ネットワークに接続する際、手作業での設定作業を全く必要としなくなっています。コンピュータを新しく接続すると、自動設定の仕組みが動作し、ルータが提供する情報をもとにして独自のアドレスを割り当てます。これは neighbor discovery (ND) プロトコルと呼ばれる仕組みです。この方法では、管理者側で行う作業は何もなく、IPv4 で自動的なアドレス割り当てを実現していた DHCP サーバのように、アドレスを割り当てるための中央サーバも不要です。

ルータがスイッチに接続されている場合でも、ルータはネットワークに接続するための情報を定期的に発信し続けます。詳しくは RFC 2462 と `radvd.conf(5)` のマニュアルページ、そして RFC 3315 をお読みください。

モビリティ

IPv6 では、1 つのネットワークインターフェイスに対して複数のアドレスを割り当てることができます。この仕組みにより、複数のネットワークに簡単に接続することができるため、携帯電話の事業者が提供する国際ローミングのように使用することができます。たとえばお使いの携帯電話を海外に持ち出した場合、その電話機は自動的にその地域の携帯電話サービスに接続しますが、どの国でも同じ番号のまま発着信できて、国内にいる場合と全く同じ使い心地で使用することができます。これと同じような仕組みを利用することができます。

セキュリティ

IPv4 ではネットワークセキュリティ (IPsec) がオプション扱いでしたが、IPv6 では IPsec が中枢機能の 1 つとなり、盗聴の可能性があり得るインターネットの世界で、機密の守られた通信を行うことができるようになっています。

後方互換性

現実的には、インターネット全体を一気に IPv4 から IPv6 に移行することは不可能です。そのため、インターネット内だけでなく、それぞれのシステムにも両方のプロトコルを共存させる仕組みが必須となります。このような要件から、互換アドレス (IPv4 アドレスを容易に IPv6 アドレスに変換できるアドレス) が用意され、トンネルを介してアクセスできる仕組みになっています。詳しくは 13.2.3項「IPv4 と IPv6 の共存」をお読みください。システム側では デュアルスタック IP と呼ばれる技術により、両方のプロトコルに対応し、どちらのプロトコルを経由してもインターネットにアクセスできる構造になっています。

マルチキャストを利用した独自サービス

IPv4 では、SMB などのサービスでブロードキャストを利用し、ローカルネットワーク内の全てのホストにパケットを送信していました。IPv6 では、より精密なアプローチである マルチキャスト を利用して、特定のグループ内の全体に情報を伝えることができるようになっています。これは全て

のホストに通知するブロードキャストとも、特定の 1 台のホストのみに通信するユニキャストとも異なる仕組みです。どのホストがグループとして割り当てられるのかは、それぞれの用途によって異なります。あらかじめ決められたグループとしては、全てのネームサーバ (全ネームサーバマルチキャストグループ) や全てのルータ (全ルータマルチキャストグループ) などがあります。

13.2.2 アドレスの種類と構造

上述のとおり、現在の IP プロトコルには 2 種類の大きな制限があります。IP アドレスの枯渇と、それに伴って発生する経路制御 (ルーティング) テーブルの複雑化、そしてその運用の煩雑さです。IPv6 では、アドレス領域を拡張することによって前者を解決していますし、後者についてはネットワークアドレスを構造化し、より整理された管理を行うと共に、マルチホーミング (複数のアドレスを 1 台のデバイスに設定し、複数のネットワークそれぞれにアクセスできる機能) でそれらを解決しています。

IPv6 を扱うにあたっては、まずアドレスの種類について知っておくことをお勧めします:

ユニキャスト

この種類のアドレスは、厳密に 1 つのネットワークインターフェイスに関連づけられるものです。このアドレス宛に発信されたパケットは、特定の 1 台に宛てて送信されます。そのため、ユニキャストのアドレスは、ネットワーク内やインターネットで、特定のホスト同士が通信する際に使用します。

マルチキャスト

この種類のアドレスは、ネットワークインターフェイスのグループに関連づけられるものです。このアドレス宛に発信されたパケットは、グループに属する全ての宛先に送信されます。マルチキャストアドレスは、主に特定のネットワークサービスが使用するもので、特定のグループに属するホストに対して通信する際に使用します。

エニーキャスト

この種類のアドレスも、ネットワークインターフェイスのグループに関連づけられるものです。ただし、このアドレス宛に発信されたパケットは、ルーティングプロトコルの仕組みに従って、最も送信者に近いグループのメンバーに配信されます。エニーキャストアドレスは、特定のサービスを提供するサーバが、インターネット内に多数存在するような場合に利用します。同じ種類のサービスのサーバは、同じエニーキャストアドレスを持ちます。通常はルーティングプロトコルの仕組みで最も近いサーバに送信されますが、そのサーバが利用できない場合は 2 番目に近いサーバに、そのサーバも利用できない場合は 3 番目に近いサーバに、のように配信されます。

IPv6 アドレスは 4 桁のフィールド 8 個で構成されています。それぞれのフィールドは 16 ビットの値を 16 進数で表す仕組みで、フィールド間はコロン (:) で区切ります。また、16 進数の上位の桁が 0 であった場合は 0 を省略して表記しますが、逆に下位の桁が 0 であった場合には省略を行いません。

また、特定のフィールドが完全な 0 であった場合にはそのフィールドの表記を省略して、`::` と表します。ただし `::` は 1 つしか設定できません。アドレス表記の例を 例13.3「IPv6 アドレスの例」 に示します (いずれも同じアドレスを表しています):

例 13.3: IPV6 アドレスの例

```
fe80 : 0000 : 0000 : 0000 : 0000 : 10 : 1000 : 1a4
fe80 :    0 :    0 :    0 :    0 : 10 : 1000 : 1a4
fe80 :                           : 10 : 1000 : 1a4
```

IPv6 アドレスは、パートごとにそれぞれの意味があります。最初のいくつかのバイトはプレフィクスとアドレスの種類を表します。中央の部分はアドレスのネットワークパートと呼ばれ、場合によっては使用しません。末尾の部分はホストパートと呼ばれ、特定のホストを表す部分になります。また、IPv6 でのネットマスクはアドレスの末尾にスラッシュを置き、その後ろにプレフィクスの長さをビット数で表します。たとえば 例13.4「プレフィクス長付きの IPv6 アドレス」 のアドレスの場合は、最初の 64 ビットまでがアドレスのネットワークパートで、残りの 64 ビットがホストパートであることを示しています。これを言い換えると、`64` とは左側から 64 ビット分までがネットワークパートであることになります。IPv4 では、論理積 (AND) を利用して同じサブネットかどうかを判断していましたが、IPv6 ではビット数のみを使用して判断します。

例 13.4: プレフィクス長付きの IPV6 アドレス

```
fe80::10:1000:1a4/64
```

IPv6 には、いくつかの定義済みプレフィクスが存在しています。そのうちのいくつかを IPv6 プレフィクス で示します。

IPV6 プレフィクス

`00`

IPv4 アドレスと IPv4 over IPv6 互換アドレスを表します。これらは IPv4 との互換性を維持するために使用されているもので、ルータが IPv6 パケットを IPv4 パケットに変換できるようにするために使用します。このほか、ループバックデバイス向けのアドレスなどもこのプレフィクスを使用しています。

最初の 1 桁が `2` または `3`

集約可能なグローバルユニキャストアドレス (Aggregatable global unicast addresses) と呼ばれるものです。IPv4 の場合と同様に、インターフェイスに対して特定のサブネットの一部を構成するために割り当てられるものです。現時点では、下記のアドレス領域が存在しています: `2001::/16` (本番用アドレス領域) および `2002::/16` (6to4 用アドレス領域)

`fe80::/10`

リンクローカルアドレスと呼ばれ、このプレフィクスを持ったアドレスは経路制御を行うべきではなく、そのため同じサブネット内にしかアクセスできないアドレスです。

`fec0::/10`

サイトローカルアドレスと呼ばれ、経路制御を行ってもかまわないものの、所属する特定の組織内でのみ使用すべきものです。実際には、`10.x.x.x` などの IPv6 アドレスにおけるプライベートアドレスとも呼べるものです。

`ff`

これらはマルチキャストアドレスです。

ユニキャストアドレスには、3 種類の基本コンポーネントが含まれています:

パブリックトポロジ

最初の部分 (上述のプレフィクスを含む) は、一般的なインターネットを介してパケットを経路制御する際に使用します。インターネットアクセスを提供する企業や団体の情報を含む部分です。

サイトトポロジ

2 つ目の部分には、パケットの配信先のサブネットに関する経路情報が含まれています。

インターフェイス ID

3 つ目の部分には、パケットの配信先となるインターフェイスを識別するための情報が含まれています。このような構造により、MAC アドレスをそのままアドレスとすることができるようになっています。MAC アドレスが全世界で唯一の番号であると仮定すると、ハードウエアの製造元が固定の識別子をデバイスに割り当てていることになりますので、設定の手順も単純化できることになります。実際のところ、最初の 64 ビットのアドレスは `EUI-64` トークンを構成する値で、最後の 48 ビットは MAC アドレスから生成し、残りの 24 ビットはトークンの種類を示す特殊な情報を設定します。また、point-to-point protocol (PPP) ベースのインターフェイスのように、MAC アドレスを持たないものに対しても、`EUI-64` トークンを割り当てることができます。

このような基本的なネットワーク構造に加えて、IPv6 では 5 種類のユニキャストアドレスが規定されています:

`::` (未指定)

このアドレスは、インターフェイスを起動する際に初期のアドレスとして使用するものです。

`::1` (ループバック)

ループバックデバイスのアドレスです。

IPv4 互換アドレス

この IPv6 アドレスは IPv4 アドレスを示すもので、96 ビットのゼロをプレフィクスとして設定します。この種類の互換アドレスはトンネリング (詳しくは 13.2.3項「IPv4 と IPv6 の共存」をお読みください) で使用され、IPv4 と IPv6 のホストが、純粋な IPv4 環境内で互いに通信できるようにするものです。

IPv6 にマッピングされた IPv4 アドレス

この種類のアドレスは、純粋に IPv4 のアドレスを IPv6 表記になおしたものです。

ローカルアドレス

ローカルアドレスとしては、下記の 2 種類のものが存在します:

リンクローカル

この種類のアドレスは、ローカルのサブネット内でのみ使用することができます。この種類のアドレスを発信元または送信先に持つパケットは、インターネットや他のサブネットに配信すべきではありません。これらのアドレスに対しては、特別なプレフィクス (`fe80::/10`) が設定され、末尾にはネットワークカードのインターフェイス ID と、残りはゼロで埋められたアドレスになります。この種類のアドレスは、同じサブネット内に存在する他のホストとの通信に使用し、自動設定を目的として使用します。

サイトローカル

この種類のアドレスは他のサブネットに配信してもかまわないものの、インターネットに対しては配信すべきではありません。企業内や団体内などの中でのみ使用すべきものです。イントラネットと呼ばれるネットワークで使用されるもので、IPv4 で言うところのプライベートアドレスと同じ意味を持ちます。また、このアドレスには特殊なプレフィクス (`fec0::/10`) が設定され、末尾にはネットワークカードのインターフェイス ID と 16 ビットのサブネット ID が入ります。なお、後の残りはゼロで埋められます。

IPv6 で全く新しい機能としては、それぞれのネットワークインターフェイスが複数のアドレスを持つのが一般的である、という機能があります。これにより、複数のネットワークにアクセスするにあたっても、同じインターフェイスを使用することができるようになります。これらのネットワークのうちいずれかは、MAC アドレスと既知のプレフィクスを利用して完全に自動化された設定を行うことができます。そのため、IPv6 が有効化されてさえいれば、ローカルネットワーク内の全てのホストがリンクローカルアドレスで通信できることになります。なお、MAC アドレスが IPv6 アドレスの一部となることから、世界中で使用される IPv6 アドレスもまた唯一のものになります。アドレスの残りの部分は サイトトポロジ とか パブリックトポロジ などと呼ばれますが、これはコンピュータの接続するネットワークに従って決まります。

特定のコンピュータが異なるネットワークに属するコンピュータと通信を行う場合、少なくとも 2 種類のアドレスを必要とします。1 つは ホームアドレス と呼ばれ、インターフェイス ID だけでなく、通常属するホームネットワークに関する識別子 (および対応するプレフィクス) を持つものです。ホームアドレスは固定のアドレスであり、通常は変化するものではありません。ただし、モバイルホスト宛の全てのパケットは、家庭内にいてもどこか別の場所にいても、必ず配信することができるようになっています。これは、IPv6 で新しい機能として導入された ステートレス自動設定 と 近隣探索 によって実現されています。ホームアドレスに加えて、モバイルホストでは接続先に対応する 1 つ以上の追加アドレスが設定されます。これらは ケア・オブ (気付) アドレス と呼ばれます。また、ホームネットワークには、そのホストが別の場所にいるような場合に、そのホスト宛の通信を別の場所に転送する機能を備えています。IPv6 環境

では、この処理は ホームエージェント と呼ばれる仕組みが実装しています。これはホームアドレス宛の全てのパケットを、トンネルを介して中継する仕組みで、逆にケア・オブアドレス宛の通信は、特にパケットを迂回させることなくモバイルのホスト宛に転送されます。

13.2.3 IPv4 と IPv6 の共存

インターネットにおいて、全てのホストに対する IPv4 から IPv6 への移行は、順次進められている状況です。この場合は、両方のプロトコルが共存することになります。1 つのシステム内での共存は、両方のプロトコルを実装する デュアルスタック の仕組みによって成り立っています。ただし、IPv6 が有効化されたホストがどのようにして IPv4 のホストと通信するのか、および IPv6 パケットがどのように現在の (主に IPv4 ベースの) ネットワークで配送されていくのかが疑問になります。このような問題を解決するために、トンネリングと互換アドレスと呼ばれる仕組みが用意されています (詳しくは 13.2.2項「アドレスの種類と構造」をお読みください。

IPv6 のホストは (世界中にある) IPv4 ネットワーク内では孤立した存在で、トンネルを通して通信を行うことになります。IPv6 パケットは IPv4 パケットとしてカプセル化され、IPv4 のネットワークを移動します。このような IPv4 ホスト同士の接続を、トンネル と呼びます。これを実現するためには、パケットには IPv6 の宛先アドレス (もしくは対応するプレフィクス) と、トンネルの出口となるリモートホストの IPv4 アドレスが含まれていなければなりません。ホストの管理者同士が合意して構築した基本的なトンネルは、スタティック (静的) トンネリング と呼ばれます。

しかしながら、スタティックトンネリングの設定やメンテナンスには双方にとって手間のかかるもので、日々の負担にもなってしまいます。そのため、IPv6 では 3 種類の ダイナミック (動的) トンネリング 方式が用意されています:

6over4
: IPv6 パケットは自動的に IPv4 パケットとしてカプセル化され、マルチキャストに対応した IPv4 ネットワーク内に送信されます。IPv6 からの見た目では、ネットワーク全体 (つまりインターネット) が巨大なローカルエリアネットワーク (LAN) であるかのように扱われます。これにより、IPv4 トンネルの受信側を決定できることになります。しかしながら、インターネットの世界ではマルチキャストが十分に普及していないことから、この方法は規模の拡大には十分に対応できない状況です。そのため、この方法はマルチキャストを利用できる小さな企業や団体などに対する解決方法にとどまっています。なお、この方式の仕様は、RFC 2529 に規定されています。

6to4
: この方法を利用すると、IPv6 アドレスから IPv4 アドレスを自動生成して、孤立した IPv6 ホストを IPv4 ネットワークと通信できるようにします。ただし、IPv6 とインターネットとの間での通信に対して、いくつかの問題が報告されています。この方式は RFC 3056 に規定されています。

IPv6 トンネルブローカー
: この方法は、IPv6 ホスト向けの専用トンネルを提供する特別なサーバを使用します。この方式は RFC 3053 に規定されています。

13.2.4 IPv6 の設定

IPv6 を設定するには、通常は個別のコンピュータに対して作業を行う必要はありません。これは、IPv6 は既定で有効化されているためです。インストール済みのシステムで IPv6 を有効化もしくは無効化するには、YaST を起動して［ネットワークの設定］モジュールを起動します。［グローバルオプション］のタブ内に、［IPv6 を有効にする］のチェックボックスがありますので、必要に応じてチェックを入れるか外すかしてください。また、再起動するまでの間、一時的に IPv6 を有効化するには、`root` で `modprobe -i ipv6` を実行します。なお、いったんモジュールを読み込んでしまうと、再起動以外の手段では読み込みを解除することができなくなりますので、ご注意ください。

IPv6 の自動設定の考え方により、ネットワークカードには リンクローカル のアドレスが割り当てられます。また、通常はワークステーション内でのルーティングテーブル管理は行いません。ルータアドバタイズメントプロトコル を利用することで、ワークステーション側からネットワークルータに問い合わせを行い、プレフィクスと使用すべきゲートウエイに関する情報を取得します。通常は radvd プログラムを利用することで、IPv6 ルータを構築します。そのほか、FRR (詳しくは https://frrouting.org/ をお読みください) を利用して、アドレスとルーティングを自動設定することもできます。

`/etc/sysconfig/network` ファイルを利用して様々な種類のトンネルを設定する方法について、詳しくは `ifcfg-tunnel` のマニュアルページ (`man ifcfg-tunnel`) をお読みください。

13.2.5 さらなる情報

これまでの章での説明は、IPv6 の一部のみをカバーしたものであって、全てを説明しているわけではありません。この新しいプロトコルに関するより詳しい情報をご希望の場合は、下記のオンライン文書やブックをご覧ください:

https://pulse.internetsociety.org
: IPv6 に関する全ての出発点です。

http://www.ipv6day.org
: ご自身で IPv6 ネットワークを始めるにあたっての情報が提供されています。

http://www.ipv6-to-standard.org/
: IPv6 対応の製品の一覧が提供されています。

https://www.bieringer.de/linux/IPv6/

Linux における IPv6 の HOWTO や、IPv6 関連の様々なリンクが提供されています。

RFC 2460

IPv6 の RFC に関する基本情報については、https://www.rfc-editor.org/rfc/rfc2460 をお読みください。

IPv6 Essentials

Silvia Hagen 氏による IPv6 の主要な要素を説明した書籍です (ISBN 0-596-00125-8) 。

13.3 名前解決

DNS は 1 つもしくは複数の名前を IP アドレスに割り当てたり、逆に IP アドレスを名前に割り当てたりすることのできる仕組みです。DNS サービスを提供する側の Linux では、この変換を bind と呼ばれる特別な種類のソフトウエアで賄います。サービスを提供する側のコンピュータを、一般に ネームサーバ と呼びます。また、DNS は階層型のシステムで、それぞれの階層をピリオドで区切ります。ただし、IP アドレスにある階層構造とは独立して運用されています。

たとえば `jupiter.example.com` のような完全な名前は、**ホスト名.ドメイン名** のような形式で記述します。この完全な名前を 完全修飾ドメイン名 (Fully Qualified Domain Name; FQDN) と呼びます。また、ドメイン名 (例: `example.com`) の末尾は トップレベルドメイン (Top Level Domain; TLD) と呼びます。上述の例では、`com` がそれにあたります。

TLD の割り当ては、歴史的な事情から複雑になっています。以前は、3 文字の TLD はアメリカ合衆国が使用し、残りの国では、2 文字の ISO 国コードを TLD として使用していました。これに加えて、2000 年にはより長い TLD が導入され、様々な用途で使用されるようになりました (例: `.info` , `.name` , `.museum` など) 。

インターネットが始まったばかりの頃 (1990 年以前) 、インターネット上の全マシンの名前を保存するために、`/etc/hosts` が使われてきました。ところが、インターネットに接続するコンピュータの台数が爆発的に増えることによって、それは一瞬で非現実的なものになってしまいました。このような経験から、ホスト名を管理するための分散データベースを開発して、それを世界中に広げることにしました。このデータベースはネームサーバに似た仕組みで、インターネット内の全てのホストを保持するには不十分であるものの、他のネームサーバにリクエストを転送できる仕組みを備えていました。

階層構造の頂点には ルートネームサーバ と呼ばれるサーバが存在しています。これらのルートネームサーバは、トップレベルドメインを管理する存在で、ネットワークインフォメーションセンター (Network Information Center; NIC) が管理しています。それぞれのルートネームサーバは、どのネームサーバがどのトップレベルドメインを管理しているのかを知っています。トップレベルドメインの NIC は https://www.internic.net です。

DNS はホスト名の解決を行うだけではありません。ドメイン全体の電子メールを受信するホストなどの情報 (Mail Exchanger (MX)) も保持しています。

お使いのマシンから IP アドレスを解決できるようにするためには、少なくとも 1 台のネームサーバの IP アドレスが必要となります。ネームサーバの設定を行うには、YaST をお使いください。また、openSUSE® Leap における具体的なネームサーバ設定については、13.4.1.4項「ホスト名と DNS の設定」をお読みください。独自のネームサーバを構築したい場合は、第19章「ドメインネームシステム」をお読みください。

`whois` と呼ばれるプロトコルは、DNS と密接な関係を持つプロトコルです。このプロトコルは、特定のドメインに対する管理情報などを問い合わせることができます。

注記: マルチキャスト DNS と .local ドメイン名について

`.local` のトップレベルドメイン名はリンクローカルドメインと呼ばれるもので、このドメイン宛の DNS リクエストは、通常の DNS リクエストではなく、マルチキャスト DNS リクエストとして送信されます。既に `.local` というドメインをネームサーバの設定で使用している場合は、`/etc/host.conf` で無効化する必要があります。詳しくは `host.conf` のマニュアルページをお読みください。

インストール時にマルチキャスト DNS を使用しないようにするには、起動パラメータに `nomdns=1` を指定してください。

マルチキャスト DNS についての詳細は、http://www.multicastdns.org をお読みください。

13.4 YaST を利用したネットワークの設定

Linux では様々な種類のネットワークに対応しています。これらのうちのほとんどは、それぞれ異なるデバイス名を使用するもので、ファイルシステム内の様々な箇所に設定ファイルが分散しています。手作業でのネットワーク設定について、詳しくは 13.6項「ネットワーク接続の手動管理」をお読みください。

リンクの確立している (ケーブルが接続されている) 全てのインターフェイスが、自動的に設定されます。インストール済みの状態でも、任意の時点で追加のハードウエアを設定することができます。下記の章では、openSUSE Leap で対応している全種類のネットワーク接続に対して、その設定方法を説明しています。

13.4.1 YaST を利用したネットワークカードの設定

YaST でイーサネットや Wi-Fi, Bluetooth カードなどを設定したい場合は、YaST を起動して［システム］›［ネットワークの設定］を選択します。モジュールを起動すると、YaST は［ネットワーク設定］ウインドウ内で 4 つのタブを表示します。それぞれ［グローバルオプション］,［概要］,［ホスト名／DNS］,［ルーティング］というタブになっています。

［グローバルオプション］のタブでは、ネットワークの設定方法や IPv6 の利用可否、そして一般的な DHCP オプションなどを設定することができます。詳しくは 13.4.1.1項「グローバルネットワークオプションの設定」をお読みください。

［概要］タブには、インストール済みのネットワークカードとその設定が表示されます。ここには、検出済みのネットワークカードが名前で表示されます。このダイアログでは新しいネットワークカードの設定のほか、既存のネットワークカードの設定変更や削除も行うことができます。自動では検出されないネットワークカードを手作業で設定するには、13.4.1.3項「未検出のネットワークカードの設定」の手順に従って作業を行ってください。また、既に設定済みのネットワークカードの設定を変更するには、13.4.1.2項「ネットワークカードの設定変更」をお読みください。

［ホスト名／DNS］のタブでは、マシンに設定するホスト名や、使用すべきネームサーバなどを設定します。詳しくは 13.4.1.4項「ホスト名と DNS の設定」をお読みください。

［ルーティング］タブでは、ルーティングに関連する設定を行います。詳しくは 13.4.1.5項「ルーティングの設定」をお読みください。

図 13.3: ネットワークの設定

13.4.1.1 グローバルネットワークオプションの設定

YaST の［ネットワークの設定］モジュール内の［グローバルオプション］タブでは、NetworkManager や IPv6 の使用可否のほか、DHCP クライアントオプションなどを設定することができます。これらの設定は、全てのネットワークインターフェイスに適用されます。

［ネットワークの設定方法］では、ネットワークの接続を管理するための方法を選択します。NetworkManager のデスクトップアプレットを使用して全てのネットワークインターフェイスを管理したい場合は、［NetworkManager サービス］を選択します。NetworkManager は有線ネットワークと無線ネットワークを切り替えて使用する場合にも便利な仕組みです。デスクトップ環境を動作させていない場合や、お使いのコンピュータが Xen サーバや仮想化システムであるような場合、もしくは DHCP や DNS などのネットワークサービスを提供しているような場合は、［Wicked サービス］を選択します。なお、NetworkManager を使用する場合は、ネットワークの設定は `nm-applet` を利用して行うことになりますので、［概要］,［ホスト名／DNS］,［ルーティング］などのタブは無効化され、選択できなくなります。NetworkManager について、詳しくは 第28章「NetworkManager の使用」をお読みください。

［IPv6 プロトコル設定］では、IPv6 プロトコルを使用するかどうかを選択します。IPv6 は、IPv4 と共に使用することができます。既定では IPv6 が有効化されています。しかしながら、IPv6 を使用していないネットワークの場合は、IPv6 プロトコルを無効化したほうが高速に動作しますので、この場合は［IPv6 を有効にする］のチェックを外してください。IPv6 を無効化すると、カーネルは IPv6 モジュールを自動では読み込まなくなります。また、この設定は次回の再起動から有効になります。

［DHCP クライアントオプション］では、DHCP クライアントに対する設定を行うことができます。［DHCP クライアント識別子］はネットワーク内で唯一とならなければならない値を入力する項目で、何も指定しない場合はネットワークインターフェイスのハードウエアアドレスを使用します。ただし、同じネットワークインターフェイスを利用して複数の仮想マシンを動作させているような場合は、複数の仮想マシンが同じハードウエアアドレスになってしまいますので、それぞれで別々の識別子を入力する必要があります (記入に際してのルールはありません。自由記入です) 。

［送信するホスト名］では、DHCP クライアントが DHCP サーバに送信する際の、ホスト名オプションの値を文字列で指定します。DHCP サーバによっては、ネームサーバのゾーン (正引きおよび逆引き) を更新しますが、この際に［送信するホスト名］の値を利用して更新を行うものがあります (動的 DNS) 。また、DHCP サーバによっては［送信するホスト名］に特定の文字列が含まれていることを確認して、応答を送信するものもあります。なお、現在のホスト名を送信したい場合は、`AUTO` のままにしておいてください (ホスト名は `/etc/hostname` に設定されます) 。また、何もホスト名を送信したくない場合は、何も入力しないでください。

また、DHCP サーバからの情報でデフォルトルートを更新したくない場合は、［DHCP で既定のルートを変更する］のチェックを外してください。

13.4.1.2 ネットワークカードの設定変更

ネットワークカードの設定を変更するには、YaST の［ネットワークの設定］から、［概要］タブを選択します。すると、検出されたカードの一覧が表示されますので、必要な項目を選んで［編集］を押します。すると、［ネットワークカードの設定］ダイアログが表示されますので、必要に応じて［一般］,［アドレス］,［ハードウエア］の各設定を変更してください。

13.4.1.2.1 IP アドレスの設定

IP アドレスそのものの設定や、IP アドレスの割り当て方法の設定を行う場合は、［ネットワークカードの設定］ダイアログ内の［アドレス］タブで行います。ここでは IPv4 と IPv6 の両方に対応しています。ここでは、［リンクおよび IP の設定無し］(ボンディングデバイスとして使用する場合に指定します) のほか、［固定 IP アドレス］(IPv4 または IPv6) や［可変 IP アドレス］のいずれかを選択します。なお、［可変 IP アドレス］を選択した場合は、［DHCP］もしくは［Zeroconf］のいずれか、もしくはその両方を選択することができます。

また、［可変 IP アドレス］を選択した場合は、［バージョン 4 のみでの DHCP］(IPv4),［バージョン 6 のみでの DHCP］(IPv6),［バージョン 4 と 6 の両方での DHCP］のいずれかを選択します。

また、インストールの時点でリンクが確立しているネットワークカードがあれば、それらのうちの最初のネットワークカードが DHCP で設定されます。

DHCP は、インターネットサービスプロバイダ (Internet Service Provider; ISP) から固定の IP アドレスが割り当てられない DSL 回線をお使いの場合には、使用しておくべき設定となります。なお、DHCP を使用する場合は、［ネットワーク設定］ダイアログ内の［グローバルオプション］内にある［DHCP クライアントオプション］についてもご確認ください。また、同じインターフェイスを利用して複数の仮想ホストを動作させるような場合は、それらを区別するために［DHCP クライアント識別子］の設定が必要となる場合があります。

また、DHCP はクライアント側を設定する際には適切な選択ではありますが、サーバとして設定する場合には不適切です。固定の IP アドレスを設定するには、下記の手順で行います:

1. YaST のネットワーク設定モジュールの［概要］タブ内で、検出されたネットワークカードのうちのいずれかを選択して、［編集］を押します。
2. ［アドレス］タブでは、［固定 IP アドレス］を選択します。
3. ［IP アドレス］の欄にアドレスを入力します。IPv4 と IPv6 の両方のアドレスを使用することができます。また、［サブネットマスク］にはネットマスクを指定します。IPv6 アドレスの場合は、［サブネットマスク］の項目には `/64` のような形式で指定します。
   また、必要であれば［ホスト名］の欄に、完全修飾ドメイン名を指定することができます。この設定は `/etc/hosts` 内に書き込まれます。
4. ［次へ］を押します。

5. 設定を反映させるには、［OK］を押します。

注記: インターフェイスの有効化とリンクの検出

ネットワークインターフェイスを有効化する際、`wicked` は物理的なリンク状況を確認して、リンクが検出できた場合にのみ IP 設定を適用します。リンクの確率とは関係なくアドレスを設定したい場合 (たとえば特定のアドレスで待ち受けるサービスをテストしたいような場合) は、`/etc/sysconfig/network/ifcfg-インターフェイス名` 設定ファイル内に `LINK_REQUIRED=no` を追加してください。

また、リンクを待機する秒数を指定したい場合は、`LINK_READY_WAIT=5` のような指定を行うこともできます。

`ifcfg-*` 設定ファイルについて、詳しくは 13.6.2.5項「/etc/sysconfig/network/ifcfg-*」および `man 5 ifcfg` をお読みください。

固定のアドレスを使用する場合は、ネームサーバやデフォルトゲートウエイも自動では設定されません。ネームサーバを設定するには 13.4.1.4項「ホスト名と DNS の設定」の手順を、デフォルトゲートウエイを設定するには 13.4.1.5項「ルーティングの設定」の手順をお読みください。

13.4.1.2.2 複数アドレスの設定

1 つのネットワークデバイスに対して、エイリアスやラベルとして複数の IP アドレスを設定することができます。

注記: エイリアス／ラベル機能について

エイリアスやラベルと呼ばれる機能は、IPv4 でのみ動作します。`iproute2` を利用することで、ネットワークインターフェイスに複数のアドレスを設定できます。

YaST を利用してネットワークカードに追加のアドレスを設定するには、下記の手順で実施します:

1. YaST の［ネットワーク設定］ダイアログ内にある［概要］タブで、検出されたネットワークカードのうちいずれかを選択し、［編集］を押します。
2. ［アドレス］›［追加アドレス］で、［追加］ボタンを押します。
3. それぞれ［IPv4 アドレスラベル］,［IP アドレス］,［ネットマスク］をそれぞれ指定します。なお、IP エイリアスの場合は `/32` のネットマスクでなければなりません。また、別名内にインターフェイス名は入れないでください。

4. 設定を有効化するには、［次へ］または［完了］を押してください。

13.4.1.2.3 デバイス名の変更と udev ルール

使用するネットワークカードの名前については、必要に応じて変更することができます。ネットワークカード同士の判別は、ハードウエア (MAC) アドレスのほか、コンピュータ内のバス ID で行います。巨大なサーバを構成するような場合は、後者を指定しておくとネットワークカードを後から接続 (ホットプラグ) することができるようになります。これらの設定を YaST で行うには、下記の手順で行います:

1. YaST の［ネットワーク設定］ダイアログ内にある［概要］タブで、検出されたネットワークカードのうちいずれかを選択し、［編集］を押します。
2. ［一般］タブに移動します。現在のデバイス名が、［udev ルール］内に表示されています。ここから［変更］を押します。
3. まずはネットワークカードの識別方法を、［MAC アドレス］もしくは［バス ID］の中から選択します。それぞれ現在の MAC アドレスとバス ID がダイアログ内に表示されます。
4. デバイス名を変更するには、［デバイス名］の欄に新しいデバイス名を入力します。
5. 設定を有効化するには、［次へ］または［完了］を押してください。

13.4.1.2.4 ネットワークカードのカーネルドライバの変更

ネットワークカードによっては、複数のカーネルドライバの中からいずれかを選択することができます。カードが既に設定済みの場合、YaST では利用可能なドライバの一覧を表示して、その中からいずれかを選択することができます。また、ここではカーネルドライバに対するオプションを設定することもできます。YaST で設定を変更するには、下記の手順で行います:

1. YaST の［ネットワーク設定］ダイアログ内の［概要］タブで、検出されたネットワークカードのうちいずれかを選択し、［編集］を押します。
2. ［ハードウエア］タブに移動します。
3. 使用すべきカーネルドライバを、［モジュール名］の欄で選択します。また、`=` = `値` の形式で、［オプション］欄に必要なオプションを指定します。複数のオプションを指定したい場合は、スペースで区切って指定してください。
4. 設定を有効化するには、［次へ］または［完了］を押してください。

13.4.1.2.5 ネットワークデバイスの有効化

`wicked` を利用する方法を選択した場合、ネットワークデバイスをシステムの起動時に開始するか、ケーブルを接続したときに開始するか、カードを検出した際に開始するか、もしくは手作業で開始するか、何も有効化しないかを選択することができます。開始方法を変更するには、下記の手順で行います:

1. YaST から［システム］›［ネットワークの設定］を選択して、検出されたネットワークカードの中からいずれかを選択し、［編集］を押します。
2. ［一般］タブ内にある［デバイスの有効化］で、いずれかを選択します。
   ［起動時］を選択すると、システムの起動時にデバイスが開始されます。［ケーブル接続時］を選択すると、物理的な接続状況を監視して、接続が確認できると開始されます。［ホットプラグ］を選択すると、インターフェイスが利用可能な状態になると開始されます。これは［起動時］の選択とほぼ同じ動作ですが、システムの起動時にデバイスが存在していなくても、エラーにならない点が異なります。また、［手動］を選択すると、`ifup` で手作業で開始する設定になります。［開始しない］を選択すると、デバイスを開始しなくなります。また、［NFSroot］の選択肢は、［起動時］とほぼ同じ動作になりますが、`systemctl stop network` を実行してもインターフェイスがシャットダウンされなくなります。また、`wicked` が有効な際は、`network` サービスが `wicked` サービスの起動も扱うようになります。この選択肢は、NFS や iSCSI を介してルートファイルシステムにアクセスする際に選択します。
3. 設定を有効化するには、［次へ］または［完了］を押してください。

ヒント: NFS をルートファイルシステムで使用する場合の注意

ルートパーティションを NFS 共有経由でマウントするシステム (ディスクを持たないシステム) の場合、NFS 共有にアクセスするネットワークデバイスについては、設定を注意して行う必要があります。

システムをシャットダウンしたり再起動したりする際、既定の処理順序はネットワークの接続を切断してからルートパーティションをマウント解除しますが、ルートパーティションが NFS である場合、この順序で処理してしまうと、ネットワークが先に切れてしまうことから、NFS のマウントを正しく解除することができなくなってしまいます。ネットワークデバイスが無効化されないようにするには、13.4.1.2.5項「ネットワークデバイスの有効化」で説明しているとおりネットワークデバイスの設定ウインドウを開いて、［デバイスの有効化］で［NFSroot］を選択してください。

13.4.1.2.6 最大転送単位サイズの設定

インターフェイスに対しては、最大転送単位 (MTU) を設定することができます。MTU とは、最も大きいパケットのサイズをバイト単位で表したもので、MTU を大きくすればするほど、帯域を効率的に使用することができるようになります。ただし、遅いインターフェイスで MTU を大きくしすぎると、その送信にかかる時間が大きくなってしまい、インターフェイスを長い時間占有してしまうことになりますので、即時性が損なわれてしまいます。

1. YaST から［システム］›［ネットワークの設定］を選択して、検出されたネットワークカードの中からいずれかを選択し、［編集］を押します。
2. ［一般］タブを選択し、［最大転送単位 (MTU)］の欄内にあるドロップダウンボックスで、値を入力するか選択を行います。
3. 設定を有効化するには、［次へ］または［完了］を押してください。

13.4.1.2.7 PCIe 多機能デバイス

LAN, iSCSI, FCoE のそれぞれに対応する多機能型デバイスを利用することができます。YaST の FCoE クライアント (`yast2 fcoe-client`) では、ユーザ側で FCoE に使用するかどうかを選択できるようにするため、追加の列内にプライベートフラグが表示されます。YaST のネットワークモジュール (`yast2 lan`) では、「ストレージのみ」に設定したネットワークインターフェイスは除外して表示します。

13.4.1.2.8 IP-over-InfiniBand (IPoIB) のための Infiniband 設定

1. YaST を起動し、［システム］›［ネットワークの設定］を選択して、表示された Infiniband デバイスを選択して［編集］を押します。
2. ［一般］タブでは、［IP-over-InfiniBand］(IPoIB) のモードのうち、［connected］(既定値) もしくは［datagram］の中からいずれかを選択します。
3. 設定を有効化するには、［次へ］または［完了］を押してください。

InfiniBand について、詳しくは `/usr/src/linux/Documentation/infiniband/ipoib.txt` をお読みください。

13.4.1.2.9 ファイアウオールの設定

『セキュリティ強化ガイド』、第24章「マスカレードとファイアウオール」、24.4項「firewalld」で説明している詳細なファイアウオール設定を行っていない場合は、ネットワークインターフェイスの設定手順のうちの 1 つで、基本的なファイアウオール設定を行うことができます。具体的には下記の手順で行います:

1. YaST を開いて、［システム］›［ネットワークの設定］を選択します。［概要］タブが表示されたら、検出済みのネットワークカードの一覧からいずれかを選択し、［編集］を押します。
2. ［ネットワークカードの設定］ダイアログ内にある［一般］タブを選択します。
3. ［ファイアウオールゾーン］内のリストボックスで、インターフェイスをどのゾーンに割り当てるのかを選択します。下記の選択肢が用意されています:

   ファイアウオールを無効にする
   : この選択肢はファイアウオールが無効化されていて、動作していない場合に表示されます。この設定は、別途のファイアウオールで保護された環境内に接続している場合にのみご利用ください。

   自動ゾーン割り当て
   : このオプションはファイアウオールが有効化されている場合にのみ表示されます。ファイアウオールが動作すると、ファイアウオールがインターフェイスを自動的にゾーンに割り当てます。 any というキーワードを含むゾーン、もしくは外部ゾーンのいずれかに割り当てられます。

   内部ゾーン (保護無し)
   : ファイアウオール自身は動作させるものの、このインターフェイスに対する保護ルールは何も適用しないゾーンです。この設定は、別途のファイアウオールで保護された環境内に接続している場合や、外部には全く接続していない場合にお使いください。

   非武装ゾーン
   : 非武装ゾーンとは、内部ネットワークの最前線に立つ追加のゾーンで、その外側には (攻撃を受けうる) インターネットが存在するゾーンです。このゾーンに割り当てられたホストは、内部ネットワークとインターネットの両方から接続することができますが、インターネット側から内部にはアクセスさせない構成を取るのが一般的です。

   外部ゾーン
   : このインターフェイスでファイアウオールを動作させ、全てのネットワークトラフィックに対して (特に攻撃などから) 保護を適用するゾーンです。この選択肢が既定値になっています。

4. 設定を有効化するには、［次へ］または［完了］を押してください。

13.4.1.3 未検出のネットワークカードの設定

ネットワークカードが正しく検出されなかった場合は、検出済みのネットワークインターフェイスの一覧内には表示されません。お使いのネットワークカード向けのドライバが存在することがわかっている場合は、手作業で設定を行うことができます。このほか、ネットワークブリッジやボンディング、TUN や TAP と呼ばれる特殊なネットワークインターフェイスについても、手作業で設定を行うことになります。具体的には、下記の手順で行います:

1. YaST から［システム］›［ネットワークの設定］›［概要］を選び、［追加］ボタンを押します。
2. ［ハードウエアダイアログ］内では、まず［デバイスの種類］を選択し、［設定名］で選択または入力を行います。お使いのネットワークカードが USB デバイスである場合は、対応するチェックボックスにチェックを入れてから［次へ］を押します。それ以外のデバイスである場合は、［モジュール名］にカーネルのモジュール名を入力し、必要であれば［オプション］に必要なオプションを設定します。
   なお、［ethtool オプション］では、`ifup` コマンドでインターフェイスを起動する際、`ethtool` コマンドに指定するオプションを設定します。利用可能なオプションの一覧については、`ethtool` のマニュアルページをお読みください。
   オプションの文字列が `-` で始まるものの場合 (たとえば `-K インターフェイス名 rx on` など) は、文字列内の 2 番目の箇所が現在のインターフェイス名に置き換えられます。それ以外の場合 (たとえば `autoneg off speed 10`) は、`ifup` コマンドを実行する際、冒頭に `-s インターフェイス名` を追加します。
3. ［次へ］を押します。
4. あとは IP アドレスやデバイスの有効化、インターフェイスに割り当てるファイアウオールのゾーンなど、残りのオプションを［一般］,［アドレス］,［ハードウエア］で設定します。設定項目についての詳細は、13.4.1.2項「ネットワークカードの設定変更」をお読みください。
5. なお、［デバイスの種類］で［無線］を選択した場合は、次のダイアログで無線関連の設定を行います。
6. 新しいネットワーク設定を有効化するには、［次へ］または［完了］を押してください。

13.4.1.4 ホスト名と DNS の設定

インストールの時点でお使いのネットワークカードが利用できる状態であり、かつネットワークの設定を変更していない場合は、お使いのコンピュータに対してホスト名が自動生成され、DHCP が有効化された状態になります。これはネットワーク環境に接続するためのネームサーバの設定についても同様で

す。ネットワークアドレスを DHCP 経由で取得していると、ネームサーバの情報を受信した際に自動的に設定を行います。何らかの理由により、手作業で設定を行いたい場合は、これらの設定を行う必要があります。

お使いのコンピュータの名前や使用するネームサーバの検索リストを変更するには、下記の手順で行います:

1. YaST を起動し、［システム］›［ネットワーク設定］を選択して、［ホスト名／DNS］タブを選択します。

2. まずは［ホスト名］の欄にホスト名を指定します。なお、ホスト名はコンピュータ全体に対する設定であり、全てのネットワークインターフェイスに設定されることに注意してください。
IP アドレスを取得するのに DHCP を使用している場合は、DHCP 側から通知された名前がホスト名として自動設定されます。このような動作は、複数のネットワークに接続する環境では、ホスト名が頻繁に変わる結果になることから、グラフィカルデスクトップ環境に悪影響を及ぼすこともありますので、無効化しておくことをお勧めします。DHCP で IP アドレスを取得する環境で、ホスト名を変更しないようにするには、［DHCP でホスト名を設定］を［いいえ］に設定してください。

3. また、［DNS 設定の修正］では、DNS まわりの設定 (ネームサーバ、検索リスト、`/run/netconfig/resolv.conf` の内容) の修正方法を設定することができます。
［既定のポリシーを使用する］を選択すると、設定内容は `netconfig` スクリプトが処理を行い、固定で設定された内容 (YaST もしくは設定ファイル内で設定した内容) と動的に取得した内容 (DHCP クライアントもしくは NetworkManager) を合成します。通常は、この既定のポリシーのままで問題ありません。
［手動のみ］を選択すると、`netconfig` が `/run/netconfig/resolv.conf` を変更しないようになります。ただし、このファイルは手作業で編集して変更することが可能です。
［カスタムポリシー］を選択した場合は、［カスタムポリシールール］で指定した文字列の合成ポリシーで制御が行われます。ここにはカンマ区切りでインターフェイス名を指定し、指定したインターフェイスからの情報を正しい情報源と見なして、設定を書き換える動作を行います。インターフェイス名は、そのまま指定するだけでなく、ワイルドカードを指定することもできます。たとえば `eth* ppp?` のように指定すると、`eth` で始まるインターフェイスと、`ppp0` から `ppp9` が該当するようになります。このほか、`/etc/sysconfig/network/config` ファイルで定義されている下記のキーワードも指定することができます:

   `STATIC`
   : 固定で設定した内容を、動的に取得した内容と合成する必要があることを指定します。

   `STATIC_FALLBACK`
   : 動的に設定が取得できない場合にのみ、固定の設定を使用するよう指定します。

さらに詳しい情報については、`netconfig` (8) のマニュアルページ (`man 8 netconfig`) をお読みください。

4. ［ネームサーバ］と［ドメイン検索］に対して、それぞれ必要な設定を行います。ネームサーバに対しては、ホスト名ではなく IP アドレス (例: 192.168.1.116) で設定してください。また、［ドメイン検索］には、ドメイン名無しでホスト名のみを指定した場合に、自動的に補完されるべきドメイン名を指定します。複数のドメインを［ドメイン検索］に指定したい場合は、カンマまたはスペースで区切ってください。

5. 設定を有効化するには、［次へ］または［完了］を押してください。

このほか、YaST を利用してコマンドラインからホスト名を変更することもできます。YaST で変更を行った場合、その設定は即時に反映されます (`/etc/hostname` を編集した場合とは異なります) 。ホスト名を変更するには、下記のコマンドを実行します:

```
# yast dns edit hostname=ホスト名
```

ネームサーバを変更したい場合は、下記のようなコマンドを実行します:

```
# yast dns edit nameserver1=192.168.1.116
# yast dns edit nameserver2=192.168.1.117
# yast dns edit nameserver3=192.168.1.118
```

13.4.1.5 ルーティングの設定

お使いのマシンを他のネットワーク内にあるマシンと通信したい場合は、ルーティング (経路制御) に関する情報を設定して、正しい経路で通信が配送されるように設定しなければなりません。DHCP をお使いの場合、この情報は自動的に提供され適用されます。固定の設定をお使いの場合は、このデータは手作業で設定しなければなりません。

1. YaST を起動して、［ネットワークの設定］› ［ルーティング］を選択します。

2. ［デフォルトゲートウエイ］の IP アドレスを入力します (必要であれば、IPv4 だけでなく IPv6 でも設定することができます) 。デフォルトゲートウエイは、その他のルーティング情報内に存在しなかった場合に、最後に使用すべき (既定の) ゲートウエイ (中継器) を意味するものです。

3. ［ルーティングテーブル］の欄を利用することで、さらに細かい設定を行うことができます。ここでは、［宛先］に通信先の IP アドレスを、［ゲートウエイ］には使用するゲートウエイの IP アドレスを、［ネットマスク］にはネットマスクをそれぞれ指定します。また、［デバイス］の欄には、使用すべきネットワークインターフェイスの名前を指定します (`-` を指定すると、任意のデバイス

を使用する意味になります) 。これらの値のうちのいずれかを省略したい場合は、マイナス記号 `-` を指定してください。また、表内にデフォルトゲートウエイを設定したい場合は、［宛先］欄に `default` と入力してください。

注記: ルートの優先制御

複数のデフォルトゲートウエイを使用する場合は、メトリック値を指定して優先順位を指定することができます。メトリック値を設定するには、［オプション］の欄に `- metric 数値` の形式で指定を行います。設定可能なメトリック値の最小は 0 で、最も小さなメトリック値のルートが最優先とされ、既定で使用されます。ネットワークデバイスの接続が切れた場合は、ルーティング情報からその経路が削除され、次に有効なルーティングを使用します。

4. お使いのシステムをルータとして設定するには、［ネットワーク設定］内にある［IPv4 転送を有効にする］と［IPv6 転送を有効にする］をそれぞれ選択してください。
5. 設定を有効化するには、［次へ］または［完了］を押してください。

13.5 NetworkManager

NetworkManager はラップトップ機など、可搬性のあるコンピュータでは便利な仕組みです。NetworkManager を利用することで、特にネットワークを頻繁に切り替えて利用するような環境の場合、ネットワークインターフェイスの設定を心配する必要がなくなります。

13.5.1 NetworkManager と `wicked` の違い

しかしながら、NetworkManager が全ての用途において適切な選択肢であるとは言えないのが現状です。そのため、現時点でも `wicked` による制御方式と NetworkManager による制御方式のいずれかを選択するようになっています。NetworkManager を利用してネットワーク接続を管理したい場合は、まず YaST のネットワーク設定モジュールで設定を行ってください。こちらについて、詳しくは 28.2項「NetworkManager の有効化と無効化」をお読みのうえ、設定を行ってください。また、NetworkManager の使用例や設定方法についての詳しい説明は、第28章「NetworkManager の使用」をお読みください。

wicked と NetworkManager には、いくつかの違いがあります:

`root` の権限

ネットワーク設定を NetworkManager で行う場合、アプレットを使用すれば、お使いのデスクトップから簡単にネットワークの接続を切り替えたり、開始もしくは停止したりすることができます。NetworkManager では、`root` の権限無しに、無線 LAN カードの接続を変更したり管理したりすることもできます。このような理由から、NetworkManager はモバイル端末では魅力的なソリューションとなります。

`wicked` でもネットワーク接続の切り替えや開始／停止などに対応していて、ユーザが明示的に介在していなくても、自分自身で管理しているかのように扱うことができます。ただし、これらの作業にはいずれも `root` の権限が必要となります。また、全ての接続をあらかじめ設定しておかなければならないことにもなりますので、モバイル環境では使いづらくなってしまいます。

ネットワーク接続の種類

`wicked` であっても NetworkManager であっても、有線 LAN だけでなく、無線 LAN の接続を扱うことができます (WEP, WPA-PSK, WPA-Enterprise などに対応しています) 。また、DHCP による自動設定や固定の設定にも対応しています。このほか、ダイヤルアップ接続や VPN にも対応しています。ただし、ブロードバンド (3G) モデムや DSL 接続を扱うことができるのは NetworkManager だけで、従来の設定方式では扱うことができません。

NetworkManager では、お使いのコンピュータで常に最適な接続を使用するように努めています。ネットワークケーブルが外れてしまったような場合でも、再接続を試みるようになっています。また、無線 LAN 接続の設定が存在する場合、設定されているものの中で最も信号強度が強いものを検出して、それに接続するように動作します。`wicked` を利用してこれを行うとすると、さらなる設定作業が必要となってしまいます。

13.5.2 NetworkManager の機能と設定ファイル

NetworkManager で作成した個別のネットワーク接続設定は、設定プロファイル内に保存されます。NetworkManager や YaST で設定したシステム接続は、`/etc/NetworkManager/system-connections/*` もしくは `/etc/sysconfig/network/ifcfg-*` 内に保存されます。GNOME の場合、全てのユーザ定義接続は GConf 内に保存されます。

何もプロファイルを設定していない場合、NetworkManager は自動的にプロファイルを作成し、そのプロファイルに `Auto $INTERFACE-NAME` という名前を設定します。これは、できる限り多くの場合において、設定を行うことなくネットワークを動作させる意図で作られたものですが、これが要件に適合しない場合は、GNOME などで提供されているネットワーク接続ダイアログを利用して、変更を行ってください。詳しくは 28.3項「ネットワーク接続の設定」をお読みください。

13.5.3 NetworkManager の機能制御とロックダウン

集中管理されているマシンなどの場合は、特定の NetworkManager 機能を無効化したり Polkit で無効化したりすることができます。たとえばユーザは管理者が設定した接続のみを変更できるようにするとか、ユーザは独自のネットワーク接続のみを利用できる、などがあります。NetworkManager のポリシーを表示もしくは変更するには、Polkit のグラフィカルな［認可］ツールを利用してください。左側のツリー表示に［network-manager-settings］という項目があるはずです。Polkit とその設定方法について、詳しくは『セキュリティ強化ガイド』、第19章「Polkit 認可フレームワーク」をお読みください。

13.6 ネットワーク接続の手動管理

ネットワークインターフェイスの手動設定は最後の選択肢として用意されています。通常は YaST の使用をお勧めしますが、下記に示す背景となる情報を知っておくことで、YaST での設定もよりわかりやすくなります。

13.6.1 `wicked` ネットワーク設定

`wicked` と呼ばれるツールとライブラリが、ネットワーク設定に対する新しいフレームワークを提供しています。

以前のネットワークインターフェイス管理では、異なるレイヤのネットワーク管理を寄せ集めて単一のスクリプトにするか、あっても 2 つの異なるスクリプトに仕立て上げるものでした。これらのスクリプトは、きちんと定義されていない方法で互いに作用するものであり、これによって予期しない問題を発生させたり、不明瞭な制約や慣習などをもたらしたりする結果になってしまっていました。また、様々なシナリオに対応するための様々なレイヤの特殊な修正により、さらにメンテナンスを複雑化させてしまっていました。アドレス設定プロトコルは dhcpcd のような実装を介して使用されていましたが、こちらも他のインフラストラクチャとの対話性は低いままでした。また、わけのわからないインターフェイス名の命名方式によって、インターフェイス名の永続性を達成するのに udev の支援を多く必要としてしまっていました。

wicked は、様々な方法でこれらの問題を切り分けるために作られています。いずれの箇所も奇抜なものではないものの、様々なプロジェクトからのアイディアをまとめて、全体的によりよいソリューションとなるよう期待が込められています。

1 つ目のアプローチは、クライアント／サーバモデルの採用です。これにより、wicked はフレームワーク全体と十分に統合された仕組みの中で、アドレス設定などの標準化された仕組みを定義することができるようになっています。たとえば特定のアドレス設定を行う場合、管理者は DHCP や IPv4 Zeroconf などを使用するよう設定する場合がありますが、この場合もサーバからアドレスの貸与情報を取得して、それを wicked のサーバプロセスに渡すことで、wicked 側がそのアドレスと経路の設定を行うようになっています。

問題を切り分けるためのもう 1 つのアプローチとしては、レイヤ構造をきちんと守っているということが挙げられます。どのような種類のネットワークインターフェイスであっても、ネットワークインターフェイスのデバイスレイヤ (VLAN, ブリッジ, ボンディング, 準仮想化デバイスなど) を設定する DBus サービスを定義することができるようになっています。アドレスの設定などの一般的な機能については、それぞれを特別に実装することなくデバイス固有のサービス上に搭載することのできる、接続サービスを用いて実装されています。

wicked フレームワークは、これら 2 つの要素を様々な DBus サービスを利用して実装しています。使用する DBus サービスはネットワークの種類によって異なりますが、ここには wicked 内での現在のオブジェクト構造について、大まかな概要を示しています。

それぞれのネットワークインターフェイスは `/org/opensuse/Network/Interfaces` の子オブジェクトとして表されます。子オブジェクトの名前は ifindex の値から設定されます。たとえばループバックインターフェイスの場合、通常は ifindex が 1 になりますので、`/org/opensuse/Network/Interfaces/1` になります。最初に登録されたイーサネットインターフェイスは、`/org/opensuse/Network/Interfaces/2` になります。

それぞれのネットワークインターフェイスには「クラス」が割り当てられ、対応する DBus インターフェイスを選択する際に使用します。既定では、各ネットワークインター他フェイスは `netif` というクラスになっていて、`wickedd` はこのクラスと互換性のある全てのインターフェイスに自動的に接続するようになっています。また、現在の実装では、下記のインターフェイスが含まれています:

org.opensuse.Network.Interface
: リンクアップやリンクダウン、MTU の割り当てなどの一般的なネットワークインターフェイス機能を表します。

org.opensuse.Network.Addrconf.ipv4.dhcp,
org.opensuse.Network.Addrconf.ipv6.dhcp,
org.opensuse.Network.Addrconf.ipv4.auto
: DHCP, IPv4 zeroconf などのアドレス設定サービスを表します。

これらに加えて、ネットワークインターフェイスによっては特別の設定機構を必要としていたり、提供していたりすることがあります。たとえばイーサネットデバイスであれば、リンク速度やチェックサムのオフロード機能などを制御することができるでしょう。これらを実現するために、イーサネットの場合は `netif` のサブクラスである `netif-ethernet` という特殊なクラスが用意されています。そのため、イーサネットインターフェイスに割り当てられている DBus インターフェイスは、上述の全てのサービスに加えて、`org.opensuse.Network.Ethernet` サービスが提供されています。これにより、`netif-ethernet` クラスに属するオブジェクトにアクセスできるようになっています。

同様に、ブリッジや VLAN 、ボンディングや InfiniBand 等のインターフェイスの種類に対しても、それぞれのクラスが用意されています。

VLAN のように、イーサネットデバイスを包含するタイプの仮想ネットワークインターフェイスである場合、これらを先に作成する必要がありますが、これは wicked でどのように扱われているのでしょうか? これを実現するために、wicked では `org.opensuse.Network.VLAN.Factory` などの factory インターフェイスを定義しています。この factory インターフェイスは、指定した種類のインターフェイスを作成する機能だけを持ち、このインターフェイスが `/org/opensuse/Network/Interfaces` のリストノードに割り付けられるようになっています。

13.6.1.1 wicked の構造と機能

`wicked` サービスは複数のパーツから構成されています。詳しくは 図13.4「wicked の構造」をお読みください。

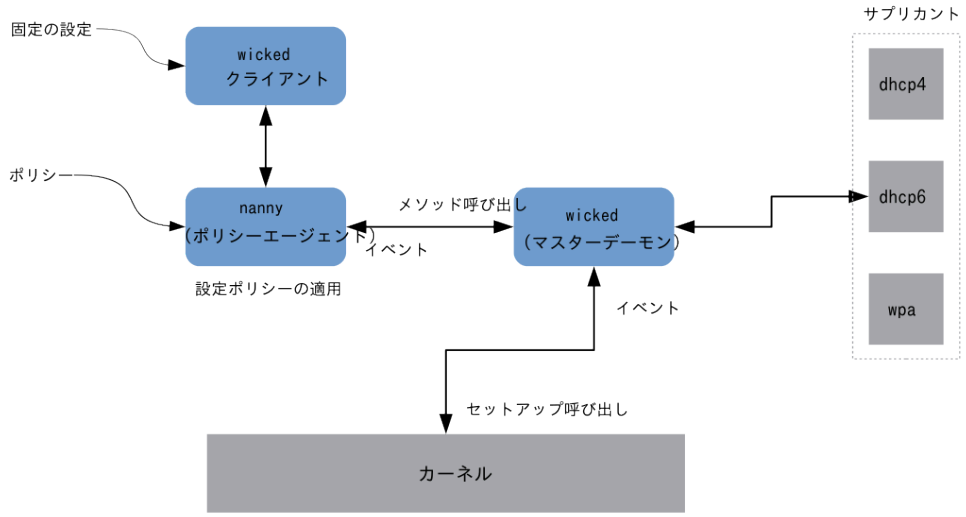


図 13.4: wicked の構造

`wicked` には現在、下記のものが含まれています:

- SUSE 形式の `/etc/sysconfig/network` ファイルを処理するための設定ファイルバックエンド
- ネットワークの設定を XML 形式で表す内部用の設定バックエンド
- イーサネットや InfiniBand, VLAN,ブリッジ, ボンディング, TUN, TAP,ダミー, macvlan, macvtap, hsi, qeth, iucv, および無線 (現時点では WPA-PSK/EAP のみ) などの「一般的な」インターフェイスの起動と停止
- 内蔵型の DHCPv4 クライアントおよび DHCPv6 クライアント

- 設定済みのインターフェイスが利用可能な状態になった場合の自動起動 (ホットプラグ) と、リンク (キャリア) を検出した際の IP 設定に対応する nanny デーモン (既定で有効化されています) 。詳しくは 13.6.1.3項「Nanny」をお読みください
- systemd と統合するための DBus サービスの集合体としての実装 (そのため、通常の `systemctl` のコマンドが `wicked` にも適用されます)

13.6.1.2 `wicked` の使用

openSUSE Leap では、`wicked` はデスクトップハードウエアやサーバハードウエアの場合に、既定で動作しています。モバイルハードウエアの場合は、NetworkManager が既定で動作しています。現時点で何が有効化され動作しているのかを調べるには、下記のコマンドを実行します:

```
systemctl status network
```

`wicked` が有効化されている場合、下記のような出力が現れます:

```
wicked.service - wicked managed network interfaces
    Loaded: loaded (/usr/lib/systemd/system/wicked.service; enabled)
    ...
```

何か異なるシステム (たとえば NetworkManager) が動作しているような場合で、`wicked` に切り替えたい場合は、まず動作しているものを停止してから `wicked` を有効化してください:

```
systemctl is-active network && \
systemctl stop      network
systemctl enable --force wicked
```

これにより wicked サービスが有効化され、`network.service` から `wicked.service` に別名リンクが設定され、次回以降の起動で開始されるようになります。

サーバプロセスを開始するには、下記のコマンドを実行します:

```
systemctl start wickedd
```

これにより、`wickedd` (メインサーバ) が起動され、関連する下記のプログラムも起動されます:

```
/usr/lib/wicked/bin/wickedd-auto4 --systemd --foreground
/usr/lib/wicked/bin/wickedd-dhcp4 --systemd --foreground
/usr/lib/wicked/bin/wickedd-dhcp6 --systemd --foreground
/usr/sbin/wickedd --systemd --foreground
/usr/sbin/wickedd-nanny --systemd --foreground
```

あとはネットワークを起動します:

```
systemctl start wicked
```

上記ではなく、別名である `network.service` のほうを起動してもかまいません:

```
systemctl start network
```

これらのコマンドは既定値を使用するか、もしくは `/etc/wicked/client.xml` 内に書かれているシステム設定を使用します。
デバッグを有効化するには、`/etc/sysconfig/network/config` 内で `WICKED_DEBUG` を設定します。たとえば下記のようになります:

```
WICKED_DEBUG="all"
```

下記のようにいくつかのデバッグ項目を省いてもかまいません:

```
WICKED_DEBUG="all,-dbus,-objectmodel,-xpath,-xml"
```

全てのインターフェイスに対する情報を表示したり、*インターフェイス名* で指定したインターフェイスに対する情報を表示したりするには、下記のいずれかを実行します:

```
wicked show all
wicked show インターフェイス名
```

XML 形式で出力させたい場合は、下記のように実行します:

```
wicked show-xml all
wicked show-xml インターフェイス名
```

いずれか 1 つのインターフェイスを起動する場合は、下記のように実行します:

```
wicked ifup eth0
wicked ifup wlan0
...
```

何も設定情報を指定しない場合、wicked クライアントは `/etc/wicked/client.xml` 内で指定されている既定の情報源を利用して設定しようとします:

1. `firmware:` iSCSI 起動ファームウエアテーブル (iBFT)
2. `compat:` `ifcfg` ファイル (互換性を維持するために実装されています)

`wicked` がインターフェイスの設定を取得した場合は、どのような内容であっても、`firmware` , `compat` の順序で計画されています。ただし将来的には変更される可能性があります。
詳しくは `wicked` のマニュアルページをお読みください。

13.6.1.3 Nanny

nanny はイベントおよびポリシー駆動型のデーモンで、ホットプラグ型のデバイスなどの非同期かつ不用意なシナリオに対応するための仕組みです。そのため、nanny デーモンは開始や再起動の遅延のほか、一時的に取り外されたデバイスにも対応することができます。また、nanny はデバイスとリンク状態の変更を監視しているほか、現在のポリシーセットで設定されている新しいデバイスを統合することができます。Nanny では指定されたタイムアウト制約によって `ifup` が既に終了してしまったような場合でも、継続して設定を行うことができます。

既定では、nanny デーモンはシステム内で有効化されています。具体的には、`/etc/wicked/common.xml` 設定ファイル内の下記で有効化されています:

```
<config>
  ...
  <use-nanny>true</use-nanny>
</config>
```

この設定により、ifup と ifreload は nanny デーモンに対して効率的な設定ポリシーを適用することになります。その後、nanny は `wickedd` をも設定しますので、ホットプラグにも対応することになります。また、nanny は裏でイベントや変更 (新しいデバイスの接続やキャリア状態の変更など) を待機する処理も行います。

13.6.1.4 複数のインターフェイスの開始

ボンディングやブリッジでは、複数のデバイスを 1 つにまとめて設定を行い (ifcfg-bondX) ますので、それらを一括で起動する必要があります。wicked では、ボンディングやブリッジなどの上位側のインターフェイス名を起動するだけで、その中にあるインターフェイスを一括で起動することができます:

```
wicked ifup br0
```

このコマンドを実行すると、ブリッジを設定して起動するまでの処理を、特に依存関係の順序 (ポートなど) を指定することなく、自動で行うことができます。

複数のインターフェイスを 1 つのコマンドで起動するには、下記のように実行します:

```
wicked ifup bond0 br0 br1 br2
```

全てのインターフェイスを一括で起動するには、下記のように実行します:

```
wicked ifup all
```

13.6.1.5 Wicked によるトンネルの使用

wicked でトンネルを使用する必要がある場合は、`TUNNEL_DEVICE` で設定を行って対応することができます。必要であれば、デバイス名を指定してトンネルとデバイスを紐づけることもできます。トンネルされたパケットは、このデバイスを介して配信されることになります。

詳しくは `man 5 ifcfg-tunnel` をお読みください。

13.6.1.6 差分変更の処理

`wicked` では、インターフェイスを再設定するにあたって、インターフェイスをいったん停止する必要はありません (カーネル側で求められている場合を除きます) 。たとえばネットワークインターフェイスに IP アドレスを追加したり、経路設定を行ったりする場合でも、インターフェイスの設定に IP アドレスを追加したあと、もう一度「ifup」操作を行うだけで済むようになっています。サーバ側では変更すべき設定のみを抽出して適用する処理を行います。これは MTU や MAC アドレスなどのリンクレベルのオプションだけでなく、アドレスや経路、アドレスの設定モード (固定のアドレス設定から DHCP への変更など) のネットワークレベルの設定に対しても、このような動作をするようになっています。

ブリッジやボンディングなど、複数の実デバイスから構成される仮想デバイスの設定に対しては、注意する必要があります。たとえばボンディングデバイスの場合、デバイスの起動中には特定の設定を行うことができません。設定しようとすると、エラーが発生することがあります。

しかしながら、ボンディングやブリッジなどの実デバイスの追加や削除、ボンディングにおけるプライマリインターフェイスの選択などは、行うことが可能です。

13.6.1.7 wicked 拡張: アドレス設定

`wicked` はシェルスクリプトによって拡張を行うことができます。これらの拡張は、`config.xml` ファイル内に定義されています。

現時点では、いくつかの拡張クラスに対応しています:

- リンク設定: これらは、クライアント側から提供されたリンクレイヤの設定にあわせてデバイスを設定し起動する機能、およびデバイスを停止するためのスクリプトです。
- アドレス設定: これらは、デバイスのアドレス設定を行うためのスクリプトです。通常はアドレスの設定や DHCP の設定などは `wicked` 自身が管理しますが、拡張として実装もされています。
- ファイアウオール拡張: これらのスクリプトは、ファイアウオールのルールを適用します。

一般的に、拡張には開始と停止のコマンドが用意されているほか、必要に応じて「PID ファイル」やスクリプトに渡される環境変数などが存在しています。

拡張がどのように動作するのかを知るには、`etc/server.xml` 内にあるファイアウオール拡張をご覧ください:

```
<dbus-service interface="org.opensuse.Network.Firewall">
 <action name="firewallUp"   command="/etc/wicked/extensions/firewall up"/>
 <action name="firewallDown" command="/etc/wicked/extensions/firewall down"/>

 <!-- default environment for all calls to this extension script -->
 <putenv name="WICKED_OBJECT_PATH" value="$object-path"/>
 <putenv name="WICKED_INTERFACE_NAME" value="$property:name"/>
 <putenv name="WICKED_INTERFACE_INDEX" value="$property:index"/>
</dbus-service>
```

拡張は `<dbus-service>` タグに割り当てられ、このインターフェイスのアクションを実行する際のコマンドを定義しています。また、アクションに対して渡される環境変数の定義を行うこともできます。

13.6.1.8 wicked 拡張: 設定ファイル

Wicked では、スクリプトを利用して設定ファイルの処理を拡張することができます。たとえば DHCP による貸与情報からの DNS の書き換えは、`extensions/resolver` が実施していて、これは `server.xml` 内に記述されています:

```
<system-updater name="resolver">
 <action name="backup" command="/etc/wicked/extensions/resolver backup"/>
 <action name="restore" command="/etc/wicked/extensions/resolver restore"/>
 <action name="install" command="/etc/wicked/extensions/resolver install"/>
 <action name="remove" command="/etc/wicked/extensions/resolver remove"/>
</system-updater>
```

`wickedd` に対して更新が届くと、貸与情報を処理して、`resolver` スクリプトの対応するコマンド (`backup`, `install` など) を実行します。ここから `/sbin/netconfig` を呼び出して DNS の設定を変更するか、手作業で `/run/netconfig/resolv.conf` を変更している場合は、このファイルを直接変更する処理を行います。

13.6.2 設定ファイル

本章では、ネットワーク関連の設定ファイルの概要とそれらの目的、そして使用する書式について説明しています。

13.6.2.1 /etc/wicked/common.xml

/etc/wicked/common.xml ファイルには、全てのアプリケーションが使用する一般的な設定が記述されています。このファイルは、同じディレクトリ内にある他の設定ファイルから取り込まれる形で使用されます。なお、全ての wicked コンポーネントを一括でデバッグする目的で使用することもできますが、このような場合は /etc/wicked/local.xml ファイルをお使いください。また、ソフトウエアの更新が行われた場合は、/etc/wicked/common.xml ファイルが上書きされ、元の設定が失われることがある点にも注意してください。なお、/etc/wicked/common.xml ファイルは既定で /etc/wicked/local.xml ファイルを取り込むようになっていますので、特別な理由が無い限り、/etc/wicked/common.xml ファイルを編集する必要はないはずです。

nanny を無効化する目的で <use-nanny> を false に設定した場合は、wickedd.service サービスの再起動に加えて、下記のコマンドを実行して全ての設定とポリシーを適用してください:

```
> sudo wicked ifup all
```

注記: 設定ファイル

wickedd , wicked , nanny の各プログラムは、自分専用の設定ファイルが存在していない場合、/etc/wicked/common.xml ファイルを読み込みます。

13.6.2.2 /etc/wicked/server.xml

/etc/wicked/server.xml ファイルは wickedd サーバプロセスが起動時に読み込むファイルです。このファイルには /etc/wicked/common.xml に対する追加の設定を記述します。具体的には、リゾルバの処理方法や DHCP などの addrconf から受信した情報の処理方法を設定します。

なお、このファイルに対して何らかの設定を追加したい場合は、/etc/wicked/server-local.xml ファイルに記述することをお勧めします。このファイルは /etc/wicked/server.xml から取り込まれるようになっています。このように別途のファイルにすることで、メンテナンス更新後も設定内容を失わないようにすることができます。

13.6.2.3 /etc/wicked/client.xml

/etc/wicked/client.xml ファイルは wicked コマンドが使用するファイルです。このファイルには ibft が管理するデバイスを検出した際に使用するスクリプトの場所や、ネットワークインターフェイスの設定などを記述します。

なお、このファイルに対して何らかの設定を追加したい場合は、`/etc/wicked/client-local.xml` ファイルに記述することをお勧めします。このファイルは `/etc/wicked/client.xml` から取り込まれるようになっています。このように別途のファイルにすることで、メンテナンス更新後も設定内容を失わないようにすることができます。

13.6.2.4 `/etc/wicked/nanny.xml`

`/etc/wicked/nanny.xml` ファイルはリンクレイヤの種類を設定するためのファイルです。このファイルに対して何らかの設定を追加したい場合は、`/etc/wicked/nanny-local.xml` ファイルに記述することをお勧めします。このように別途のファイルにすることで、メンテナンス更新後も設定内容を失わないようにすることができます。

13.6.2.5 `/etc/sysconfig/network/ifcfg-*`

これらのファイルには、ネットワークインターフェイスに対する従来の設定が含まれています。

注記: `wicked` と `ifcfg-*` ファイルの関係性について

`wicked` は `compat:` プレフィクスを指定した場合、これらのファイルを読み込むようになります。openSUSE Leap の既定の `/etc/wicked/client.xml` 設定では、`wicked` が `/etc/wicked/ifconfig` 内にある XML ファイルを読み込むよりも前に、これらのファイルを読み込むようになっています。

また、`--ifconfig` スイッチはほぼテスト用にのみ提供されているものです。これを指定した場合、`/etc/wicked/ifconfig` 内に設定されている既定の設定ソースは適用されなくなります。

`ifcfg-*` ファイルには起動モードや IP アドレスなどの情報が含まれています。設定可能なパラメータについて、詳しくは `ifup` のマニュアルページをお読みください。これに加えて、特定の 1 つのインターフェイスにのみ一般的な設定を適用したい場合は、`dhcp` ファイルや `wireless` ファイルにある変数のうちのほとんどを使用することができます。ただし、`/etc/sysconfig/network/config` の変数のうちのほとんどはグローバルなものであり、ifcfg ファイルでは上書きできません。たとえば `NETCONFIG_*` 変数などがそれに該当します。

`macvlan` や `macvtab` インターフェイスを設定したい場合は、`ifcfg-macvlan` および `ifcfg-macvtap` の各マニュアルページをお読みください。たとえば `ifcfg-macvlan0` というファイルで macvlan インターフェイスを使用したい場合は、下記のように記述します:

```
STARTMODE='auto'
```

```
MACVLAN_DEVICE='eth0'
#MACVLAN_MODE='vepa'
#LLADDR=02:03:04:05:06:aa
```

`ifcfg.template` に関する詳細は、13.6.2.6項「`/etc/sysconfig/network/config`, `/etc/sysconfig/network/dhcp`, `/etc/sysconfig/network/wireless`」をお読みください。

13.6.2.6 `/etc/sysconfig/network/config`, `/etc/sysconfig/network/dhcp`, `/etc/sysconfig/network/wireless`

`config` ファイルには `ifup`, `ifdown`, `ifstatus` などのコマンドに対する一般的な設定が含まれています。また `dhcp` ファイルには DHCP に関する設定が、`wireless` には無線 LAN カードに関する設定がそれぞれ含まれています。なお、3 つの設定ファイルには、それぞれコメントで変数の説明が書かれています。`/etc/sysconfig/network/config` ファイル内の設定のうちのいくつかは `ifcfg-*` ファイルでも使用することができます。なお、この場合は `ifcfg-*` 側での設定が優先されます。また、`/etc/sysconfig/network/ifcfg.template` ファイルには、各インターフェイスに対して設定することのできる変数の一覧が書かれています。ただし、`/etc/sysconfig/network/config` 内のほとんどの変数はグローバルなものであり、ifcfg ファイルでは上書きできません。たとえば `NETWORKMANAGER` 変数や `NETCONFIG_*` 変数などがそれに該当します。

注記: DHCPv6 の使用について

DHCPv6 を使用する場合、ネットワーク内にある少なくとも 1 台のルータから、このネットワークが DHCPv6 で管理されている旨を表す RA を送信する必要があります。

ルータを正しく設定することができないようなネットワークをお使いの場合は、`ifcfg` ファイル内のオプションで `DHCLIENT6_MODE='managed'` を指定することで、従来のバージョンの動作に戻すことができます。インストールシステムを起動させる際に設定したい場合は、起動パラメータに下記の設定を追加してください:

```
ifcfg=eth0=dhcp6,DHCLIENT6_MODE=managed
```

13.6.2.7 `/etc/sysconfig/network/routes` および `/etc/sysconfig/network/ifroute-*`

TCP/IP パケットのスタティックルーティング (固定の経路制御) に対しては、それぞれ `/etc/sysconfig/network/routes` ファイルと `/etc/sysconfig/network/ifroute-*` ファイルを使用します。さまざまなシステム処理で必要なスタティックルーティングは `/etc/sysconfig/network/`

`routes` ファイル側に設定します。これにはたとえばホスト宛のルーティングのほか、ゲートウエイ経由でのルーティング、ネットワーク宛のルーティングなどを設定することができます。インターフェイス単位でルーティングを設定したい場合は、`/etc/sysconfig/network/ifroute-*` (ここで、ワイルドカード (`*`) 部分にはインターフェイスの名前を入れます) ファイルを設定してください。いずれの場合も、設定ファイルの内容は下記のようになります:

```
# 宛先              ゲートウエイ        ネットマスク        インターフェイス  オプション
```

一番左の列には宛先を記述します。ここにはネットワークやホストのアドレスのほか、到達可能な ネームサーバが存在する場合は、完全修飾ネットワーク名もしくは完全修飾ホスト名を指定することができます。なお、ネットワークを指定する場合は、CIDR 形式 (アドレスの後ろにプレフィクス長を付ける形式) で記述します。たとえば IPv4 であれば 10.10.0.0/16 のような形式に、IPv6 であれば fc00::/7 のような形式になります。また、`default` キーワードを指定するとデフォルトゲートウエイを指定することができます。ゲートウエイの無いデバイスの場合は、0.0.0.0/0 もしくは ::/0 のように明示的に宛先を指定してください。

左から 2 番目の列にはゲートウエイを指定します。特定のホストやネットワーク、もしくはデフォルト (既定値) として使用するものを指定します。

左から 3 番目の列は廃止予定とされているものです。以前は IPv4 のネットマスクを指定していた箇所になります。IPv6 の場合は一番左の列に宛先とプレフィクス長 (CIDR 形式) を指定しますので、使用していません。IPv4 で CIDR 形式を使用している場合や、IPv6 の場合は、ここにハイフン (`-`) を入れておいてください。

左から 4 列目にはインターフェイスの名前を指定します。ハイフン (`-`) を指定した場合は省略している意味になりますが、`/etc/sysconfig/network/routes` で使用した場合は、予期しない動作になってしまう場合があります。詳しくは `routes` のマニュアルページをお読みください。

左から 5 列目 (オプション) には、特殊なオプション設定を記述します。詳しくは `routes` のマニュアルページをお読みください。

例 13.5: 一般的なネットワークインターフェイスとスタティックルーティングの設定

```
# --- CIDR 表記での IPv4 ルーティング設定:
# Destination     [Gateway]         -                   Interface
127.0.0.0/8       -                 -                   lo
204.127.235.0/24  -                 -                   eth0
default           204.127.235.41    -                   eth0
207.68.156.51/32  207.68.145.45     -                   eth1
192.168.0.0/16    207.68.156.51     -                   eth1

# --- 古い（廃止予定の）ネットマスク形式での IPv4 ルーティング設定"
# Destination     [Dummy/Gateway]   Netmask             Interface
#
127.0.0.0         0.0.0.0           255.255.255.0       lo
204.127.235.0     0.0.0.0           255.255.255.0       eth0
default           204.127.235.41    0.0.0.0             eth0
```

```
207.68.156.51     207.68.145.45     255.255.255.255    eth1
192.168.0.0       207.68.156.51     255.255.0.0        eth1

# --- IPv6 のルーティング設定 (CIDR 表記のみ使用できます):
# Destination     [Gateway]                 -          Interface
2001:DB8:100::/64 -                         -          eth0
2001:DB8:100::/32 fe80::216:3eff:fe6d:c042 -           eth0
```

13.6.2.8 /var/run/netconfig/resolv.conf

ホストが属するドメイン名を指定するには、`/var/run/netconfig/resolv.conf` ファイル (キーワード `search`) を使用します。この `search` オプションでは、それぞれ 256 文字までのドメイン名を最大 6 つまで指定することができます。完全修飾ではない名前を解決する場合、`search` で指定したドメインをそれぞれ後ろに付けて解決を行おうとします。また、このファイルでは `nameserver` オプションを利用して、最大 3 つまでのネームサーバを指定することができます。コメントを指定したい場合はハッシュ記号 (`#`) もしくはセミコロン (`;`) を行頭に記述します。たとえば 例13.6「/var/run/netconfig/resolv.conf」のようになります。

なお、`/etc/resolv.conf` ファイルは手作業で作成したりしないでください。このファイルは `netconfig` で生成されるものであるほか、`/run/netconfig/resolv.conf` へのシンボリックリンクになっているためです。YaST を利用せずに DNS の設定を行いたい場合は、`/etc/sysconfig/network/config` ファイル内にある下記の項目を、手作業で編集してください:

`NETCONFIG_DNS_STATIC_SEARCHLIST`
: ホスト名の参照時に使用する DNS ドメイン名のリスト

`NETCONFIG_DNS_STATIC_SERVERS`
: ホスト名の参照時に使用するネームサーバの IP アドレスのリスト

`NETCONFIG_DNS_FORWARDER`
: DNS フォワーダの使用形態の指定 (例: `bind` や `resolver` など)

`NETCONFIG_DNS_RESOLVER_OPTIONS`
: `/var/run/netconfig/resolv.conf` に書き込むべき任意のオプション指定。例:

  ```
  debug attempts:1 timeout:10
  ```

  詳しくは `resolv.conf` のマニュアルページをお読みください。

`NETCONFIG_DNS_RESOLVER_SORTLIST`
: 最大 10 項目まで。例:

  ```
  130.155.160.0/255.255.240.0 130.155.0.0
  ```

詳しくは `resolv.conf` のマニュアルページをお読みください。

netconfig による DNS 設定機能を無効化したい場合は、`NETCONFIG_DNS_POLICY=''` を指定してください。また、`netconfig` の詳細については、`netconfig(8)` のマニュアルページ (`man 8 netconfig`) をお読みください。

例 13.6: `/var/run/netconfig/resolv.conf`

```
# ドメイン名
search example.com
#
# 使用するネームサーバの指定
nameserver 192.168.1.116
```

13.6.2.9 `/sbin/netconfig`

`netconfig` はさまざまな追加のネットワーク設定を管理するためのモジュール型ツールです。固定で指定されている設定と DHCP や PPP などで取得した動的な設定を、あらかじめ指定したポリシーで合成したりすることができます。また、設定ファイルの修正を行う netconfig のモジュールを呼び出すことで、システムに設定を反映させることができるほか、必要なサービスを再起動したりなどの処理を行うことができます。

`netconfig` では、主に下記に示す 3 つのアクションを使用します。`netconfig modify` や `netconfig remove` は DHCP や PPP などのデーモンが使用し、必要なネットワーク設定を行ったり削除したりすることができます。ユーザ側でのみ使用されるコマンドとしては、`netconfig update` があります:

`modify`

`netconfig modify` コマンドは現在のインターフェイスやサービス固有の設定を修正したり、ネットワークの設定を更新したりすることができるコマンドです。netconfig は標準入力のほか、`--lease-file ファイル名` を指定すれば、ファイルから設定を読み込むこともできます。内部的には、システムの再起動を行うまで (もしくは次の modify や remove を行うまで) の範囲で保存が行われるほか、同じインターフェイスや同じサービスに対する設定が既に存在した場合、既存の設定は上書きされるようになっています。インターフェイスは `-i インターフェイス名` パラメータで、サービスは `-s サービス名` パラメータでそれぞれ指定します。

`remove`

`netconfig remove` コマンドは、指定したインターフェイスやサービスの組み合わせに対して、動的に編集した内容を削除してネットワーク設定を更新するコマンドです。インターフェイスは `-i インターフェイス名` パラメータで、サービスは `-s サービス名` パラメータでそれぞれ指定します。

`update`

`netconfig update` コマンドは、現在の設定を利用してネットワークの設定を更新するためのコマンドです。これはポリシーや固定の設定を変更した場合に使用するものです。特定のサービス (例: `dns` , `nis` , `ntp`) のみを更新したい場合は、`-m モジュール名` パラメータをお使いください。

netconfig のポリシーと固定の設定は手作業で設定することができるほか、YaST で設定することもできます。いずれも `/etc/sysconfig/network/config` ファイルを編集することになります。DHCP や PPP など自動設定ツールで提供される動的な設定は、それぞれ `netconfig modify` や `netconfig remove` のコマンドを介して動的に配信されます。NetworkManager が有効化されている場合は、netconfig (ポリシーモードが `auto` である場合) は NetworkManager 側の設定のみを使用し、従来の ifup 方式を利用したインターフェイスの設定は無視されるようになります。NetworkManager が何も設定を提供しない場合は、代替として固定の設定を使用します。NetworkManager と `wicked` を混在させて使用することはできません (サポートしていません)。

`netconfig` に関する詳細は、`man 8 netconfig` をお読みください。

13.6.2.10 `/etc/hosts`

例13.7「/etc/hosts」に示しているとおり、このファイルには IP アドレスとそれに対応するホスト名を記述します。ネットワーク内にネームサーバが存在していない場合、IP で接続する全てのホストをここに記述する必要があります。各行には IP アドレスと完全修飾ホスト名、およびホスト名単体をそれぞれ指定します。なお、IP アドレスは行頭に配置しなければならず、各項目の間はスペースもしくはタブで区切ります。コメントを記述したい場合は、行頭に `#` 記号を入力します。

例 13.7: `/etc/hosts`

```
127.0.0.1 localhost
192.168.2.100 jupiter.example.com jupiter
192.168.2.101 venus.example.com venus
```

13.6.2.11 `/etc/networks`

ここでは、ネットワーク名とネットワークアドレスを記述します。書式は `hosts` と同様ですが、ネットワーク名のほうを左側に記述します。詳しくは 例13.8「/etc/networks」をご覧ください。

例 13.8: `/etc/networks`

```
loopback    127.0.0.0
localnet    192.168.0.0
```

13.6.2.12 /etc/host.conf

このファイルでは、ホスト名やネットワーク名を解決するリゾルバライブラリの制御を行います。このファイルは libc4 や libc5 とリンクされているプログラムのみが使用します。現行の glibc プログラムの場合は `/etc/nsswitch.conf` ファイルで設定を行います。このファイルでは、それぞれのパラメータは別々の行に記述します。コメントは行頭に `#` を入力します。表13.2「/etc/host.conf で使用するパラメータ」には、利用可能なパラメータの一覧が示されています。また、`/etc/host.conf` の設定例は例13.9「/etc/host.conf」にあります。

表 13.2: /ETC/HOST.CONF で使用するパラメータ

order hosts , bind	名前解決の際のサービスの利用順序を指定します。指定可能な値は下記のとおりです (複数のものを指定する場合は、スペースもしくはカンマで区切ります):
	hosts : `/etc/hosts` ファイル内を検索します
	bind : ネームサーバにアクセスを行います
	nis : NIS を使用します
multi on / off	`/etc/hosts` ファイル内のホストに対して、複数の IP アドレスを設定できるようにするかどうかを指定します。
nospoof on spoofalert on / off	これらのパラメータは、ネームサーバの なりすまし に対する設定を表しますが、ネットワークの設定に対しては特に影響がありません。
trim ドメイン名	ホスト名の解決を行ったあと、ホスト名に指定したドメイン名が含まれていれば、そのドメイン名部分を分離する指定です。このオプションは、`/etc/hosts` ファイル内にローカルのドメイン名が存在しているものの、ドメイン名をそのまま認識させておきたいような場合にのみ使用します。

例 13.9: /etc/host.conf

```
# ネームサーバが存在する
order hosts bind
# 複数アドレスの許可
multi on
```

13.6.2.13 /etc/nsswitch.conf

GNU C Library 2.0 およびそれ以降のバージョンでは、Name Service Switch (NSS) と呼ばれる仕組みが導入されています。詳細については `nsswitch.conf(5)` のマニュアルページ、もしくは The GNU C Library Reference Manual (GNU C ライブラリリファレンスマニュアル) をお読みください。

問い合わせの順序は `/etc/nsswitch.conf` ファイルで設定します。`nsswitch.conf` の設定例については 例13.10「/etc/nsswitch.conf」をご覧ください。コメントは行頭に `#` を入力します。この例の `hosts` では、`/etc/hosts` (`files`) に問い合わせたのち、DNS (詳しくは第19章「ドメインネームシステム」をお読みください) に問い合わせを行います。

例 13.10: /etc/nsswitch.conf

```
passwd:     compat
group:      compat

hosts:      files dns
networks:   files dns

services:   db files
protocols:  db files
rpc:        files
ethers:     files
netmasks:   files
netgroup:   files nis
publickey:  files

bootparams: files
automount:  files nis
aliases:    files nis
shadow:     compat
```

NSS で利用可能な「データベース」の一覧は 表13.3「/etc/nsswitch.conf で使用できるデータベース」に示されています。また、NSS データベースで設定可能なオプションの一覧は、表13.4「NSS「データベース」での設定オプション」に示されています。

表 13.3: /ETC/NSSWITCH.CONF で使用できるデータベース

`aliases`	`sendmail` で実装されているメール別名機能。詳しくは `man 5 aliases` をお読みください。
`ethers`	イーサネットアドレス。
`netmasks`	ネットワークとそれらのサブネットマスク。サブネットを設定する場合にのみ使用します。

`group`	`getgrent` で問い合わせることのできるユーザグループ。詳しくは `group` のマニュアルページをお読みください。
`hosts`	`gethostbyname` などの関数で問い合わせることのできるホスト名と IP アドレス。
`netgroup`	アクセス権を制御するためのネットワーク内のホストとユーザのリスト。詳しくは `netgroup(5)` のマニュアルページをお読みください。
`networks`	`getnetent` で問い合わせることのできるネットワーク名とアドレス。
`publickey`	NFS および NIS+ で使用される Secure_RPC 向けの公開鍵と機密鍵。
`passwd`	`getpwent` で問い合わせることのできるユーザパスワード。詳しくは `passwd(5)` のマニュアルページをお読みください。
`protocols`	`getprotoent` で問い合わせることのできるネットワークプロトコル。詳しくは `protocols(5)` のマニュアルページをお読みください。
`rpc`	`getrpcbyname` などの関数で問い合わせることのできるリモートプロシージャコールの名前とアドレス。
`services`	`getservent` で問い合わせることのできるネットワークサービス。
`shadow`	`getspnam` で問い合わせることのできるユーザのパスワード。詳しくは `shadow(5)` のマニュアルページをお読みください。

表 13.4: NSS「データベース」での設定オプション

`files`	`/etc/aliases` 等のファイルへの直接アクセス
`db`	データベース経由でのアクセス

`nis`, `nisplus`	NIS (詳しくは『セキュリティ強化ガイド』、第3章「NIS の使用」をお読みください)
`dns`	`hosts` と `networks` に対する拡張としてのみ使用することができます
`compat`	`passwd`, `shadow`, `group` に対する拡張としてのみ使用することができます

13.6.2.14 `/etc/nscd.conf`

このファイルは nscd (ネームサービスキャッシュデーモン; name service cache daemon) を設定するために使用するものです。詳しくは `nscd(8)` や `nscd.conf(5)` のマニュアルページをお読みください。既定では、`passwd`, `groups`, `hosts` の各システム項目がキャッシュ対象となります。これは NIS や LDAP などのディレクトリサービスの性能を確保するためには重要な設定となります。キャッシュを使用しないと、名前やグループ、ホストにアクセスがあるごとにネットワークへの接続が発生することになってしまうためです。

`passwd` に対するキャッシュ機能を有効化した場合、新しくローカルユーザを追加した場合は、それが反映されるまでに 15 秒程度の時間がかかります。この待機を行いたくない場合は、下記のようにして nscd を再起動してください:

```
> sudo systemctl restart nscd
```

13.6.2.15 `/etc/hostname`

`/etc/hostname` には完全修飾ホスト名 (FQHN) が含まれています。完全修飾ホスト名とはホスト名にドメイン名が付加されたものを意味しています。このファイルには、設定したい完全修飾ホスト名を 1 行だけ記述してください。また、このファイルはマシンの起動時に読み込まれます。

13.6.3 設定のテスト

設定ファイル内に独自の設定を行う前に、事前にテストを行っておくことをお勧めします。テスト設定を行うには `ip` コマンドを、接続をテストするには `ping` コマンドを使用します。

`ip` コマンドは、設定ファイルに設定を保存することなく、ネットワークの設定を直接変更します。そのため、変更した設定はシステムを再起動したり仮想アダプタのリセットをしたりすると、元に戻ってしまいます。

注記: `ifconfig` と `route` の廃止予定について

`ifconfig` と `route` の各ツールは、廃止予定としてマークされています。代わりに `ip` コマンドをお使いください。また、`ifconfig` コマンドは、インターフェイス名の長さが 9 文字までに制限されています。

13.6.3.1 `ip` を利用したネットワークインターフェイスの設定

`ip` コマンドは、ネットワークデバイスやルーティング、ポリシールーティングやトンネルなどを設定したり、設定を取得したりするためのツールです。

`ip` コマンドは非常に複雑なツールです。一般的には `ip オプション オブジェクト コマンド` の形式で実行します。オブジェクトには下記のものがあります:

link
: このオブジェクトは、ネットワークデバイスそのものを表します。

address
: このオブジェクトは、デバイスの IP アドレスを表します。

neighbor
: このオブジェクトは、ARP や NDISC のキャッシュ項目などを表します。

route
: このオブジェクトは、ルーティングテーブルの項目を表します。

rule
: このオブジェクトは、ルーティングポリシーデータベース内のルールを表します。

maddress
: このオブジェクトは、マルチキャストのアドレスを表します。

mroute
: このオブジェクトは、マルチキャストのルーティングキャッシュ項目を表します。

tunnel
: このオブジェクトは、IP 経由でのトンネルを表します。

何もコマンドを指定しない場合は、指定のコマンド (通常は `list`) が指定されているものと見なします。

デバイスの状態を変更するには、下記のコマンドを実行します:

```
> sudo ip link set デバイス名
```

たとえばデバイス eth0 を無効化したい場合は、下記のように入力して実行します:

```
> sudo ip link set eth0 down
```

有効化したい場合は、下記のように入力して実行します:

```
> sudo ip link set eth0 up
```

ヒント: NIC デバイスの切断について

デバイスを無効化するには、下記のように入力して実行します:

```
> sudo ip link set デバイス名 down
```

ただし、このコマンドはネットワークインターフェイスをソフトウエアレベルで無効化するだけです。イーサネットケーブルが取り外された場合や、スイッチの電源が切れた場合のように、リンクを失った場合の処理を行わせたい場合は、下記のように入力して実行します:

```
> sudo ip link set デバイス名 carrier off
```

なお、`ip link set デバイス名 down` は `デバイス名` を経由する全ての経路 (ルーティング) 情報を削除しますが、`ip link set デバイス名 carrier off` ではそのようなことは行いません。また `carrier off` コマンドは、対応するネットワークデバイスのドライバ側での対応が必要となります。

デバイスに対する物理的な接続を復帰させるには、下記のように入力して実行します:

```
> sudo ip link set デバイス名 carrier on
```

デバイスを有効化したあとは、様々な設定作業を行うことができます。IP アドレスを設定するには、下記のように入力して実行します:

```
> sudo ip addr add IP_アドレス + dev デバイス名
```

たとえば標準ブロードキャストオプション (オプション `brd`) を指定して、インターフェイス eth0 に 192.168.12.154/30 というアドレスを設定するには、下記のように入力して実行します:

```
> sudo ip addr add 192.168.12.154/30 brd + dev eth0
```

ネットワーク接続を正しく動作させるには、通常はデフォルトゲートウエイの設定も行わなくてはなりません。お使いのシステムにデフォルトゲートウエイを設定するには、下記のようなコマンドを入力して実行します:

```
> sudo ip route add default via デフォルトゲートウエイの_IP_アドレス
```

全てのデバイスを一覧表示するには、下記のように入力して実行します:

```
> sudo ip link ls
```

動作中のインターフェイスのみを表示したい場合は、下記のように入力して実行します:

```
> sudo ip link ls up
```

特定のデバイスに対する統計情報を表示するには、下記のように入力して実行します:

```
> sudo ip -s link ls デバイス名
```

仮想ネットワークデバイスなどの追加情報を表示するには、下記のように入力して実行します:

```
> sudo ip -d link ls デバイス名
```

上記に加えてデバイスに設定されたネットワーク層 (IPv4, IPv6) のアドレスも表示したい場合は、下記のように入力して実行します:

```
> sudo ip addr
```

上記の出力には、各デバイスの MAC アドレスに関する情報も表示されます。また、全ての経路 (ルーティング) 情報を表示するには、下記のように入力して実行します:

```
> sudo ip route show
```

`ip` コマンドの使用方法について、詳しくは `ip help` を実行すると表示されるヘルプ、もしくは `man 8 ip` コマンドで表示されるマニュアルページをお読みください。このほか、`ip` コマンドのサブコマンドの後ろに `help` オプションを指定することもできます。たとえば下記のように入力して実行することができます:

```
> sudo ip addr help
```

それ以外にも、`/usr/share/doc/packages/iproute2/ip-cref.pdf` ファイルには `ip` コマンドのマニュアルも用意されています。

13.6.3.2 ping による通信テスト

`ping` コマンドは、TCP/IP の接続が動作しているかどうかをテストすることができる標準的なツールです。このツールは ICMP プロトコルを利用して、ECHO_REQUEST データグラムと呼ばれる小さなデータパケットを宛先のホストに送信し、相手側からの即時の応答を待ちます。応答があると、`ping` はその旨を示すメッセージを表示します。これにより、ネットワークが正しく動作していることを確認することができます。

`ping` は 2 台のコンピュータ間での接続機能テストを行うだけではありません。基本的な接続品質に関する情報も提供します。例13.11「ping コマンドの出力」には `ping` の出力例を示していますが、出力の末尾から 2 行目には、送信したパケット数と損失数、および `ping` の実行にかかった時間が表示されます。

また、宛先の指定には IP アドレスだけでなくホスト名を指定することもできます。たとえば`ping example.com` や `ping 192.168.3.100` のように実行することができます。また、このプログラムは `Ctrl` – `C` を押すまでパケットを送り続けます。

接続ができるかどうかだけを調べたい場合は、`-c` オプションでパケットの送信回数を制限してください。たとえば 3 回パケットを送信したい場合は、`ping -c 3 example.com` のように入力して実行します。

例 13.11: PING コマンドの出力

```
ping -c 3 example.com
PING example.com (192.168.3.100) 56(84) bytes of data.
64 bytes from example.com (192.168.3.100): icmp_seq=1 ttl=49 time=188 ms
64 bytes from example.com (192.168.3.100): icmp_seq=2 ttl=49 time=184 ms
64 bytes from example.com (192.168.3.100): icmp_seq=3 ttl=49 time=183 ms
--- example.com ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2007ms
rtt min/avg/max/mdev = 183.417/185.447/188.259/2.052 ms
```

既定の送信間隔は 1 秒に設定されています。この間隔を変更したい場合は、`-i` オプションを指定してください。たとえば送信間隔を 10 秒にしたい場合は、`ping -i 10 example.com` のように入力して実行します。

なお、システムに複数のネットワークデバイスが存在する場合は、インターフェイスを指定して送信したほうが便利である場合があります。これを行うには、`-I` とデバイス名を指定してください。たとえば `ping -I wlan1 example.com` のように入力して実行します。

`ping` の使用方法について、詳しくは `ping -h` で表示されるヘルプ、もしくは `ping (8)` のマニュアルページをお読みください。

ヒント: IPv6 アドレスへの ping

IPv6 アドレスの場合は、`ping6` コマンドを使用します。ただし、リンクローカルアドレスに対して送信したい場合は、`-I` でインターフェイス名を指定してください。たとえば下記のコマンドが成功した場合は、`eth1` を介して指定したアドレスに到達できる意味になります:

```
ping6 -I eth1 fe80::117:21ff:feda:a425
```

13.6.4 ユニットファイルと起動スクリプト

上述までの設定のほかに、マシンを起動する際にネットワークサービスを読み込むための systemd ユニットファイルと各種スクリプトが存在しています。これらはシステムが `multi-user.target` ターゲットに切り替わる際に開始されます。これらのユニットファイルやスクリプトファイルのいくつかは、ネットワークプログラム向けのユニットファイルと起動スクリプト で説明しています。なお、`systemd` についての詳細は、第10章「systemd デーモン」をお読みください。また、`systemd` のターゲットに関する詳細は、`systemd.special` のマニュアルページ (`man systemd.special`) をお読みください。

ネットワークプログラム向けのユニットファイルと起動スクリプト

`network.target`

`network.target` はネットワーク処理のための systemd ターゲットですが、この意味合いはシステム管理者が設定した内容によって異なります。
詳しくは https://www.freedesktop.org/wiki/Software/systemd/NetworkTarget/ をお読みください。

`multi-user.target`

`multi-user.target` は systemd ターゲットのうちの 1 つで、全てのネットワークサービスを有効化したマルチユーザシステムを開始するためのものです。

`rpcbind`

RPC のプログラム番号をユニバーサルアドレスに変換する `rpcbind` ユーティリティを起動します。NFS サーバなどで必要となります。

`ypserv`

NIS サーバを開始します。

`ypbind`

NIS クライアントを開始します。

`/etc/init.d/nfsserver`
: NFS サーバを開始します。

`/etc/init.d/postfix`
: postfix プロセスを制御します。

13.7 基本的なルータの構築

ルータとはネットワークデバイスの一種で、一方のインターフェイスからデータ (ネットワークパケット) を受信して、他方 (複数の場合もある) に送信して中継する機器です。ルータはローカルのネットワークからリモートのネットワーク (インターネット) に接続する際に一般的に使用されているほか、ローカルのネットワーク同士を接続する場合にも用いられることがあります。openSUSE Leap でも、NAT (ネットワークアドレス変換; Network Address Translation) や高度なファイアウオール機能の付属したルータを構築することができます。

openSUSE Leap をルータにするには、下記の基本的な手順を踏む必要があります。

1. まずはパケットの転送機能を有効化します。たとえば `/etc/sysctl.d/50-router.conf` 内で下記のように設定します:

   ```
   net.ipv4.conf.all.forwarding = 1
   net.ipv6.conf.all.forwarding = 1
   ```

   あとはそれぞれのインターフェイスに対して、IPv4 と IPv6 の固定アドレスを設定します。パケットの転送機能を有効化すると、IPv6 の RA (ルータ告知; Router Advertisement) などのいくつかの仕組みが無効化され、既定のルートも自動では設定されなくなります。

2. 多くの場合 (たとえば複数のインターフェイスを介して同じネットワークに到達できるような場合や、VPN を一般的に使用している「一般的なマルチホームホスト」の場合など) において、IPv4 のリバースパスフィルタを無効化しなければなりません (この機能は、IPv6 では現在存在していません):

   ```
   net.ipv4.conf.all.rp_filter = 0
   ```

   ファイアウオールの設定でフィルタすることもできます。

3. 外部やアップリンク、ISP などから IPv6 RA を受け付けて、IPv6 のデフォルトルートを作成するようにするには、下記のように設定します:

   ```
   net.ipv6.conf.${ifname}.accept_ra = 2
   net.ipv6.conf.${ifname}.autoconf = 0
   ```

> (注意:「`eth0.42`」のようなインターフェイスの場合は、`eth0/42` のように sysfs パス形式で記述する必要があります)

さらなるルータの動作に転送ポリシーなどについて、詳しくは https://www.kernel.org/doc/Documentation/networking/ip-sysctl.txt をお読みください。

IPv6 を内部 (もしくは DMZ) インターフェイスに提供し、自分自身を IPv6 ルータとして動作させて「autoconf ネットワーク」を提供するには、`radvd` をインストールして `/etc/radvd.conf` を下記のように設定する必要があります:

```
interface eth0
{
    IgnoreIfMissing on;         # インターフェイスが存在していなくてもエラーにしない

    AdvSendAdvert on;           # RA の送信の有効化
    AdvManagedFlag on;          # DHCPv6 での IPv6 アドレスの管理
    AdvOtherConfigFlag on;      # DNS, NTP などは DHCPv6 のみ

    AdvDefaultLifetime 3600;    # クライアント側でのデフォルトルートの有効期間 (1 時間)

    prefix 2001:db8:0:1::/64    # (/64 は既定値であり、 autoconf でも必要となる)
    {
        AdvAutonomous off;          # アドレス autoconf の無効化 (DHCPv6 のみ)

        AdvValidLifetime 3600;      # プレフィクス (autoconf アドレス) の有効期間 (1 時間)
        AdvPreferredLifetime 1800; # プレフィクス (autoconf アドレス) の更新期間はその半分 (30
 分)
    }
}
```

また、NAT を利用して LAN から WAN にパケットを転送し、それと共にアドレス変換を行い、さらに WAN 側からの入力パケットをブロックしたい場合は、下記のように実行します:

```
> sudo firewall-cmd --permanent --zone=external --change-interface=WAN_インターフェイス
> sudo firewall-cmd --permanent --zone=external --add-masquerade
> sudo firewall-cmd --permanent --zone=internal --change-interface=LAN_インターフェイス
> sudo firewall-cmd --reload
```

13.8 ボンドデバイスの設定

システムによっては、一般的なイーサネットデバイスにおける標準的なセキュリティや可用性の要件を越えて、ネットワーク接続を実装する要望がある場合があります。このような場合は、複数のイーサネットデバイスをまとめて、1 つのボンドデバイスにすることができます。

ボンドデバイスの設定はボンドモジュールを設定して行います。また、このモジュールの動作はモードによって切り替えることができます。既定では `active-backup` (アクティブ-バックアップ) モードに設定されていて、一方のデバイスに障害が発生した場合に、他方のデバイスに切り替える動作を行います。モードには下記の設定が用意されています:

[0] (balance-rr)
: パケットはラウンドロビン形式で、最初のインターフェイスから最後のインターフェイスまで、パケットを 1 つずつ送りながらインターフェイスを切り替える動作になります。これにより、冗長化と負荷分散の両方を実現することができます。

[1] (active-backup)
: いずれか 1 つのネットワークインターフェイスのみを有効に設定します。そのネットワークインターフェイスに障害が発生すると、異なるインターフェイスが有効化されます。この設定は openSUSE Leap における既定値で、冗長化のみを実現することができます。

[2] (balance-xor)
: パケットはボンド内に含まれているデバイス数を基準に、利用可能な全てのインターフェイスに分散されます。接続するスイッチ側での対応が必要です。冗長化と負荷分散の両方を提供します。

[3] (broadcast)
: 全てのパケットを全てのインターフェイスに送信します。接続するスイッチ側での対応が必要です。冗長化のみを実現することができます。

[4] (802.3ad)
: 同じ速度と二重通信方式を共有する複数のインターフェイスを、1 つにまとめる設定です。インターフェイスドライバ側に `ethtool` のサポートが必要となるほか、接続するスイッチ側で IEEE 802.3ad 動的リンク集約の設定を行う必要があります。冗長化と負荷分散の両方を提供します。

[5] (balance-tlb)
: 順応型送信負荷分散を行います。インターフェイスドライバ側に `ethtool` のサポートが必要となりますが、スイッチ側での対応は不要です。冗長化と負荷分散の両方を提供します。

[6] (balance-alb)
: 順応型負荷分散を行います。インターフェイスドライバ側に `ethtool` のサポートが必要となりますが、スイッチ側での対応は不要です。冗長化と負荷分散の両方を提供します。

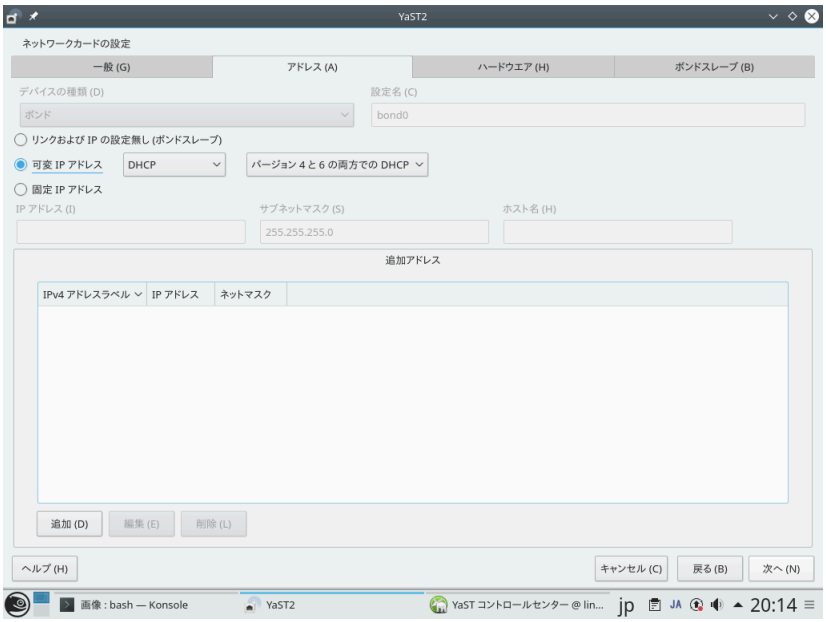
モードに関する詳細な説明については、https://www.kernel.org/doc/Documentation/networking/bonding.txt をお読みください。

ヒント: ボンドと Xen

ボンドデバイスの使用は、複数のネットワークカードを接続しているマシンでのみ設定することができる仕様です。ほとんどの設定において、ボンドデバイスは Dom0 でのみ使用すべきものであることを意味します。VM ゲスト システムに対して複数のネットワークカードを割り当てている場合にのみ、VM ゲスト 内でボンディングを設定することができることになります。

ボンドデバイスを設定するには、下記の手順で行います:

1. [YaST] › [システム] › [ネットワークの設定] を実行します。
2. [追加] を押し、[デバイスの種類] に [ボンド] を選択して、[次へ] を押します。



3. ボンドデバイスへの IP アドレスの割り当て方法を指定します。下記の 3 種類の中から選択します:

   - IP の設定無し
   - 可変 IP アドレス (DHCP もしくは Zeroconf)
   - 固定 IP アドレス

   お使いの環境に適切なものを選択してください。
4. また、[Bond ポート] のタブでは、ボンドデバイスに含めるべきインターフェイスを、チェックボックスで選択します。
5. [ボンドドライバのオプション] を編集して、モードを設定します。

6. なお、［ボンドドライバのオプション］の末尾に `miimon=100` のパラメータが存在していることを確認してください。このパラメータを指定しないと、定期的なチェックが行われなくなります。

7. ［Next］を押して進み、［OK］を押して YaST を終了すると、デバイスを作成することができます。

13.8.1 ボンドポートのホットプラグ

ネットワーク環境 (たとえば高可用性が必要な環境) によっては、障害の発生したネットワークインターフェイスを別のものに交換する必要がある場合があります。このような要件を満たすには、ボンドポートのホットプラグを設定して解決する必要があります。

ボンドの設定を通常通りに行います (詳しくは `man 5 ifcfg-bonding` をお読みください) 。たとえば下記のようになります:

```
ifcfg-bond0
          STARTMODE='auto' # or 'onboot'
          BOOTPROTO='static'
          IPADDR='192.168.0.1/24'
          BONDING_MASTER='yes'
          BONDING_SLAVE_0='eth0'
          BONDING_SLAVE_1='eth1'
          BONDING_MODULE_OPTS='mode=active-backup miimon=100'
```

ボンドポート側には `STARTMODE=hotplug` と `BOOTPROTO=none` を指定します:

```
ifcfg-eth0
          STARTMODE='hotplug'
          BOOTPROTO='none'

ifcfg-eth1
          STARTMODE='hotplug'
          BOOTPROTO='none'
```

`BOOTPROTO=none` を指定すると、`ethtool` を使用して設定が行われますが、`ifup eth0` で起動が行われなくなります。これはボンドポートのインターフェイスはボンドデバイス側で制御されるべきであるためです。

`STARTMODE=hotplug` を指定すると、ボンドポートインターフェイスが利用可能な状態になった際に、自動的にボンドデバイスに参加するようになります。

また、`/etc/udev/rules.d/70-persistent-net.rules` 内にある `udev` のルールを、MAC アドレスではなくバス ID を指定する形式に変更する必要があります (udev の `KERNELS` キーに、`hwinfo --netcard` で出力される "SysFS のバス ID"を指定します) 。これにより、障害の発生した

ハードウエア (同じスロットにありながら、MAC アドレスの異なるハードウエア) を交換することができるようになり、MAC アドレスが変更されることによるボンドドライバ側の問題を回避できるようになります。

たとえば下記のようになります:

```
SUBSYSTEM=="net", ACTION=="add", DRIVERS=="?*",
KERNELS=="0000:00:19.0", ATTR{dev_id}=="0x0", ATTR{type}=="1",
KERNEL=="eth*", NAME="eth0"
```

システムの起動時点では、systemd の `network.service` はホットプラグ指定されたボンドポートを待機しませんが、ボンドインターフェイスを動作させるためには、少なくとも 1 つ以上のボンドポートが必要となります。ボンドポートのいずれかが取り外される (NIC ドライバからバインド解除される、つまり NIC のドライバが `rmmod` されるか、PCI ホットプラグで取り外される) と、カーネルはボンドデバイスについても自動的に削除を行います。新しいカードがシステムに接続される (置き換えとして同じスロットに接続される) と、`udev` はバス ID ベースの固定名を付与して `ifup` を呼び出します。`ifup` 側では、ボンドデバイス側への参加を自動的に行います。

13.8.2 予測可能なインターフェイス命名方式

ネットワークインターフェイスに対して恒久的な名前を設定する機能とボンディングを併用しようとすると問題が発生します。これは、インターフェイスがボンドデバイスに参加する際、MAC アドレスはボンドデバイスのものに一時的に上書きされるため、`70-persistent-net.rules` において MAC アドレスベースの命名を行っている場合、うまく動作しなくなってしまうためです。

なお、MAC アドレス変更済みの NIC に対して `uevent` が実行された場合、`udev` は暫定的なインターフェイス名 (例: `rename4`) を設定します。また、MAC アドレスベースのルールを避けて別の方式で恒久的な名前を設定しようとするのも現実的ではありません。このような事情から、恒久的な命名方式は廃止予定とされ、新たに予測可能な命名方式に置き換えられるようになっています。これにより設定の可能性が広がっているほか、MAC アドレスにも依存しなくなっています。予測可能な命名方式は、新規システムのインストール時に **起動オプション** に `net.ifnames=1` を設定することで有効化できます。

手順 13.1: インストール済みのシステムにおける命名方式の有効化

1. `biosdevname` パッケージがインストールされている場合、アンインストールします。

   ```
   > sudo zypper rm biosdevname
   ```

2. `/etc/udev/rules.d` 内に既存の命名方式が設定されている場合は、まずバックアップしておきます。たとえば下記のように実行します:

   ```
   > sudo cp /etc/udev/rules.d/70-persistent-net.rules /backup
   ```

注記

`/etc/udev/rules.d/70-persistent-net.rules` 内の既存の命名方式があれば、これらを削除して予測可能なインターフェイス命名方式に切り替えます。

3. initrd を生成し直します。

```
> sudo dracut -f
```

4. あとは YaST ブートローダモジュールを起動して、カーネルのコマンドラインに `net.ifnames=1` を追加します。作業が終わったら［OK］を押してシステムを再起動してください。

5. なお、Wicked をネットワークマネージャとして使用している場合は、ネットワークインターフェイスの設定も変更しておく必要があります。`yast lan` モジュールを実行して設定し直すか、もしくは `/etc/sysconfig/network/ifcfg-*` のファイル名を変更してください。ファイル名を変更したあとは、下記のようにして Wicked を再起動してください。

```
> sudo systemctl restart wicked.service
```

13.9 ネットワークチーミング によるチームデバイス (チーミング／ボンディング) の設定

「リンクアグリゲーション」とは、複数のネットワーク接続を組み合わせて (もしくは束ねて) 1 つの論理レイヤを提供する一般的な用語です。これは「チャンネルチーミング」,「イーサネットボンディング」,「ポートトランキング」などと呼ばれることもありますが、いずれも同じような意味であり、同じような意図で作られている仕組みです。

これらの考え方は一般に「ボンディング」として知られているもので、元々は Linux カーネル内に組み込まれている機能を意味します (ボンディングについての詳細は、13.8項「ボンドデバイスの設定」をお読みください) 。ネットワークチーミング という用語は、この考え方をさらに進化させた実装を意味しています。

ボンディングと ネットワークチーミング の大きな違いとして挙げられるのは、チーミングは teamd のインスタンスに対してインターフェイスを提供するための小さなカーネルモジュールが複数存在するという点です。それ以外の処理は全てユーザスペース側で行います。ボンディングはこれとは異なり、全ての機能をカーネル側で行っています。詳しい比較については、表13.5「ボンディングとチーミングの機能比較」をお読みください。

表 13.5: ボンディングとチーミングの機能比較

機能	ボンディング	チーミング
ブロードキャスト型／ラウンドロビン型送信ポリシー	はい	はい
アクティブ-バックアップ型送信ポリシー	はい	はい
LACP (802.3ad) サポート	はい	はい
ハッシュベースの送信ポリシー	はい	はい
ユーザ側でのハッシュ関数の設定	いいえ	はい
送信負荷分散 (TLB) サポート	はい	はい
LACP 向けの送信負荷分散サポート	いいえ	はい
ethtool リンク監視	はい	はい
ARP リンク監視	はい	はい
NS/NA (IPV6) リンク監視	いいえ	はい
TX/RX パスにおける RCU ロッキング	いいえ	はい
ポートの優先順位と固着設定	いいえ	はい
個別／ポート別リンク監視設定	いいえ	はい
複数リンクの監視設定	制限あり	はい
VLAN サポート	はい	はい
複数デバイスのスタック	はい	はい
情報源: https://libteam.org/files/teamdev.pp.pdf		

ボンディングと ネットワークチーミング の実装は、同時に使用することもできます。また、ネットワークチーミング は既存のボンディングに対する代替としても使用することができますが、置き換えるようなものではありません。

ネットワークチーミング には様々な用途があります。本章では下記にある最も重要な 2 つの用途を説明しています:

- 異なるネットワークデバイス間の負荷分散
- デバイスに障害が発生した場合の、一方から他方へのネットワークデバイスの切り替え

現時点では、チーミングデバイスを作成するための YaST モジュールは用意されていません。ネットワークチーミング をご利用になる場合は、手作業で設定を行う必要があります。なお、下記には全てのネットワークチーミング 設定で利用することのできる、一般的な手順を説明しています:

手順 13.2: 一般的な手順

1. まずは `libteam-tools` パッケージをインストールします:

   ```
   > sudo zypper in libteam-tools
   ```

2. 次に `/etc/sysconfig/network/` ディレクトリ内に設定ファイルを作成します。一般的には `ifcfg-team0` のようなファイル名で作成します。複数の ネットワークチーミング デバイスを作成する場合は、末尾の数字を 1 つずつ増やしながら作成してください。
   この設定ファイルには、マニュアルページで説明されている各種の指定を記述します。詳しくは `man ifcfg` および `man ifcfg-team` をお読みください。また、お使いのシステム内には、設定ファイルのサンプル (`/etc/sysconfig/network/ifcfg.template`) も用意されていますので、こちらもあわせてお読みください。

3. チーミングデバイス内に組み込むインターフェイスの設定ファイルを削除します。たとえば `ifcfg-eth0` や `ifcfg-eth1` などを削除してください。
   なお、いずれのファイルとも、バックアップを取ってから削除することをお勧めします。Wicked 側では、チーミングに必要なパラメータで設定ファイルを再作成します。

4. なお、必要であれば Wicked の設定ファイルとして取り込まれていることを確認します:

   ```
   > sudo wicked show-config
   ```

5. ネットワークチーミング デバイス `team0` を開始するため、下記のコマンドを実行します:

   ```
   > sudo wicked ifup team0
   ```

   デバッグ情報を取得する必要がある場合は、`all` サブコマンドの後ろに `--debug all` を指定してください。

6. ネットワークチーミング デバイスの状態を確認します。具体的には下記のコマンドを実行します:

   - まずは Wicked の teamd インスタンスの状態を確認します:

     ```
     > sudo wicked ifstatus --verbose team0
     ```

   - インスタンス全体の状態を確認します:

     ```
     > sudo teamdctl team0 state
     ```

   - teamd インスタンスの systemd の状態を確認します:

     ```
     > sudo systemctl status teamd@team0
     ```

   それぞれは、お使いの環境によって出力が少しずつ異なります。

7. 何らかの理由で `ifcfg-team0` を書き換える必要がある場合は、書き換えた後に下記のコマンドを実行して、設定ファイルを再読み込みさせてください:

   ```
   > sudo wicked ifreload team0
   ```

ただし、チーミングデバイスの開始や停止に際しては、`systemctl` を 使用しないでください 。代わりに、上述のとおり `wicked` コマンドを使用してください。

チーミングデバイスを削除するには、下記の手順で行います:

手順 13.3: チーミングデバイスの削除

1. まずは ネットワークチーミング デバイス (例: `team0`) を停止します:

   ```
   > sudo wicked ifdown team0
   ```

2. 設定ファイル `/etc/sysconfig/network/ifcfg-team0` を `/etc/sysconfig/network/.ifcfg-team0` のように名前変更します。ファイル名の冒頭にドットを入れることで、wicked からは読み込みができなくなります。完全に不要になった段階で、ファイルを削除してください。

3. 設定を再読み込みします:

   ```
   > sudo wicked ifreload all
   ```

13.9.1 使用例: ネットワークチーミング による負荷分散

負荷分散は帯域を増やすための仕組みです。下記のような設定で ネットワークチーミング デバイスを作成すると、負荷分散の機能を設定することができます。デバイスの開始や停止、設定ファイルの配置などについては、手順13.2「一般的な手順」をお読みください。また、設定後は `teamdctl` の出力もご確認ください。

例 13.12: ネットワークチーミング による負荷分散の設定例

```
STARTMODE=auto ❶
BOOTPROTO=static ❷
IPADDRESS="192.168.1.1/24" ❷
IPADDR6="fd00:deca:fbad:50::1/64" ❷

TEAM_RUNNER="loadbalance" ❸
TEAM_LB_TX_HASH="ipv4,ipv6,eth,vlan"
TEAM_LB_TX_BALANCER_NAME="basic"
TEAM_LB_TX_BALANCER_INTERVAL="100"

TEAM_PORT_DEVICE_0="eth0" ❹
TEAM_PORT_DEVICE_1="eth1" ❹

TEAM_LW_NAME="ethtool" ❺
TEAM_LW_ETHTOOL_DELAY_UP="10" ❻
TEAM_LW_ETHTOOL_DELAY_DOWN="10" ❻
```

❶ チーミングデバイスの開始方法を制御しています。`auto` はインターフェイスを自動起動する意味になり、システムの起動時にネットワークサービスが開始されると、インターフェイスが自動的に設定され開始されます。
デバイスを自動起動せず、手動で起動する必要がある場合は、`STARTMODE` の値を `manual` にしてください。

❷ 固定の IP アドレスを設定しています (ここでは IPv4 アドレスとして `192.168.1.1` を、IPv6 アドレスとして `fd00:deca:fbad:50::1` をそれぞれ設定しています) 。
ネットワークチーミング デバイスで動的な IP アドレスを使用する場合は、`BOOTPROTO="dhcp"` を追加したあと、`IPADDRESS` と `IPADDR6` の行を削除 (もしくはコメントアウト) してください。

❸ `TEAM_RUNNER` の値を `loadbalance` に設定することで、負荷分散モードを有効化しています。

❹ ネットワークチーミング デバイスを構成する 1 つ以上のデバイスを指定しています。

❺ 従属するデバイスの状態を開始するためのリンク監視機構を定義しています。既定値の `ethtool` では、デバイスのリンクと存在のみを監視します。通常はこの設定で問題ありませんが、デバイスから本当にパケットを送受信できるかどうかはチェックしていないことに注意してください。

この接続の信頼性を高めたい場合は、`arp_ping` オプションを設定してください。これは特定のホスト (ホストは `TEAM_LW_ARP_PING_TARGET_HOST` で設定します) に対して ping を送信するための仕組みで、この ping が成功することをもって、ネットワークが正常に接続されていると判断します。

❻ リンクが確立し (もしくはリンクが外れ) てから、リンク監視機構がそれを確認するまでの時間をミリ秒単位で指定します。

13.9.2 使用例: ネットワークチーミング による冗長構成

冗長構成は ネットワークチーミング デバイスで高可用性を実現するための仕組みで、予備のネットワークデバイスを用意して障害に耐える仕組みを作るものです。予備のネットワークデバイスは常に動作し続け、メインのデバイスに障害が発生すると通信を引き継ぐようになります。

下記のような設定で ネットワークチーミング デバイスを作成すると、冗長構成の機能を設定することができます。デバイスの開始や停止、設定ファイルの配置などについては、手順13.2「一般的な手順」をお読みください。また、設定後は `teamdctl` の出力もご確認ください。

例 13.13: ネットワークチーミング デバイスに対する冗長構成 の設定例

```
STARTMODE=auto ❶
BOOTPROTO=static ❷
IPADDR="192.168.1.2/24" ❷
IPADDR6="fd00:deca:fbad:50::2/64" ❷

TEAM_RUNNER=activebackup ❸
TEAM_PORT_DEVICE_0="eth0" ❹
TEAM_PORT_DEVICE_1="eth1" ❹

TEAM_LW_NAME=ethtool ❺
TEAM_LW_ETHTOOL_DELAY_UP="10" ❻
TEAM_LW_ETHTOOL_DELAY_DOWN="10" ❻
```

❶ チーミングデバイスの開始方法を制御しています。`auto` はインターフェイスを自動起動する意味になり、システムの起動時にネットワークサービスが開始されると、インターフェイスが自動的に設定され開始されます。
デバイスを自動起動せず、手動で起動する必要がある場合は、`STARTMODE` の値を `manual` にしてください。

❷ 固定の IP アドレスを設定しています (ここでは IPv4 アドレスとして `192.168.1.2` を、IPv6 アドレスとして `fd00:deca:fbad:50::2` をそれぞれ設定しています) 。
ネットワークチーミング デバイスで動的な IP アドレスを使用する場合は、`BOOTPROTO="dhcp"` を追加したあと、`IPADDRESS` と `IPADDR6` の行を削除 (もしくはコメントアウト) してください。

❸ `TEAM_RUNNER` の値を `activebackup` に設定することで、冗長構成モードを有効化しています。

❹ ネットワークチーミング デバイスを構成する 1 つ以上のデバイスを指定しています。

❺ 従属するデバイスの状態を開始するためのリンク監視機構を定義しています。既定値の `ethtool` では、デバイスのリンクと存在のみを監視します。通常はこの設定で問題ありませんが、デバイスから本当にパケットを送受信できるかどうかはチェックしていないことに注意してください。
この接続の信頼性を高めたい場合は、`arp_ping` オプションを設定してください。これは特定のホスト (ホストは `TEAM_LW_ARP_PING_TARGET_HOST` で設定します) に対して ping を送信するための仕組みで、この ping が成功することをもって、ネットワークが正常に接続されていると判断します。

❻ リンクが確立し (もしくはリンクが外れ) てから、リンク監視機構がそれを確認するまでの時間をミリ秒単位で指定します。

13.9.3 使用例: チーミングデバイスでの VLAN

VLAN とは 仮想ローカルエリアネットワーク (Virtual Local Area Network) の略で、複数の 論理イーサネットを単一の物理イーサネットで賄う仕組みです。この仕組みにより、1 つのネットワークを複数の異なるネットワークに分割して、スイッチ側では同じ VLAN 同士の通信のみを中継するような動作を実現することができます。

下記の使用例では、チーミングデバイスから 2 つの固定 VLAN を作成します:

- `vlan0`: IP アドレス=`192.168.10.1`
- `vlan1`: IP アドレス=`192.168.20.1`

下記の手順で行います:

1. まずはスイッチ側で VLAN タグ機能を有効化します。また、チーミングデバイスで負荷分散を実施している場合は、Link Aggregation Control Protocol (LACP) (802.3ad) に対応したスイッチでなければなりません。詳しくはハードウエアのマニュアルをお読みください。

2. 次にチーミングデバイスで負荷分散や冗長構成を使用するかどうかを決めます。それぞれの設定を行うには、13.9.1項「使用例: ネットワークチーミング による負荷分散」や 13.9.2項「使用例: ネットワークチーミング による冗長構成」をお読みください。

3. `/etc/sysconfig/network` ディレクトリ内に `ifcfg-vlan0` という設定ファイルを作成し、下記の内容を記述します:

```
STARTMODE="auto"
```

```
BOOTPROTO="static" ❶
IPADDR='192.168.10.1/24' ❷
ETHERDEVICE="team0" ❸
VLAN_ID="0" ❹
VLAN='yes'
```

❶ 固定の IP アドレスである旨を指定しています (実際のアドレスは `IPADDR` で指定します) 。

❷ IP アドレスとネットマスクを指定しています。

❸ VLAN インターフェイスに使用する実インターフェイスを指定しています。ここではチーミングデバイス (`team0`) を指定しています。

❹ VLAN のユニーク ID を指定します。なお、ファイル名と `VLAN_ID` の値は、`ifcfg-vlanVLAN_ID の値` のように対応させておくことをお勧めします。この例では `VLAN_ID` を `0` に設定していますので、ファイル名は `ifcfg-vlan0` のようになります。

4. 設定ファイル `/etc/sysconfig/network/ifcfg-vlan0` を `/etc/sysconfig/network/ifcfg-vlan1` にコピーして、コピーした先のファイルを下記のとおり変更します:

   - `IPADDR` の値を `192.168.10.1/24` から `192.168.20.1/24` に変更します。
   - `VLAN_ID` を `0` から `1` に変更します。

5. あとは 2 つの VLAN を開始します:

```
# wicked ifup vlan0 vlan1
```

6. `ifconfig` の出力を確認します:

```
# ifconfig -a
[...]
vlan0     Link encap:Ethernet  HWaddr 08:00:27:DC:43:98
          inet addr:192.168.10.1 Bcast:192.168.10.255 Mask:255.255.255.0
          inet6 addr: fe80::a00:27ff:fedc:4398/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:12 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:0 (0.0 b)  TX bytes:816 (816.0 b)

vlan1     Link encap:Ethernet  HWaddr 08:00:27:DC:43:98
          inet addr:192.168.20.1 Bcast:192.168.20.255 Mask:255.255.255.0
          inet6 addr: fe80::a00:27ff:fedc:4398/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:12 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
```

```
          RX bytes:0 (0.0 b)  TX bytes:816 (816.0 b)
```

13.10 ネットワークブリッジデバイスの設定

ネットワークブリッジとは仮想的なリンク層 (レイヤ 2) デバイスで、複数のネットワークセグメント同士を接続して、単一の論理ネットワークにするための仕組みです。ブリッジは仮想的なネットワークスイッチとして振る舞うので、流れるトラフィックを監視してハードウエア MAC アドレスをベースにした動的なフレーム転送を行います。一般的には、物理ネットワークデバイスを複数の仮想マシンで共有するような仮想化環境 (KVM や Xen など) での使用が考えられます。

ネットワークブリッジは YaST を利用してグラフィカルに設定できるほか、手作業で `/etc/sysconfig/network/` ディレクトリ内に設定ファイルを作成して設定する方法もあります。

通常は YaST ネットワークデバイスモジュールを利用して設定するのがお勧めです。

手順 13.4: YAST によるブリッジインターフェイスの作成

1. まずは［YaST］›［システム］›［ネットワーク設定］を選択します。
2. ［追加］を押し、［デバイスの種類］で［ブリッジ］を選択したあと、［次へ］を押します。
3. ［設定名］の欄にインターフェイス名 (例: `br0`) を入力して［次へ］を押します。
4. ［アドレス］タブでは、まずネットワークブリッジインターフェイスの IP アドレス設定方法を選択 (`DHCP` もしくは［固定 IP アドレス］) し、後者であれば IP アドレスを設定します。
5. ［ブリッジ接続デバイス］のタブに切り替えます。ここでは、ブリッジに接続する物理ネットワークカード、もしくは VLAN インターフェイスをチェックボックスで選択します。
6. ［次へ］を押すとネットワーク設定の概要に戻ります。あとは［OK］を押すと、設定を保存してネットワークサービスが再起動されます。

13.10.1 手作業によるネットワークブリッジの作成

YaST を使用せずに手作業でネットワークブリッジを作成する場合、ネットワークディレクトリ内に各種の制御パラメータを記述したファイルを作成します。本章の例では、`br0` という名前のブリッジインターフェイスを作成し、`eth0` という物理インターフェイスを追加するまでの手順を説明しています。

手順 13.5: WICKED インターフェイス定義によるブリッジの設定

1. まずはネットワークブリッジデバイスの設定ファイルとして、新しい設定プロファイル `/etc/sysconfig/network/ifcfg-br0` を作成します:

```
STARTMODE='auto'
```

```
BOOTPROTO='static'
IPADDR='ブリッジに割り当てる_IP_アドレス'
BRIDGE='yes'
BRIDGE_PORTS='eth0'
BRIDGE_STP='off'
```

2. また、ブリッジに追加する物理ネットワークインターフェイスの設定も修正します。具体的には、IP アドレスの設定があれば削除するほか、アドレスの自動取得の設定があれば、これを無効化して純粋な物理ポートとして動作するように設定します:

```
STARTMODE='auto'
BOOTPROTO='none'
```

3. 固定 IP の設定を行う場合、必要であればデフォルトゲートウエイも明示的に設定します。デフォルトゲートウエイは `/etc/sysconfig/network/ifroute-br0` という別のファイルに記述します:

```
default デフォルトゲートウエイの_IP_アドレス - br0
```

警告: ルータ設定の競合について

デフォルトゲートウエイを設定するにあたっては、あらかじめ物理インターフェイス (例: `ifroute-eth0`) 側で設定していないかどうかをご確認ください。物理インターフェイス側で設定してしまうと、カーネル内でルーティング競合フラグが設定され、期待通りに動作しなくなってしまいます。

4. あとはネットワークの設定を再読み込みして、変更内容を反映させます:

```
> sudo systemctl restart network
```

13.11 Open vSwitch を利用したソフトウエア定義型ネットワーク

ソフトウエア定義型ネットワーク (Software-Defined Networking; SDN) とは、トラフィックの送受信を制御するシステム (制御プレーン) と、実際にトラフィックを配送するシステム (データプレーン もしくは 転送プレーン) を分離する仕組みです。これにより、従来は単一で柔軟性に欠けるスイッチを、スイッチ (データプレーン) とコントローラ (制御プレーン) に分割できることになります。また、このモデルを使用すると、コントローラ側ではプログラム的な処理ができるようになりますので、非常に柔軟で構成変更にも素早く対応できる仕組みにすることができます。

Open vSwitch は OpenFlow プロトコルとの互換性がある、分散型仮想マルチレイヤスイッチを実装するソフトウエアです。OpenFlow では、コントローラアプリケーションに対してスイッチの設定変更を受け付けることができます。また OpenFlow は TCP プロトコル上で動作する仕組みで、様々なハードウエアやソフトウエア内にも実装されています。そのため、単一のコントローラから複数の様々なスイッチを扱うことができるようになります。

13.11.1 Open vSwitch の利点

Open vSwitch でソフトウエア定義型ネットワークを利用すると、特に仮想マシンと併用する場合に様々な利点をもたらします:

- ネットワークの状態を簡単に識別できるようになります。
- ネットワークと現在の状態を、一方のホストから他方のホストに移動できるようになります。
- ネットワーク上での動作は追跡可能であり、外部のソフトウエアから応答することができます。
- ネットワークパケット内のタグを適用したり制御したりすることができますので、どのマシンが通信を行っているのかを判別できるほか、他のネットワークとの関係を管理することもできます。タグ付けのルールは設定および移行することができます。
- Open vSwitch は GRE (汎用ルーティングカプセル化 (Generic Routing Encapsulation) プロトコルに対応しています。そのため、たとえばプライベートな VM 同士を接続したりすることができるようになります。
- Open vSwitch は単独でも利用することができますが、ネットワークハードウエアと併用するように設計されていて、ハードウエアスイッチを制御することもできるようになっています。

13.11.2 Open vSwitch のインストール

1. Open vSwitch と各種補助パッケージをインストールします:

   ```
   # zypper install openvswitch openvswitch-switch
   ```

   Open vSwitch と KVM ハイパーバイザを併用する場合は、`tunctl` もインストールしてください。また、Open vSwitch と Xen ハイパーバイザを併用する場合は、`openvswitch-kmp-xen` もインストールしてください。

2. Open vSwitch サービスを有効化します:

   ```
   # systemctl enable openvswitch
   ```

3. コンピュータを再起動するか、もしくは `systemctl` コマンドで Open vSwitch サービスを即時に起動します:

```
# systemctl start openvswitch
```

4. Open vSwitch が正しく有効化されているかどうかを確認するには、下記のコマンドを実行します:

```
# systemctl status openvswitch
```

13.11.3 Open vSwitch デーモンとユーティリティの概要

Open vSwitch は複数のコンポーネントから構成されています。カーネルモジュールのほか、様々なユーザスペースコンポーネントが用意されています。カーネルモジュールはデータパスの高速化のために用意されているものですが、最小限の Open vSwitch インストールでは必ずしも必要なものではありません。

13.11.3.1 デーモン

Open vSwitch の中枢は 2 つのデーモンから構成されています。`openvswitch` サービスを起動すると、それら 2 つを間接的に起動することになります。

メインの Open vSwitch デーモン (`ovs-vswitchd`) は、スイッチの実装を提供するデーモンです。もう 1 つの Open vSwitch データベースデーモン (`ovsdb-server`) は、Open vSwitch の設定と状態を保存するデータベース機能を提供します。

13.11.3.2 ユーティリティ

Open vSwitch には様々な作業を支援するための各種ユーティリティが用意されています。下記のユーティリティー覧が全てではありませんが、主要なコマンドのみについて説明を記しています。

`ovsdb-tool`

Open vSwitch のデータベースを作成／アップグレード／整理／問い合わせすることができるユーティリティです。Open vSwitch データベースでトランザクション処理を行います。

`ovs-appctl`

`ovs-vswitchd` や `ovsdb-server` デーモンの設定を行います。

`ovs-dpctl`, `ovs-dpctl-top`

データパスの作成や修正、可視化や削除などを行います。このツールを使用することで、`ovs-vswitchd` の設定を行うことができるほか、データパスの管理も行うことができます。そのため、通常は分析目的でのみ使用します。

`ovs-dpctl-top` は `top` コマンドのような出力をするユーティリティで、データパスの可視化を行うことができます。

`ovs-ofctl`

OpenFlow プロトコルに準拠する任意のスイッチを管理します。`ovs-ofctl` は Open vSwitch を取り扱うだけではありません。

`ovs-vsctl`

設定データベースに対する高レベルなインターフェイスを提供するユーティリティです。データベースへの問い合わせや、データベースの修正を行うことができます。実際には `ovs-vswitchd` の状態の表示や設定などを行うことになります。

13.11.4 Open vSwitch を利用したブリッジの作成

下記の設定例では、openSUSE Leap の既定のネットワークサービスである Wicked を使用しています。Wicked についての詳細は、13.6項「ネットワーク接続の手動管理」をお読みください。

Open vSwitch をインストールして起動してある状態から、下記のようにして行います:

1. お使いの仮想マシンで使用するブリッジを作成するには、まず下記のような内容で設定ファイルを作成します:

```
STARTMODE='auto'❶
BOOTPROTO='dhcp'❷
OVS_BRIDGE='yes'❸
OVS_BRIDGE_PORT_DEVICE_1='eth0'❹
```

❶ ネットワークサービスの起動時にブリッジを自動作成します。

❷ IP アドレスを設定するためのプロトコルの指定です。

❸ Open vSwitch のブリッジとして使用する設定です。

❹ ブリッジに追加すべきデバイスです。複数のデバイスを追加する場合は、下記の書式で行を追加していってください:

```
OVS_BRIDGE_PORT_DEVICE_サフィックス='デバイス'
```

`サフィックス` には任意の英数字が入ります。ただし、`サフィックス` の値は重複させないでください。

`/etc/sysconfig/network` 内に、`ifcfg-br0` というファイル名で保存します。`br0` の箇所には、任意の名前を指定してかまいません。ただし、`ifcfg-` の部分は変えないでください。オプション類について、詳しくは `ifcfg` のマニュアルページ（`man 5 ifcfg`）および `ifcfg-ovs-bridge` のマニュアルページ（`man 5 ifcfg-ovs-bridge`）をお読みください。

2. あとはブリッジを起動します:

```
# wicked ifup br0
```

Wicked の処理が完了するとブリッジの名前が表示され、`up` という動作中を示すメッセージが表示されます。

13.11.5 KVM による Open vSwitch の直接使用

13.11.4項「Open vSwitch を利用したブリッジの作成」に示す手順でブリッジを作成したら、あとは Open vSwitch を使用して KVM や QEMU の仮想マシンのネットワークアクセスを管理します。

1. Wicked の機能を最大限に生かすため、作成したブリッジの設定を一部変更します。作成した `/etc/sysconfig/network/ifcfg-br0` ファイルを開いて、下記のような行を追加します:

```
OVS_BRIDGE_PORT_DEVICE_2='tap0'
```

また、`BOOTPROTO` を `none` に設定します。たとえば下記のようになります:

```
STARTMODE='auto'
BOOTPROTO='none'
OVS_BRIDGE='yes'
OVS_BRIDGE_PORT_DEVICE_1='eth0'
OVS_BRIDGE_PORT_DEVICE_2='tap0'
```

新しいインターフェイス `tap0` は、次の手順で設定を行います。

2. `tap0` デバイスに対する設定ファイルを追加します:

```
STARTMODE='auto'
BOOTPROTO='none'
TUNNEL='tap'
```

上記の内容を記述したファイルを、`/etc/sysconfig/network` ディレクトリ内の `ifcfg-tap0` というファイル名に保存します。

ヒント: 他のユーザからの TAP デバイスへのアクセス許可

`root` 以外のユーザで起動した仮想マシンから、TAP デバイスを使用できるようにするには、下記のような行を追加します:

```
TUNNEL_SET_OWNER=ユーザ名
```

グループ全体に対してアクセスを許可するには、下記のような行を追加します:

```
TUNNEL_SET_GROUP=グループ名
```

3. 最後に、`OVS_BRIDGE_PORT_DEVICE` の最初の行に設定したデバイスに対する設定を開きます。今回の例では `eth0` になります。`/etc/sysconfig/network/ifcfg-eth0` ファイルを開いて、下記のオプションを設定します:

```
STARTMODE='auto'
BOOTPROTO='none'
```

ファイルが存在していない場合は、新しく作成してください。

4. あとは Wicked を利用してブリッジインターフェイスを再起動します:

```
# wicked ifreload br0
```

これにより、新しく設定したブリッジポートデバイスについても、再読み込みが行われます。

5. 仮想マシンを起動するには、たとえば下記のように実行します:

```
# qemu-kvm \
-drive file=ディスクイメージのパス❶ \
-m 512 -net nic,vlan=0,macaddr=00:11:22:EE:EE:EE \
-net tap,ifname=tap0,script=no,downscript=no❷
```

❶ 起動したい QEMU ディスクイメージのパスを指定します。

❷ 上記で作成した TAP デバイス (`tap0`) を指定します。

KVM/QEMU での使用方法について、詳しくは『**仮想化ガイド**』をお読みください。

13.11.6 libvirt での Open vSwitch の使用

13.11.4項「Open vSwitch を利用したブリッジの作成」に示す手順でブリッジを作成したら、libvirt で管理されている既存の仮想マシンをブリッジに追加することができるようになります。libvirt には Open vSwitch ブリッジのサポートが既に含まれているため、13.11.4項「Open vSwitch を利用したブリッジの作成」でブリッジを作成すれば、あとは特殊な変更を行うことなくブリッジを使用できるようになります。

1. 設定を行いたい仮想マシンのドメイン XML ファイルを開きます:

   ```
   # virsh edit VM_名
   ```

   VM_名 の欄には、仮想マシンの名前を指定します。これにより、既定のテキストエディタが開きます。

2. <interface type="..."> で始まり、</interface> で終わるネットワークセクションを探します。
   ネットワークセクション内の下記の箇所を、下記のように変更します:

   ```
   <interface type='bridge'>
     <source bridge='br0'/>
     <virtualport type='openvswitch'/>
   </interface>
   ```

   重要: virsh iface-* と Open vSwitch の 仮想マシンマネージャ との互換性について

   現時点では、libvirt での Open vSwitch の互換性は virsh iface-* ツールや仮想マシンマネージャ を介しては開示されていません。これらのツールを使用してしまうと、設定が壊れてしまうことがあります。

3. 後は通常通り仮想マシンを起動もしくは再起動します。

libvirt の使用方法についての詳細は、『仮想化ガイド』をお読みください。

13.11.7 さらなる情報

SDN に関する詳細は、https://docs.openvswitch.org/en/latest/#documentation にある Open vSwitch プロジェクト Web サイト内のドキュメンテーションをお読みください。

14 UEFI (Unified Extensible Firmware Interface)

改訂履歴

2025-06-06

UEFI (Unified Extensible Firmware Interface) はシステムに付属するファームウエアと、システム内の全てのハードウエアコンポーネント、そしてオペレーティングシステムとの間に存在するインターフェイスを意味する用語です。

UEFI は現在の PC システムに順次取り入れられている仕組みで、従来の PC-BIOS を置き換える目的で作られています。たとえば UEFI では 64 ビットシステムを正しくサポートしているほか、最も重要な点として、「Secure Boot」(ファームウエアバージョン 2.3.1c もしくはそれ以降が必要) と呼ばれる仕組みにも対応しています。最終的には、全ての x86 プラットフォームで利用できるようになるはずのものです。

UEFI には、これらに加えて下記の利点が用意されています:

- GUID Partition Table (GPT) を利用した、巨大なディスク (2 TiB 以上) からの起動に対応しています。
- CPU に依存しないアーキテクチャとドライバに対応しています。
- ネットワーク機能にも対応した柔軟な OS 起動前環境 (pre-OS) に対応しています。
- CSM (互換性サポートモジュール; Compatibility Support Module) により、PC-BIOS を疑似できるようになっています。これにより、古いオペレーティングシステムにも対応できます。

さらに詳しい情報については、https://ja.wikipedia.org/wiki/Unified_Extensible_Firmware_Interface をお読みください。下記の章では、UEFI に関する一般的な説明ではなく、openSUSE Leap ではこれらがどのように実装されているかを説明しています。

14.1 Secure Boot

UEFI の世界では、起動処理時に不正なコードが紛れ込むことのないよう、信頼の鎖 (くさり) を構成して解決しています。このとき、openSUSE Leap の用語で「プラットフォーム」は信頼の鎖の基礎となる存在ですが、これは一般にメインボードとそこに搭載されたファームウエアを意味します。これはハードウエアの製造元と言い換えることもできますので、ハードウエイの製造元からコンポーネントの製造元、そして OS の製造元へと信頼の鎖が繋がることになります。

信頼そのものは公開鍵認証の仕組みを利用しています。ハードウエアの製造元は、プラットフォーム鍵 (PK; Platform Key) をファームウエア内に配置し、これを信頼の鎖の基礎とします。オペレーティングシステムなどの製造元との信頼関係は、それらの鍵をプラットフォーム鍵で署名することによって成立させます。

最後にファームウエアは、それらの「信頼済み」鍵のいずれかで署名されたコードだけを実行することによって、セキュリティを成立させます。このコードには OS のブートローダのほか、PCI Express カードやディスク内に配置されたいくつかのドライバ、そしてファームウエア更新そのものも含まれます。

Secure Boot を使用するには、まずファームウエアが信頼する鍵で OS ローダに署名し、OS ローダが読み込むカーネルを信頼する必要があります。

鍵交換鍵 (KEK; Key Exchange Key) を UEFI の鍵データベースに追加することができます。この方法により、PK で署名されている他の証明書を使用することができるようになります。

14.1.1 openSUSE Leap での実装

Microsoft 社の鍵交換鍵 (KEK) が既定でインストールされています。

注記: GUID Partitioning Table (GPT) が必要となる件について

Secure Boot の機能は UEFI/x86_64 環境にインストールする場合は、既定で有効化されます。具体的には、［ブートローダの設定］ダイアログ内の［ブートコードのオプション］タブにある［Secure Boot サポートを有効にする］に既定でチェックが入っています。ファームウエア側で Secure Boot を有効にしていれば起動を行うことができますし、無効化しても起動することができます。

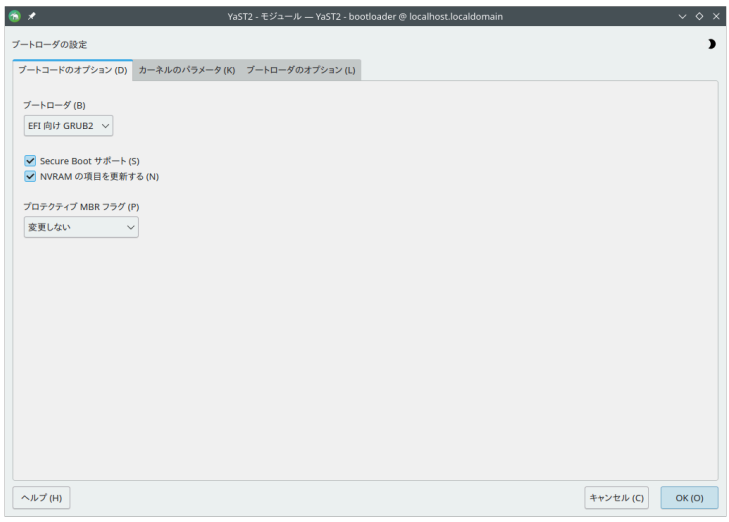


図 14.1: SECURE BOOT サポート

> Secure Boot の機能を利用するには、パーティション方式を古いマスターブートレコード (MBR; Master Boot Record) 形式から、GUID パーティションテーブル (GPT; GUID Partition Table) に置き換える必要があります。YaST がインストール時に EFI モードを検出すると、GPT でパーティションを作成しようとします。また、UEFI では、FAT 形式でフォーマットされた EFI システムパーティション (ESP; EFI System Partition) 内に EFI プログラムを配置する必要があります。

さらに、UEFI の Secure Boot を利用するには、ファームウエア側で信頼されている鍵で署名されたブートローダを使用する必要があります。また、この鍵はあらかじめ信頼されていなければなりません。この信頼を得るには、2 種類の方法があります。1 つ目の方法は、ハードウエアの製造元に依頼して、SUSE の鍵を信頼してもらう方法です。SUSE ではブートローダに対して署名を付与しています。もう 1 つの方法は、Microsoft 社の Windows ロゴ認証プログラムに申し込んでブートローダの証明を得て、Microsoft 社から SUSE の署名鍵を認証してもらう (つまり、KEK で署名してもらう) 方法です。現在は SUSE 社が UEFI 署名サービス (この場合は Microsoft 社) を介して、署名を付与してもらっています。

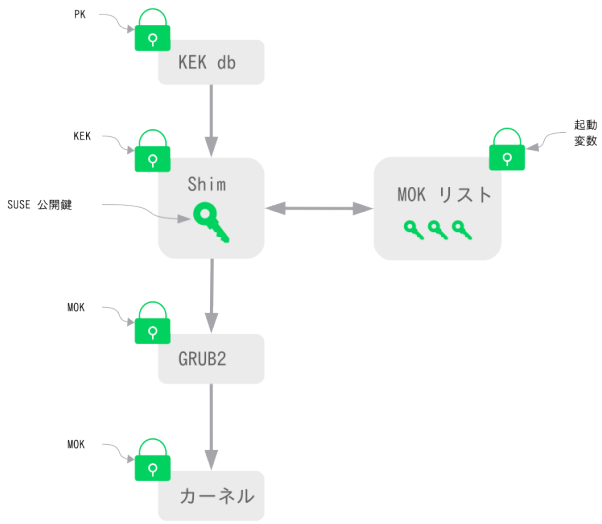


図 14.2: UEFI: SECURE BOOT の処理

実装レイヤでは SUSE は、既定でインストールされる `shim` ローダを使用しています。これは法的な問題を回避できる賢いソリューションであるだけでなく、証明書と署名の手順を非常に簡単にすることができます。`shim` ローダが行う処理は、GRUB 2 のようなブートローダを読み込んで、それを検証するだけです。このブートローダは、SUSE の鍵で署名されたカーネルのみを読み込むようになっています。

信頼済みのユーザという観点では、下記の 2 種類のユーザが存在します:

- 1 つ目は鍵の所有者です。プラットフォーム鍵 (PK) はほぼ全てのことが許可されます。鍵交換鍵 (KEK) は PK の変更を除き、PK と同じ内容が許可されます。
- 2 つ目はマシンに対して物理的なアクセスのできるユーザです。このユーザはマシンを再起動することができるほか、UEFI の設定を行うことができます。

UEFI では、これらのユーザの要件を満たすため、2 種類の変数を提供しています:

- 1 つ目は「認証済み変数」と呼ばれるもので、起動処理 (起動サービス環境 (Boot Services Environment)) 内でも起動済みの OS 内でも、変更できる変数です。ただし、変数の値は古い値と同じ鍵で署名されている場合にのみ変更することができます。また、より高いシリアル番号の値を追記するか、もしくはより高いシリアル番号の値にのみ置き換えることができます。
- 2 つ目は「起動サービスのみの変数」と呼ばれるもので、起動処理中に実行される任意のコードからのみアクセスすることができます。起動処理が終わると、OS の起動前にブートローダ側で `ExitBootServices` を呼び出さなければなりません。その後、これらの変数にはアクセスできなくなりますので、OS 側からもアクセスができなくなります。

UEFI 鍵リストは最初の種類のものであり、これによってオンラインでの更新や追加、鍵やドライバ／ファームウエアのフィンガープリントのブラックリストなどを行うことができます。また、2 番目の「起動サービスのみの変数」では、Secure Boot を安全かつオープンソースに適したやり方で実装し、それによって GPLv3 との互換性も確保することができています。

SUSE では `shim` と呼ばれる、SUSE と Microsoft が署名している小さくてシンプルな EFI ブートローダを使用しています。

これにより、`shim` が読み込みと実行ができるようになっています。

`shim` を読み込んで実行すると、読み込むべきブートローダが信頼のできるものかどうかを確認しようとします。既定の動作では、`shim` は独立した SUSE の内蔵署名を使用します。これに加えて `shim` は、既定の SUSE 鍵を上書きする追加の鍵を「登録」します。下記では、この鍵を略して「マシン所有者鍵」(MOK) と呼んでいます。

あとはブートローダがカーネルを検証して起動し、カーネルが同様にモジュールを検証して読み込みます。

14.1.2 MOK (マシン所有者鍵; Machine Owner Key)

起動処理時に使用するカーネルやドライバ、もしくはその他のコンポーネントを入れ替えたい場合は、マシン所有者鍵 (MOKs) を使用する必要があります。これを行うには、`mokutils` と呼ばれるツールを利用します。

MOK の登録を行うには `mokutil` と呼ばれるツールを使用します。要求は `MokNew` と呼ばれる UEFI のランタイム (RT) 変数内に保存されます。次回以降の起動では `shim` ブートローダが `MokNew` を検出し、`MokManager` を読み込んでいくつかの選択肢を表示します。ここから［Enroll key from disk］,［Enroll hash from disk］と選択していくことで、MokList 内にキーを追加できるようになります。`MokNew` 変数内にキーをコピーするには、［Enroll MOK］オプションをお使いください。

ディスクからのキーの登録は一般に、`grub2` の読み込みが失敗し、MokManager に戻された場合に実施できます。まだ `MokNew` が存在していない場合は、UEFI パーティション内でキーを探すことができます。

14.1.3 独自のカーネルの起動

下記の説明は、https://ja.opensuse.org/openSUSE:UEFI#.E7.8B.AC.E8.87.AA.E3.81.AE.E3.82.AB.E3.83.BC.E3.83.8D.E3.83.AB.E3.81.AE.E8.B5.
をベースにしています。

Secure Boot は、独自にコンパイルしたカーネルを起動できないようにする仕組みではありません。自分の証明書で署名したあと、その証明書の発行者をファームウエアか MOK にインストールすることで、解決することができます。

1. まずは署名に使用する独自の X.509 鍵と証明書を作成します:

   ```
   openssl req -new -x509 -newkey rsa:2048 -keyout key.asc \
     -out cert.pem -nodes -days 666 -subj "/CN=$USER/"
   ```

   証明書の作成方法について、詳しくは https://ja.opensuse.org/openSUSE:UEFI_%E3%82%A4%E3%83%A1%E3%83%BC%E3%82%B8%E3%83%95%E3%82%A1%E3%82%A4%E3%83%AB%E7%BD%B2%E5%90%8D%E3%83%84%E3%83%BC%E3%83%AB をお読みください。

2. 鍵と証明書を PKCS#12 形式にパッケージ化します:

   ```
   > openssl pkcs12 -export -inkey key.asc -in cert.pem \
     -name kernel_cert -out cert.p12
   ```

3. pesign で使用する NSS データベースを生成します:

   ```
   > certutil -d . -N
   ```

4. NSS データベース内に PKCS#12 に含まれる鍵と証明書を取り込みます:

   ```
   > pk12util -d . -i cert.p12
   ```

5. `pesign` を利用して、新しい署名をカーネルに追加します:

```
> pesign -n . -c kernel_cert -i arch/x86/boot/bzImage \
  -o vmlinuz.signed -s
```

6. カーネルイメージに含まれる署名を一覧表示します:

```
> pesign -n . -S -i vmlinuz.signed
```

あとは通常通り、/boot ディレクトリ内にカーネルをインストールすることができます。ただし、カーネルには独自の証明書による署名が設定されているため、これをファームウエアか MOK に取り込む必要があります。

7. UEFI ファームウエアまたは MOK に取り込むため、証明書を DER 形式に変換します:

```
> openssl x509 -in cert.pem -outform der -out cert.der
```

8. 簡単にアクセスできるよう、ESP に証明書をコピーします:

```
> sudo cp cert.der /boot/efi/
```

9. MOK を自動的に取り扱うには、`mokutil` を使用します。

- a. 証明書を MOK に取り込みます:

```
> mokutil --root-pw --import cert.der
```

`--root-pw` オプションを指定することで、`root` ユーザ経由でアクセスすることができます。

b. 登録の準備ができた証明書の一覧を確認します:

```
> mokutil --list-new
```

c. システムを再起動します。`shim` から MokManager が起動されるはずです。`root` のパスワードを入力して、MOK の一覧内に証明書を取り込みます。

d. 新しく取り込んだ鍵が登録されていることを確認します:

```
> mokutil --list-enrolled
```

- a. 上記以外にも、MOK を手作業で処理することもできます:
まずは再起動します。

b. GRUB 2 のメニューが表示されたら、' `c` ' キーを押します。

c. 下記のように入力します:

```
chainloader $efibootdir/MokManager.efi
boot
```

d. [Enroll key from disk] を選択します。

e. `cert.der` ファイルを指定して `Enter` を押します。

f. 後は表示された手順に従います。通常は「0」を押してから「y」を押すと、確認を完了することができます。
それ以外にも、ファームウエア側のメニューで、署名データベース内に新しい鍵を追加することができるものもあります。

14.1.4 非同梱ドライバの使用について

Secure Boot が有効化されている場合、インストール時に非同梱ドライバ (openSUSE Leap には含まれていないドライバ) の追加を行うことはできません。SolidDriver/PLDP 向けの署名鍵は、既定では信頼されていません。

Secure Boot が有効化されている場合、インストール時にサードパーティ製のドライバをインストールする方法が 2 種類用意されています。いずれも下記に注意してください:

- インストールを行う前に、ファームウエアやシステム管理ツールを介して、ファームウエアデータベース内に必要な鍵を取り込んでおいてください。この設定は、使用しているハードウエアによって異なります。詳しくはハードウエアの製造元にお問い合わせください。
- https://drivers.suse.com/ もしくはハードウエアの製造元が提供する起動可能な ISO を利用して、初回の起動時に必要な鍵を登録します。

起動可能な ISO を利用して MOK リスト内にドライバの鍵を登録するには、下記の手順を実施します:

1. 空きの CD/DVD メディアに ISO イメージを書き込みます。
2. 標準のインストールメディアを利用するか、もしくはネットワークのインストールサーバを利用してインストールを開始します。
ネットワーク経由でインストールする場合は、起動パラメータに `install=` オプションで URL を指定します。
光学メディアからインストールを行う場合、インストールプログラムは最初にドライバキットから起動し、その後製品のインストールディスクを挿入してください。
3. initrd には更新済みのドライバが含まれていますので、それを利用してインストールすることができます。

詳しくは https://drivers.suse.com/doc/Usage/Secure_Boot_Certificate.html をお読みください。

14.1.5 機能と制限

Secure Boot モードで起動する場合、下記の機能が提供されます:

- UEFI の既定の場所に対するブートローダのインストール、および EFI の起動項目に対する維持や復元の仕組み
- UEFI 経由での再起動
- 従来の BIOS にフォールバックすることのない、UEFI での Xen ハイパーバイザの起動
- UEFI の IPv6 PXE 起動
- UEFI ビデオモードサポート (UEFI 経由でカーネルがビデオモードを取得し、同じパラメータで KMS モードを設定する機能)
- USB デバイス経由の UEFI 起動
- openSUSE Leap 15.3 およびそれ以降のバージョンでは、Kexec と Kdump は Secure Boot モードにも対応します。

Secure Boot モードでの起動時には、下記の制限事項が適用されます:

- Secure Boot が容易に突破されないようにするため、Secure Boot の動作中はいくつかのカーネル機能が無効化されます。
- ブートローダ、カーネル、カーネルモジュールは、いずれも署名されていなければなりません。
- ハイバネーション (suspend to disk) が無効化されます。
- root ユーザであっても `/dev/kmem` および `/dev/mem` にアクセスできなくなります。
- root ユーザであっても I/O ポートへのアクセスができなくなります。また、X11 のグラフィカルドライバはカーネルドライバを使用しなければなりません。
- sysfs を介した PCI BAR アクセスができなくなります。
- ACPI の `custom_method` が利用できなくなります。
- asus-wmi モジュール向けの debugfs が利用できなくなります。
- `acpi_rsdp` がカーネルに対して何も影響を及ぼさなくなります。

14.2 Secure Boot 失効リスト

UEFI Secure Boot 失効リストは `dbx` (Secure Boot Forbidden Signature Database (Secure Boot 禁止署名データベース)) としても知られ、コンピュータ内の UEFI ファームウエアでは重要なセキュリティコンポーネントです。このリストは、システムの起動時に信頼のできないソフトウエアの読み込みや実行を阻止するための仕組みで、システムのセキュリティ向上に有用なものです。具体的には、`dbx` は下記のようなことを行います:

- システム起動時のマルウエア阻止: `dbx` は、オペレーティングシステムが動作するよりも前の段階で、不正なコードがシステムに読み込まれて実行されるのを防ぎます。
- 信頼の連鎖: システムの起動時、それぞれのコンポーネントは次に読み込まれるコンポーネントの検証を実施します。`dbx` を利用することで、そのような連鎖内に不正なコードが存在していれば、それを即時に停止する機能を提供します。
- ロールバック (巻き戻し) 攻撃からの保護: 古いバージョンのファームウエアやブートローダへの巻き戻しを防止することで、古いバージョンに存在する脆弱性からの保護を提供します。
- セキュリティの拡張: `dbx` は新しく発見された脆弱性から、お使いのシステムを保護するのに重要な仕組みです。このリストを更新しないと、お使いのシステムが既知の脆弱性から保護されなくなってしまいます。

14.2.1 オンラインの失効リスト更新の適用

前提条件

- お使いのシステムで Secure Boot を有効化しておく必要があります。
- 更新をダウンロードするため、インターネットへの接続が必要です。

1. まずは失効リストの現在のバージョンを確認します:

```
>  fwupdmgr get-devices
      LENOVO 21AAS05L00
      │
      ├─11th Gen Intel Core™ i7-11800H @ 2.30GHz:
      │     Device ID:          4bde70ba4e39b28f9eab1628f9dd6e6244c03027
      │     Current version:    0x00000052
      │     Vendor:             Intel
      │     GUIDs:              a6bd4ca5-75a6-5796-b564-66b5cab1b11b ← CPUID
\PRO_0&FAM_06&MOD_8D
      │                         d9dd5e77-df17-5bab-b5ec-22827598bfed ← CPUID
\PRO_0&FAM_06&MOD_8D&STP_1
      │     Device Flags:       • Internal device
```

```
      |     Device Requests:    • Message
```

2. まずは LVFS (Linux Vendor Firmware Service) 失効リストリポジトリを有効化します:

```
> fwupdmgr enable-remote lvfs
```

3. リポジトリのメタデータを更新します:

```
> fwupdmgr refresh
```

4. 失効リストの更新を適用します:

```
> fwupdmgr update
```

5. システムを再起動して、最新バージョンが適用されたことを確認します:

```
> fwupdmgr get-devices
```

14.2.2 オンラインの失効リスト更新の適用

オフライン環境で失効リスト更新を行いたい場合は、openSUSE Leap が提供する情報で Secure Boot 失効リストを更新することができます。これにより、既知のセキュリティ問題から保護できます。この手順によってシステムが起動できなくなるようなことはありませんので、安全です。

1. まずは失効リストの現在のバージョンを確認します:

```
> fwupdmgr get-devices
        LENOVO 21AAS05L00
        |
        ├─11th Gen Intel Core™ i7-11800H @ 2.30GHz:
        |     Device ID:          4bde70ba4e39b28f9eab1628f9dd6e6244c03027
        |     Current version:    0x00000052
        |     Vendor:             Intel
        |     GUIDs:              a6bd4ca5-75a6-5796-b564-66b5cab1b11b ← CPUID
\PRO_0&FAM_06&MOD_8D
        |                         d9dd5e77-df17-5bab-b5ec-22827598bfed ← CPUID
\PRO_0&FAM_06&MOD_8D&STP_1
        |     Device Flags:       • Internal device
        |     Device Requests:    • Message
```

2. 次に openSUSE Leap が提供する更新を一覧表示します:

```
> ls /usr/share/dbxtool/
```

3. お使いのアーキテクチャに合致する最新の更新ファイルを探します。たとえば `DBXUpdate-date-architecture.cab` のようなファイルになります。

4. 選択した更新ファイルをインストールします:

```
> fwupdmgr install /usr/share/dbxtool/DBXUpdate-date-architecture.cab
```

5. システムを再起動して、最新バージョンが適用されたことを確認します:

```
> fwupdmgr get-devices
```

14.3 さらなる情報

- https://uefi.org: UEFI の Web ページです。ここには最新の UEFI 仕様に関する情報が記載されています。
- Olaf Kirch 氏および Vojtěch Pavlík 氏によるブログ投稿 (本章の内容は、この投稿をベースにしています):
  - https://www.suse.com/c/uefi-secure-boot-plan/
  - https://www.suse.com/c/uefi-secure-boot-overview/
  - https://www.suse.com/c/uefi-secure-boot-details/
- https://ja.opensuse.org/openSUSE:UEFI openSUSE での UEFI

15 特殊なシステム機能

改訂履歴

2024-05-13

本章では、仮想コンソールやキーボードレイアウトなどのソフトウエアパッケージについて説明を行っています。特に `bash` , `cron` , `logrotate` については、直近のリリースサイクルで変更や機能追加が行われていることから、これらを詳しく説明しています。これらのソフトウエアは、いずれもシステムに密接に繋がっていることから、小さな変更やあまり重要ではない変更についても、ユーザ側に影響があるためです。また、本章では言語や国固有の設定 (I18N や L10N と呼ばれるもの) についても説明しています。

15.1 特殊なソフトウエアパッケージについての情報

本章では、`bash` , `cron` , `logrotate` , `locate` , `ulimit`, `free` の各ツールに対する基本的な情報を提供しています。

15.1.1 bash パッケージと /etc/profile の関係

bash は既定のシステムシェルです。ログインシェルとして使用した場合、いくつかの設定ファイルを読み込みます。設定ファイルの処理順序は、下記のとおりです:

1. `/etc/profile`
2. `~/.profile`
3. `/etc/bash.bashrc`
4. `~/.bashrc`

独自の設定を行いたい場合は、`~/.profile` ファイル、もしくは `~/.bashrc` ファイルに設定を記述してください。また、これらのファイルを正しく処理するようにするため、`/etc/skel/.profile` もしくは `/etc/skel/.bashrc` にある雛形のファイルをホームディレクトリにコピーしてお使いください。また、システムを更新した後は、`/etc/skel` からファイルをコピーし直して、必要な変更を加えることをお勧めします。以前の設定を失わないようにするため、具体的には下記のように実行してください:

```
> mv ~/.bashrc ~/.bashrc.old
```

```
> cp /etc/skel/.bashrc ~/.bashrc
> mv ~/.profile ~/.profile.old
> cp /etc/skel/.profile ~/.profile
```

あとは `*.old` ファイル内にある変更を、新しいファイルに反映させてください。

15.1.2 cron パッケージ

`cron` システムは、指定した時刻に自動的に裏でコマンドを実行するための仕組みです。`cron` では特殊な書式のタイムテーブルと、あらかじめ決められたスケジュールのファイルを利用して、時刻の指定と実行すべきコマンドを指定します。一般ユーザでも同様に、独自のタイムテーブルを設定することができます。

cron のタイムテーブルは `/var/spool/cron/tabs` ディレクトリ内に配置されます。システム全体のタイムテーブルは `/etc/crontab` ファイルにあります。`/etc/crontab` 内では、日時の指定の後ろにユーザ名を指定して、そのユーザでコマンドを実行するようにすることができます。たとえば 例 15.1「/etc/crontab の内容」では、`root` を指定しています。また、パッケージ固有のタイムテーブルは `/etc/cron.d` ディレクトリ内に同じ形式で配置します。詳しくは `cron` のマニュアルページ (`man cron`) をお読みください。

例 15.1: /ETC/CRONTAB の内容

```
1-59/5 * * * *   root   test -x /usr/sbin/atrun && /usr/sbin/atrun
```

なお、`/etc/crontab` ファイルは `crontab -e` コマンドで編集することはできません。このファイルはエディタなどで直接編集して保存してください。

また、パッケージによっては `/etc/cron.hourly` 、`/etc/cron.daily` 、`/etc/cron.weekly` 、`/etc/cron.monthly` などのディレクトリにシェルスクリプトをインストールするものがあります。これらのディレクトリ内のスクリプトは、`/usr/lib/cron/run-crons` ファイルが制御しています。この `/usr/lib/cron/run-crons` ファイルは、メインのタイムテーブル (`/etc/crontab`) 内に設定されていて、15 分に一度動作するようになっています。これにより、見逃されやすい処理も正しく実行できるようになっています。

`hourly` や `daily` のほか、その他の定期的なメンテナンススクリプトを独自の時間に実行したい場合は、`/etc/crontab` 内に項目 (ジョブ) を追加して、タイムスタンプファイルを削除するように設定してください (たとえば 例15.2「/etc/crontab: タイムスタンプファイルの削除」では、`hourly` を毎時 59 分に、`daily` を毎日午前 2:14 に削除したりしています) 。

例 15.2: /ETC/CRONTAB: タイムスタンプファイルの削除

```
59 *  * * *     root  rm -f /var/spool/cron/lastrun/cron.hourly
```

```
14 2  * * *     root  rm -f /var/spool/cron/lastrun/cron.daily
29 2  * * 6     root  rm -f /var/spool/cron/lastrun/cron.weekly
44 2  1 * *     root  rm -f /var/spool/cron/lastrun/cron.monthly
```

上記以外の方法としては、`/etc/sysconfig/cron` ファイル内の `DAILY_TIME` を設定する方法があります。この変数は、`cron.daily` を開始すべき時刻を指定するものです。また、`MAX_NOT_RUN` は、ユーザが `DAILY_TIME` で指定していた時刻にコンピュータを起動していなかった場合、どれだけの日数が経過するとジョブを即時に実行するかを指定します。なお、`MAX_NOT_RUN` の最大値は 14 日です。

15.1.3 cron の状態メッセージの停止

cron の状態メールでメールボックスが溢れないようにする目的で、新規インストールでは `/etc/sysconfig/cron` 内の `SEND_MAIL_ON_NO_ERROR` が " `no` " に設定されるようになっています。ただし、" `no` " に設定した場合でも、cron のマニュアルページに書かれているとおり、`MAILTO` アドレスに対しては実行結果のメールが配信されます。

古いバージョンから更新している場合は、お使いの環境の要件にあわせて値を指定してください。

15.1.4 ログファイル: logrotate パッケージ

様々なシステムサービス (デーモン) だけでなく、カーネル自身も、様々な状態やイベントを通知する目的で、ログファイルに書き込みを行います。このログファイルが存在するおかげで、システム管理者はその当時のシステムの状態を確認できるほか、様々なトラブルの原因調査と解決を行うことができるようになっています。これらのログファイルは通常、FHS で規定されている `/var/log` 内に保存され、日々の運用でサイズが増加していきます。`logrotate` パッケージは、そのようなサイズの肥大化を解決するための仕組みです。詳しくは **『システム分析／チューニングガイド』、第3章「システムログファイル」、3.3項「`logrotate` によるログの管理」** をお読みください。

15.1.5 `locate` コマンド

`locate` コマンドはファイルを素早く見つけるためのコマンドですが、インストール済みのソフトウエアを見つけるためだけのコマンドではありません。必要であれば `mlocate` (`findutils-locate` パッケージの後継パッケージ) をインストールすることで、様々なファイルを検索できるようになります。なお、`updatedb` プロセスは、毎晩もしくはシステムの起動後 15 分経過後に自動で開始されるプログラムです。

15.1.6 `ulimit` コマンド

`ulimit` (ユーザ制限 (User Limits)) を利用することで、システム資源の使用範囲に制限を設定したり、それらの制限を表示したりすることができます。`ulimit` は特に、アプリケーションに対して利用を許すメモリサイズを制限する用途で役に立ちます。これを利用することで、アプリケーションがシステム資源を使用しすぎることで、オペレーティングシステムの動作を遅くしてしまったり、場合によってはハングアップ (反応がなくなる) させてしまったりするようなことを防げます。

`ulimit` には様々なオプションが用意されています。たとえばメモリの使用を制限するには、表 15.1「`ulimit` : ユーザに対する資源の制限」に示されているオプションをお使いください:

表 15.1: `ulimit` : ユーザに対する資源の制限

`-m`	最大常駐セットサイズ
`-v`	シェルに対して開放する最大仮想メモリサイズ
`-s`	最大スタックサイズ
`-c`	作成されるコアファイルの最大サイズ
`-a`	現在の制限を全て表示する

システム全体に対する設定は、`/etc/profile` 内で行われています。ただし、このファイルを直接修正することはお勧めしません。なぜなら、システムのアップグレードを行うと、上記のファイルが上書きされてしまうためです。システム全体のプロファイル設定をカスタマイズしたい場合は `/etc/profile.local` ファイルに、ユーザ単位で設定を行いたい場合は `~ユーザ名/.profile` ファイルにそれぞれ記述してください。

例 15.3: `ulimit` : `~/.bashrc` 内での設定

```
# 最大常駐セットサイズ（物理メモリ）の制限:
ulimit -m 98304

# 仮想メモリの制限:
ulimit -v 98304
```

メモリの割り当てはキロバイト (KB) 単位で行います。詳しくは `man bash` をお読みください。

重要: `ulimit` のサポートについて

シェルによっては `ulimit` ディレクティブに対応していないものがあります。また、PAM の仕組みを利用することで、`ulimit` に代わる広範囲な調整機能を利用することができます (`pam_limits` など) 。

15.1.7 `free` コマンド

`free` コマンドは、物理メモリやスワップ領域の空き容量と使用済み容量を表示するほか、カーネルが消費しているバッファやキャッシュのサイズを表示することができるツールです。利用可能なメモリ量の考え方は、統一されたメモリ管理機構が生まれるよりも前に遡ります。メモリを空けておくなんてもったいない の考え方は Linux にも当てはまり、不要なメモリを解放するよりは、できる限りキャッシュとして使用したほうが良い、という動作になっています。

なお、カーネル側ではアプリケーションなのかユーザデータなのかを直接知っているわけではありません。アプリケーションやユーザデータは ページキャッシュ という仕組みで管理されています。そのため、メモリが枯渇すると、まず `mmap` コマンドを利用して準備したそれらを、スワップ領域かファイルに書き込んで開放する動作を行います。

また、カーネル側ではそのほかのキャッシュ機構も用意されています。そのうちの 1 つがslab キャッシュ で、ネットワークアクセスに対するキャッシュを補完します。これにより、`/proc/meminfo` のカウンタ間での差を説明することができます。なお、slab キャッシュ のうちのほとんどの統計情報については、`/proc/slabinfo` からアクセスすることができます。

ただし、どれだけのメモリが使用されているのかを調べたいような場合は、従来通り `/proc/meminfo` ファイルが適切でしょう。

15.1.8 マニュアルページと info ページ

GNU アプリケーションによっては、マニュアルページがメンテナンスされなくなってしまっているものもあります (例: tar) 。これらのコマンドについては `--help` オプションを指定して実行することで、コマンドの概要説明を得ることができます。また、より詳しい説明を読みたい場合は、info ページをお使いください。info ページは GNU 製のハイパーテキストシステムで、info ページの説明そのものも info ページで提供されています。詳しくは `info info` と入力して実行してください。また、info ページは emacs 内に表示することもできます。こちらについては、`emacs -f info` と入力して実行してください。このほか、tkinfo, xinfo やデスクトップ環境のヘルプシステムでも、info ページを表示することができます。

15.1.9 `man` コマンドでのマニュアルページの選択

マニュアルページを読むには、`man` ページ名 のように入力して実行します。なお、異なるセクションに同じ名前のマニュアルページが存在する場合は、セクション番号の一覧が表示され、いずれかを選択するように促されます。この場合、何も入力せずにしばらく経過すると、最初に一致したマニュアルページを表示します。

このような既定の動作を変更したい場合は、`MAN_POSIXLY_CORRECT=1` を `~/.bashrc` のようなシェル設定ファイルで指定してください。

15.1.10 GNU Emacs の設定

GNU Emacs は複雑な作業環境です。下記の章では、GNU Emacs を起動する際に処理される設定ファイルについて説明しています。さらに詳しい説明については、https://www.gnu.org/software/emacs/ をお読みください。

Emacs はその起動時に、ユーザやシステム管理者、ディストリビューションの作成者が、カスタマイズや事前設定を行う目的で、いくつかの設定ファイルを読み込みます。設定ファイルである `~/.emacs` は `/etc/skel` 内にあるものを雛形として、各ユーザのホームディレクトリにコピーされます。また `.emacs` は、そのファイル内の指定で `/etc/skel/.gnu-emacs` を読み込むようになっています。プログラムをカスタマイズしたい場合は、まず `.gnu-emacs` ファイルをホームディレクトリにコピー (例: `cp /etc/skel/.gnu-emacs ~/.gnu-emacs`) してから、コピー先のファイルに対して設定を行ってください。

また `.gnu-emacs` では、`~/.gnu-emacs-custom` ファイルを `custom-file` として指定しています。emacs 内で `customize` オプションを使用して設定を行った場合、その設定は `~/.gnu-emacs-custom` 内に保存されます。

openSUSE Leap では、`emacs` パッケージをインストールすると、`/usr/share/emacs/site-lisp` ディレクトリ内に `site-start.el` ファイルをインストールします。この `site-start.el` ファイルは `~/.emacs` よりも前に読み込まれるファイルで、たとえば `psgml` などの emacs アドオンパッケージに同梱される特別な設定ファイルを、自動的に読み込むようにする目的で作られています。この種類の設定ファイルは `/usr/share/emacs/site-lisp` 内にも存在していて、それぞれは `suse-start-` で始まるファイル名になっています。ローカルのシステム管理者であれば、システム全体の設定を `default.el` で行うことができます。

これらのファイルについてのより詳しい説明は、emacs 内の info ファイルで提供されています。詳しくは Init File : `info:/emacs/InitFile` をお読みください。また、これらのファイルの読み込みを無効化する方法についても、ここに記されています。

emacs のコンポーネントは、下記の複数のパッケージに分割されています:

- 基本パッケージ: `emacs`
- X11 サポート 付き の emacs (通常はこちらがインストールされます): `emacs-x11`
- X11 サポート 無し の emacs : `emacs-nox`
- info 形式でのオンライン文書 : `emacs-info`

- Emacs Lisp 形式でのコンパイル前のライブラリファイル (実行時には不要です) : `emacs-el`
- 必要であればインストールできる各種のアドオンパッケージ: `emacs-auctex` (LaTeX), `psgml` (SGML および XML), `gnuserv` (クライアント／サーバ操作) など

15.2 仮想コンソール

Linux はマルチユーザ／マルチタスクシステムです。これらの仕組みによる利点は、PC 1 台の中だけでも生かすことができます。たとえばテキストモードでは、6 個の仮想コンソールを利用することができます。仮想コンソール間の切り替えは `Alt`–`F1` から `Alt`–`F6` までのキー入力で行うことができます。なお、7 番目のコンソールは X 向けに、10 番目のコンソールはカーネルメッセージの表示向けにそれぞれ予約されています。

X 環境をシャットダウンすることなく仮想コンソールを切り替えたい場合は、`Ctrl`–`Alt`–`F1` から `Ctrl`–`Alt`–`F6` までのキー入力を行ってください。なお、X に戻る場合は `Alt`–`F7` を押してください。

15.3 キーボードマッピング

プログラムによるキーボードマッピングを標準化するため、下記のファイルに対して変更が加えられています:

```
/etc/inputrc
/etc/X11/Xmodmap
/etc/skel/.emacs
/etc/skel/.gnu-emacs
/etc/skel/.vimrc
/etc/csh.cshrc
/etc/termcap
/usr/share/terminfo/x/xterm
/usr/share/X11/app-defaults/XTerm
/usr/share/emacs/バージョン番号/site-lisp/term/*.el
```

これらの変更は、`terminfo` の項目を使用するアプリケーションか、もしくはこれらの設定ファイルを直接使用するアプリケーション (`vi` , `emacs` など) にのみ影響があります。システムに付属していないアプリケーションについては、これらの既定値にあわせて調整してください。

また、X 環境下では合成キー (マルチキー) の有効化を行うことができます。詳しくは `/etc/X11/Xmodmap` をお読みください。

それ以外の設定については、X キーボード拡張 (X Keyboard Extensions (XKB)) を利用して行うことができます。

ヒント: さらなる情報

XKB に関する詳細は、`/usr/share/doc/packages/xkeyboard-config` 内にある文書 (`xkeyboard-config` パッケージ) をお読みください。

15.4 言語および国固有の設定

システムとは、非常に幅広い意味で国際化され、地域の要件にあわせた変更が加えられます。つまり、国際化 (internationalization (I18N)) は特定の地域への対応 (localization (L10N) を含む意味になります。ちなみに、I18N や L10N とは、最初と最後の文字だけを残して、その途中の文字を文字数で省略した表現です。

`LC_` 変数に対する設定は、`/etc/sysconfig/language` ファイル内で行います。ここでの設定は 言語サポート という意味だけでなく、メッセージ (言語), 文字セット , 並び替え順序 , 日付と時刻の表示形式 , 数字の表記方法 , 通貨単位 などの分野があります。それぞれの分野に対する設定は、直接指定することができるほか、`language` ファイル内のマスター変数で間接的に使用することもできます (詳しくは `locale` のマニュアルページをお読みください) 。

変数の一覧

`RC_LC_MESSAGES` , `RC_LC_CTYPE` , `RC_LC_COLLATE` , `RC_LC_TIME` , `RC_LC_NUMERIC` , `RC_LC_MONETARY`

これらの変数は `RC_` のプレフィクスを取り除いてシェルに渡され、それぞれの分類に対応する意味を表すようになっています。関連するシェルのプロファイルを下記に示します。現在の設定は、`locale` コマンドで表示することができます。

`RC_LC_ALL`

この変数が設定されている場合、上述の各変数の値をこの値で上書きします。

`RC_LANG`

上述までの変数のいずれも設定されていない場合、この変数を代替として使用します。既定では `RC_LANG` のみが設定されます。これにより、ユーザが独自の設定を行いやすくしています。

`ROOT_USES_LANG`

`yes` もしくは `ctype` (既定値) を指定する変数です。`yes` を指定すると、`root` に対して言語や国に固有の設定を適用するようになりますが、そうでない場合、システム管理者は POSIX 環境を使用するようになります。

なお、これらの変数は YaST sysconfig エディタで設定することができます。これらの変数の値は言語コードや国コード、エンコーディングや修飾子などが含まれます。具体的には、下記のような書式になります:

```
LANG=<言語>[[_<国コード>].<エンコーディング>[@<修飾子>]]
```

15.4.1 システム全体の言語設定

`systemd` は起動時の早い段階で `/etc/locale.conf` を読み込みます。このファイル内に書かれたロケール設定は、個別に設定した場合を除いて、サービスやユーザに継承されます。

注記: openSUSE Leap における古い設定ファイルの動作について

以前のバージョンでは、openSUSE Leap は `/etc/sysconfig/language` , `/etc/sysconfig/keyboard` , `/etc/sysconfig/console` の各ファイルから設定を読み込んでいました。openSUSE Leap 15.0 15 では、これらのファイルは廃止予定としてマークされるようになっています。`systemd` でも、これらのファイルを読み込むことは行っていません。SUSE Linux Enterprise Server 15 およびそれ以降では `/etc/locale.conf` を読み込みます。

ただし、`/etc/sysconfig/language` 内での設定はシステム全体のロケールを上書きするために使用されているほか、ユーザのシェルに対して異なるロケール設定を行うためにも使用されています (詳しくは 15.4.2項「いくつかの例」をお読みください) 。

システム全体のロケールを設定するには、下記のいずれかを実施します:

- `/etc/locale.conf` ファイル内に設定を書き込みます。各行は環境変数のような書式で変数設定を行います (変数の一覧については `man 5 locale.conf` をお読みください):

  ```
  LANG=ja_JP.UTF-8
  ```

  さらに細かい設定を行いたい場合は、1 行に 1 つの変数を記述する方式で、設定を追加していってください。

- `localectl` コマンドを使用してもかまいません:

  ```
  # localectl set-locale LANG=ja_JP.UTF-8
  ```

  こちらの場合でも、`localectl set-locale` コマンドの後ろに追加の変数を指定することができます。

後方互換性を維持するため、systemd パッケージの更新時にこれらの変数設定が存在すると、新しいほうの設定を書き込むようになっています。

15.4.2 いくつかの例

言語と国コードについては、必ず指定してください。言語は ISO 639 (一覧は https://ja.wikipedia.org/wiki/ISO_639-1%E3%82%B3%E3%83%BC%E3%83%89%E4%B8%80%E8%A6%A7 にあります) を、国コードは ISO 3166 (一覧は https://ja.wikipedia.org/wiki/ISO_3166-1 にあります) をそれぞれお使いください。

なお、`/usr/lib/locale` 以下にあるディレクトリの言語および国コードなどの組み合わせのみを使用することができます。また、追加の定義ファイルを作成したい場合は、`localedef` コマンドをお使いのうえ、`/usr/share/i18n` 内にファイルを作成してください。定義ファイルは `glibc-i18ndata` に含まれているものです。たとえば `en_US.UTF-8` (イギリスおよびアメリカ英語) を作成したい場合は、下記のように実行します:

```
localedef -i en_US -f UTF-8 en_US.UTF-8
```

`LANG=en_US.UTF-8`

: インストール時にアメリカ英語を選択すると、上記のような既定の設定になります。他の言語を選択した場合は言語と国コードが変わるものの、文字エンコーディングの UTF-8 についてはそのまま保持されます。

`LANG=en_US.ISO-8859-1`

: この設定は、言語を英語に、国をアメリカに、文字セットを `ISO-8859-1` にする設定です。この文字セットにはユーロ記号などが含まれていませんが、`UTF-8` では正しく動作しないプログラムには有用な設定です。なお、文字セットの部分 (この場合は `ISO-8859-1`) については、emacs のようなプログラム側で使用されます。

`LANG=en_IE@euro`

: 上記の設定は、言語の設定内でユーロ記号を使用できるように明示的に指定しているものです。UTF-8 には既にユーロ記号が含まれていますので、この設定は既に古い方式の設定になります。このような設定は、ISO-8859-15 には対応しているものの、UTF-8 には対応していないアプリケーションを使用する際に便利です。

/etc/sysconfig/language に対する設定は、下記の処理順序で反映されます:

- bash の場合: /etc/profile が /etc/profile.d/lang.sh を読み込み、そのファイル内で /etc/sysconfig/language を読み込みます。
- tcsh の場合: ログイン時に /etc/csh.login が /etc/profile.d/lang.csh を読み込み、そのファイル内で /etc/sysconfig/language を読み込みます。

このような仕組みにより、/etc/sysconfig/language を変更すると、手作業で反映させる処理を行うことなく、次回のログイン時に言語設定が反映されるようになっています。

ユーザ側では、システムに設定された言語設定を ~/.bashrc で上書きすることができます。たとえばシステム全体に en_US が設定されている場合、この設定をプログラムメッセージに対して使用したくない場合は、LC_MESSAGES=es_ES を設定することで、英語ではなくスペイン語で表示することができるようになります。

15.4.3 ~/.i18n 内でのロケール (言語) 設定

システム側で設定された既定値では不都合がある場合は、~/.i18n ファイルを bash のスクリプト形式で記述することで、設定を変更することができます。~/.i18n 内での記述は /etc/sysconfig/language の設定を上書きすることができます。なお、変数名は /etc/sysconfig/language のものから、RC_ の接頭辞を取り除いて指定してください。たとえば RC_LANG ではなく LANG のようになります:

```
LANG=cs_CZ.UTF-8
LC_COLLATE=C
```

15.4.4 言語に対応するための設定

メッセージ に対する言語の設定には、フォールバック (後退) という仕組みが用意されています。これはたとえば、優先して使用したい言語のメッセージがアプリケーション側に用意されていない場合、提供されている言語の中からいずれか (一般的には en (英語)) を選択して表示することができます。たとえば LANG に en_US を設定していて、/usr/share/locale/en_US/LC_MESSAGES ディレクトリ内にメッセージファイルが存在しない場合、/usr/share/locale/en/LC_MESSAGES などをフォールバックとして使用することができます。

フォールバックの仕組みでは複数のものを指定することができます。たとえばブルトン語からフランス語に、もしくはガリシア語からスペイン語、ポルトガル語にフォールバックさせたい場合は、下記のように設定します:

```
LANGUAGE="br_FR:fr_FR"
```

```
LANGUAGE="gl_ES:es_ES:pt_PT"
```

また、ノルウェー語の変形であるニーノシュクとブークモールを使用したいような場合は、下記のように指定します (一般的なノルウェー語を表す `no` の意味も含まれます):

```
LANG="nn_NO"
```

```
LANGUAGE="nn_NO:nb_NO:no"
```

もしくは、下記のように指定してもかまいません:

```
LANG="nb_NO"
```

```
LANGUAGE="nb_NO:nn_NO:no"
```

ただし、ノルウェー語では `LC_TIME` も異なる形式になることに注意してください。

なお、言語の設定によって数値の桁区切り文字が正しく認識されない、という問題が発生する場合があります。これは `LANG` に 2 桁の言語コードのみを指定した場合 (例: `de`) に起こります。glibc では `/usr/share/lib/de_DE/LC_NUMERIC` に区切り文字の設定があるため、`de` では区切り文字が読み込まれないためです。この場合は、`LC_NUMERIC` に `de_DE` のような正しい言語コードおよび国コードを入力して、区切り文字を読み込めるようにしてください。

15.4.5 さらなる情報

- The GNU C Library Reference Manual (英語) 内の「Locales and Internationalization」章。この文書は `glibc-info` パッケージ内に含まれています。
- Markus Kuhn 氏による UTF-8 and Unicode FAQ for Unix/Linux (英語): 現時点では https://www.cl.cam.ac.uk/~mgk25/unicode.html にあります。

16 udev による動的なカーネルデバイス管理

改訂履歴

2024-05-13

カーネルは、それが実行中であったとしても、ほぼ全てのデバイスを追加したり削除したりすることができます。また、デバイスの状態の変化 (デバイスの接続や取り外し) についても、その情報をユーザスペース側に配信する必要がありますし、接続時や認識時に設定を行う必要もあります。また、デバイスによっては、ユーザ側にその認識状態を知らせる必要があるものもあります。`udev` では、`/dev` ディレクトリ内にあるデバイスノードファイルやシンボリックリンクファイルを動的に管理するのに必要な、インフラストラクチャを提供しています。また、`udev` のルールでは、カーネルのデバイスイベントを処理する外部プログラムを定義する機能も提供しています。これにより `udev` は、カーネルのデバイス処理の一部として独自のスクリプトを動作させることができるほか、デバイスの処理時に必要な追加のデータを要求したり、取り込んだりすることができるようになっています。

16.1 /dev ディレクトリ

`/dev` ディレクトリ内にあるデバイスノードは、対応するカーネルデバイスへのアクセス機能を提供します。`udev` を使用する環境では、`/dev` ディレクトリはカーネル側での現状を反映したものになっています。それぞれのカーネルデバイスはデバイスファイルという形で提供され、システムからデバイスが取り外されると、デバイスノードも削除されるようになっています。

`/dev` ディレクトリの内容は一時的なファイルシステムとして用意されているもので、システムが起動するたびに新しくデバイスファイルを作成しています。手作業でデバイスファイルを作成したり変更したりしても、そのような構造から、システムを再起動するとリセットされてしまいます。`/dev` 内に、カーネル側の認識とは無関係に、固定で配置すべきファイルやディレクトリがある場合は、systemd-tmpfiles を利用して作成する必要があります。設定ファイルはそれぞれ `/usr/lib/tmpfiles.d/` と `/etc/tmpfiles.d/` にあります。詳しくは `systemd-tmpfiles(8)` のマニュアルページをお読みください。

16.2 カーネルの uevent と udev

必要なデバイス情報は `sysfs` ファイルシステム経由で開示されています。それぞれのデバイスに対して、カーネルが認識し準備を行うと、デバイス名のディレクトリを作成します。ここには属性ファイルのほか、デバイス固有のプロパティファイルなどが含まれています。

デバイスが追加されたり削除されたりすると、カーネルは uevent を送信し、`udev` に変更を通知します。`udev` デーモンは、その起動時に `/usr/lib/udev/rules.d/*.rules` および `/etc/udev/rules.d/*.rules` にあるファイルから、設定されている全てのルールを読み込み、メモリ内に保持し

ます。ルールファイルを変更したり、追加もしくは削除を行ったりした場合は、`udevadm control --reload` コマンドでデーモン側に再読み込みを依頼することができます。`udev` のルールとその文法について、詳しくは 16.6項「udev ルールによるカーネルのデバイスイベント処理の調整」をお読みください。

受信したそれぞれのイベントは、読み込んだルールに対して適合可否を判断します。ルール側では環境キーを追加したり変更したりすることができるほか、作成すべきデバイスノードの名前を指定したり、ノードを指し示すシンボリックリンクを追加したり、デバイスノードの作成後に実行すべきプログラムなどを指定したりすることができます。なお、`uevent` はカーネルの netlink ソケットを介して受信されます。

16.3 ドライバ／カーネルモジュール／デバイス

カーネルのバスドライバがデバイスを探索します。カーネルは検出されたデバイスごとに内部デバイス構造を作成し、ドライバコアが uevent を `udev` デーモンに送信します。バスデバイスは特殊な書式の ID で識別され、その ID にはデバイスの種類が示されています。これらの ID には製造元 (ベンダ) と製品 ID のほか、サブシステム固有の値が含まれています。また、バスごとに ID の割り当てルールが存在していて、それらは `MODALIAS` と呼ばれています。カーネルはデバイスの情報を取得して `MODALIAS` ID 文字列を生成し、イベントと共にその文字列を送信します。たとえば USB マウスの場合、下記のようになります:

```
MODALIAS=usb:v046DpC03Ed2000dc00dsc00dp00ic03isc01ip02
```

それぞれのデバイスドライバには、自身が扱うことのできる既知の別名一覧が含まれています。この一覧はカーネルモジュールファイル自身に含まれています。depmod プログラムは、利用可能な全てのモジュール内にある ID の一覧を読み込んで、`/lib/modules` 内にある対応するディレクトリ内に `modules.alias` ファイルを作成します。このような構造になっていることから、モジュールの読み込みは、それぞれのイベント時に `MODALIAS` キーを取得し、`modprobe` を呼び出すだけの簡単な作業で済むようになっています。`modprobe $MODALIAS` を呼び出すと、モジュール側で提供されている別名と、デバイス向けに作成した別名を比較して、一致する項目があれば読み込むようになっています。これらの全ては `udev` によって発動される処理です。

16.4 システムの起動と初期デバイス設定

システムの起動時、`udev` が動作する前に発生したデバイスのイベントは、読み込まれずに失われてしまいます。これは、これらのイベントを処理するインフラストラクチャがルートファイルシステム内に存在しているため、起動処理中には利用できないためです。このような問題を解決するため、カーネル側では `sysfs` ファイルシステム内の各デバイスのディレクトリ内に `uevent` ファイルを用意しています。

このファイルに add を書き込むと、カーネルは起動時に送信したものと同じイベントを再送するようになっています。/sys ディレクトリ内にある全ての uevent ファイルに対して、同じことを繰り返すことで、必要なデバイスノードの作成とデバイスの設定を行うことができるようになっています。

たとえば起動時に USB マウスを接続していた場合、ドライバは起動時には利用できないため、初期の起動ロジックでは準備が行われません。そのため、デバイスの検出イベントは失われ、デバイスに対応するカーネルモジュールの検出も失敗します。そのため udev では、ルートファイルシステムの準備が完了した段階で、接続されている全てのデバイスを手作業で検索するのではなく、カーネルから発せられた起動時の全てのイベントを取得します。これにより、USB マウスのイベントが届くようになりますので、ルートファイルシステム内のカーネルモジュールを読み込んで初期化することができるようになっています。

ユーザスペース側では、起動時にデバイスを接続しても起動後にデバイスを接続しても、目に見える違いはありません。いずれの場合も、同じルールが適用され、同じ設定プログラムが動作します。

16.5 動作中の udev デーモンの監視

udevadm monitor プログラムを利用することで、ドライバコアのイベントや udev のイベント処理のタイミングを視覚化することができます。

```
UEVENT[1185238505.276660] add   /devices/pci0000:00/0000:00:1d.2/usb3/3-1 (usb)
UDEV  [1185238505.279198] add   /devices/pci0000:00/0000:00:1d.2/usb3/3-1 (usb)
UEVENT[1185238505.279527] add   /devices/pci0000:00/0000:00:1d.2/usb3/3-1/3-1:1.0 (usb)
UDEV  [1185238505.285573] add   /devices/pci0000:00/0000:00:1d.2/usb3/3-1/3-1:1.0 (usb)
UEVENT[1185238505.298878] add   /devices/pci0000:00/0000:00:1d.2/usb3/3-1/3-1:1.0/input/
input10 (input)
UDEV  [1185238505.305026] add   /devices/pci0000:00/0000:00:1d.2/usb3/3-1/3-1:1.0/input/
input10 (input)
UEVENT[1185238505.305442] add   /devices/pci0000:00/0000:00:1d.2/usb3/3-1/3-1:1.0/input/
input10/mouse2 (input)
UEVENT[1185238505.306440] add   /devices/pci0000:00/0000:00:1d.2/usb3/3-1/3-1:1.0/input/
input10/event4 (input)
UDEV  [1185238505.325384] add   /devices/pci0000:00/0000:00:1d.2/usb3/3-1/3-1:1.0/input/
input10/event4 (input)
UDEV  [1185238505.342257] add   /devices/pci0000:00/0000:00:1d.2/usb3/3-1/3-1:1.0/input/
input10/mouse2 (input)
```

UEVENT の行には、カーネルが netlink 経由で送信したイベントが表示されています。UDEV の行には、完了した udev のイベントハンドラが表示されています。それぞれのタイミングはマイクロ秒単位で示されています。UEVENT から UDEV までの時間は、udev がイベントを処理するのにかかった時間か、もしくは udev が関連するイベントを待機していて遅延した時間を示しています。たとえばハードディスクのパーティションに関するイベントは、メインのディスクのイベント内で、ハードウエアに対して問い合わせを行った結果を待たなければならないため、その分だけ遅延することになります。

`udevadm monitor --env` を実行すると、完全なイベント環境を表示することができます:

```
ACTION=add
DEVPATH=/devices/pci0000:00/0000:00:1d.2/usb3/3-1/3-1:1.0/input/input10
SUBSYSTEM=input
SEQNUM=1181
NAME="Logitech USB-PS/2 Optical Mouse"
PHYS="usb-0000:00:1d.2-1/input0"
UNIQ=""
EV=7
KEY=70000 0 0 0 0
REL=103
MODALIAS=input:b0003v046DpC03Ee0110-e0,1,2,k110,111,112,r0,1,8,amlsfw
```

`udev` はメッセージを syslog にも送信します。どのメッセージを syslog に送信するかを決定する、既定の syslog の優先順位は、`udev` の設定ファイルである `/etc/udev/udev.conf` で設定します。実行中のデーモンのログ優先順位は、`udevadm control --log_priority=` レベル/番号 で設定することができます。

16.6 udev ルールによるカーネルのデバイスイベント処理の調整

`udev` のルールは、カーネルがイベント自身に付与するプロパティ情報のほか、カーネルが `sysfs` で開示する任意の情報を条件として設定することができます。このほか、外部プログラムから追加の情報を取得することもできます。また、それぞれのイベントは提供されている全てのルールに対して適合可否を判断します。全てのルールは `/usr/lib/udev/rules.d/` (既定のルール) および `/etc/udev/rules.d` (システム固有の設定) 内に配置されます。

ルールファイルの各行には、少なくとも 1 つ以上のキーと値の対が含まれています。キーには 2 種類のものがあり、それぞれマッチ (適合可否) キーと代入キーと呼ばれています。ルール内のマッチキーで指定された全ての条件が満たされると、そのルールが適用され、代入キーに対して値が割り当てられます。また、マッチキーのルールではデバイスノードの名前を指定することができるほか、ノードを指し示すシンボリックリンクを追加したり、イベント処理の一部として指定したプログラムを起動したりすることができます。なお、いずれのルールにも該当しない場合は、デバイスノードを作成する際に既定のデバイス名が使われます。ルールの文法やマッチキーの一覧、およびデータの取り込み方法などについては、`udev` のマニュアルページをお読みください。下記の例では、`udev` のルールに関する基本的な文法の説明を行っています。ルールの例は、`/usr/lib/udev/rules.d/50-udev-default.rules` にある `udev` の既定のルールから引用されているものです。

例 16.1: udev ルールの例

```
# コンソール
```

```
KERNEL=="console", MODE="0600", OPTIONS="last_rule"

# シリアルデバイス
KERNEL=="ttyUSB*", ATTRS{product}=="[Pp]alm*Handheld*", SYMLINK+="pilot"

# プリンタ
SUBSYSTEM=="usb", KERNEL=="lp*", NAME="usb/%k", SYMLINK+="usb%k", GROUP="lp"

# カーネルファームウエアローダ
SUBSYSTEM=="firmware", ACTION=="add", RUN+="firmware.sh"
```

`console` のルールには 3 種類のキーが設定されています。1 つ目はマッチキー（ `KERNEL` ）で、残りの 2 つは代入キーになっています（ `MODE` , `OPTIONS` ）。`KERNEL` のマッチキーは、種類が `console` である任意のデバイスに適合するものです。この場合は厳密一致で比較処理が行われ、比較が成功した場合にのみルールが実行されます。`MODE` キーはデバイスノードに対するアクセス権の設定を示すもので、この場合はこのデバイスの所有者のみが読み書きできるルールになります。最後の `OPTIONS` キーは、この種類の任意のデバイスに対して、最後に適用すべきルールであることを示しています。この種類のデバイスの場合、これ以降のルールは適用されなくなります。

`serial devices` のルールは既に `50-udev-default.rules` から削除されているルールですが、ルールを理解するには格好の材料です。ここには 2 種類のマッチキー（ `KERNEL` と `ATTRS` ）と、1 種類の代入キー（ `SYMLINK` ）が書かれています。`KERNEL` キーはデバイスの種類が `ttyUSB` である任意のデバイスに適合するものです。ここでは `*` というワイルドカードを指定しているため、厳密一致ではなく複数の種類にマッチできるようになっています。2 つ目のマッチキーである `ATTRS` では、種類が `ttyUSB` である任意のデバイスに対して、`sysfs` 内にある `product` という属性ファイルを読み込んで、特定の文字列に合致するかどうかを調べています。代入キー（ `SYMLINK` ）は、このデバイスに対するシンボリックリンクを `/dev/pilot` として作成するよう指示しています。このキーで使用されている（ `+=` ）という演算子は `udev` に対して、以前のルールや後続のルールで他のシンボリックリンクを作成している場合でも、そのルールを適用することを示しています。なお、このルールには 2 種類のマッチキーがありますので、両方の条件に合致した場合にのみルールが適用されることにも注意してください。

`printer` ルールは USB プリンタを扱うためのルールで、ルール全体を適用する際の条件となる 2 種類のマッチキー（ `SUBSYSTEM` および `KERNEL` ）が含まれています。また、3 種類の代入キーは、それぞれこの種類のデバイスの命名ルール（ `NAME` ）やデバイスのシンボリックリンクの作成方法（ `SYMLINK` ）、そしてこの種類のデバイスに対するグループメンバー設定（ `GROUP` ）が書かれています。また、`KERNEL` 内には `*` というワイルドカードが含まれているため、複数の `lp` プリンタデバイスに合致する仕組みになっていのす。また、`NAME` と `SYMLINK` には置換が設定されていて、これによって内部デバイス名からそれぞれを拡張できるようになっています。たとえば `lp` という内部名の最初の USB プリンタであれば、`/dev/usblp0` のような名前になります。

`kernel firmware loader` は `udev` から実行時に外部のヘルパースクリプトを実行して、追加のファームウエアを読み込むためのものです。`SUBSYSTEM` のマッチキーは `firmware` サブシステムを検索します。`ACTION` キーは、`firmware` サブシステムに属する任意のデバイスが追加されたかどうかを判断しています。`RUN+=` キーは `firmware.sh` スクリプトの実行を指定していて、これによってファームウエアの読み込みを行います。

全てのルールに対して適用される、一般的な規約は下記の通りです:

- それぞれのルールには、カンマ区切りで 1 つまたは複数のキー／値の対を指定します。
- キーの操作は演算子で決定します。`udev` では複数の演算子に対応しています。
- それぞれの値は引用符で括らなければなりません。
- ルールファイル内の 1 行は 1 つのルールに対応します。ルールが 1 行に収まらない場合は、シェルでの記法と同じく、行末に `\` を記述して次の行に続けます。
- `udev` のルールでは、シェル形式のパターンマッチに対応しています。具体的には `*`, `?`, `[]` の各パターンを利用することができます。
- `udev` では置換に対応しています。

16.6.1 udev ルール内での演算子の使用

代入キーの場合は、代入するキーの種類によって利用できる演算子が異なります。マッチキーの場合は通常、値と比較するための様々な演算子を利用することができます。マッチキーでは、下記の演算子を指定することができます:

`==`
: 一致しているかどうかを判断します。値にパターンマッチが指定されている場合は、そのパターンに一致するかどうかを判断します。

`!=`
: 不一致を判断します。値にパターンマッチが指定されている場合は、そのパターンに一致していないかどうかを判断します。

代入キーの場合は、下記の演算子を指定することができます:

`=`
: キーに対して値を代入します。以前に値や値のリストが代入されていた場合は、その値は消去され、指定した単一の値が代入されます。

`+=`
: 項目の一覧内に値を追加します。

`:=`
: 最終値を代入します。後続のルールで変更ができなくなります。

16.6.2 udev ルール内での置換の使用

`udev` ではプレースホルダや置換の仕組みを利用することができます。これは他のスクリプトでも利用できる仕組みに似たもので、`udev` のルールでは下記のものを使用することができます:

`%r` , `$root`
: デバイスのディレクトリ (既定では `/dev`)

`%p` , `$devpath`
: `DEVPATH` の値

`%k` , `$kernel`
: `KERNEL` の値もしくは内部デバイス名

`%n` , `$number`
: デバイスの番号

`%N` , `$tempnode`
: デバイスファイルの一時的な名前

`%M` , `$major`
: デバイスのメジャー番号

`%m` , `$minor`
: デバイスのマイナー番号

`%s{属性名}` , `$attr{属性名}`
: `sysfs` の属性の値 (`属性名` で名前を指定します)

`%E{変数名}` , `$env{変数名}`
: 環境変数の値 (`変数名` で名前を指定します)

`%c` , `$result`
: `PROGRAM` の出力

`%%`

`%` 文字そのもの

`$$`

`$` 文字そのもの

16.6.3 udev マッチキーの使用

マッチキーは `udev` のルールを適用する際、条件を指定するものです。下記のマッチキーを使用することができます:

`ACTION`

イベントアクションの名前 (デバイスを追加する場合は `add` に、デバイスを削除する場合は `remove` になります)

`DEVPATH`

イベントデバイスのデバイスパス (たとえば `DEVPATH=/bus/pci/drivers/ipw3945` のように指定すると、ipw3945 ドライバに関連する全てのイベントに合致するようになります)

`KERNEL`

イベントデバイスの内部 (カーネル) 名

`SUBSYSTEM`

イベントデバイスのサブシステム名 (たとえば `SUBSYSTEM=usb` のように指定すると、USB デバイスに関連する全てのイベントに合致するようになります)

`ATTR{ファイル名}`

イベントの `sysfs` 属性 (たとえば `vendor` 内に特定の文字列が含まれているかどうかを調べるには、`ATTR{vendor}=="On[sS]tream"` のように指定します)

`KERNELS`

`udev` に対して、該当するデバイス名を上位方向に検索させる指定

`SUBSYSTEMS`

`udev` に対して、該当するデバイスのサブシステム名を上位方向に検索させる指定

`DRIVERS`

`udev` に対して、該当するデバイスのドライバ名を上位方向に検索させる指定

`ATTRS{ファイル名}`

`udev` に対して、該当する `sysfs` の属性名を上位方向に検索させる指定

`ENV{キー}`

環境変数の値 (たとえば `ENV{ID_BUS}="ieee1394"` のように指定すると、FireWire バス ID に関連する全てのイベントに合致するようになります)

`PROGRAM`

`udev` に対して外部プログラムを実行する指定 (成功を表す場合、プログラムは 0 を返さなければなりません。また、プログラムが STDOUT に出力した実行結果は、`RESULT` キーでチェックすることができます)

`RESULT`

直近の `PROGRAM` の出力 (`PROGRAM` キーと同じルール内に含めるか、後続のルールで指定する必要があります)

16.6.4 udev 代入キーの使用

上述のように合致すべき条件を指定するマッチキーとは異なり、代入キーでは `udev` が管理するデバイスノードに対して、値や名前、アクションなどを代入します。

`NAME`

作成すべきデバイスノードの名前 (ルール内でデバイスノード名が設定されると、`NAME` キーがある他のルールは全て無視されるようになります)

`SYMLINK`

このノードに対して作成すべき関連シンボリックリンクの名前 (特定のデバイスノードに対して、複数のルールでシンボリックリンクを作成することができます。また、1 つのルール内でも、スペースで区切って指定することで、複数のシンボリックリンクを作成することができます。

`OWNER, GROUP, MODE`

新しいデバイスノードのアクセス権 (ここで何らかの値を指定すると、内蔵されている既定のルールが上書きされます)

`ATTR{キー}`

イベントデバイスの `sysfs` 属性に対して書き込むべき値 (`==` 演算子を使用した場合、このキーはマッチキーとして作用し、特定の `sysfs` 属性と比較が行われることに注意してください)

`ENV{キー}`

`udev` に対して、この環境変数に設定すべき値 (`==` 演算子を使用した場合、このキーはマッチキーとして作用し、特定の `sysfs` 属性と比較が行われることに注意してください)

RUN

udev に対して、このデバイス向けに実行すべきプログラムの一覧への追加 (ただし、後続の処理が遅延したりすることのないよう、短時間で終了する処理にしてください)

LABEL

GOTO でジャンプできる先のラベル指定

GOTO

udev に対して、ラベルで指定した位置までいくつかのルールを読み飛ばす指定

IMPORT{種類}

外部プログラムの出力などをイベント環境に読み込む指定 (udev は複数の種類の変数を取り込むことができます。種類を指定しない場合、udev はファイルのアクセス権をもとに実行可否を自身で判断するようになります)

- program を指定すると、udev は外部プログラムを実行して出力を取り込みます。
- file を指定すると、udev はテキストファイルから取り込みを行います。
- parent を指定すると、udev は親デバイスに保存されているキーを取り込みます。

WAIT_FOR_SYSFS

udev に対して、指定した sysfs ファイルが作成されるまで待機する指定 (たとえば WAIT_FOR_SYSFS="ioerr_cnt" のように指定すると、udev は ioerr_cnt ファイルが作成されるまで待機します)

OPTIONS

OPTION キーには下記のような値を指定することができます:

- last_rule を指定すると、udev は後続の全てのルールを無視するようになります。
- ignore_device を指定すると、udev はこのイベントを完全に無視するようになります。
- ignore_remove を指定すると、udev はこのデバイスに対して後から発生する削除イベントを無視するようになります。
- all_partitions を指定すると、udev はブロックデバイス内に存在する全てのパーティションに対して、デバイスノードを作成するようになります。

16.7 デバイスの命名の恒久化

動的なデバイスディレクトリと udev ルールの構造により、デバイスの認識順序や接続方式によらず、全てのディスクデバイスに対して、一貫した命名を行うことができるようになっています。カーネルが作成するそれぞれのブロックデバイスは、バスやドライブの種類、ファイルシステムに対して特別な知識のあるツールが検証して作成することになります。カーネルが提供するデバイスノード名と共に、udev ではそのデバイスを指し示す一貫したシンボリックリンクを作成します:

```
/dev/disk
|-- by-id
|   |-- scsi-SATA_HTS726060M9AT00_MRH453M4HWHG7B -> ../../sda
|   |-- scsi-SATA_HTS726060M9AT00_MRH453M4HWHG7B-part1 -> ../../sda1
|   |-- scsi-SATA_HTS726060M9AT00_MRH453M4HWHG7B-part6 -> ../../sda6
|   |-- scsi-SATA_HTS726060M9AT00_MRH453M4HWHG7B-part7 -> ../../sda7
|   |-- usb-Generic_STORAGE_DEVICE_02773 -> ../../sdd
|   `-- usb-Generic_STORAGE_DEVICE_02773-part1 -> ../../sdd1
|-- by-label
|   |-- Photos -> ../../sdd1
|   |-- SUSE10 -> ../../sda7
|   `-- devel -> ../../sda6
|-- by-path
|   |-- pci-0000:00:1f.2-scsi-0:0:0:0 -> ../../sda
|   |-- pci-0000:00:1f.2-scsi-0:0:0:0-part1 -> ../../sda1
|   |-- pci-0000:00:1f.2-scsi-0:0:0:0-part6 -> ../../sda6
|   |-- pci-0000:00:1f.2-scsi-0:0:0:0-part7 -> ../../sda7
|   |-- pci-0000:00:1f.2-scsi-1:0:0:0 -> ../../sr0
|   |-- usb-02773:0:0:2 -> ../../sdd
|   |-- usb-02773:0:0:2-part1 -> ../../sdd1
`-- by-uuid
    |-- 159a47a4-e6e6-40be-a757-a629991479ae -> ../../sda7
    |-- 3e999973-00c9-4917-9442-b7633bd95b9e -> ../../sda6
    `-- 4210-8F8C -> ../../sdd1
```

16.8 udev で使用されるファイル

`/sys/*`

Linux カーネルが提供する仮想ファイルシステムで、既知のすべてのデバイスに対する情報を保持しています。この情報は、udev が /dev 内にデバイスノードを作成する際に用いられます。

`/dev/*`

動的に作成されたデバイスノードと、systemd-tmpfiles で指定された固定の内容が含まれます。詳しくは systemd-tmpfiles(8) のマニュアルページをお読みください。

下記のファイルやディレクトリは、udev インフラストラクチャにとって重要なものとなっています:

/etc/udev/udev.conf
: udev のメインの設定ファイルです。

/etc/udev/rules.d/*
: システム固有の udev マッチングルールが含まれています。ここに独自のルールを設定することで、/usr/lib/udev/rules.d/* にある既定のルールを変更したり、上書きしたりすることができます。
ファイルはアルファベット／数字順に読み込まれます。また、低い優先順位のファイルに書かれたルールは、より高い優先順位のファイルに書かれたルールによって変更されたり、上書きされたりします。数字の小さいものほど高い優先順位が割り当てられます。

/usr/lib/udev/rules.d/*
: 既定の udev イベントマッチングルールが含まれています。このディレクトリ内にあるファイルは、それぞれのパッケージが所有しているファイルであるため、将来の更新によって上書きされることがあります。そのため、このディレクトリ内にあるファイルに対しては、追加や削除、編集などを行ってはなりません。追加や削除、編集などを行いたい場合は、/etc/udev/rules.d ディレクトリをお使いください。

/usr/lib/udev/*
: udev のルールから呼び出されるヘルパープログラムです。

/usr/lib/tmpfiles.d/ および /etc/tmpfiles.d/
: /dev 以下に配置する固定の内容を設定する場所です。

16.9 さらなる情報

udev のインフラストラクチャについて、詳しくは下記のマニュアルページをお読みください:

udev
: udev に関する一般的な情報のほか、ルールやその他の重要な設定問題について説明しています。

udevadm
: udevadm は udev の実行時の振る舞いを制御する際に使用するもので、カーネルイベントの要求やイベントキューの管理、シンプルなデバッグ機構などが提供されています。

udevd
: udev イベント管理デーモンに関する情報が記されています。

III サービス

17 SLP 349

18 NTP を利用した時刻同期 353

19 ドメインネームシステム 363

20 DHCP 389

21 Samba 406

22 NFS によるファイル共有 433

23 autofs によるオンデマンド型のマウント 455

24 Apache HTTP サーバ 463

25 YaST による FTP サーバの設定 507

26 squid: キャッシュ機能付きプロキシサーバ 512

17 SLP

改訂履歴

2024-05-13

ネットワーククライアントの設定にあたっては、ネットワーク側で提供されている様々なサービス (印刷関連のプロトコルや LDAP など) に関する知識が必要となります。「Service Location Protocol」(SLP) はそのような目的で開発された仕組みで、ネットワークサービスを利用するクライアントの設定をより簡単にすることができるようになっています。SLP では、ローカルのネットワーク内に存在する全てのクライアントに対して、選択したサービスが利用できる旨と、その設定データを提供することができます。SLP に対応したアプリケーションであれば、その情報をもとに自動設定を行うことができます。

openSUSE® Leap では SLP 経由でインストールソースの情報を提供し、その情報をもとにインストールを行うことができます。また、多数のシステムサービスが SLP と統合しています。SLP を利用することで、お使いのシステム内に存在するインストールサーバやファイルサーバ、印刷サーバなどの中央サーバに関する情報を、ネットワークに接続されたクライアントに通知することができます。SLP に対応するサービスとしては、cupsd, login, ntp, openldap2-client, postfix, rpasswd, rsyncd, saned, sshd (fish 経由), vnc, ypserv などがあります。

ネットワーククライアント側では、SLP のサービスを利用するにあたって必要なパッケージが既定でインストールされています。逆に SLP を介してサービスを 提供する 場合は、`openslp-server` パッケージをインストールする必要があります。

17.1 SLP フロントエンド `slptool`

`slptool` ツールは SLP のサービスを問い合わせたり、登録したりすることができるコマンドラインツールです。問い合わせの機能は調査を行うような場合に便利です。最も重要な `slptool` サブコマンドを下記に示します。`slptool --help` を実行すると、利用可能の全てのオプションと機能を一覧表示することができます。

findsrvtypes
: ネットワーク内で利用可能な全てのサービスタイプを一覧表示します。

```
> slptool findsrvtypes
service:install.suse:nfs
service:install.suse:ftp
service:install.suse:http
service:install.suse:smb
```

```
service:ssh
service:fish
service:YaST.installation.suse:vnc
service:smtp
service:domain
service:management-software.IBM:hardware-management-console
service:rsync
service:ntp
service:ypserv
```

findsrvs サービスタイプ

サービスタイプ を提供する全てのサーバを一覧表示します。

```
> slptool findsrvs service:ntp
service:ntp://ntp.example.com:123,57810
service:ntp://ntp2.example.com:123,57810
```

findattrs サービスタイプ // ホスト

指定した ホスト の サービスタイプ に対して、属性情報を表示します。

```
> slptool findattrs service:ntp://ntp.example.com
(owner=tux),(email=tux@example.com)
```

register サービスタイプ // ホスト : ポート "(属性=値),(属性=値)"

指定した ホスト の サービスタイプ を、属性の一覧と共に登録します。

```
slptool register service:ntp://ntp.example.com:57810 \
"(owner=tux),(email=tux@example.com)"
```

deregister サービスタイプ // ホスト

指定した ホスト の サービスタイプ の登録を解除します。

```
slptool deregister service:ntp://ntp.example.com
```

さらに詳しい情報については、`slptool --help` をお読みください。

17.2 SLP を介したサービス情報の提供

SLP 経由でサービス情報を公開するには、SLP デーモン (`slpd`) を動作させなければなりません。openSUSE Leap でのほとんどのサービスと同様に、`slpd` は個別の起動スクリプトで制御しています。なお、インストールを行っても、デーモンは既定で有効化されません。現在のセッションでサービスを起動するには、`sudo systemctl start slpd` を実行します。システムの起動時に `slpd` を開始したい場合は、`sudo systemctl enable slpd` を実行してください。

openSUSE Leap では、多数のアプリケーションが `libslp` ライブラリを利用して SLP に対応しています。特定のサービスが SLP サポート付きでコンパイルされていない場合は、下記のいずれかの方法で SLP に対応させることができます:

`/etc/slp.reg.d` を利用した固定登録

新しいサービスそれぞれに対して、個別の登録ファイルを作成して対応します。下記の例では、スキャナサービスを登録しています:

```
## このシステム内にある saned サービスを登録しています。
## en は英語を、 65535 はタイムアウトの無効化を意味しています。
## タイムアウトを無効化することで、サービスの更新を不要にしています。
service:scanner.sane://$HOSTNAME:6566,en,65535
watch-port-tcp=6566
description=SANE scanner daemon
```

このファイル内で最も重要な行は `service:` で始まる service URL の行です。ここにはサービスの種類 (`scanner.sane`) のほか、サーバ内でサービスが提供されているアドレスが示されています。`$HOSTNAME` を指定すると、自動的に自分自身の完全修飾ホスト名に置き換えられます。その後ろにはコロンに続いて、サービスが待ち受けている TCP ポートが書かれています。その後ろにはカンマ区切りでサービスの提供言語と、登録の有効期間が秒単位で示されています。登録の有効期間は `0` から `65535` までの間で指定します。ただし、`0` は登録の無効化を、`65535` は無期限の登録を表します。
また、登録ファイルには `watch-port-tcp` と `description` という行があります。`watch-port-tcp` は SLP によるサービスの告知と関係するサービスを連携させる仕組みで、`slpd` がサービスの状態をチェックしてサービスを告知するようになります。また、2 つ目の値はブラウザ側に表示するテキストで、サービスに関するより詳しい説明を記述します。

ヒント: YaST と SLP の関係について

インストールサーバや YOU (YaST オンライン更新) などの場合、YaST が仲介役となってサービスを登録することができます。この登録は、モジュールのダイアログ内で SLP を有効化することで、自動的に行うことができます。YaST 側では、これらのサービスに対する登録ファイルを作成します。

`/etc/slp.reg` を利用した固定登録

`/etc/slp.reg.d` とこの方式の違いは、全てのサービスを 1 つのファイルにまとめる、という点のみです。

`slptool` を利用した動的な登録

設定ファイルを使用せずに動的に登録を行う必要がある場合は、`slptool` コマンドラインユーティリティをお使いください。同ユーティリティを使用することで、`slpd` を再起動したりすることなく、既存のサービスの登録を解除することもできます。詳しくは 17.1項「SLP フロントエンド `slptool`」をお読みください。

17.2.1 SLP インストールサーバの構築

お使いのネットワーク内で SLP を介してインストールデータを告知することで、ネットワーク経由でのインストールをより簡単に行うことができるようになります。特に、インストールメディアの存在するサーバの IP アドレスやパスなどを、クライアント側で自動的に取得できる点が便利です。

17.3 さらなる情報

RFC 2608, 2609, 2610

RFC 2608 には SLP に対する一般的な説明があります。また、RFC 2609 にはサービス URL の書式説明が、RFC 2610 には DHCP 経由での SLP の説明があります。

http://www.openslp.org

OpenSLP プロジェクトのホームページです。

`/usr/share/doc/packages/openslp`

このディレクトリには、`openslp-server` パッケージに付属するドキュメンテーションが配置されています。また、openSUSE Leap での詳細は、`README.SUSE` ファイルに書かれているほか、RFC や 2 種類の入門文書なども用意されています。SLP の機能を利用したいプログラマ向けには、`openslp-devel` パッケージ内に含まれている Programmers Guide (プログラマガイド) をお読みになることをお勧めします。

18 NTP を利用した時刻同期

改訂履歴

2025-04-25

NTP (Network Time Protocol) はネットワークを介してシステムの時刻を同期させるためのプロトコルです。まず、コンピュータは信頼できる時刻の発信源となるサーバから、時刻を取得します。次にコンピュータ自身が時刻の発信源となって、ネットワークに時刻を提供します。つまり、目的は 2 つ存在することになります。絶対時刻を管理することと、ネットワーク内にある全マシンの時刻を同期させることです。

システム時刻の正確性は、様々な状況下で重要となります。内蔵のハードウエア時計は、データベースやクラスタなどのアプリケーションの要件に適合するほど正確なものではありませんし、だからといって手作業で時刻を修正してしまったりすると、時刻が逆戻りすることによって、重要なアプリケーションの動作に問題が発生したりしてしまいます。ネットワークに接続されているコンピュータであれば、通常は全てのマシンのシステム時刻を同期させる必要がありますが、この場合も手作業での修正は好ましくありません。NTP ではこれらの問題を解決する仕組みを備えています。NTP では、ネットワーク経由でシステムの時刻を信頼できる時刻に少しずつ調整する仕組みを提供しています。このほか、ラジオ制御の時計など、ローカルの参照時計を管理する機能もあります。

openSUSE Leap 15 以降では、`chrony` が NTP の既定の実装になっています。`chrony` は 2 種類のパーツから構成されています。1 つは `chronyd` と呼ばれるデーモンで、システムの起動時に開始されるものです。もう 1 つは `chronyc` と呼ばれるもので、`chronyd` の状態を監視したり、動作中にパラメータを変更したりするためのコマンドラインインターフェイスです。

openSUSE Leap 15.2 以降では、YaST モジュールの NTP クライアントは `chrony` をデーモンを使用せずに実行するよう設定する際、cron デーモンではなく systemd-timer を使用するようになっています。

注記

Active Directory を利用して時刻同期を行う場合は、『セキュリティ強化ガイド』、第8章「Active Directory サポート」、8.3.3項「[Windows ドメインメンバーシップ] を利用した Active Directory ドメインへの参加」、[Windows ドメインメンバーシップ] を利用した Active Directory ドメインへの参加 に示されている手順に従ってください。

18.1 YaST を利用した NTP クライアントの設定

chrony パッケージに付属する NTP デーモン (chronyd) は、ローカルコンピュータのハードウエア時計を参照先として使用するように事前設定されています。ハードウエア時計の正確さは、その時刻の発信源に大きく依存しています。たとえば原子時計や GPS レシーバは正確な時刻源となりますが、一般的な PC に搭載されている RTC チップは、信頼できる時刻源とは言えません。このような場合は、YaST を利用して時刻同期を行ってください。

YaST NTP クライアント設定 ([ネットワークサービス] › [NTP 設定]) のウインドウでは、NTP デーモンの起動のタイミングや設定元の種類、そして独自のタイムサーバの指定を行うことができます。

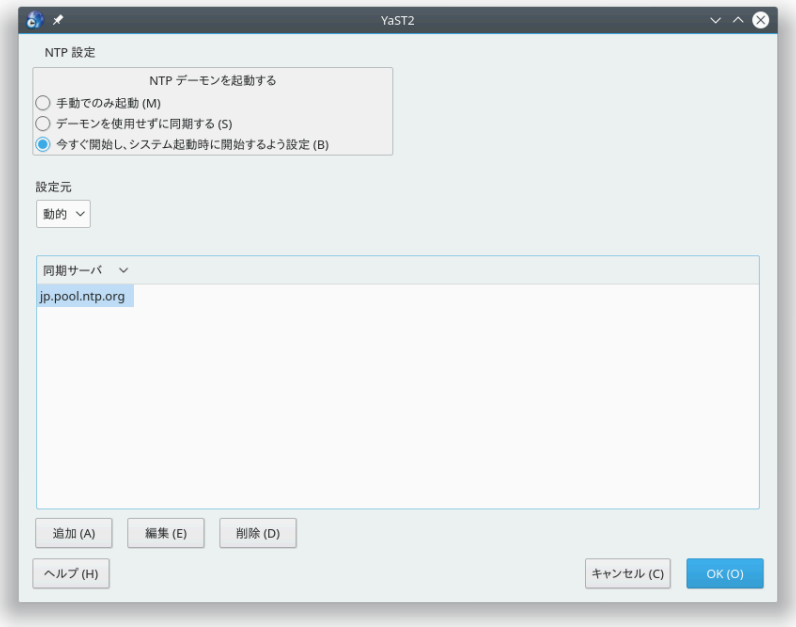


図 18.1: NTP 設定ウインドウ

18.1.1 NTP デーモンの開始

NTP デーモンの起動については、下記のいずれかを選択することができます:

[手動でのみ起動]

必要な場合にのみ chrony デーモンを手作業で起動したい場合は、[手動でのみ起動] を選択してください。

[デーモンを使用せずに同期する]

`chrony` を恒久的に起動したりせず、定期的にシステム時刻を設定したい場合は、[デーモンを使用せずに同期する] を選択してください。なお、[同期間隔 [分]] で間隔を指定することもできます。

[今すぐ開始し、システム起動時に開始するよう設定]

システムの起動時に `chronyd` を開始したい場合は、[今すぐ開始し、システム起動時に開始するよう設定] を選択してください。こちらを選択しておくことをお勧めします。

18.1.2 設定元の種類

[設定元] のドロップダウンボックスでは、[動的] または [静的] のいずれかを選択します。お使いのサーバが固定の (公開されている) NTP サーバを利用する場合は [静的] を、DHCP を介して NTP サーバの情報を提供しているネットワークの場合は [動的] を選択してください。

18.1.3 タイムサーバの設定

時刻の問い合わせ先となるタイムサーバの設定は、[NTP 設定] ウインドウの下半分に書かれています。この一覧は、[追加]，[編集]，[削除] の各ボタンで変更することができます。

新しいタイムサーバを追加するには、[追加] を押します:

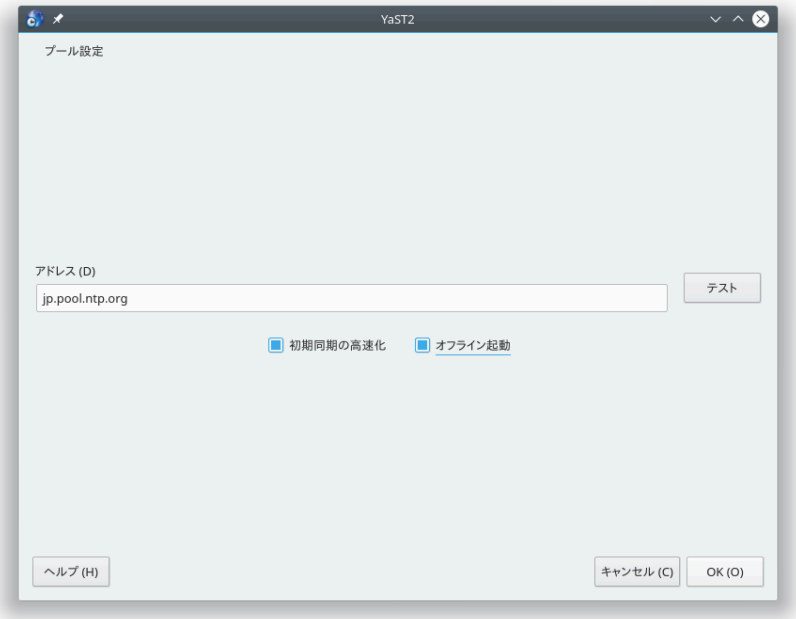


図 18.2: タイムサーバの追加

1. ［アドレス］欄には、そのマシンの同期先となるタイムサーバの URL か、タイムサーバのプールの URL を指定します。入力が終わったら［テスト］を押すと、タイムサーバとの通信を確認することができます。

2. `chronyd` の起動時に多くの要求を送信して同期を高速化するには、［初期同期の高速化］を選択します。

3. ［オフライン起動］を選択すると、システムの起動時に `chronyd` を素早く起動することができます。これは、システムの起動時にインターネットに接続できない環境のほか、NetworkManager でネットワーク接続を管理しているような場合に便利な機能です。

4. ［OK］を押して閉じます。

18.2 ネットワーク内での NTP の手動設定

`chronyd` は `/etc/chrony.conf` にある設定ファイルを読み込みます。お使いのコンピュータの同期を維持するため、`chrony` に対してどのタイムサーバを使用するのかを指定します。タイムサーバの指定は完全修飾ホスト名のほか、IP アドレスでも指定することができます。たとえば下記のようになります:

```
0.suse.pool.ntp.org
1.suse.pool.ntp.org
2.suse.pool.ntp.org
3.suse.pool.ntp.org
```

下記のようにして pool (プール) を指定することもできます。プールは複数の IP アドレスに展開される仕組みです:

```
pool pool.ntp.org
```

ヒント: 同じネットワーク内のコンピュータ

同じネットワーク内にある複数のコンピュータの時刻を同期するにあたっては、それら全てを外部のサーバと同期させる構成はお勧めできません。その代わりに、いずれか 1 台のコンピュータを外部のタイムサーバと同期させ、残りのコンピュータをそのコンピュータに同期させることをお勧めします。この場合は、サーバ側の `/etc/chrony.conf` 内に `local` ディレクティブを指定して、信頼できるタイムサーバと区別するようにしてください:

```
local stratum 10
```

`chrony` を開始するには、下記のように実行します:

```
systemctl start chronyd.service
```

`chronyd` の準備が完了すると、時刻が安定して受信され、ローカルコンピュータの時計の調整を行うためのドリフトファイルが作成されるまで、しばらくの時間がかかります。ドリフトファイルは、コンピュータが起動されてからのハードウエア時計のズレの計算結果を表したファイルです。補正は直ちに開始されるため、これによってシステム時刻の安定性を高めるようになっています。

システムの起動時に `chrony` サービスを開始するには、下記のように実行します:

```
systemctl enable chronyd.service
```

警告: `yast-timesync.service` サービスとの競合について

`chronyd.service` に加えて、openSUSE Leap には `yast-timesync.service` という名前のサービスが含まれています。`yast-timesync.service` はタイマーで 5 分おきに動作するよう設定され、`chronyd` に `-q` オプションを付けて動作させたあと、終了するようになっています。ただし、`chronyd` は同時に 1 つまでしか起動できない仕様であることから、`chronyd` サービスを動作させている場合は、このサービスを有効化または起動してはなりません。

18.3 NTS の設定

Network Time Protocol (NTP) はネットワーク内の 1 つまたは複数のホストに対して、システム時刻を同期し、正確に維持するためのプロトコルです。本記事では、Network Time Security (NTS) を利用した NTP 機密保持機能の設定方法を説明しています。

NTP プロトコルにはセキュリティ機能が含まれていないため、タイムサーバとクライアントとの間でやり取りされる通信に認証や暗号化を追加できません。Network Time Security (NTS) はそのような NTP に対して、セキュリティ機能を提供します。`chrony` は NTS に対応していますので、時刻ソースからの情報を認証し、特定の種類のネットワーク攻撃から保護できるようになっています。

下記の手順では、より安全な時刻同期のため、タイムサーバとクライアントとの間で設定すべき事項について概要を説明しています。

手順 18.1: NTS タイムサーバの設定

1. NTS を利用して時刻の更新を行うようタイムサーバを設定しておくことをお勧めします。これにより、同期処理の最初から安全に時刻を取得できるようになります。まずは `/etc/chrony.conf` 内にある NTS 非対応の時刻ソースをコメントアウトし、少なくとも 1 つ以上の NTS 対応サーバを指定します。たとえば下記のようになります:

```
server time.cloudflare.com iburst nts
```

ヒント

`nts` オプションを指定すると、NTS に対応しているサーバであれば NTS を使用するようになります。対応していないサーバの場合は従来の NTP として通信します。

2. あとは `chronyd` サービスを再起動します。

```
> sudo systemctl restart chronyd.srvice
```

3. あとは設定済みの時刻ソースとの同期状況を確認します。

```
> chronyc sources -v
MS Name/IP address         Stratum Poll Reach LastRx Last sample
===============================================================================
^? time.cloudflare.com           3   6     1     2   -947ms[ -947ms] +/-   12ms
^? pyrrha.fi.muni.cz             2   6     1     1   -948ms[ -948ms] +/-   39ms
^* whitesoft-intex16.c.cbsn>     1   6     1     2   -948ms[ -948ms] +/- 5444us
^? mail.combatostrich.dev        2   6     1     1   -948ms[ -948ms] +/-   28ms
```

注記

`^*` で始まる行が、最適な時刻ソースとして判断されたサーバを表します。

あとは NTS モードで同期しているかどうかを確認します。

```
> chronyc -N authdata
Name/IP address             Mode KeyID Type KLen Last Atmp  NAK Cook CLen
=========================================================================
[...]
time.cloudflare.com          NTS     1   15  256    3    0    0    8   96
```

4. なお、サーバ側で `allow` オプションを指定していると、そのタイムサーバと同期可能なクライアントを指定することができます。たとえば下記のようになります:

```
allow 192.168.1.0/24
```

5. タイムサーバがファイアウォール内で動作している場合は、NTP と NTS の両方を許可するように設定してください。既定で NTP はポート 123, NTS はポート 4460 を使用します。

6. また、TLS 証明書と対応する機密鍵を取得し、それらを `/var/lib/chrony/` 内にコピーします。また、それらのファイルが `chrony` から読み取れる用に設定します。たとえば下記のようになります:

```
> sudo install -m 0440 -o chrony -g chrony nts.key /var/lib/chrony/
> sudo install -m 0440 -o chrony -g chrony nts.crt /var/lib/chrony/
```

ヒント

TLS 証明書に関する詳細情報については、こちらの記事 (https://documentation.suse.com/smart/security/html/tls-certificates/index.html) をお読みください。

7. また、`/etc/chrony.conf` ファイルを編集して、`ntsdumpdir /var/lib/chrony` オプションを有効化してください。これに加えて、TLS 鍵と証明書のパスを指定してください。

```
ntsdumpdir /var/lib/chrony
ntsserverkey /var/lib/chrony/nts.key
ntsservercert /var/lib/chrony/nts.crt
```

8. あとは `chronyd` サービスを再起動します。

```
> sudo systemctl restart chronyd.service
```

手順 18.2: NTS クライアントの設定

1. まずは既存の NTP タイムソースを無効化します。たとえば下記のように設定します:

```
#server 192.168.1.1 iburst
```

タイムソースの設定は `/etc/chrony.conf` 内に直接記述するか、もしくは `/etc/chrony.d/` ディレクトリ内にファイルを作成して記述します。

2. クライアント側では、サーバ側に設定されている TLS 証明書の発行元となるルート証明機関 (CA) を信頼するように設定する必要があります。証明機関 (CA) ストアの管理方法に関する詳細は、こちらの記事 (https://documentation.suse.com/smart/security/html/tls-certificates/index.html#tls-certificates-store) で説明しています。

3. あとはクライアント側の `chrony` 設定ファイルである `/etc/chrony.conf` に対して、手順 18.1「NTS タイムサーバの設定」で設定した NTS タイムサーバを指定します。具体的には下記のように設定します:

```
server サーバアドレス iburst nts
```

4. あとは `chronyd` サービスを再起動します。

```
> sudo systemctl restart chronyd.service
```

5. 再起動が終わったら下記を実行して、クライアント側で正しく時刻ソースが設定されているかどうか、および接続認証が働いているかどうかを確認します。

```
> sudo chronyc sources -v
> sudo chronyc -N authdata
```

6. NTS タイムサーバ側では、NTS 接続に対するクライアント統計情報を確認します。

```
> sudo chronyc -N clients -k
```

18.4 `chronyc` による動作中の chronyd の設定

`chronyd` の動作中に何らかの変更を行いたい場合は、`chronyc` を使用することができます。`chronyc` では、`chronyd` の操作に関する状態レポートを生成することもできます。

`chronyc` は対話モードと非対話モードの両方に対応しています。`chronyc` を対話モードで起動したい場合は、コマンドラインに `chronyc` とだけ入力してください。これによりプロンプトが表示され、コマンドの入力を待機するようになります。たとえば NTP の対向でオンラインのものとオフラインのものの数を知りたい場合は、下記のように実行します:

```
# chronyc
chronyc> activity
200 OK
4 sources online
2 sources offline
1 sources doing burst (return to online)
1 sources doing burst (return to offline)
0 sources with unknown address
```

`chronyc` のプロンプトを終了するには、`quit` もしくは `exit` と入力してください。

対話プロンプトが必要ない場合は、コマンドを直接指定して実行することもできます:

```
# chronyc activity
```

注記: 一時的な変更

`chronyc` を利用した変更は恒久的なものではありません。`chronyd` を再起動すると、元の設定に戻ってしまいます。設定を恒久化させたい場合は、`/etc/chrony.conf` を編集してください。

利用可能な `chronyc` のコマンド一覧について、詳しくはマニュアルページ (`man 1 chronyc`) をお読みください。

18.5 動作中の動的な時刻同期

システムの起動時にネットワークの接続が利用できないような環境では、`chronyd` は起動できるものの、設定ファイル内に書かれたタイムサーバの DNS 名が解決できないことになります。

`chronyd` では `server` , `pool` , `peer` の各ディレクティブで指定したタイムサーバ名を、成功するまで間隔を増やしながら解決を試みます。

`chronyd` の起動時にはタイムサーバに接続できないことがわかっている場合は、下記のようにして `offline` オプションを指定してください:

```
server サーバアドレス offline
```

上記のように設定すると、`chronyd` は下記のコマンドを送信するまで、サーバへの問い合わせを行わないようになります:

```
# chronyc online サーバアドレス
```

なお、`auto_offline` オプションを指定すると、`chronyd` はタイムサーバに対して 2 回リクエストを送信しても応答がない場合、そのタイムサーバがオフライン状態にあるものと判断するようになります。これにより、ネットワーク接続が切れている状態でも `offline` コマンドを実行する必要がなくなります。

18.6 ローカル参照時計の設定

ソフトウエアパッケージ `chrony` は、他のプログラム (たとえば `gpsd`) を利用して SHM や SOCK ドライバ経由でタイミングデータを取得することができます。`/etc/chrony.conf` 内に `refclock` ディレクティブを指定することで、ハードウエア参照時計を同期先として使用することができます。このディ

レクティブには、2 種類の必須パラメータがあります。1 つはドライバ名で、もう 1 つはドライバ固有のパラメータです。2 つのパラメータは 0 個以上の `refclock` オプションの後ろに続きます。`chronyd` では、下記のドライバに対応しています:

- PPS - カーネルの `Pulse Per Second` API 向けのドライバです。たとえば下記のように設定します:

```
refclock PPS /dev/pps0 lock NMEA refid GPS
```

- SHM - NTP 共有メモリドライバです。たとえば下記のように設定します:

```
refclock SHM 0 poll 3 refid GPS1
refclock SHM 1:perm=0644 refid GPS2
```

- SOCK - Unix ドメインソケットドライバです。たとえば下記のように設定します:

```
refclock SOCK /var/run/chrony.ttyS0.sock
```

- PHC - PTP ハードウエアクロックドライバです。たとえば下記のように設定します:

```
refclock PHC /dev/ptp0 poll 0 dpoll -2 offset -37
refclock PHC /dev/ptp1:nocrossts poll 3 pps
```

個別のドライバのオプションについて、詳しくは `man 8 chrony.conf` をお読みください。

19 ドメインネームシステム

改訂履歴

2024-05-13

DNS (ドメインネームシステム; Domain Name System) は、ドメイン名やホスト名を IP アドレスに解決するために必要なシステムです。この仕組みにより、たとえば IP アドレス 192.168.2.100 とホスト名 `jupiter` を結びつけることができるようになります。お使いの環境で独自のネームサーバを構築する前に、まずは 13.3項「名前解決」に書かれている DNS の一般的な情報をお読みください。また、下記の設定例では、既定のネームサーバである bind を使用して説明しています。

19.1 DNS で使用される用語

ゾーン

ドメイン名は複数のゾーンに分割することができます。たとえば `example.com` というドメイン名の場合、`com` ドメイン内の `example` というセクション (ゾーン) であることになります。

DNS サーバ

DNS サーバとは、ドメインに対して名前と IP アドレスの情報を保持するサーバを意味します。DNS サーバには、必要に応じて複数のゾーン (プライマリゾーン／セカンダリゾーン) が設定されるほか、ゾーンを持たないキャッシュ専用のセカンダリサーバもあります。

プライマリゾーン DNS サーバ

プライマリゾーンには、特定のドメイン内での全ホストについて、その最新情報が登録されています。

セカンダリゾーン DNS サーバ

セカンダリゾーンはプライマリゾーンのコピーを意味します。DNS サーバはプライマリサーバから、セカンダリゾーンとして設定されたゾーンの転送を行います。プライマリゾーンは、その情報が有効である限り (有効期限が切れない範囲) において、信頼できる情報源であるものとされます。プライマリサーバと通信ができなくなり、その情報の有効期限が切れると、DNS サーバはそのゾーンに対する応答を行わなくなります。

フォワーダ

フォワーダとは、お使いの DNS サーバがその問い合わせに応答できない場合に、その問い合わせの転送先となる DNS サーバを意味します。1 つの設定内で複数の設定ソースを利用したい場合は、`netconfig` を使用します (詳しくは `man 8 netconfig` をお読みください) 。

レコード

レコードとは名前や IP アドレスに関する情報を意味します。対応するレコードとその書式については、BIND のドキュメンテーションに説明があります。下記にいくつかの特殊なレコードを示します:

NS レコード

NS レコードとは、指定したドメイン (ゾーン) を受け持つネームサーバを示すためのレコードです。

MX レコード

MX (メール交換; Mail eXchange) レコードとは、インターネットを介して電子メールを交換する際、そのドメイン宛のメールを受信するメールサーバを示すためのレコードです。

SOA レコード

SOA (権威開始; Start Of Authority) レコードとは、ゾーンファイル内で最初に作成すべきレコードです。SOA レコードは、複数のコンピュータ間でデータを同期する際に使用します。

19.2 インストール

DNS サーバをインストールするには、YaST を起動して［ソフトウエア］›［ソフトウエア管理］を選択します。その後［表示］›［パターン］を選択して［DHCP および DNS サーバ］を選択します。あとは［了解］を押して必要なパッケージをインストールします。

上記以外にも、下記のコマンドラインを入力してもかまいません:

```
> sudo zypper in -t pattern dhcp_dns_server
```

19.3 YaST での設定

ローカルネットワーク内で DNS サーバを設定したい場合は、YaST の DNS モジュールをお使いください。初めてこのモジュールを起動した場合はウイザードが表示され、サーバの管理に関わるいくつかの決定を行うことになります。この初期設定を完了すると、基本的なサーバ設定が作成されます。ACL やログ、TSIG 鍵やその他のオプションなど、より高度な設定を行いたい場合は、熟練者モードをお使いください。

19.3.1 ウイザードによる設定

このウイザードは 3 つの手順 (ダイアログ) から構成されています。熟練者モードへの移行は、それが可能になった段階で表示されるようになっています。

1. 初めて本モジュールを起動した場合は、図19.1「DNS サーバのインストール: フォワーダの設定」に示されている［フォワーダの設定］ダイアログが表示されます。ここでは、［ローカル DNS の解決ポリシー］で下記のいずれかを選択することができます:

   - ［フォワーダの合成を無効にする］
   - ［自動的に合成する］
   - ［フォワーダの合成を有効にする］
   - ［独自設定］: これを選択すると、［カスタムポリシー］を指定することができるようになります。既定では (［自動的に合成する］が選択されるため) 、［カスタムポリシー］が `auto` に設定されますが、ここにはインターフェイス名のほか、`STATIC` や `STATIC_FALLBACK` のような特殊ポリシー名を指定することができます。

   ［ローカル DNS 解決器のフォワーダ］では、どのサービスを使用するのかを選択します。［システムのネームサーバを使用する］, ［このネームサーバ (bind)］, ［ローカルの dnsmasq サーバ］の中からいずれかを選択してください。
   これら全ての設定に関する詳細は、`man 8 netconfig` をお読みください。

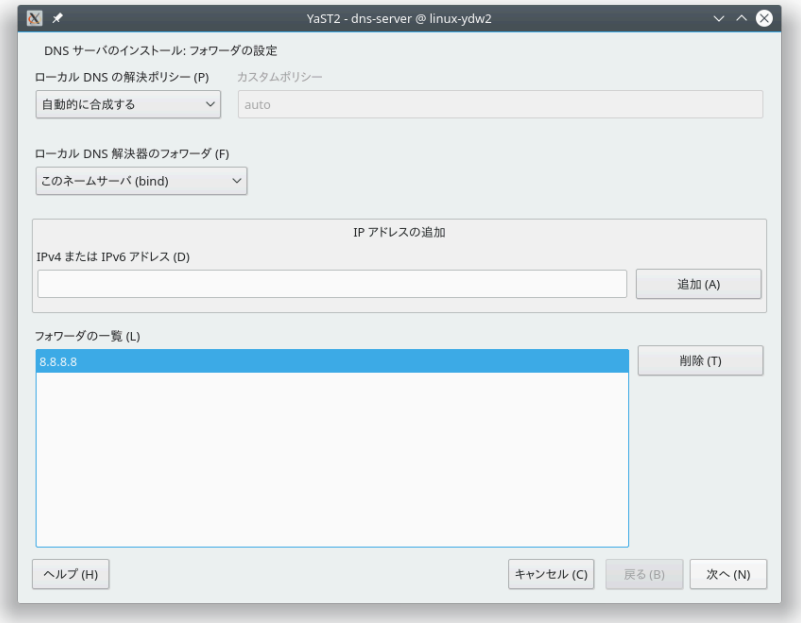


図 19.1: DNS サーバのインストール: フォワーダの設定

フォワーダとは、お使いの DNS サーバがその問い合わせに応答できない場合に、その問い合わせの転送先となる DNS サーバを意味します。フォワーダの IP アドレスを入力して、［追加］を押してください。

2. ［DNS ゾーン］ダイアログではゾーンファイル (詳しくは 19.6項「ゾーンファイル」をお読みください) の管理を行います。新しいゾーンを作成するには、［名前］欄にゾーンの名前を入力します。なお、逆引きゾーンを作成する場合は、ゾーンの名前が `.in-addr.arpa` で終わらなければなりません。また、［種類］(プライマリ／セカンダリ／転送) も選択する必要があります。詳しくは 図19.2「DNS サーバのインストール: DNS ゾーン」をお読みください。既存のゾーンを編集したい場合は、ゾーンを選択して［編集］を押します。ゾーンを削除したい場合は、ゾーンを選択して［削除］を押します。

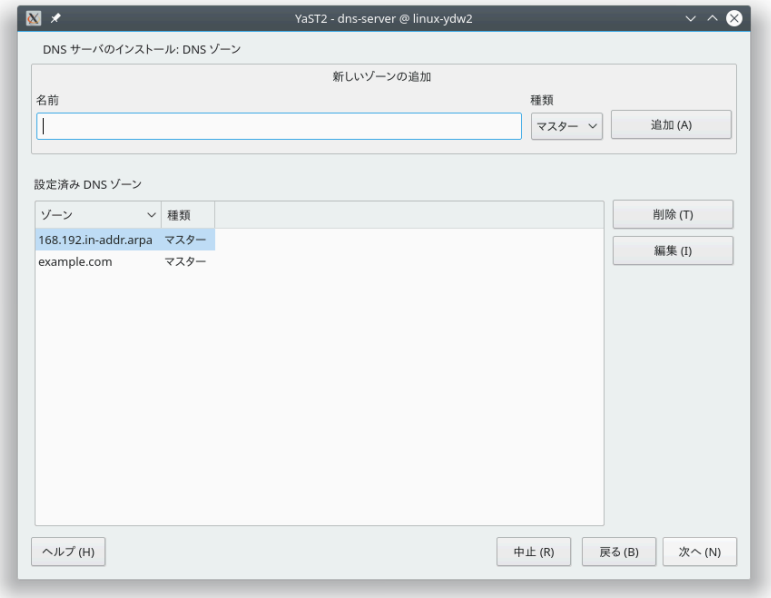


図 19.2: DNS サーバのインストール: DNS ゾーン

3. 最後のダイアログでは残りの設定を行います。[ファイアウオールでポートを開く] を選択すると、ファイアウオールで DNS のポートを開くことができます。また、システムの起動時に DNS サーバを開始するかどうかを設定することもできる ([オン] または [オフ]) ほか、LDAP サポートを有効化することもできます。詳しくは 図19.3「DNS サーバのインストール: ウイザードの完了」をご覧ください。

図 19.3: DNS サーバのインストール: ウイザードの完了

19.3.2 熟練者向け設定

モジュールを起動すると、YaST は複数の設定オプションを表すウインドウを表示します。これらの設定を行うことで、DNS サーバの基本的な機能を適用することができるようになります:

19.3.2.1 起動

[起動] では、DNS サーバをシステムの起動時に開始するかどうかを選択することができます。DNS サーバを今すぐに開始したい場合は、[今すぐ開始] を押してください。DNS サーバを停止するには、[今すぐ停止] を押してください。現在の設定を保存するには、[設定保存後に再読み込みする] を選択してください。ファイアウオールで DNS のポートを開くには、[ファイアウオールでポートを開く] を選択して、[ファイアウオールの詳細] で必要に応じて設定を変更してください。

[LDAP サポートを有効にする] を選択すると、ゾーンファイルが LDAP データベース内に保存されるようになります。LDAP データベースに書き込まれたゾーンデータを修正した場合は、DNS サーバの再起動や再読み込みでそれが反映されるようになります。

19.3.2.2 フォワーダ

ローカルの DNS サーバが特定の問い合わせに応答できない場合、その問い合わせを［フォワーダ］で設定したサーバに転送することができます。このフォワーダは、［フォワーダの一覧］内に手作業で追加して設定します。ダイアルアップ接続の場合など、固定では設定できない環境の場合は、［netconfig］が設定を取り扱います。netconfig について、詳しくは `man 8 netconfig` をお読みください。

19.3.2.3 基本オプション

このセクションでは基本的なサーバ設定を行います。［オプション］で設定したい項目を一覧から選んで、右側に設定したい値を入力してください。入力が終わったら［追加］を押すと、設定を追加することができます。

19.3.2.4 ログ

DNS サーバでのログ記録を設定するには、［ログ］を選択します。［ログの種類］では、DNS サーバからのログの保存先を指定します。［システムログ］を選択するとシステム全体のログファイルに、［ファイル］を選択すると指定したファイルに保存することができます。後者の場合はファイル名のほか、メガバイト単位で最大ファイルサイズと、残しておくべき過去のログファイル数 (バージョン) を指定することができます。

さらに詳しいオプションが［追加ログ］内に用意されています。［すべての DNS への問い合わせ］を選択すると、届いた全ての問い合わせをログに記録するようになりますので、ログファイルのサイズが大きくなることに注意してください。そのため、デバッグの目的がある場合を除いて、有効化することはお勧めできません。また、DHCP サーバと DNS サーバの間でのゾーン更新時にログを記録したい場合は、［ゾーン更新をログに記録］を選択してください。また、プライマリサーバからセカンダリサーバへのゾーン転送をログに記録したい場合は、［ゾーン転送をログに記録］を選択してください。詳しくは 図 19.4「DNS サーバ: ログ」をお読みください。

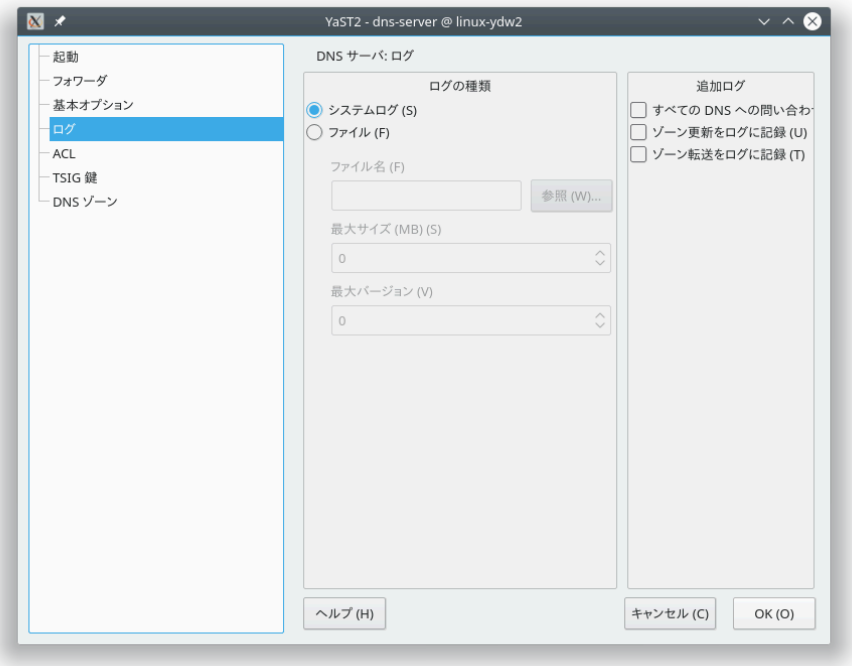


図 19.4: DNS サーバ: ログ

19.3.2.5 ACL

このダイアログでは ACL (アクセス制御リスト; Access Control List) を定義して、アクセス制限を設定することができます。［名前］欄に他と重複しない名前を設定し、［値］欄に IP アドレス (ネットマスクは設定してもいなくてもかまいません) を指定することで、ACL を定義することができます。値は下記のように記述します:

```
{ 192.168.1/24; }
```

設定ファイルの書式により、アドレスの末尾にはセミコロンを付けるほか、値を波括弧で括ってください。

19.3.2.6 TSIG 鍵

TSIG (トランザクション署名; Transaction SIGnatures) の主な目的は、DHCP サーバと DNS サーバとの間で機密を保持する通信を行うことにあります。詳しくは 19.8項「トランザクションの暗号化」で説明しています。

TSIG 鍵を生成するには、［鍵 ID］の欄に他と重複しない名前を入力し、［ファイル名］の欄に保存先のファイル名を指定したあと、［生成］を押します。

以前に作成した鍵を使用したい場合は、［鍵 ID］の欄には何も入力せず、［ファイル名］の欄で既存のファイルのファイル名を入力してください。入力が終わったら［追加］を押します。

19.3.2.7 DNS ゾーン (セカンダリゾーンの追加)

セカンダリゾーンを追加するには、［DNS ゾーン］を選択して、［種類］で［セカンダリ］を選択します。その後ゾーンの名前を入力して［追加］を押します。

［ゾーンエディタ］サブダイアログ内の［プライマリ DNS サーバの IP］では、セカンダリサーバが情報を取得する際のプライマリサーバを指定します。また、サーバへのアクセスを制限したい場合は、表示された ACL の一覧から必要なものを選択します。

19.3.2.8 DNS ゾーン (プライマリゾーンの追加)

プライマリゾーンを追加するには、［DNS ゾーン］を選択して［種類］で［プライマリ］を選択します。その後ゾーンの名前を入力して［追加］を押します。なお、プライマリゾーンを追加する場合は、逆引きゾーンも必要となります。たとえば `example.com` というゾーンを追加して、そこに `192.168.1.0/24` というサブネット内のホストを設定する場合、その IP アドレスを範囲内とする逆引きゾーンを追加する必要があります。この逆引きゾーンは、一般に `1.168.192.in-addr.arpa` という名前であるべきものです。

19.3.2.9 DNS ゾーン (プライマリゾーンの編集)

プライマリゾーンを編集するには、［DNS ゾーン］を選択して表の中からプライマリゾーンを選択し、［編集］を押します。ダイアログには下記のようなページが用意されています: ［基本］(最初に表示されるページ) , ［NS レコード］, ［MX レコード］, ［SOA］, ［レコード］

図19.5「DNS サーバ: ゾーンエディタ (基本)」に示されている基本設定ダイアログでは、動的な DNS 更新の可否とクライアントやセカンダリサーバに対するゾーン転送の設定を行うことができます。ゾーンを動的に更新できるようにするには、［動的な更新の許可］を選択したあと、対応する TSIG 鍵を選択する必要があります。鍵は更新処理の開始よりも前に定義しておかなければなりません。ゾーン転送を有効化するには、対応する ACL を選択します。ACL はこの時点までに設定しておかなければなりません。

［基本］ダイアログでは、ゾーンの転送可否を設定します。表示された ACL を選択して、ゾーンのダウンロードを許可したい相手を選択してください。

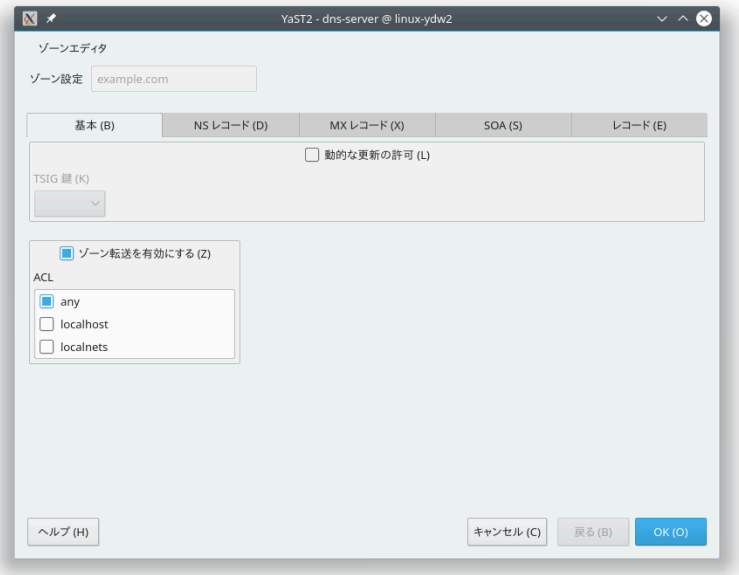


図 19.5: DNS サーバ: ゾーンエディタ (基本)

ゾーンエディタ (NS レコード)

［NS レコード］ダイアログでは、指定したゾーンでのネームサーバを設定することができます。なお、この一覧内には自分自身も含めておく必要があることに注意してください。レコードを追加するには、［追加するネームサーバ］の欄に名前を入力して［追加］を押します。詳しくは 図 19.6「DNS サーバ: ゾーンエディタ (NS レコード)」 をご覧ください。

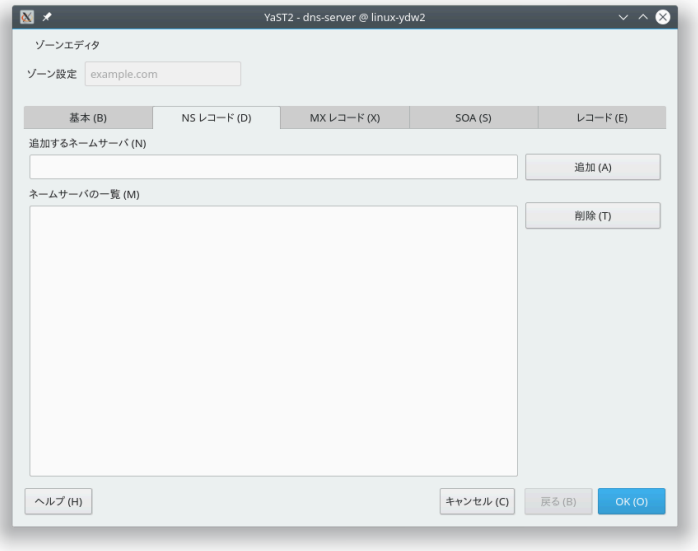


図 19.6: DNS サーバ: ゾーンエディタ (NS レコード)

ゾーンエディタ (MX レコード)

既存の一覧内に現在のゾーンに対するメールサーバを追加するには、まず対応するテキストボックス内にアドレスと優先度を入力します。それぞれ入力が終わったら［追加］を押します。詳しくは 図19.7「DNS サーバ: ゾーンエディタ (MX レコード)」をご覧ください。

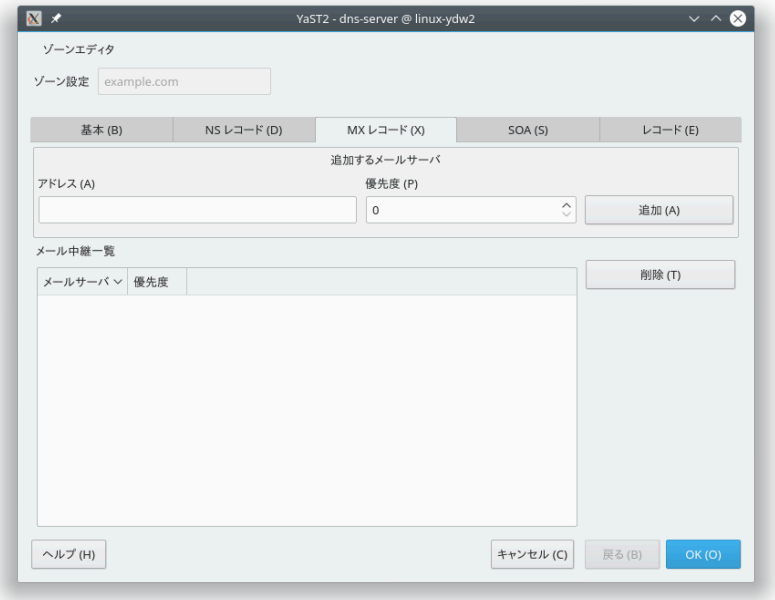


図 19.7: DNS サーバ: ゾーンエディタ (MX レコード)

ゾーンエディタ (SOA)

このページでは、SOA (権威開始; Start Of Authority) レコードを作成します。個別のオプションに関する説明については、例19.6「/var/lib/named/example.com.zone ファイル」をお読みください。ただし、LDAP で管理されている動的ゾーンの場合は、SOA レコードの変更には対応していません。

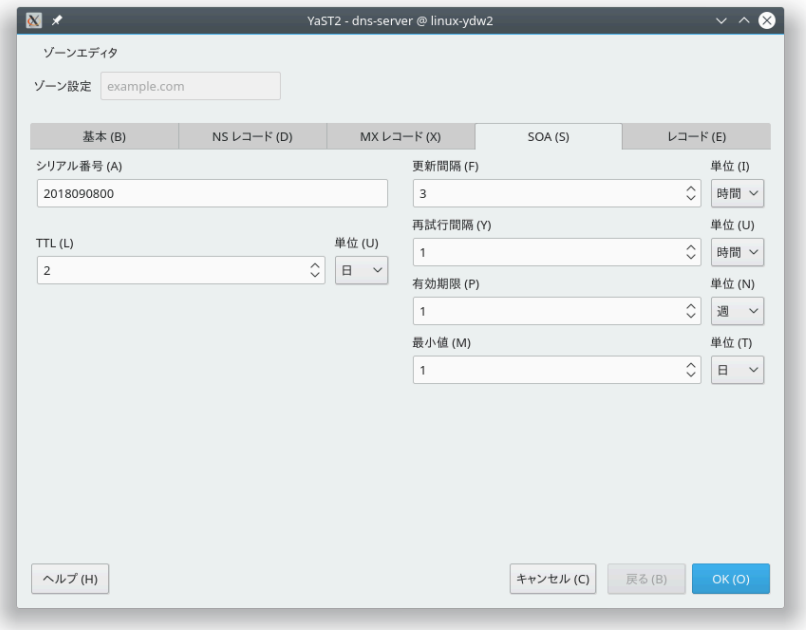


図 19.8: DNS サーバ: ゾーンエディタ (SOA)

ゾーンエディタ (レコード)

このダイアログでは、名前解決を管理します。[レコードキー] にはホスト名を入力して、種類を選択します。種類に [A] を選択した場合は、[レコードキー] がホスト名、[値] が IP (IPv4) アドレスになります。同様に [AAAA] の場合は、[レコードキー] がホスト名、[値] が IPv6 アドレスになります。[CNAME] の場合は [レコードキー] と [値] の両方がホスト名となり、一方から他方への別名定義を作成することができます。[NS] と [MX] については、それぞれ [NS レコード] と [MX レコード] のタブで提供されているものと同じです。なお、[PTR] は逆引きゾーンで使用するものです。これは `A` レコードの逆を意味するもので、たとえば下記のようになります:

```
hostname.example.com. IN A 192.168.0.1
1.0.168.192.in-addr.arpa IN PTR hostname.example.com.
```

19.3.2.9.1 逆引きゾーンの追加

逆引きゾーンを追加するには、下記の手順で行います:

1. [YaST] › [DNS サーバ] › [DNS ゾーン] を選択します。
2. まだ (逆引きではない) プライマリゾーンを作成していない場合は、プライマリゾーンを追加して [編集] を押します。

3. ［レコード］タブに移動し、［レコードキー］と［値］をそれぞれ入力して［追加］を押します。作業が終わったら［OK］を押します。YaST が NS サーバの設定についてエラーメッセージを表示した場合は、［NS レコード］のタブでサーバを追加してください。

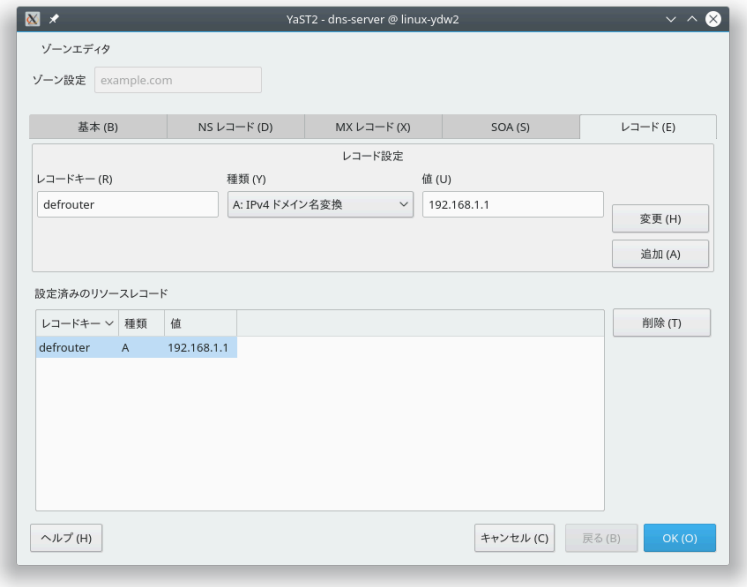


図 19.9: プライマリゾーンに対するレコードの追加

4. ［DNS ゾーン］ウインドウに戻ったら、逆引きのプライマリゾーンを追加します。

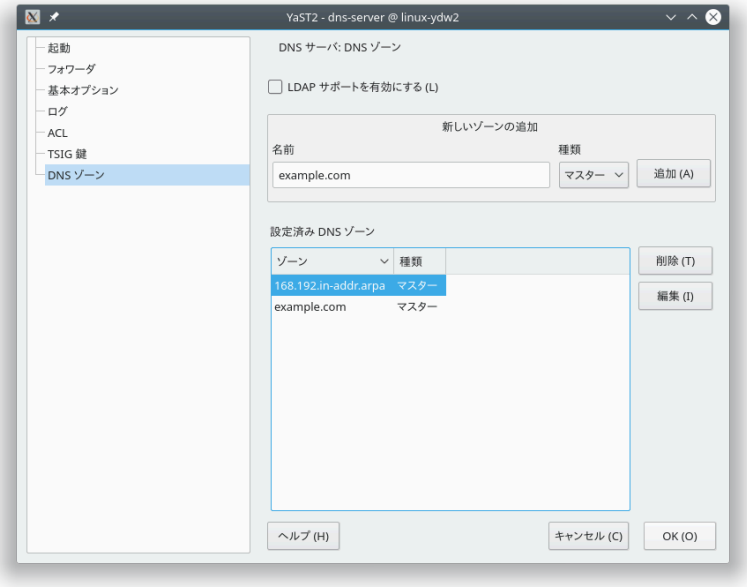


図 19.10: 逆引きゾーンの追加

5. 逆引きゾーンを選択して［編集］し、［レコード］タブに移動します。逆引きゾーンの場合、［PTR: 逆変換］というレコードの種類が表示されるはずです。それぞれ対応する［レコードキー］と［値］を入力して［追加］を押し、作業が終わったら［OK］を押して閉じます。

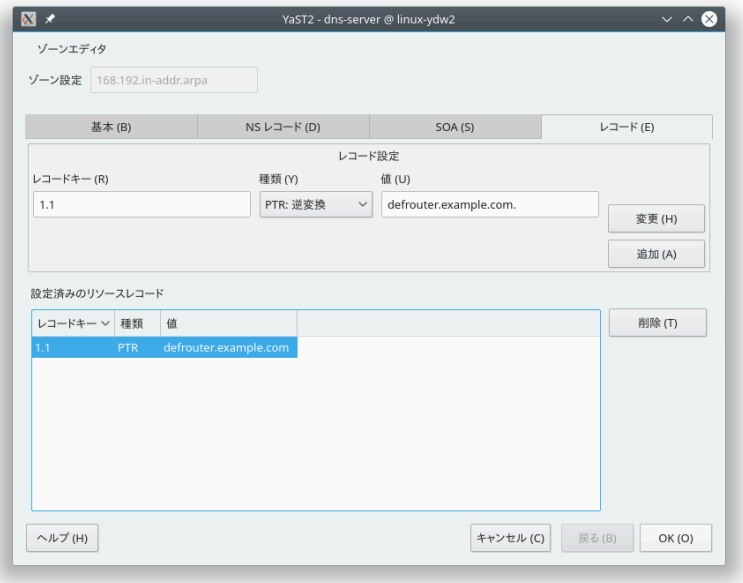


図 19.11: 逆引きレコードの追加

なお、必要であれば NS レコードも追加してください。

ヒント: 逆引きゾーンの編集

正引きのゾーンを追加したら、メインメニューに戻って逆引きゾーンを選択して、編集を行ってください。この中の［基本］タブ内には、［以下のものから自動的にレコードを生成］というチェックボックスがありますので、このチェックを入れてから対応する正引きゾーンを選択すると、自動的にレコードを生成することができるようになります。これを設定しておくことで、正引きゾーンで何らかの変更を行うと、それにあわせて逆引きゾーンが更新されるようになります。

19.4 BIND ネームサーバの起動

openSUSE® Leap システムでは、ネームサーバ BIND (Berkeley Internet Name Domain) を事前に設定した状態でインストールすることができます。これにより、インストールを行うだけで起動できるようになっています。インターネットに接続されている環境であれば、`/var/run/netconfig/resolv.conf` 内に設定するネームサーバのアドレスに、`localhost` を表す `127.0.0.1` を指定することで、プロバイダから提供される DNS 情報を利用することなく、名前が解決できるようになります。こ

れは、BIND にルートネームサーバの情報が含まれていることによるもので、それなりに遅い処理にはなりますが、いちおう解決することができます。通常はプロバイダのネームサーバを `/etc/named.conf` 内の `forwarders` に指定することで、効率的で素早く、かつ安全な解決を行います。このような構成で構築したネームサーバは、キャッシュのみの ネームサーバと呼ばれます。ここから独自のゾーンを設定すれば、通常の DNS サーバと同じ構成になります。簡単な設定例については、`/usr/share/doc/packages/bind/config` をお読みください。

ヒント: ネームサーバ情報の自動調整について

インターネットやネットワークの接続方法によっては、現在の状況に合わせてネームサーバの情報を修正する必要がある場合があります。これを行いたい場合は、`/etc/sysconfig/network/config` 内の `NETCONFIG_DNS_POLICY` の値を、`auto` に設定してください。

ただし、公式にドメインを取得した場合を除いて、一般的なドメインに対するネームサーバを設定すべきではありません。また、ドメインを取得している場合でも、プロバイダ側に管理を委託しているような場合も、独自にそのドメインに対するサーバを構築することはお勧めできません。なぜなら、BIND でそれらのドメインを設定してしまうと、ドメイン宛の問い合わせを転送しなくなってしまうためです。逆に、たとえばプロバイダ内に設置している Web サーバが存在するような場合、独自に構築したドメインにはアクセスできません。

ネームサーバを開始するには、`root` で `systemctl start named` のコマンドを実行します。実行した後は、`systemctl status named` コマンドで named (ネームサーバのプロセス名) の状態を確認することができます。ローカルでネームサーバの動作を確認するには、`host` や `dig` のコマンドを使用します。たとえば `localhost` のホストを問い合わせると、`127.0.0.1` という応答が返るはずです。応答がない場合や、正しい応答になっていない場合は、`/var/run/netconfig/resolv.conf` 内のネームサーバが正しく設定されていないか、もしくはファイルそのものが存在していないものと思われます。たとえば `host 127.0.0.1` のように実行してもエラーが返される場合は、何らかの問題があるものと考えられます。エラーメッセージを表示するには、`systemctl status named` を実行して、まずはサーバそのものが動作しているかどうかを確認してください。ネームサーバが動作していない場合や、正しくない応答を返しているような場合は、`journalctl -e` の出力結果をお読みください。

プロバイダのネームサーバ (もしくは既にネットワーク内で動作しているネームサーバ) をフォワーダとして使用するには、IP アドレスまたはホスト名を `forwarders` 内の `options` セクションに設定してください。なお、例19.1「named.conf 内でのフォワーダオプション」で設定しているアドレスは、あくまでも例として示しているものです。必要に応じて修正してください。

例 19.1: NAMED.CONF 内でのフォワーダオプション

```
options {
        directory "/var/lib/named";
```

```
        forwarders { 10.11.12.13; 10.11.12.14; };
        listen-on { 127.0.0.1; 192.168.1.116; };
        allow-query { 127/8; 192.168/16 };
        notify no;
};
```

`options` の項目は `localhost` や `0.0.127.in-addr.arpa` の後に続く項目です。また、「.」内の `type hint` は、必ず設定しておくべき項目です。これらについては変更することなく、そのまま残しておいてください。また、各項目の末尾には「;」を付けるとともに、必要な箇所に波括弧を入力してください。`/etc/named.conf` やゾーンファイルを編集した後は、BIND に対して再読み込みを指示するため、`systemctl reload named` を実行してください。再読み込みは、ネームサーバを一旦停止して起動し直すことでも対応することができます (`systemctl restart named`) 。また、サーバを停止したい場合は、`systemctl stop named` を実行してください。

19.5 /etc/named.conf 設定ファイル

BIND ネームサーバ自身に対する全ての設定は、`/etc/named.conf` 内に保存されます。自身が処理すべきドメインのゾーンデータ (ホスト名や IP アドレスなどの情報) については、`/var/lib/named` ディレクトリ内の個別のファイル内に保存されます。これらの詳細は後述します。

大きく分けると、`/etc/named.conf` には 2 種類の領域があります。1 つ目の領域は一般的な設定を記述する `options` セクション、もう 1 つは各ドメインに対する `zone` 項目です。`logging` セクションと `acl` (アクセス制御リスト) セクションは任意で設定します。コメント行は `#` 記号もしくは `//` を行頭に入力します。最小限の設定の `/etc/named.conf` は、例19.2「基本的な /etc/named.conf」のようになります。

例 19.2: 基本的な /ETC/NAMED.CONF

```
options {
        directory "/var/lib/named";
        forwarders { 10.0.0.1; };
        notify no;
};

zone "localhost" in {
       type master;
       file "localhost.zone";
};

zone "0.0.127.in-addr.arpa" in {
        type master;
        file "127.0.0.zone";
```

```
};

zone "." in {
        type hint;
        file "root.hint";
};
```

19.5.1 主要な設定オプション

directory " ディレクトリ ";

BIND がゾーンデータを含むファイルを検索するディレクトリを指定します。通常は `/var/lib/named` を指定します。

forwarders { IP_アドレス ; };

ネームサーバが DNS の問い合わせを直接解決できない場合、その問い合わせを転送する先のネームサーバ (多くはプロバイダのネームサーバ) を指定します。IP_アドレス はたとえば、`192.168.1.116` のように指定します。

forward first;

ルートネームサーバを利用して解決しようとする前に、フォワーダを利用して DNS の問い合わせを解決するようにします。`forward first` ではなく `forward only` を指定すると、ルートネームサーバを使用せずにフォワーダのみを使用するようになります。これは特にファイアウォールを利用するような場合に便利です。

listen-on port 53 { 127.0.0.1; IP_アドレス ; };

BIND に対してネットワークインターフェイスを指定するほか、クライアントからの問い合わせを受け付けるポートを指定します。`53` は既定のポートであるため、`port 53` を明示的に指定する必要はありません。また、`127.0.0.1` を指定して、ローカルホストからのアクセスについては必ず受け付けるようにしてください。この設定を全く行わない場合は、全てのインターフェイスを利用して問い合わせを受け付けるようになります。

listen-on-v6 port 53 {any; };

BIND に対して IPv6 のクライアントからの問い合わせを受け付けるように指定しています。`any` (任意のアドレス) 以外には `none` (全てのアドレスを拒否) を指定することもできます。IPv6 については、ワイルドカードアドレスを指定することもできます。

query-source address * port 53;

この項目は、ファイアウォールが外部宛の DNS リクエストをブロックしているような場合に必要となります。この設定により、BIND はポート 53 からリクエストを送信するようになり、1024 以上のポートを使用しなくなります。

query-source-v6 address * port 53;
: BIND に対して、IPv6 の問い合わせで使用するポートを指定しています。

allow-query { 127.0.0.1; NET ; };
: どのクライアントから DNS のリクエストを受け付けるのかを指定します。NET には 192.168.2.0/24 のように、アドレス情報を指定してください。/24 はネットマスクの省略表記で、この例の場合は 255.255.255.0 を意味します。

allow-transfer ! *;;
: どのホストからのゾーン転送要求を受け入れるのかを指定します。! * の例では、全てのホストからの要求を拒否します。この項目を指定しない場合、ゾーン転送は全てのホストから実施できるようになります。

statistics-interval 0;
: 上記の項目を設定しない場合、BIND はシステムのジャーナルに対して、1 時間おきに統計情報を出力するようになります。0 を指定すると、この統計情報を省略することができます。0 以外の値を指定する場合は、分単位で指定します。

cleaning-interval 720;
: このオプションは、BIND がキャッシュを清掃する時間間隔を指定します。この清掃が行われると、システムのジャーナルに対しても出力が行われます。時間は分単位で指定します。既定値は 60 分です。

interface-interval 0;
: BIND では新しいネットワークインターフェイスや存在しなくなったインターフェイスを、定期的に検索します。この値を 0 にすると、BIND は新しいインターフェイスを使用しなくなり、起動時に検出されたインターフェイスのみを使用するようになります。それ以外の場合は分単位で指定してください。既定値は 60 分です。

notify no;
: no を指定すると、ゾーンデータに修正が発生したような場合や、ネームサーバが再起動されたような場合に、他のネームサーバに通知を送信しなくなります。

利用可能なオプションの一覧については、man 5 named.conf のマニュアルページをお読みください。

19.5.2 ログ

BIND では、ログ記録を行う種類と方法、場所をそれぞれ設定することができます。通常は既定値のままでかまいません。なお、例19.3「ログ記録を無効化するための設定」には一切のログ記録を無効化するための最も簡単な設定が示されています。

例 19.3: ログ記録を無効化するための設定

```
logging {
        category default { null; };
};
```

19.5.3 ゾーンの項目

例 19.4: EXAMPLE.COM に対するゾーン項目

```
zone "example.com" in {
      type master;
      file "example.com.zone";
      notify no;
};
```

例19.4「example.com に対するゾーン項目」に示されているとおり、まず `zone` の後ろには、管理対象のドメイン名を指定 (`example.com`) し、それに続いて `in` と波括弧を記述します。セカンダリゾーン を定義するには、`type` を `secondary` にして `primary` でプライマリサーバを指定してください (詳しくは 例19.5「example.net に対するゾーン項目」をご覧ください) 。

例 19.5: EXAMPLE.NET に対するゾーン項目

```
zone "example.net" in {
      type secondary;
      file "secondary/example.net.zone";

      masters { 10.0.0.1; };
};
```

それぞれの項目の意味は下記の通りです:

type primary;
: `primary` を指定することで、BIND は指定したゾーンをローカルのネームサーバ内で処理するようになります。なお、ゾーンファイルは正しい形式で作成している必要があります。

type secondary;
: このゾーンは、あらかじめ他のネームサーバから情報を転送して提供するものであることを示しています。この宣言を行った場合、`primary_servers` もあわせて指定しなければなりません。

type hint;
: ゾーン `.` に対して設定されている `hint` とは、ルートネームサーバを設定する際に使用するものです。このゾーンの設定は、変更したりせずそのままにしておいてください。

file `example.com.zone` および file「secondary/example.net.zone」;

この項目は、指定したドメインに対するゾーンデータが、どこに配置されているのかを示すものです。セカンダリゾーンの場合は、ゾーンデータをプライマリサーバから取得するため、file で指定したファイルが存在している必要はありません。ただし、セカンダリゾーンの場合は `secondary` ディレクトリ内に配置して区別するようにしてください。

primary_servers { `SERVER_IP_ADDRESS` ; };

この項目はセカンダリゾーンの場合にのみ指定すべき項目です。ネームサーバのゾーンデータを、どこから取得するのかを指定します。

allow-update {! *; };

この項目は外部からの書き込みアクセスを制御するもので、どのクライアントから DNS の項目を修正できるのかを指定します。セキュリティ上の理由から、必要な場合を除いて許可すべきではないものです。この項目を指定しない場合、全ての修正が拒否されます。なお、上記の `! *` のように指定しても同じ意味になり、全てのクライアントからの修正を拒否します。

19.6 ゾーンファイル

ゾーンファイルに対しては、2 種類のものが必要となります。1 つはホスト名に IP アドレスを割り当てるもので、もう 1 つはその逆に IP アドレスにホスト名を割り当てるものです。

ヒント: ゾーンファイル内でのドット (ピリオド) の使用について

ゾーンファイル内では、`"."` は特別な意味を持ちます。たとえばホスト名の末尾がドットで終わっていない場合、ゾーンのドメイン名が後ろに付いているものと見なされます。完全修飾ドメイン名の場合は、必ずドット (`.`) で終わらなければなりません。ドットで終わらないと、さらにドメイン名が付与されてしまいます。また、"." を誤った位置に配置したり、記述すべき箇所に記述しなかったりすると、ネームサーバの設定エラーを引き起こします。

まずは `example.com.zone` というゾーンファイルについて説明を行います。このゾーンファイルは `example.com` というドメインに対するゾーンファイルです。例19.6「/var/lib/named/example.com.zone ファイル」に例を示しています。

例 19.6: /VAR/LIB/NAMED/EXAMPLE.COM.ZONE ファイル

```
$TTL 2D ❶
example.com. IN SOA      dns  root.example.com. ( ❷
             2003072441  ; シリアル番号 ❸
             1D          ; 更新間隔 ❹
             2H          ; 再試行間隔 ❺
```

```
                1W          ; 有効期限 ❻
                2D )        ; ネガティブキャッシュ ❼

                IN NS       dns ❽
                IN MX       10 mail dns ❾
gate            IN A        192.168.5.1 ❿
                IN A        10.0.0.1
dns             IN A        192.168.1.116
mail            IN A        192.168.3.108
jupiter         IN A        192.168.2.100
venus           IN A        192.168.2.101
saturn          IN A        192.168.2.102
mercury         IN A        192.168.2.103
ntp             IN CNAME    dns ⓫
dns6            IN A6  0    2002:c0a8:174::
```

❶ `$TTL` は、このファイル内の全ての項目に対して適用される、既定の有効期間 (Time-to-Live) を設定します。この例では、各項目は 2 日間有効となっています (`2 D`) 。

❷ この行が SOA (権威開始; Start Of Authority) 制御レコードの開始位置です:

- まずは対象のドメイン (`example.com`) を最初に指定します。ここではドメイン名を再度追加されてしまうことのないよう、`"."` で終わらせる必要があります。また、1 文字で `@` を指定することもできますが、この場合は `/etc/named.conf` から対応する項目を抽出する意味になります。
- `IN SOA` の後ろには、このゾーンを担当するプライマリネームサーバの名前を指定します。この項目は `"."` で終わっていませんので、`dns` は `dns.example.com` に展開されます。
- ネームサーバの管理を行っている管理者の電子メールアドレスを指定します。ただし、`@` はゾーンファイル内では特別な意味を持ちますので、代わりに `"."` を使用します。たとえば `root@example.com` の場合は、`root.example.com.` となります。また、ドメイン名を再度追加されてしまうことのないよう、末尾には `"."` を付けなければなりません。
- その後の `(` から `)` までが SOA レコードとなります。

❸ **シリアル番号** は 10 桁の番号を指定しています。この数値は、ゾーンファイルが変更されるたびに変えなければなりません。この数値はセカンダリネームサーバに対して変更を通知するためのもので、一般的には YYYYMMDDNN の書式で指定します (YYYY = 年, MM = 月, DD = 日, NN = 順序番号) 。なお、NN は 1 日の間に複数回の変更を行った場合に使用します。

❹ **更新間隔** は、セカンダリネームサーバがゾーンの **シリアル番号** を確認する時間間隔を指定します。この例では 1 日になっています。

❺ **再試行間隔** はセカンダリネームサーバで何らかのエラーが発生した場合に、プライマリネームサーバにアクセスする時間間隔を指定します。この例では 2 時間になっています。

6 **有効期限** は、セカンダリサーバからプライマリサーバにアクセスすることができなくなった場合に、キャッシュ済みのデータを廃棄するまでの時間を指定します。この例では 1 週間になっています。

7 SOA レコードの最後の項目は **ネガティブキャッシュ TTL** で、他のコンピュータからの問い合わせで、解決することのできなかった DNS の問い合わせをキャッシュしておく時間を指定します。

8 `IN NS` の行は、このドメインに対するネームサーバを指定するための項目です。`dns` は `"."` で終わっていないため、`dns.example.com` に展開されます。また、この行は複数のサーバ (プライマリ、セカンダリなど) を指定する目的で、複数指定することもできます。この場合は `IN NS` の行を複数記述してください。また、`/etc/named.conf` で `notify` に `no` を指定していない場合、ゾーンの更新が発生すると、ここに記述している全てのネームサーバに対して通知が行われます。

9 MX レコードは、このドメイン (`example.com`) に対するメールを受け取って処理したり、転送したりするためのメールサーバを指定します。この例では `mail.example.com` がメールサーバに指定されています。ホスト名の前に設定されている数値は優先順位を表す値で、複数の MX レコードが存在した場合、その中で最も小さい優先順位の値のメールサーバを優先して使用します。もしもそのサーバ宛のメール配送が失敗した場合は、次に小さい優先順位のメールサーバを使用します。

10 この行以降が実際のアドレスレコードで、1 つ以上の IP アドレスをホスト名に割り当てています。それぞれのホスト名には `"."` が含まれていないため、それぞれに対して `example.com` が設定されます。なお、`gate` には 2 つの IP アドレスが割り当てられていますが、これは 2 枚のネットワークカードが接続されていることによるものです。ホストに対して割り当てるアドレスが従来のもの (IPv4) であった場合、レコードは `A` レコードになります。IPv6 アドレスを指定する場合は、`AAAA` レコードになります。

注記: IPv6 の書式について

IPv6 アドレスを指定する場合は、IPv4 とは異なる書式で指定します。IPv6 アドレスは IPv4 アドレスよりも長いものの、中間部分にゼロが入ることが多いため、その部分を省略して表記するためです。IPv6 アドレスのうち、「0」が連続する部分については、コロンを 2 つ付けて省略します。

```
pluto     AAAA 2345:00C1:CA11::1234:5678:9ABC:DEF0
pluto     AAAA 2345:00D2:DA11::1234:5678:9ABC:DEF0
```

11 `ntp` を `dns` の別名として使用する設定です。(`CNAME` は canonical name (正式名称) の略です) 。

疑似ドメインである `in-addr.arpa` は、IP アドレスからホスト名への逆変換を行う際に使用するドメインです。このドメインの前にアドレスを逆順で記述してドメイン名とします。たとえば `192.168` のアドレスに対する逆引きドメインは `168.192.in-addr.arpa` になります。詳しくは 例19.7「逆引き参照」をご覧ください。

例 19.7: 逆引き参照

```
$TTL 2D ❶
168.192.in-addr.arpa.   IN SOA dns.example.com. root.example.com. ( ❷
                        2003072441      ; シリアル番号
                        1D              ; 更新間隔
                        2H              ; 再試行間隔
                        1W              ; 有効期限
                        2D )            ; ネガティブキャッシュ TTL

                        IN NS           dns.example.com. ❸

1.5                     IN PTR          gate.example.com. ❹
100.3                   IN PTR          www.example.com.
253.2                   IN PTR          cups.example.com.
```

❶ $TTL は、このファイル内の全ての項目に対して適用される、既定の有効期間 (Time-to-Live) を設定します。

❷ この行は `192.168` のネットワークに対する逆引き参照を定義しています。そのため、ゾーン名 (ドメイン名) は `168.192.in-addr.arpa` になります。そのため、逆引きゾーンでホスト名を記述する際は、必ず完全修飾ドメイン名でなければならず、かつ `"."` で終わらなければなりません。残りの項目については、上記の `example.com` に対する例と同じです。
このレコード内の各項目に対する詳細は、例19.6「/var/lib/named/example.com.zone ファイル」をお読みください。

❸ この行では、このゾーンに対するネームサーバを指定しています。前述の通り、ホスト名は完全修飾ドメイン名で記述しなければならず、かつ `"."` で終わらなければなりません。

❹ この行以降は IP アドレスと対応するホスト名を記述しています。IP アドレスについては末尾だけを逆順で記述し、`"."` を最後に付けていません。これにより、自動的にゾーン名が末尾に追加されることになり、`.in-addr.arpa` で終わる名前になるようになっています。

なお、BIND のバージョンが異なっていても、ゾーンの転送は問題なく実施することができます。

19.7 ゾーンデータの動的な更新

動的な更新 とは、プライマリサーバ内のゾーンファイルの項目を、動的に追加したり変更したり削除したりする機能を意味します。この仕組みは RFC 2136 に規定されていて、各ゾーンの項目内に `allow-update` や `update-policy` のルールを指定することで、有効化することができます。ただし、動的に更新するゾーンについては、手作業でゾーンファイルを編集すべきではありません。

DNS サーバに対して更新要求を送信するには、`nsupdate` コマンドを使用します。このコマンドの正確な書式については、nsupdate のマニュアルページ (`man 8 nsupdate` をご覧ください。なお、セキュリティ上の理由から、19.8項「トランザクションの暗号化」で説明している TSIG 鍵の使用をお勧めします。

19.8 トランザクションの暗号化

トランザクションの暗号化は、共有機密鍵をベースにしたトランザクション署名 (TSIG) を利用して行います (そのため、TSIG 鍵とも呼ばれます) 。本章では、このような鍵を生成し、使用する方法について説明しています。

トランザクションの暗号化は、サーバ間での通信のほか、ゾーンデータの動的な更新を行う場合に必要となります。鍵に依存したアクセス制御を設定することで、IP アドレスだけに依存するよりもずっと機密を保持できるようになります。

TSIG 鍵を生成するには、下記のコマンドを実行します (詳しくは `man tsig-keygen` をお読みください):

```
> sudo tsig-keygen -a hmac-md5 host1-host2 > host1-host2.key
```

上記を実行すると `host1-host2.key` というファイルが作成され、下記のような内容が書き込まれます:

```
key "host1-host2" {                         |
    algorithm hmac-md5;
    secret "oHpBLgtcZso6wxnRTWdJMA==";
};
```

このファイルは通常、機密の保持される方法 (例: scp) で対向のホストに転送します。また、`host1` と `host2` との間で機密が保持できるようにするため、鍵は両方のサーバの `/etc/named.conf` 内に含めなければなりません。

```
key host1-host2 {
 algorithm hmac-md5;
 secret "ejIkuCyyGJwwuN3xAteKgg==";
```

```
};
```

警告: /etc/named.conf のアクセス権について

/etc/named.conf に対するアクセス権 (パーミッション) が正しく制限されていることを確認してください。このファイルの既定のアクセス権は 0640 で、所有者は root 、グループは named になっているはずです。また、/etc/named.conf から別のファイルを参照するように指定することで、異なるアクセス権のファイルを取り込むこともできます。別のファイルを参照するには、下記のように指定します:

```
include  "ファイル名"
```

ファイル名 の欄には、絶対パスで参照すべきファイルを指定します。

サーバ host1 に対して、host2 (この例では、アドレス 10.1.2.3) 向けの鍵を使用するようにするには、サーバ側の /etc/named.conf 内に下記のルールを含めなければなりません:

```
server 10.1.2.3 {
  keys { host1-host2. ;};
};
```

host2 側の設定についても、同様の項目を含めなければなりません。

あとは TSIG 鍵を、IP アドレスやアドレスの範囲を指定している任意の ACL (アクセス制御リスト (Access Control List) 、ファイルのアクセス権 (パーミッション) と混同しないでください) に設定して、許可を付与します。たとえば下記のようになります:

```
allow-update { key host1-host2. ;};
```

さらに詳しい説明については、BIND Administrator Reference Manual 内の update-policy をお読みください。

19.9 DNS セキュリティ

DNSSEC や DNS セキュリティは、RFC 2535 に規定されています。DNSSEC 向けのツールは、BIND マニュアルに説明があります。

機密を保持すべきゾーンに対しては、1 つまたは複数のゾーン鍵を結びつけて使用します。これらはホスト鍵を作成する際に使用した dnssec-keygen で生成することができます。現時点では、DSA 暗号化アルゴリズムを使用してこれらの鍵を生成します。対応するゾーン内の $INCLUDE ルールで、対応するゾーン内に公開鍵を結びつけます。

`dnssec-signzone` コマンドを使用すると、生成鍵のセット (`keyset-` ファイル) を作成することができます。これらを安全な手段で親ゾーンに渡して、署名をしてもらってください。署名の結果は `/etc/named.conf` 内の各ゾーンで取り込むように設定します。

19.10 さらなる情報

より詳しい情報については、`bind-doc` パッケージ内の `/usr/share/doc/packages/bind/arm` ディレクトリに含まれる、BIND Administrator Reference Manual (BIND 管理者向けリファレンスマニュアル) をお読みください。また、マニュアルや BIND のマニュアルページから参照されている、各種の RFC についてもお読みになることをお勧めします。また、openSUSE Leap での最新情報については、`/usr/share/doc/packages/bind/README.SUSE` をお読みください。

20 DHCP

改訂履歴

2025-02-28

Dynamic Host Configuration Protocol (DHCP) はそれぞれのコンピュータに対して個別の設定を行うことなく、ネットワークの設定を集中管理するための仕組みです。DHCP を使用するように設定したコンピュータは、自分自身のアドレスを固定で持つことはなく、サーバからの指示に従って自動的に設定することができます。また、クライアント側で NetworkManager を使用している場合は、クライアント側の設定を行う必要はありません。この仕組みは、特に 1 つのインターフェイスで複数のネットワークに接続するような場合に有用です。なお、DHCP サーバを動作させているマシンでは、NetworkManager を使用してはなりません。

DHCP サーバを使用するもう 1 つの用途としては、ネットワークカードのハードウエアアドレス (通常は固定のアドレスが割り当てられています) を利用して各クライアントを識別し、常に同じアドレスを割り当てたり、動的にアドレスを割り当てたりすることがあげられます。動的にアドレスを割り当てる場合でも、DHCP サーバ側では貸与期限を越えて同じアドレスを割り当てようとします。この方式は、クライアント数よりもアドレス数のほうが多い場合にのみ成立します。

DHCP はシステム管理者にとっても利点があります。アドレスやネットワークの設定変更は面倒な作業になりがちですが、サーバの設定ファイルだけで集中管理すれば、クライアント側の設定を一斉に変えることができますので、クライアントの台数が多くなればなるほどメリットとなります。また、新しいマシンのネットワークへの接続に際しても、プールから IP アドレスを取るだけの簡単な話になってしまいます。DHCP サーバ側で適切な設定を提供することで、様々なネットワークに接続するラップトップに対しても、非常に便利な機能となります。

本章では、DHCP サーバとクライアントは同じサブネット (`192.168.2.0/24`) 内で動作するものとし、ルータが `192.168.2.1` にあるものとします。また、DHCP サーバの IP アドレスは `192.168.2.254` で、それぞれ `192.168.2.10` から `192.168.2.20` までと、`192.168.2.100` から `192.168.2.200` までのアドレス範囲を提供するものとします。

DHCP サーバは IP アドレスとネットマスクの情報を提供するだけではありません。ホスト名やドメイン名、デフォルトゲートウエイやネームサーバのアドレスなどをクライアントに提供して、設定させることができます。また、これに加えて、DHCP サーバではクライアントが現在時刻を問い合わせるためのタイムサーバや、印刷サーバなどの情報を集中管理して提供することもできます。

20.1 YaST を利用した DHCP サーバの設定

DHCP サーバをインストールするには、YaST を起動して［ソフトウエア］›［ソフトウエア管理］を選択します。その後［表示］›［パターン］を選択して［DHCP および DNS サーバ］を選択します。あとは［了解］を押して必要なパッケージをインストールします。

重要: LDAP サポートについて

YaST DHCP モジュールでは、サーバの設定をローカル (DHCP サーバを動作させるマシン自身) に保存することができるほか、LDAP サーバで設定データを管理させることもできます。LDAP を使用するには、DHCP サーバを設定する前に LDAP の環境を設定してください。

LDAP について、詳しくは『セキュリティ強化ガイド』、第5章「389 Directory Server を利用した LDAP サービス」をお読みください。

YaST の DHCP モジュール（ `yast2-dhcp-server` ）では、ローカルネットワークに対する DHCP サーバの設定を行うことができます。このモジュールは、ウイザードモードのほか、熟練者向けの設定モードで動作させることができます。

20.1.1 初期設定 (ウイザード)

このモジュールを初めて起動した場合は、ウイザードが表示されます。ここでは、サーバの管理に関わるいくつかの基本的な決定を行います。この初期設定を完了することで、サーバの基本設定を作成することができます。また、熟練者モードを使用することで、より高度な設定作業を行うこともできます。具体的には下記のように進めていきます:

1. まずは DHCP サーバが待ち受けるべきネットワークインターフェイスを一覧から選択し、［次へ］を押します。詳しくは 図20.1「DHCP サーバ: カードの選択」をご覧ください。

注記: DHCP と `firewalld` について

openSUSE Leap 15.7 では、［ファイアウオールで選択したインターフェイスを開く］のオプションが正しく動作しない問題が確認されています。手作業で DHCP のポートを開くには、下記のように実行します:

```
> sudo firewall-cmd --zone=public --permanent --add-service=dhcp
> sudo firewall-cmd --reload
```

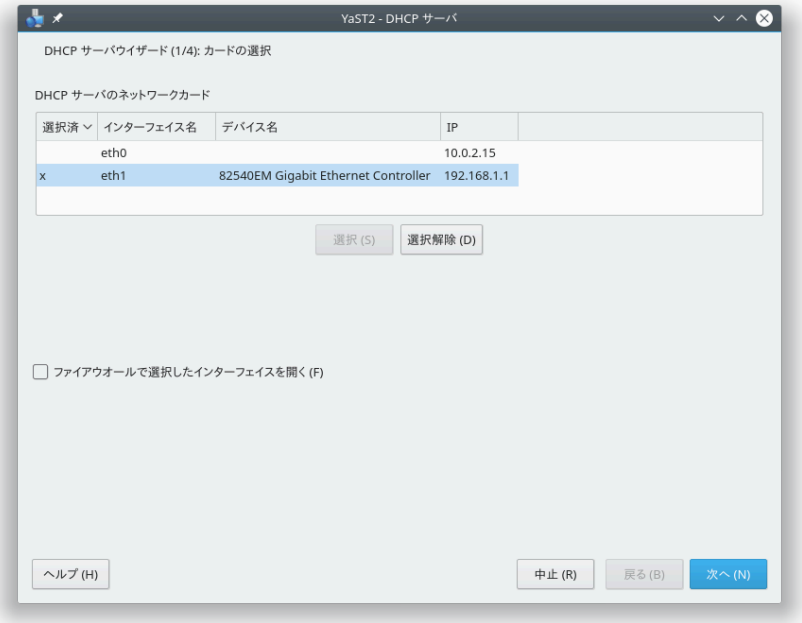


図 20.1: DHCP サーバ: カードの選択

2. まずは LDAP サーバ内に DHCP の設定を保存するかどうかを、チェックボックスで選択します。テキストボックスは、DHCP サーバが管理する全てのクライアントに対する情報を設定します。ドメイン名やタイムサーバのアドレス、プライマリおよびセカンダリのネームサーバ、印刷 (プリント) サーバや WINS サーバ (Windows と Linux のクライアントが共存している場合) やゲートウエイ、貸与時間を設定します。詳しくは 図20.2「DHCP サーバ: グローバル設定」をご覧ください。

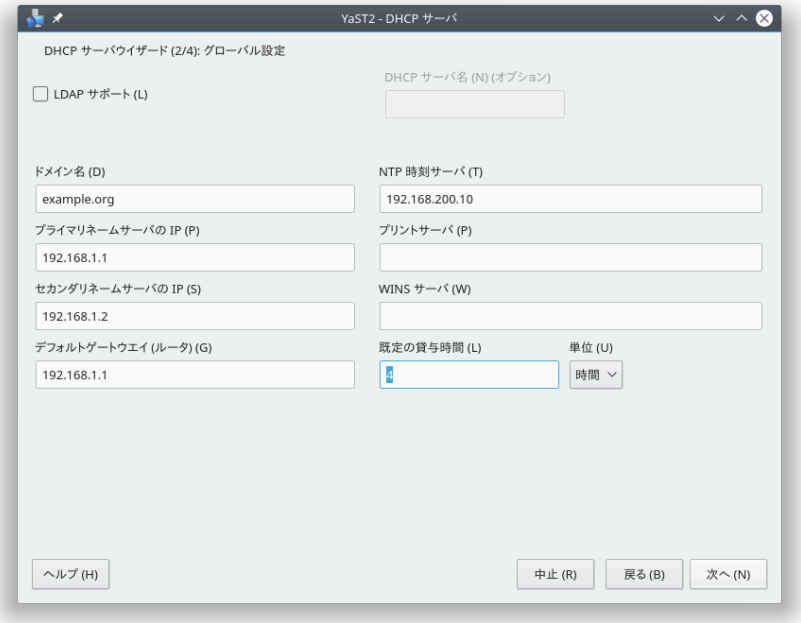


図 20.2: DHCP サーバ: グローバル設定

3. ここでは、動的な IP アドレスをどのようにしてクライアントに配布するかを設定します。まずはサーバがクライアントに対して配布する IP アドレスの範囲を指定します。配布するアドレスは、同じネットマスク内のものに限られることに注意してください。また、クライアント側から再貸与を送信させるまでの貸与時間を設定することもできます。必要であれば、サーバ側で IP アドレスを予約しておく最大の貸与時間を設定することもできます。詳しくは 図20.3「DHCP サーバ: 動的 DHCP」 をご覧ください。

図 20.3: DHCP サーバ: 動的 DHCP

4. ここでは DHCP サーバの開始方法を設定します。システムの起動時に DHCP サーバを自動的に開始するか、もしくは必要に応じて開始する (テスト用途など) かを選択することができます。サーバの設定が終わったら、［完了］を押してください。詳しくは 図20.4「DHCP サーバ: 起動」をご覧ください。

図 20.4: DHCP サーバ: 起動

5. 上述の手順で動的なアドレス割り当てを設定するだけでなく、サーバに対してほぼ固定のアドレスを割り当てさせる方法があります。下半分に表示されたテキストボックスを利用して、固定のアドレスを割り当てるべきクライアントを指定していってください。具体的には、[名前] と [IP アドレス] がクライアントに対して割り当てる名前とアドレスを、[ハードウエアアドレス] と [ネットワークの種類] (トークンリングもしくはイーサネット) がそのクライアントを識別するための情報になっています。上半分に表示されているクライアントの設定には、[追加] を押すことで追加することができます。追加済みの設定を編集したり削除したりするには、対象を上半分で選択して編集を行い、[変更] もしくは [一覧から削除] を押します。詳しくは 図20.5「DHCP サーバ: ホスト管理」をご覧ください。

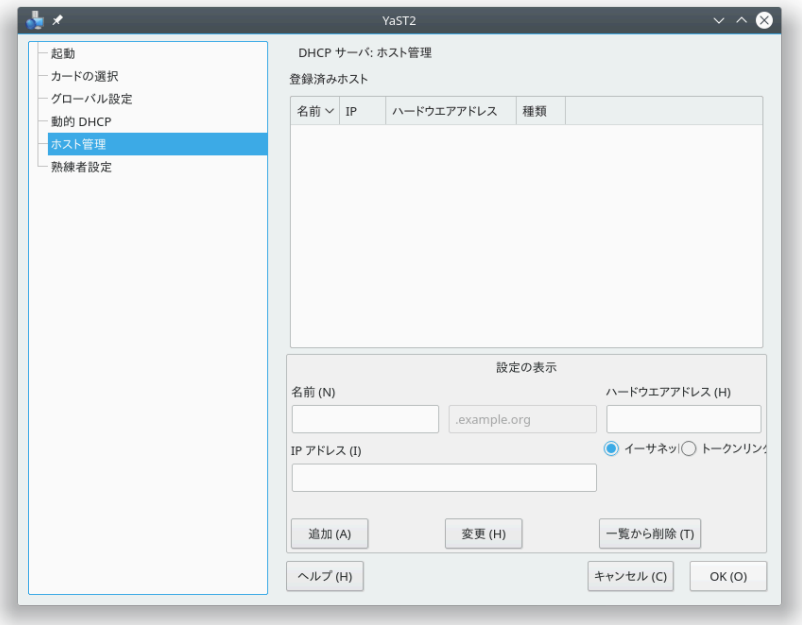


図 20.5: DHCP サーバ: ホスト管理

20.1.2 DHCP サーバ設定 (熟練者向け)

上記までの設定方法に加えて、YaST ではより詳しい設定を行うことのできる熟練者モードが用意されています。熟練者モードを起動するには、[起動] ダイアログ内にある [DHCP サーバ熟練者設定] ボタンを押してください (詳しくは 図20.4「DHCP サーバ: 起動」をご覧ください) 。

chroot 環境と宣言

最初に表示されたダイアログでは、[今すぐ開始] を押すことで、既存の設定ファイルを編集できるようにすることができます。また、このモードではさらに詳しい設定である chroot 環境 (chroot jail とも呼びます) の使用可否を選択することができるようになっています。これは DHCP サーバのコンピュータを保護するための仕組みで、外部から DHCP サーバが脆弱性に対する攻撃を受けても、システム全体には被害が及ばないように緩和するためのものです。ダイアログの下半分には、宣言に関するツリービューが表示されています。それぞれ [追加] , [削除] , [編集] でツリーを操作することができます。詳しくは 図20.6「DHCP サーバ: chroot jail と宣言」をご覧ください。なお、[追加] を押した場合は、追加する宣言の種類を選択することになります。また、[詳細] ボタンを押すと、サーバのログファイルを表示したり、TSIG 鍵の管理を行ったり、DHCP サーバの設定にあわせてファイアウオールの設定を調整したりすることもできます。

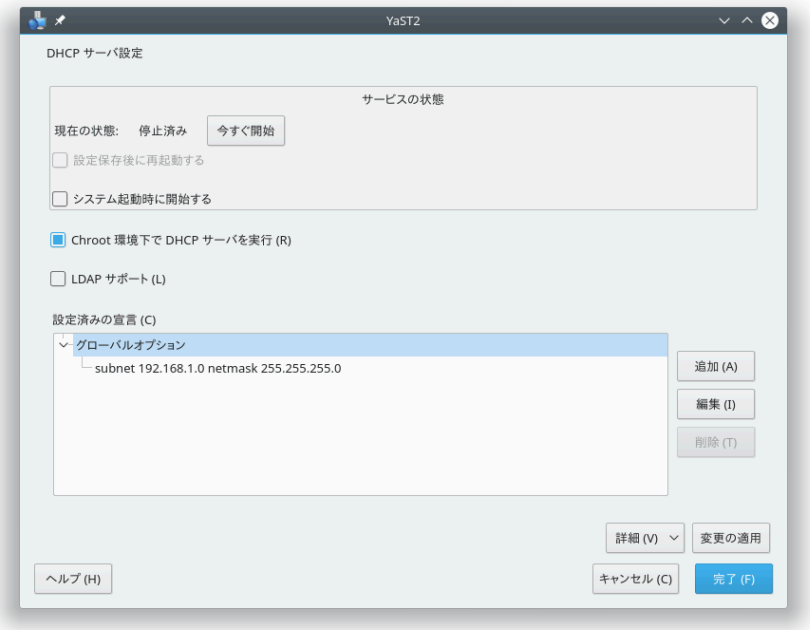


図 20.6: DHCP サーバ: CHROOT JAIL と宣言

宣言の種類の選択

DHCP サーバの［グローバルオプション］には、いくつかの宣言を追加することができます。このダイアログでは、［サブネット］,［ホスト］,［共有ネットワーク］,［グループ］,［グループ］,［アドレスのプール］,［クラス］のいずれかを追加することができます。この例ではサブネットを選択しています (図20.7「DHCP サーバ: 宣言の種類の選択」をご覧ください) 。

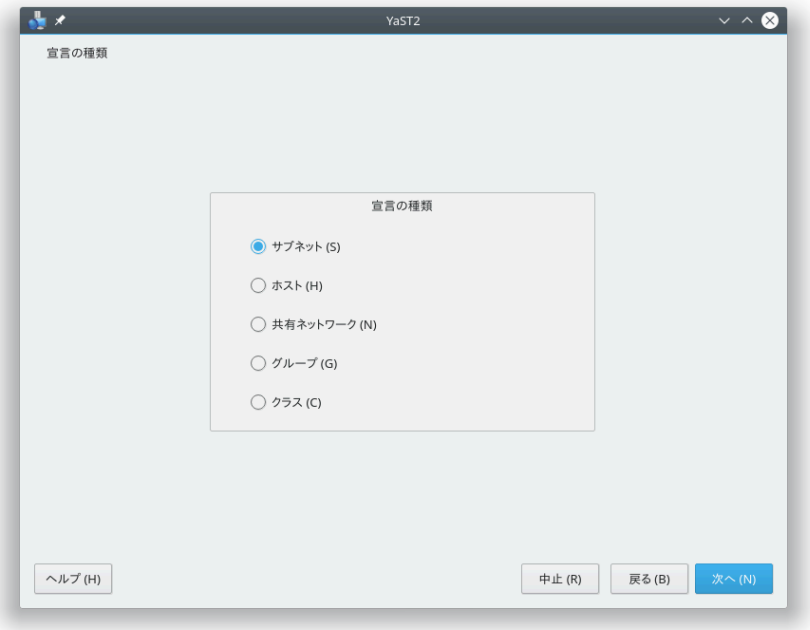


図 20.7: DHCP サーバ: 宣言の種類の選択

サブネットの設定

このダイアログでは、IP アドレスとネットマスクからなる新しいサブネットを宣言することができます。ダイアログの中央部では、［追加］,［編集］,［削除］の各ボタンを利用して、サブネットに対するオプションを指定することができます。また、サブネットに対してダイナミック DNS を設定するには、［ダイナミック DNS］を押します。

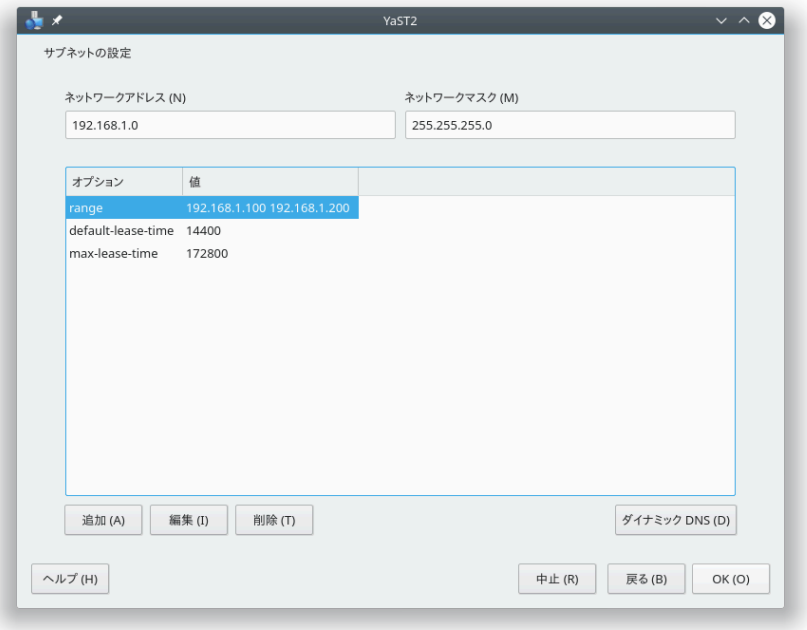


図 20.8: DHCP サーバ: サブネットの設定

TSIG 鍵管理

以前のダイアログでダイナミック DNS を設定するように選択している場合は、ここでゾーン転送の機密を保持するための鍵を管理することができます。ここで［OK］を押すと次のダイアログが表示され、ダイナミック DNS を利用するインターフェイスを設定することができるようになります (詳しくは 図20.10「DHCP サーバ: ダイナミック DNS のためのインターフェイス設定」をご覧ください) 。

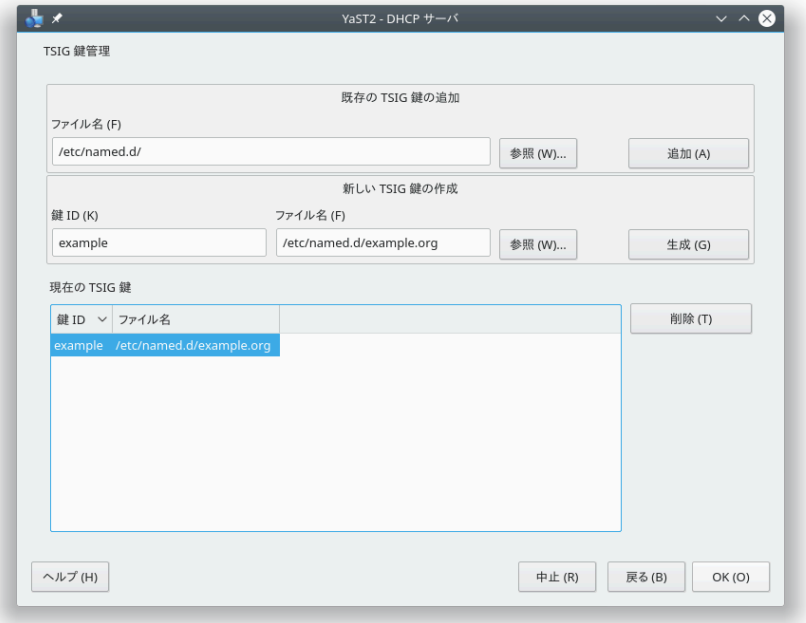


図 20.9: DHCP サーバ: TSIG 設定

ダイナミック DNS: インターフェイス設定

ここでは［このサブネットのダイナミック DNS を有効にする］を選択することで、動的な DNS の更新を行うことができるようになります。選択を行ったら、正引きと逆引きの各ゾーンに対して使用する TSIG 鍵をドロップダウンから選択します。なお、DNS サーバと DHCP サーバとの間で、同じ鍵を設定しなければならないことに注意してください。また、［グローバルダイナミック DNS 設定の更新］を選択することで、ダイナミック DNS の環境に合わせて、グローバルな DHCP サーバの設定を自動調整することもできます。さらに、更新すべきダイナミック DNS の正引きおよび逆引きのゾーンを指定したあと、［OK］を押してください。これにより、サブネットの設定ダイアログ (図20.8「DHCP サーバ: サブネットの設定」) に戻ることができます。再度［OK］を押すと、熟練者向け設定ダイアログに戻ることができます。

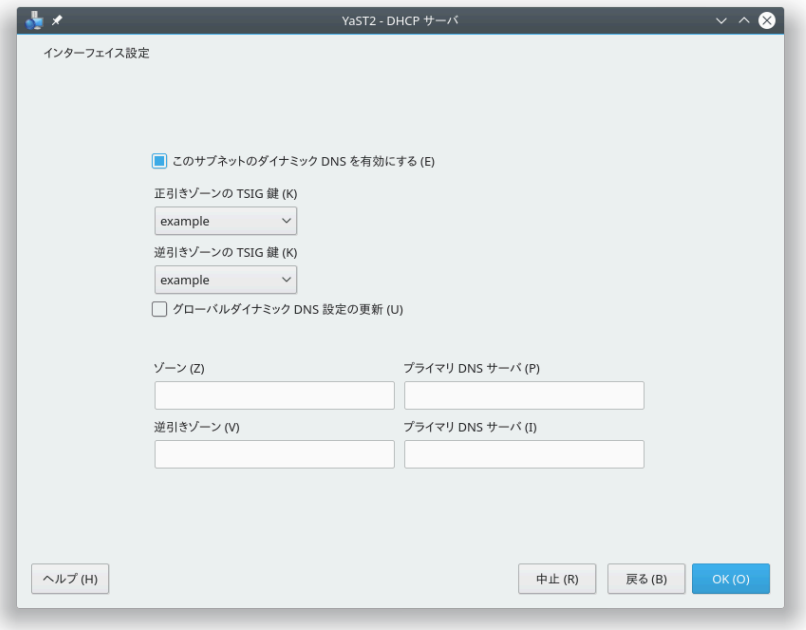


図 20.10: DHCP サーバ: ダイナミック DNS のためのインターフェイス設定

注記: `ignore client-updates` オプションについて

ゾーンに対してダイナミック DNS を有効化している場合、YaST ではクライアントの互換性問題を改善するため、自動的に `ignore client-updates` オプションを設定します。このオプションは不要であれば無効化できます。

ネットワークインターフェイスの設定

DHCP サーバが待ち受けるべきネットワークインターフェイスの選択と、ファイアウオールの設定調整を行うには、熟練者向け設定ダイアログ内の［カードの選択］で行います。表示されたインターフェイスの一覧から、DHCP サービスへのアクセスを許可するインターフェイスを 1 つまたは複数選択します。全てのサブネット内のクライアントがサーバと通信できるようにするには、サーバ側のファイアウオールも調整する必要があります。

注記: DHCP と `firewalld` について

openSUSE Leap 15.7 では、［ファイアウオールで選択したインターフェイスを開く］のオプションが正しく動作しない問題が確認されています。手作業で DHCP のポートを開くには、下記のように実行します:

```
> sudo firewall-cmd --zone=public --permanent --add-service=dhcp
```

```
> sudo firewall-cmd --reload
```

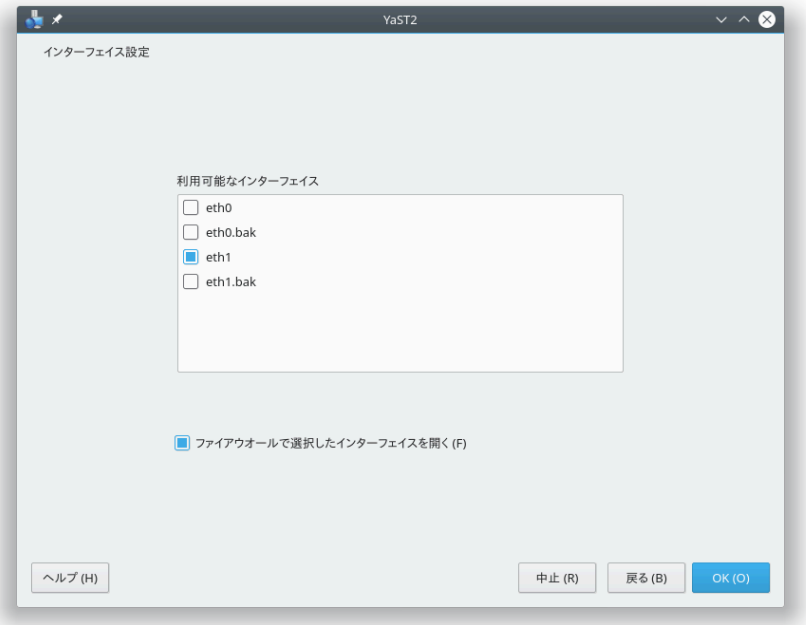


図 20.11: DHCP サーバ: ネットワークインターフェイスとファイアウォール

全ての設定手順が完了したら、[OK] を押してダイアログを閉じてください。新しい設定が書き込まれ、サーバが再起動されます。

20.2 DHCP ソフトウエアパッケージ

openSUSE Leap では、DHCP サーバと DHCP クライアントの両方が提供されています。提供されている DHCP サーバは `dhcpd` (Internet Systems Consortium 提供) で、DHCP クライアントは `dhcp-client` (こちらも Internet Systems Consortium 提供) というパッケージ名になっています。また、ネットワークを設定するためのツールとして `wicked` が用意されています。

注記: ISC DHCP Server の KEA DHCP への置き換えについて

ISC (Internet Systems Consortium) は新しい DHCP サーバである KEA の開発を行っています。詳しくは https://www.isc.org/kea/ をお読みください。

既定では、`wicked` ツールと `wickedd-dhcp4` および `wickedd-dhcp6` の各サービスがインストールされています。各システムでは、DHCP サーバの監視を行う目的で、自動的にどのシステムでも動作するようになっています。これらのサービスを利用するにあたっては、ほとんどの環境で設定ファイルを用意する必要がなく、そのままの状態で問題なく動作するようになっています。より複雑な環境の場合は、`/etc/dhclient.conf` や `/etc/dhclient6.conf` などの設定ファイルで制御できる ISC `dhcp-client` をお使いください。

20.3 DHCP サーバ dhcpd

DHCP システムの中枢は Dynamic Host Configuration Protocol (DHCP) デーモンと呼ばれるものです。このサーバは設定ファイル `/etc/dhcpd.conf` の設定に従って、アドレスを 貸し出し たあと、それがどのように使用されているのかを監視します。このファイルのパラメータや値を変更することで、システム管理者が様々な方法でプログラムの動作を変更することができます。まずは 例20.1「設定ファイル /etc/dhcpd.conf」 にある `/etc/dhcpd.conf` の設定例をご覧ください。

例 20.1: 設定ファイル /ETC/DHCPD.CONF

```
default-lease-time 600;          # 10 分
max-lease-time 7200;             # 2  時間

option domain-name "example.com";
option domain-name-servers 192.168.1.116;
option broadcast-address 192.168.2.255;
option routers 192.168.2.1;
option subnet-mask 255.255.255.0;

subnet 192.168.2.0 netmask 255.255.255.0
 {
  range 192.168.2.10 192.168.2.20;
  range 192.168.2.100 192.168.2.200;
 }
```

これは DHCP サーバからネットワークに対して、IP アドレスを割り当てるための必要十分な設定ファイルです。なお、各行の末尾にはセミコロンを忘れずに付けてください。正しく付けておかないと、dhcpd が起動しなくなってしまいます。

このサンプルファイルは、大きく分けて 3 つのセクションに分割することができます。最初のセクションは、IP アドレスを貸し出してから、クライアント側から更新要求を受け取るまでの既定の貸与時間を秒単位で指定しています (`default-lease-time`) 。また、このセクションでは、DHCP サーバが更新を受け取るまでに IP アドレスを予約しておくべき最大の時間も指定しています (`max-lease-time`) 。

2 番目のセクションでは、グローバルな範囲でいくつかのネットワークパラメータを定義しています。

- `option domain-name` の行では、ネットワーク内で使用すべき既定のドメイン名を指定しています。
- `option domain-name-servers` の行では、最大 3 つまで、ホスト名から IP アドレスを、もしくは IP アドレスからホスト名を解決することのできる DNS サーバを指定することができます。実際には、お使いのマシンにあるネームサーバと DHCP を設定する前から使用していたネットワーク内の DNS サーバを指定するのが一般的です。またネームサーバは、ホスト名と動的なアドレスとの間での解決を行うこともできます。ネームサーバの構築について、詳しくは 第19章「ドメインネームシステム」をお読みください。
- `option broadcast-address` の行では、クライアント側で使用すべきブロードキャストアドレスを指定しています。
- `option routers` の行では、設定された自分のアドレスとサブネットマスクと、宛先のアドレスとを比較した結果、ローカルネットワーク内に存在していない宛先にデータパケットを送信する際、経由すべきゲートウエイ (ルータ) を指定しています。小さなネットワーク環境の場合、このゲートウエイはインターネットに接続しているルータのアドレスになります。
- `option subnet-mask` の行では、クライアントに対して割り当てるネットマスクを指定しています。

ファイル内の最後のセクションはネットワークとサブネットマスクを定義しているものです。この定義の中には、DHCP サーバがクライアントに対して割り当てるアドレスの範囲を指定する必要があります。例 20.1「設定ファイル /etc/dhcpd.conf」の例では、`192.168.2.10` から `192.168.2.20` まで、および `192.168.2.100` から `192.168.2.200` までが割り当ての範囲になっています。

必要に応じていくつかの行を変更したら、あとは `systemctl start dhcpd` コマンドを実行して DHCP デーモンを起動するだけです。このコマンドを実行すると、即時にサービスが開始されます。なお、`rcdhcpd check-syntax` を実行することで、簡単な文法チェックを行うこともできます。設定内に何らかの問題があって起動時にエラーメッセージが表示された場合は、`journalctl` コマンドを利用することで、メインのシステムログを表示させて、原因を探ることができます (詳しくは 第11章「journalctl : systemd ジャーナルへの問い合わせコマンド」をお読みください) 。

なお、既定の openSUSE Leap システムでは、セキュリティ上の理由から、DHCP デーモンが chroot 環境で実行されます。設定ファイルは、chroot 環境から読み込むことができるようにするため、あらかじめコピーしておかなければなりません。ただし、通常は `systemctl start dhcpd` を実行することで、設定ファイルが自動的にコピーされますので、特に気にする必要はありません。

20.3.1 固定 IP アドレスのクライアント

DHCP では、特定のクライアントに対して固定のアドレスを定義して、割り当てることができます。ただし、アドレスはプール内で事前に明示的に割り当てておかなければなりません。動的なアドレスとは異なり、固定のアドレスでは有効期限が切れることはありません。

クライアントに対して固定のアドレスを割り当てるにあたって、クライアントの識別はハードウエアアドレスを使用します (ハードウエアアドレスはグローバルにユニークなアドレスで、6 バイトの値から構成されています。たとえば `00:30:6E:08:EC:80` のように表記します) 。固定のアドレスを割り当てるには、例20.2「設定ファイルへの追加」のような内容を 例20.1「設定ファイル /etc/dhcpd.conf」にある設定ファイルに追加する必要があります。DHCP サーバ側では、このクライアントからアクセスがあると、常に同じデータを返すようになります。

例 20.2: 設定ファイルへの追加

```
host jupiter {
hardware ethernet 00:30:6E:08:EC:80;
fixed-address 192.168.2.100;
}
```

最初の行にはクライアントの名前 ((`host` ホスト名 、ここでは `jupiter`) が記述され、2 行目には MAC アドレスが記述されています。Linux コンピュータであれば、MAC アドレスは `ip link show` に続いて、インターフェイス名 (例: `eth0`) を指定することで、表示することができます。このコマンドを実行すると、下記のような内容が出力されます:

```
link/ether 00:30:6E:08:EC:80
```

上記の例では、MAC アドレスが `00:30:6E:08:EC:80` のクライアントに対して、IP アドレス `192.168.2.100` とホスト名 `jupiter` を自動的に割り当てる設定になります。なお、`hardware` の後ろはほぼ全ての場合において `ethernet` になりますが、IBM システムなどでは `token-ring` を指定する必要がある場合もあります。

20.3.2 openSUSE Leap 版について

セキュリティを向上させる理由から、ISC 提供の DHCP サーバの openSUSE Leap 版では、Ari Edelkind 氏による non-root/chroot (root 以外のユーザで動作させ、かつ chroot 環境を使用する) 修正が適用されています。これにより、dhcpd は `nobody` というユーザ権限で、かつ chroot 環境内 (`/var/lib/dhcp`) で動作するように設定されています。また、この構成を利用するため、設定ファイルである `dhcpd.conf` は、`/var/lib/dhcp/etc` 内に配置しなければならなくなっています。通常は、サービスの起動時にスクリプト側で自動的にコピーするように設計されているため、特に特殊な作業を行う必要はありません。

この機能に関するサーバ側の動作を制御したい場合は、`/etc/sysconfig/dhcpd` ファイルを編集してください。chroot 環境を利用したくない場合は、`/etc/sysconfig/dhcpd` ファイル内の `DHCPD_RUN_CHROOTED` の変数を、「no」に設定してください。

また、chroot 環境下で dhcpd がホスト名を解決できるようにするため、下記の設定ファイルも chroot 環境内にコピーされるようになっています:

- `/etc/localtime`
- `/etc/host.conf`
- `/etc/hosts`
- `/var/run/netconfig/resolv.conf`

これらのファイルは、起動スクリプト内で `/var/lib/dhcp/etc/` 内にコピーされるようになっています。`/etc/ppp/ip-up` のようなスクリプトでこれらのファイルを動的に変更するような場合は、それらが反映されなくなってしまいますが、設定ファイル内では IP アドレスのみを指定していることから、ほとんど影響はありません。

また、chroot 環境内にコピーすべき追加のファイルが存在する場合は、それらのファイルを `/etc/sysconfig/dhcpd` 内の `DHCPD_CONF_INCLUDE_FILES` 内で指定してください。また、DHCP のログ機構に対して、syslog デーモンの再起動後も正しく動作するようにするには、`/etc/sysconfig/syslog` 内の `SYSLOGD_ADDITIONAL_SOCKET_DHCP` を設定してください。

20.4 さらなる情報

DHCP に関する詳しい情報は、Internet Systems Consortium (https://www.isc.org/dhcp/) で提供されています。ここには `dhcpd`，`dhcpd.conf`，`dhcpd.leases`，`dhcp-options` の各マニュアルページも用意されています。

21 Samba

改訂履歴

2023-08-23

Samba を使用することで、Unix マシンを MacOS や Windows, OS/2 マシンに対するファイルサーバや印刷サーバに仕立て上げることができます。Samba は今や本格的な使用にも耐える複雑な製品になっています。Samba の設定は YaST で行うことができるほか、設定ファイルを手作業で修正して設定することもできます。

重要: SMB1 のサポート廃止について

Samba バージョン 4.17 およびそれ以降の openSUSE Leap では、SMB1 プロトコルが無効化されています。

21.1 用語

Samba の文書や YaST のモジュールでは、それぞれ下記のような用語を使用します。

SMB プロトコル

Samba では NetBIOS サービスをベースにした SMB (Server Message Block) というプロトコルを使用しています。Microsoft 社はソフトウエアの製造元に向けてプロトコルを公開しているため、他のオペレーティングシステムからでも Microsoft のオペレーティングシステムに接続できるようになっています。なお、Samba では SMB プロトコルを TCP/IP プロトコル上で動作させていることから、Samba に接続するクライアントには TCP/IP プロトコルをインストールしておかなければなりません。

CIFS プロトコル

CIFS (Common Internet File System) プロトコルは SMB1 としても知られているプロトコルで、SMB プロトコルの初期のバージョンです。CIFS は TCP/IP を介して遠隔のファイルシステムにアクセスするためのプロトコルを規定していて、共同作業のグループを作成したり、ネットワークを介して文書を共有したりすることができます。

なお、SMB1 は Microsoft Windows Vista™ 以降で提供されるようになった SMB2 に取って代わられています。その後、Microsoft Windows 8™ と Microsoft Windows Server 2012 では SMB3 が提供されるようになっています。なお、最近のバージョンの Samba では、セキュリティ上の理由から、SMB1 は無効化されています。

NetBIOS

NetBIOS は、ネットワーク内でのコンピュータ同士の名前解決や通信を行うために設計されたソフトウエアインターフェイス (API) です。NetBIOS はネットワーク内でホスト名の重複が生じないように予約する処理を行います。予約が完了したホスト同士は、NetBIOS プロトコル内でホスト名によるアクセスができるようになります。なお、名前を管理するための中央サーバのような仕組みは存在せず、ネットワーク内で既に使用されている場合を除いて、任意の名前を予約することができます。NetBIOS は様々なネットワークプロトコルに対応していますが、その中の比較的シンプルな実装として、NetBEUI と呼ばれる、ルーティングに対応していないプロトコルにも対応しています (NetBIOS API とよく混同されます) 。このほか NetBIOS は Novell 社提供の IPX/SPX プロトコルでも利用できますが、Samba のバージョン 3.2 以降では IPv4 および IPv6 上での NetBIOS にのみ対応しています。

なお TCP/IP で送信される NetBIOS のホスト名は、`/etc/hosts` や DNS で管理される名前とは独立して動作します。また NetBIOS では、独自の独立した名前付け規約が存在しています。ですが、通常は DNS のホスト名と NetBIOS の名前を一致させて、管理上の混乱を避けておくことをお勧めします。Samba でも特に指定のない限り、DNS 名と NetBIOS の名前は一致するように動作します。

Samba サーバ

Samba サーバは SMB/CIFS サービスや NetBIOS over IP のホスト名管理サービスを、クライアントに対して提供する仕組みです。Linux では、Samba サーバは SMB/CIFS サービスを提供する `smbd` 、ホスト名管理サービスを提供する `nmbd` 、そして認証機能を提供する `winbind` の 3 つから構成されています。

Samba クライアント

Samba クライアントは、SMB プロトコルを介して Samba サーバが提供するサービスを使用するシステムを意味します。Windows や MacOS などの一般的なオペレーティングシステムが SMB プロトコルに対応しています。なお、TCP/IP プロトコルも全てのコンピュータにインストールしておかなければなりません。Samba は様々な Unix 系オペレーティングシステムに対して、クライアント機能を提供しています。Linux の場合は、Linux のシステムレベルで SMB の資源を利用することができるよう、SMB 向けのカーネルモジュールが用意されています。そのため、Samba クライアント側では、デーモンを動作させる必要がありません。

共有

SMB サーバは、共有 と呼ばれるものを介してクライアントに資源を提供します。共有はサーバ内の特定のディレクトリ (およびそれ以下のディレクトリ) を表すものであるほか、プリンタを表す場合もあります。また、共有には 共有名 を付けて公開し、名前でアクセスを行います。共有名には任意の名前を付けることができますので、ディレクトリ名と必ずしも一致する必要はありません。また、プリンタについても名前を設定します。クライアント側からは名前でプリンタにアクセスします。

また、共有名がドル記号 (`$`) で終わるものは隠し共有として扱われます。このような共有は Windows コンピュータから共有名を参照した場合、一覧表示には現れないものになります。

DC

ドメインコントローラ (DC) は、ドメイン内のアカウントを処理するサーバです。データを複製する目的で、1 つのドメイン内に複数のドメインコントローラを設置することもあります。

21.2 Samba サーバのインストール

Samba サーバをインストールするには、YaST を起動して［ソフトウエア］›［ソフトウエア管理］を選択し、表示されたウインドウ内で［表示］›［パターン］を選んで［ファイルサーバ］を選択します。あとは［了解］を押して必要なパッケージをインストールします。

21.3 Samba の開始と停止

Samba サーバは、システムの起動時に自動的に開始することができるほか、手作業で開始するように設定することができます。開始と停止のポリシーは、21.4.1項「YaST を利用した Samba サーバの設定」で説明しているとおり、YaST の Samba サーバ設定内で指定することができます。

コマンドラインから Samba サービスを停止するには、`systemctl stop smb nmb` を実行します。Samba サービスを開始するには、`systemctl start nmb smb` を実行します。`smb` サービスを開始することで、`winbind` も必要に応じて自動的に開始されます。

ヒント: `winbind`

`winbind` は個別のサービスとして提供されているもので、パッケージとしても `samba-winbind` という別の名前のパッケージになっています。

21.4 Samba サーバの設定

openSUSE® Leap 内での Samba サーバは、2 種類の異なる方法で設定することができます。1 つは YaST を利用する方法、もう 1 つは手作業で設定する方法です。手作業で設定を行うことで、より細かい設定を行いことができますが、YaST GUI ほどの利便性はありません。

21.4.1 YaST を利用した Samba サーバの設定

Samba サーバを設定するには、YaST を起動して［ネットワークサービス］›［Samba サーバ］を選択します。

21.4.1.1 Samba の初期設定

初めてモジュールを起動した場合は、［Samba インストール］のダイアログが表示され、サーバの管理に関わる基本的な設定を行うことができます。また設定の最後では、Samba の管理者パスワード (［Samba の管理者 (root) パスワード］) を入力します。2 回目以降のモジュール起動では、［Samba の設定］ダイアログが表示されます。

［Samba インストール］ダイアログは 2 つの手順から構成され、必要であれば詳細な設定を行うことができるようになっています:

ワークグループまたはドメイン名

［ワークグループまたはドメイン名］では、新しいワークグループまたはドメインの名前、もしくは既存の名前を選択して［次へ］を押します。

Samba サーバの種類

次の手順では、サーバをプライマリドメインコントローラ (PDC) として動作させるか、もしくはバックアップドメインコントローラ (BDC) として動作させるか、もしくはドメインコントローラとして動作させないようにするかを選択することができます。選択を行ったら［次へ］を押します。

詳細なサーバ設定を行いたくない場合は、［OK］を押してそのまま閉じてください。最後に表示されたポップアップボックスで［Samba の管理者 (root) パスワード］を入力します。

全ての設定は、［Samba の設定］ダイアログ内で後から変更することができます。ダイアログには［起動］,［共有］,［識別情報］,［信頼するドメイン］,［LDAP の設定］の各タブが用意されています。

21.4.1.2 サーバ側における最新バージョンの SMB サーバへの対応について

最新バージョンの openSUSE Leap やその他の最新 Linux ディストリビューションをお使いの場合、安全性の低い SMB1/CIFS プロトコルが既定で無効化されています。しかしながら、既存の Samba サーバの場合、SMB1 プロトコルのみを使用するように構成されているものも存在しています。このようなサーバに接続しているクライアントに対応するには、少なくとも SMB 2.1 プロトコルに対応できるように設定する必要があります。

環境によっては SMB1 プロトコルのみを使用できるものもあります。たとえば SMB1 や CIFS の Unix 拡張に依存して動作しているような場合です。これらの拡張は新しいバージョンのプロトコルには移植されていませんので、そのような環境をお持ちの場合は、環境を変更するか、もしくは 21.5.2項「クライアント側での SMB1/CIFS 共有のマウント」をお読みください。

SMB 2.1 プロトコルに対応するよう設定するには、`/etc/samba/smb.conf` ファイル内のグローバルパラメータに `server max protocol = SMB2_10` を設定します。設定可能な値の一覧について、詳しくは `man smb.conf` をお読みください。

21.4.1.3 高度な Samba の設定

Samba サーバモジュールを初めて起動した場合も、初回起動時の設定を行うと、21.4.1.1項「Samba の初期設定」にあるとおりの［Samba の設定］ダイアログが表示されるようになります。ここから Samba サーバの設定を変更することができます。

設定の変更が終わったら、［OK］を押すと設定を保存することができます。

21.4.1.3.1 サーバの開始

［開始］タブでは、Samba サーバの開始方法を設定することができます。システムの起動時にサービスを開始するように設定するには、［システム起動時］を選択します。手作業で開始するように設定するには、［手動］を選択します。Samba サーバの開始に関する詳細は、21.3項「Samba の開始と停止」をお読みください。

このタブでは、ファイアウオールのポートを開く作業を行うこともできます。ポートを開くには、［ファイアウオールでポートを開く］を選択します。複数のネットワークインターフェイスを接続している場合は、［ファイアウオールの詳細］を押して、Samba サービスを提供するネットワークインターフェイスを選択することができます。インターフェイスを選択したら［OK］を押して閉じてください。

21.4.1.3.2 共有

［共有］タブでは、有効にする Samba の共有を選択することができます。なお、homes や printers のように、いくつかの事前設定が用意されています。共有を選択して［状態切り替え］を押すことで、［有効］と［無効］の間を切り替えることができます。また、新しい共有を追加するには［追加］を、選択した共有を削除するには［削除］を押します。

ユーザが独自にディレクトリを指定して共有できるようにするには、［ユーザにディレクトリの共有を許可する］を選択し、［許可するグループ］内に共有の設定を許可するグループを指定します。たとえば `users` のように指定した場合はローカルのグループを、`ドメインユーザ` のように指定した場合はドメインのグループを指定することができます。また、共有を設定するには、ファイルシステム側のアクセス

権で許可しておかなければならないことにも注意してください。このほか、［最大共有数］を指定することで、作成することのできる共有数の最大値を制限することができます。認証を行わずにユーザの作成した共有にアクセスできるようにするには、［ゲストアクセスを許可］を選択してください。

21.4.1.3.3 識別情報

［識別情報］タブでは、ホストの所属するドメイン（［基本設定］）やネットワーク内での代替ホスト名の使用（［NetBIOS ホスト名］）を設定することができます。このほか、名前解決に対して Microsoft Windows Internet Name Service (WINS) を使用するよう設定することもできます。この場合は、［ホスト名の解決に WINS を使用する］を選択し、［DHCP で WINS サーバのアドレスを取得］を必要に応じて選択してください。TDB データベースではなく LDAP を使用するような場合など、より高度なグローバル設定やユーザ認証設定を行いたい場合は、［詳細設定］を押します。

21.4.1.3.4 信頼するドメイン

他のドメインのユーザがお使いのドメインにアクセスできるようにするには、［信頼するドメイン］のタブで適切な設定を行います。新しいドメインを追加するには［追加］を押します。選択したドメインを削除するには［削除］を押します。

21.4.1.3.5 LDAP の設定

［LDAP 設定］のタブでは、認証時に LDAP のサーバを使用するように設定することができます。LDAP サーバとの通信をテストするには、［接続のテスト］を押します。より高度な設定や既定値の設定を行うには、［詳細設定］を押します。

LDAP の設定に関する詳細は、『セキュリティ強化ガイド』、第5章「389 Directory Server を利用した LDAP サービス」をお読みください。

21.4.2 手作業によるサーバの設定

Samba をサーバとして使用したい場合は、`samba` をインストールしてください。Samba のメインの設定ファイルは `/etc/samba/smb.conf` です。このファイルは論理的に 2 つのパートから構成されていて、`[global]` セクションには全体的な設定が含まれています。また、下記の既定のセクションには、個別のファイルおよびプリンタの共有が含まれています:

- [homes]
- [profiles]

- [users]
- [groups]
- [printers]
- [print$]

このような構造になっていることから、共有に対する設定は個別に行うことができるだけでなく、`[global]` セクションで一括に設定することもできるようになっています。これにより、設定ファイルを読みやすくしています。

21.4.2.1 global セクション

`[global]` セクション内の下記のパラメータは、お使いのネットワーク環境や要件に応じて、修正する必要があります。これにより、Windows 環境の他のマシンから、お使いの Samba サーバにアクセスすることができるようになります。

`workgroup = WORKGROUP`

この行は、Samba サーバをワークグループに割り当てる設定です。`WORKGROUP` の箇所は、お使いのネットワーク環境に合わせて設定してください。また、お使いの Samba サーバは、ネットワーク内に同じ名前のものが存在していない限り、DNS ホスト名でアクセスすることができます。ホスト名が利用できない場合は、`netbiosname=ホスト名` を指定して対応することもできます。このパラメータについて、詳しくは `smb.conf` のマニュアルページをお読みください。

`os level = 20`

このパラメータは、お使いの Samba サーバがワークグループ内の LMB (ローカルマスタブラウザ; Local Master Browser) になるかどうかを設定するものです。`2` のように非常に小さい値を設定すると、もしも Samba サーバの設定を誤っていても、既存の Windows ネットワークに悪影響を及ぼさないようにすることができます。この設定値に関する詳細は、Samba 3 Howto の Network Browsing 章をお読みください。Samba 3 Howto については 21.9項「さらなる情報」をご覧ください。

他の SMB サーバ (Windows 2000 Server など) がネットワーク内に存在しない場合で、お使いの Samba サーバに対してローカル環境内に存在するコンピュータの管理を行わせたい場合は、`os level` の値をより大きく (例: `65`) してください。これにより、Samba サーバがローカルネットワーク内の LMB として動作するようになります。

この設定を変更する場合は、既存の Windows ネットワーク環境に対してどのような影響が発生しうるのかをよくお確かめください。可能の限り、個別の環境を構築してテストするか、業務上の影響がない日を選んで設定してください。

`wins support` および `wins server`

お使いの Samba サーバを WINS サーバのある Windows ネットワークに接続するには、`wins server` オプションを追加して、WINS サーバのアドレスを指定してください。
もしもお使いの Windows マシンが異なるサブネット内に存在している環境で、Samba サーバとの間で相互に通信を行う必要がある場合も、WINS サーバを設定する必要があります。
Samba サーバを WINS サーバとしても動作させるには、`wins support = Yes` を設定してください。ただし、ネットワーク内の 1 台のみで、この設定を有効化してください。また、`wins server` と `wins support` は、同じ `smb.conf` ファイル内で同時に設定してはなりません。

21.4.2.2 共有

下記の例では、CD-ROM ドライブとユーザのホームディレクトリ (`homes`) を、SMB クライアントに対して公開する設定を示しています。

[cdrom]

CD-ROM ドライブが意図せずに公開されてしまうことを防ぐため、これらの行は行頭にコメント文字 (この場合はセミコロン) を追加して無効化されています。Samba で CD-ROM ドライブを共有するには、行頭のセミコロンを削除してください。

例 21.1: CD-ROM ドライブの共有

```
[cdrom]
       comment = Linux CD-ROM
       path = /media/cdrom
       locking = No
```

`[cdrom]` と `comment`

`[cdrom]` では、ネットワーク内で全ての SMB クライアントに対して表示される、共有名を設定しています。また、`comment` 以下の値はコメント欄で、共有に関するより詳しい説明文を設定します。

`path = /media/cdrom`

`path` の指定を行うことで、指定したディレクトリ (この場合は `/media/cdrom`) を公開することになります。

既定の設定は非常に限定された構成になっているため、この種類の共有は、このシステム内に存在するユーザにのみ提供されます。この共有を誰にでも利用できるようにするには、設定内に `guest ok = yes` を追加してください。この設定により、ネットワーク内の誰にでも読み込むことができるようになります。ただし、このパラメータはよく注意してお使いください。また、このパラメータは `[global]` セクション内でも設定することができます。

`[homes]`

`[homes]` の共有は、ここでは特別な意味があります。Linux ファイルサーバ内にアカウントとパスワードが存在し、ホームディレクトリが存在するユーザであれば、そのホームディレクトリに接続できるようになります。

例 21.2: [HOMES] 共有

```
[homes]
        comment = Home Directories
        valid users = %S
        browseable = No
        read only = No
        inherit acls = Yes
```

[homes]

SMB サーバに接続するユーザ名と同じ名前の共有が存在する場合を除き、`[homes]` では動的に共有が生成されます。生成される共有名は、ユーザ名と同じになります。

`valid users = %S`

`%S` は Samba サーバとの接続が確立した際に、共有の名前に置き換えられる項目です。`[homes]` セクションでこれを指定した場合は、ユーザ名に置き換えられます。そのため、ユーザ名と同じ名前の共有は、そのユーザでのみアクセスすることができます。

`browseable = No`

この設定は、ネットワーク環境内では表示を行わないようにする設定です。

`read only = No`

既定では、Samba は `read only = Yes` パラメータと同じ効力になっていて、公開された共有への書き込みを禁止しています。この共有に対する書き込みを許可するには、`read only = No` を設定してください。なお、`writable = Yes` でも同じ意味になります。

`create mask = 0640`

MS Windows NT ベースではないシステムの場合、Unix のアクセス権 (パーミッション) に相当する考え方はなく、ファイルを作成する際にもアクセス権を設定することができません。`create mask` パラメータでは、新しく作成したファイルに対して割り当てるアクセス権を指定します。この設定は書き込みのできる共有に対してのみ適用されます。そのため、上記の設定を行うと、所有者が読み込みと書き込みの権限を、所有者のプライマリグループに属するユーザが読み込みの権限を持つようになります。ただし、`valid users = %S` という設定が存在しているため、グループに対して読み込みアクセスを許可していても、実際にはアクセスができなくなります。グループに対して読み込みや書き込みのアクセスを許可するには、`valid users = %S` の行を無効化してください。

警告: Samba における NFS マウントの共有設定について

`NFS` でマウントされているディレクトリは、Samba で共有してしまうとデータの損失が発生する可能性がありますので、サポート対象外となります。対象のファイルが存在するサーバで直接 Samba を動作させるか、`iSCSI` などをお使いください。

21.4.2.3 セキュリティレベル

セキュリティを強化する目的で、それぞれの共有にはパスワードを設定して保護することができます。SMB では、アクセス許可を確認するための様々な方法が用意されています:

ユーザレベルセキュリティ (`security = user`)
: この設定は、SMB に対してユーザの考え方を導入します。各ユーザはサーバに対して、あらかじめ登録を行ってパスワードを設定する必要があります。登録完了後は、設定したユーザ名に従って個別にアクセスを許可するようになります。

ADS レベルセキュリティ (`security = ADS`)
: このモードでは、Samba を Active Directory 環境内のドメインメンバーとして動作させます。このモードでは、Samba サーバ内に Kerberos をインストールして設定しておく必要があります。また、Samba サーバは ADS の領域内に参加させておく必要もあります。ADS への参加は、YaST の［Windows ドメインメンバーシップ］モジュールで実施することができます。

ドメインレベルセキュリティ (`security = domain`)
: このモードは、Samba サーバを Windows ドメインに参加させた場合にのみ正しく動作します。Samba は指定されたユーザ名とパスワードを、Windows NT Server のメンバーサーバと同様に、Windows のプライマリもしくはバックアップドメインコントローラに送信して認証作業を行うようになります。また、このモードではパスワードの暗号化を `yes` に設定する必要があります。

上記のユーザレベル、ADS レベル、ドメインレベルのセキュリティ設定は、サーバ全体に対して適用されます。一方の共有をユーザレベルに、他方の共有を ADS レベルに、などの設定を行うことはできません。しかしながら、システムに複数の IP アドレスが設定されていれば、それぞれに対して別々の Samba サーバを起動することは可能です。

本件について、より詳しい情報は Samba 3 HOWTO に記載されています。また、1 つのシステム内で複数のサーバを動作させる場合は、`interfaces` と `bind interfaces only` のオプション設定にご注意ください。

21.5 クライアントの設定

クライアント側から Samba サーバへのアクセスは、TCP/IP 経由でのみ実施することができます。NetBEUI や IPX などのプロトコルは、Samba では使用できません。

21.5.1 YaST での Samba クライアントの設定

Samba クライアントを設定することで、Samba サーバや Windows サーバにあるファイルやプリンタなどの資源にアクセスすることができるようになります。また、［ネットワークサービス］›［Windows ドメインメンバーシップ］ダイアログを使用することで、Windows ドメインや Active Directory ドメインに参加することもできます。また、［Linux の認証に SMB の情報を使用する］を選択すると、Samba や Windows, Kerberos サーバなどの認証を使用することができるようになります。

［熟練者向け設定］を押すと、より高度な設定オプションにアクセスすることができます。たとえば［サーバディレクトリのマウント］の表では、サーバのホームディレクトリの認証を指定して自動的にマウントする設定を行うことができます。この方法により、CIFS で提供されているホームディレクトリをユーザからアクセスできるようになります。詳しくは `pam_mount` のマニュアルページをお読みください。

全ての設定が完了したら、［OK］を押して閉じることで設定を保存することができます。

21.5.2 クライアント側での SMB1/CIFS 共有のマウント

SMB1 (CIFS) は SMB ネットワークプロトコルの初代となるバージョンで、古くて安全性も保たれておらず、作成者である Microsoft 自身も廃止対象としているものです。また、このようなセキュリティ面の理由から、openSUSE Leap での `mount` コマンドも、既定では SMB 2.1, SMB 3.0, SMB 3.02 の新しいプロトコルバージョンを利用して SMB 共有をマウントするようになっています。

しかしながら、このような変更は `mount` コマンドを利用して、かつ `/etc/fstab` を利用した場合にのみ適用されます。下記のいずれかの形態で使用した場合は、従来通り SMB1 を利用することができます:

- `smbclient` ツールを使用した場合。
- openSUSE に同梱されている Samba サーバソフトウエアを使用した場合。

また、既定の設定のままで使用すると、SMB1 にのみ対応している環境では、接続が失敗することになります。SMB1 にのみ対応している環境には、下記のようなものがあります:

- 新しい SMB プロトコルバージョンに対応していない SMB サーバを利用して構築している場合。Windows では、Windows 7 もしくは Windows Server 2008 以降で SMB 2.1 に対応しています。
- SMB1 や CIFS の Unix 拡張に依存した構成になっている場合。これらの拡張は、新しいプロトコルバージョンには移植されていません。

重要: セキュリティ面の危険性について

下記の手順を実施してしまうと、セキュリティ面の問題を突くことができる環境を構成することになります。セキュリティ面の問題について、詳しくは https://blogs.technet.microsoft.com/filecab/2016/09/16/stop-using-smb1/ (英語) をお読みください。

できる限り早急にお使いのサーバをアップグレードし、新しいバージョンの SMB プロトコルに対応するようにしてください。

openSUSE Leap での新しいプロトコルバージョンの有効化について、詳しくは 21.4.1.2項「サーバ側における最新バージョンの SMB サーバへの対応について」をお読みください。

現在の openSUSE Leap カーネルで SMB1 共有を有効化する必要がある場合は、`mount` コマンドを実行する際に `vers=1.0` というオプションを追加してください:

```
# mount -t cifs //ホスト/共有名 /マウントポイント -o username=ユーザ_ID,vers=1.0
```

それ以外にも、openSUSE Leap では SMB1 を全ての共有に対して有効化することもできます。これを行うには、`/etc/samba/smb.conf` の `[global]` セクション内に下記を追加します:

```
client min protocol = CORE
```

21.6 ログインサーバとしての Samba

企業内の環境などでは、一般にユーザアカウントとパスワードを一括で管理する機能が求められます。Windows ベースのネットワークの場合、この処理はプライマリドメインコントローラ (PDC) で処理を行います。PDC として設定されている Windows Server が存在していればそれでもかまいませんが、Samba サーバを PDC にすることもできます。このような構成は、`smb.conf` 内の `[global]` セクションを、例21.3「smb.conf の global セクション」のように設定することで可能になります。

例 21.3: SMB.CONF の GLOBAL セクション

```
[global]
```

```
    workgroup = WORKGROUP
    domain logons = Yes
    domain master = Yes
```

なお、ユーザアカウントとパスワードは、Windows 側の暗号化方式に準拠させるため、事前の準備が必要となります。この作業は `smbpasswd -a ユーザ名` を実行することで行うことができます。また、Windows のドメインの考え方では、コンピュータ向けのドメインアカウントを作成する必要もあります。コンピュータ向けのアカウントを作成するには、下記を実行します。

```
useradd ホスト名
smbpasswd -a -m ホスト名
```

コンピュータアカウントを作成する場合、`useradd` コマンドの末尾にドル記号を付ける必要があります。`smbpasswd` コマンドでは、`-m` オプションを指定すれば自動的に追加されます。また、`/usr/share/doc/packages/samba/examples/smb.conf.SUSE` にある設定例では、これを自動化するための設定が書かれています。

```
add machine script = /usr/sbin/useradd -g nogroup -c "NT Machine Account" \
-s /bin/false %m
```

Samba 側で上記のスクリプトが正しく実行されるようにするため、必要な管理者権限のある Samba ユーザを選択して、そのユーザを `ntadmin` グループに追加してください。この Linux グループに属する全てのユーザが `Domain Admin` になるようにするには、下記のコマンドを実行します:

```
net groupmap add ntgroup="Domain Admins" unixgroup=ntadmin
```

21.7 Active Directory ネットワーク環境での Samba サーバ

Linux サーバと Windows サーバを共存させているような環境では、それぞれが個別の認証システムや認証ネットワークを構成できるだけでなく、中央の 1 つの認証システムを利用するように構成することができます。Samba では Active Directory ドメインとの協調作業を行うことができることから、openSUSE Leap サーバを Active Directory (AD) のドメインに参加させることができます。

Active Directory ドメインに参加するには、下記の手順を実施します:

1. `root` でログインして、YaST を起動します。
2. [ネットワークサービス] › [Windows ドメインメンバーシップ] を選択します。
3. [Windows ドメインメンバーシップ] が表示されたら、[ドメインまたはワークグループ] の欄に、参加したいドメインを入力します。

図 21.1: WINDOWS ドメインメンバーシップの決定

4. お使いのサーバの Linux 認証に SMB の認証を使用したい場合は、[Linux の認証に SMB の情報を使用する] を選択します。

5. [OK] を押して閉じ、ドメインへの参加確認メッセージが表示されたら、それに応答します。

6. あとは Active Directory サーバの Windows 管理者のユーザ名と、パスワードを入力して [OK] を押します。
これでお使いのサーバは、認証データを Active Directory ドメインコントローラから取得できるようになっています。

それ以外にも、`realmd` ツールを利用して Active Directory に参加することもできます。詳しくは 21.7.1項「Active Directory を管理するための `realmd` の使用」をお読みください。

ヒント: 識別情報のマッピング

Samba サーバが複数台存在するような環境では、UID と GID がそれぞれ別々に管理されることになります。ユーザに対する UID の割り当ては、初回のログイン時に割り当てられることから、サーバ間では同じユーザ名でも UID が異なる結果になってしまいます。この問題を解決するには、識別情報のマッピング設定を行う必要があります。詳しくは https://www.samba.org/samba/docs/man/Samba-HOWTO-Collection/idmapper.html をお読みください。

21.7.1 Active Directory を管理するための `realmd` の使用

`realmd` はネットワーク認証やドメインメンバーシップを設定するための DBus サービスです。

21.7.1.1 Active Directory ドメインの検出

`realmd` は DNS SRV レコードを読み込んで設定／使用可能なドメインもしくはレルムの一覧を検出します。まずは設定したい Active Directory のドメインに対して、DNS SRV レコードが設定されていることを確認します。たとえば domain.example.com というドメイン名であれば、下記のような名前の SRV レコードになります:

```
_ldap._tcp.dc._msdcs.domain.example.com.
```

この DNS レコードは Active Directory 付属の DNS サーバで自動的に作成されるものです。

特定のドメイン名を検出するには、下記のようなコマンドを入力して実行します:

```
> sudo realm discover --verbose domain.example.com

* Resolving: _ldap._tcp.dc._msdcs.domain.example.com
 * Sending MS-CLDAP ping to: 192.168.20.10
 * Sending MS-CLDAP ping to: 192.168.12.12
 * Successfully discovered: domain.example.com
...
```

Active Directory のドメインに参加するには、下記のようなコマンドを入力して実行します:

```
> sudo realm join --verbose domain.example.com
```

Active Directory のドメインに参加が完了したあとは、ドメインアカウントに対してログインの許可を設定するだけです。具体的には下記のように入力して実行します:

```
> sudo realm permit --realm domain.example.com --all
```

特定のユーザに対してのみログインを許可したい場合は、下記のようなコマンドを入力して実行します:

```
> sudo realm permit --realm domain.example.com DOMAIN\\ユーザ名 DOMAIN\\ユーザ名
```

すべてのドメインアカウントに対してログインを拒否するように設定したい場合は、下記のようなコマンドを入力して実行します:

```
> sudo realm deny --realm domain.example.com --all
```

21.8 高度なトピック

本章では、Samba スイートのクライアントとサーバにおける、さらに高度な技術に関する説明を行っています。

21.8.1 systemd を利用した CIFS ファイルシステムの自動マウント

`systemd` を利用することで、CIFS 共有を起動時にマウントすることができます。これを行うには、下記の手順を実施します:

1. まずはマウントポイントを作成します:

   ```
   > mkdir -p マウントポイント
   ```

   以降は `マウントポイント` に `/cifs/shared` を指定したものとします。

2. 次に `systemd` のユニットファイルを作成します。このときユニットファイルのファイル名には、マウントポイントの "/" を "-" に置き換えたものを使用します。たとえば下記のようになります:

   ```
   > sudo touch /etc/systemd/system/cifs-shared.mount
   ```

   上記で作成したファイルには、下記の内容を記述します:

   ```
   [Unit]
   Description=CIFS share from The-Server

   [Mount]
   What=//The-Server/Shared-Folder
   Where=/cifs/shared
   Type=cifs
   Options=rw,username=vagrant,password=admin

   [Install]
   WantedBy=multi-user.target
   ```

3. サービスを有効化します:

   ```
   > sudo systemctl enable cifs-shared.mount
   ```

4. サービスを開始します:

   ```
   > sudo systemctl start cifs-shared.mount
   ```

   サービスが正しく開始できたかどうかを確認するには、下記のコマンドを実行します:

   ```
   > sudo systemctl status cifs-shared.mount
   ```

5. CIFS で正しくマウントできたかどうかを確認するには、下記のように実行します:

   ```
   >  cd /cifs/shared
   > ls -l
   ```

```
total 0
-rwxrwxrwx. 1 root    root    0 10月 24 22:31 hello-world-cifs.txt
drwxrwxrwx. 2 root    root    0 10月 24 22:31 subfolder
-rw-r--r--. 1 vagrant vagrant 0 10月 28 21:51 testfile.txt
```

21.8.2 btrfs による透過型ファイル圧縮

Samba では、btrfs ファイルシステム内に存在する共有に対して、クライアント側からファイルやディレクトリの圧縮フラグを遠隔で操作できるようになっています。Windows エクスプローラでは、［ファイル］›［プロパティ］›［詳細設定］内で、透過圧縮の設定を行うことができます:

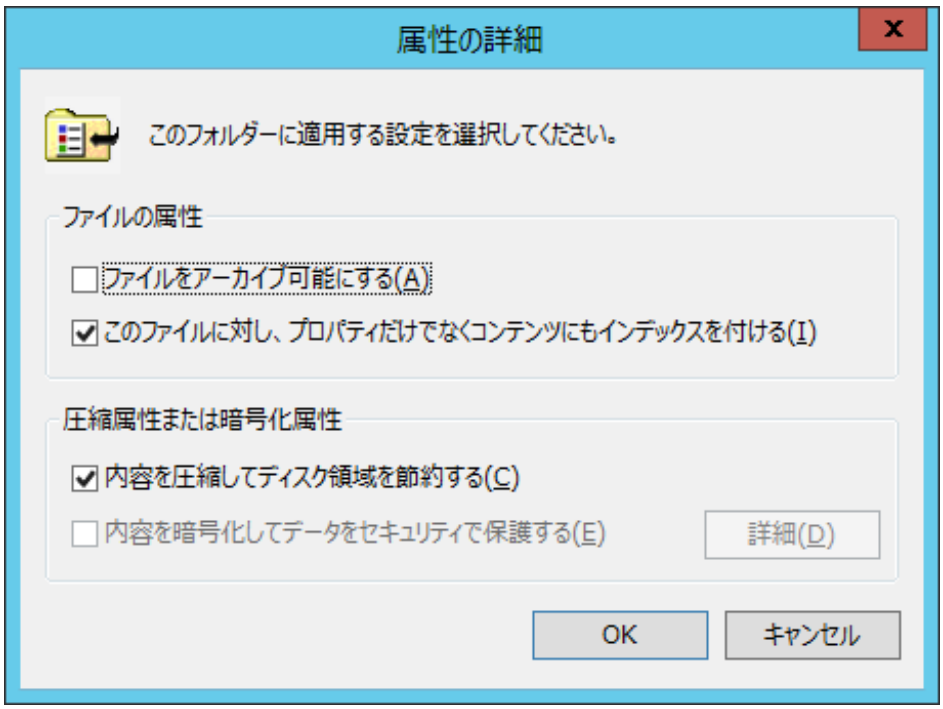


図 21.2: WINDOWS エクスプローラの［属性の詳細］ダイアログ

圧縮フラグの設定されたファイルは、そのファイルに対してアクセスや修正があるごとに、透過的に圧縮もしくは展開されます。これは通常ストレージの容量を削減する効果がありますが、ファイルにアクセスする際に CPU の負荷がかかります。また、新しいファイルやディレクトリは、FILE_NO_COMPRESSION を指定して作成していない限り、親ディレクトリの圧縮フラグを引き継ぐようになっています。

Windows エクスプローラでは、圧縮されたファイルと圧縮されていないファイルを、視覚的に区別できる形で表示します:

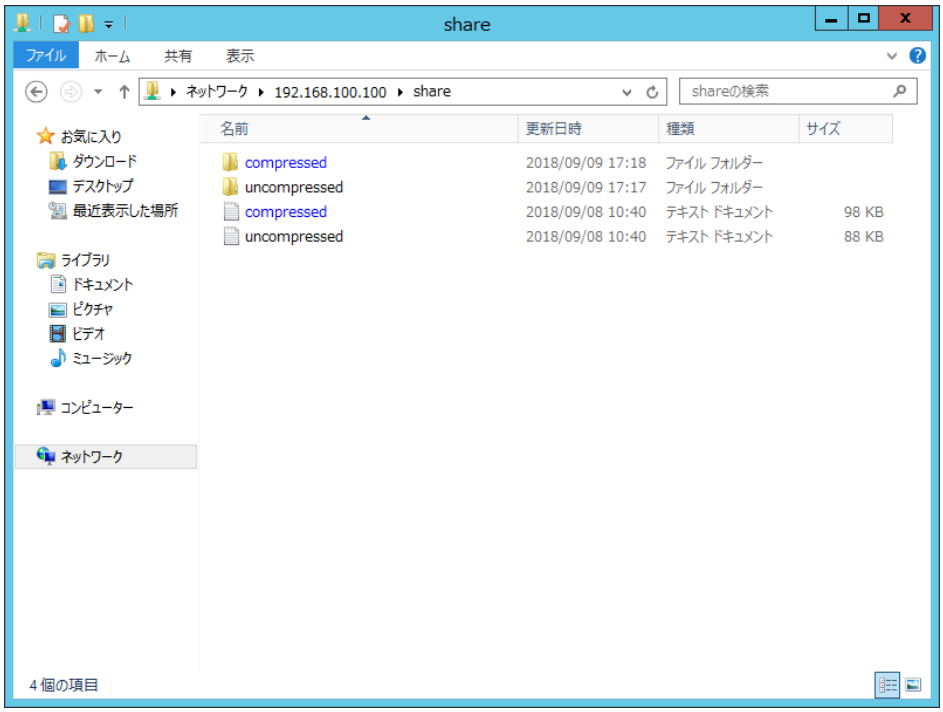


図 21.3: 圧縮されたファイルに対する WINDOWS エクスプローラのディレクトリー覧表示

Samba 共有での圧縮機能を有効化するには、

```
vfs objects = btrfs
```

を `/etc/samba/smb.conf` 内の共有設定内に追加するか、もしくは YaST から［ネットワークサービス］›［Samba サーバ］›［Add］を選択して、［btrfs の機能を使用する］にチェックを入れてもかまいません。

21.8.3 スナップショット

スナップショットはシャドウコピーとも呼ばれ、特定の時点におけるファイルシステムのサブボリュームのコピーを意味します。Linux では、Snapper を利用することで、これらのスナップショットを管理することができます。スナップショットは btrfs ファイルシステムのほか、シン・プロビジョン型の LVM ボリュームでも利用することができます。Samba スイートでは、サーバとクライアントとの間で FSRVP プロトコルを利用することで、リモートのスナップショットを管理することができます。

21.8.3.1 以前のバージョン

Samba サーバにおけるスナップショットは、リモートの Windows クライアントから見ると、ファイルやディレクトリの「以前のバージョン」として見えるようになっています。

Samba サーバでスナップショット機能を有効化するには、下記の条件を満たさなければなりません:

- SMB のネットワーク共有が、btrfs のサブボリューム内に存在しなければなりません。
- SMB ネットワーク共有のパスに対して、対応する Snapper の設定ファイルを用意する必要があります。Snapper の設定ファイルを作成するには、下記のように実行します:

  ```
  > sudo snapper -c <設定名> create-config /path/to/share
  ```

  Snapper について、詳しくは 第3章「Snapper によるシステムの復元とスナップショット管理」をお読みください。
- スナップショットのディレクトリツリーは、対応するユーザに対してアクセスを許可しなければなりません。詳しくは vfs_snapper のマニュアルページ内にある PERMISSIONS セクション (`man 8 vfs_snapper`) をお読みください。

リモートからのスナップショット機能に対応するには、`/etc/samba/smb.conf` ファイルを修正する必要があります。これは［YaST］›［ネットワークサービス］›［Samba サーバ］から行うことができるほか、手作業でも設定することができます。具体的には、対応する共有に対して、下記の設定を追加します:

```
vfs objects = snapper
```

なお、`smb.conf` を手作業で修正した場合は、Samba サービスを再起動する必要があります:

```
> sudo systemctl restart nmb smb
```

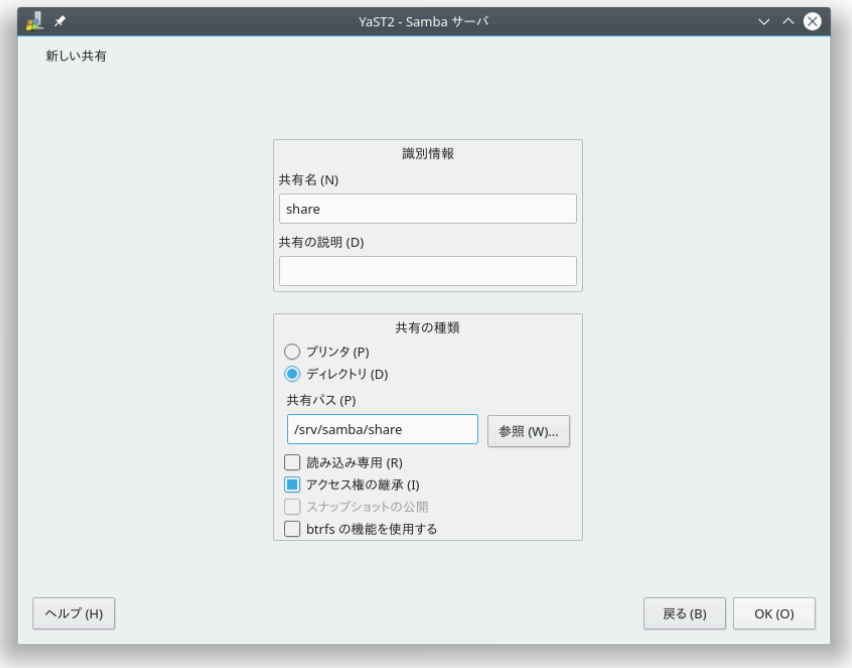


図 21.4: スナップショットを有効化した新しい SAMBA 共有の追加

設定作業を行うことで、Windows エクスプローラのファイルやディレクトリの［以前のバージョン］タブ内に、Snapper が作成したスナップショットが表示されるようになります。

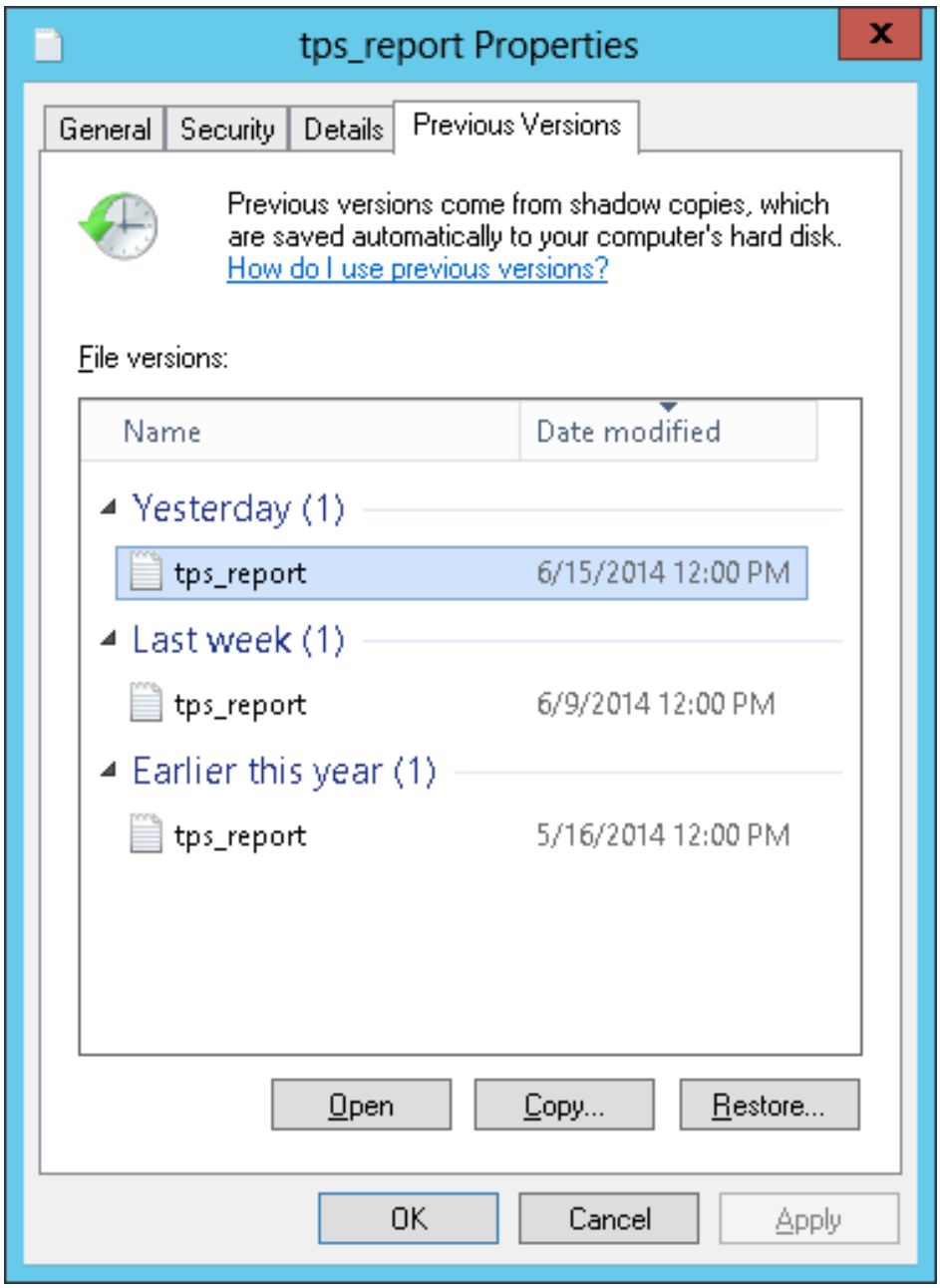


図 21.5: WINDOWS エクスプローラでの［以前のバージョン］タブ

21.8.3.2 リモート共有のスナップショット

既定でのスナップショットは、Samba サーバ内でコマンドラインユーティリティを使用するか、もしくは Snapper のタイムライン機能を利用することでのみ、作成もしくは削除することができます。

Samba では、ファイルサーバリモート VSS プロトコル (File Server Remote VSS Protocol; VSRVP) を利用することで、リモートのホストから共有に対するスナップショットを作成したり、削除したりできるようにすることができます。

21.8.3.1項「以前のバージョン」に書かれている設定と事前要件に加えて、`/etc/samba/smb.conf` 内に下記のグローバル設定を追加する必要があります:

```
[global]
rpc_daemon:fssd = fork
registry shares = yes
include = registry
```

これにより、Samba の `rpcclient` や Windows Server 2012 に付属する `DiskShadow.exe` などの FSRVP クライアントから、Samba の特定の共有に対してスナップショットを作成することができるほか、スナップショットの削除や新しい共有としての公開などを行うことができるようになります。

21.8.3.3 `rpcclient` による Linux からのリモートスナップショット管理

`samba-client` パッケージには、リモートの Windows Server や Samba サーバに対して、対応する共有のスナップショットを作成したり公開したりするためのリクエストを送信することができる、FSRVP クライアントが含まれています。共有のマウントやバックアップなどは、openSUSE Leap に付属する既存のツールを利用することができます。また、サーバへのリクエストは `rpcclient` バイナリを利用して行います。

例 21.4: `rpcclient` による WINDOWS SERVER 2012 共有に対するスナップショット要求

> `EXAMPLE` ドメイン内にある `win-server.example.com` サーバに対して、Administrator で接続する場合の例です:
>
> ```
> # rpcclient -U 'EXAMPLE\Administrator' ncacn_np:win-server.example.com[ndr64,sign]
> Enter EXAMPLE/Administrator's password:
> ```
>
> `rpcclient` で SMB の共有が見えるかどうかを確認します:
>
> ```
> # rpcclient $> netshareenum
> netname: windows_server_2012_share
> remark:
> path:   C:\Shares\windows_server_2012_share
> password:       (null)
> ```
>
> SMB 共有がスナップショットの作成に対応しているかどうかを確認します:
>
> ```
> # rpcclient $> fss_is_path_sup windows_server_2012_share \
> UNC \\WIN-SERVER\windows_server_2012_share\ supports shadow copy requests
> ```
>
> スナップショット共有の作成を要求します:
>
> ```
> # rpcclient $> fss_create_expose backup ro windows_server_2012_share
> ```

```
13fe880e-e232-493d-87e9-402f21019fb6: shadow-copy set created
13fe880e-e232-493d-87e9-402f21019fb6(1c26544e-8251-445f-be89-d1e0a3938777): \
\\WIN-SERVER\windows_server_2012_share\ shadow-copy added to set
13fe880e-e232-493d-87e9-402f21019fb6: prepare completed in 0 secs
13fe880e-e232-493d-87e9-402f21019fb6: commit completed in 1 secs
13fe880e-e232-493d-87e9-402f21019fb6(1c26544e-8251-445f-be89-d1e0a3938777): \
share windows_server_2012_share@{1C26544E-8251-445F-BE89-D1E0A3938777} \
exposed as a snapshot of \\WIN-SERVER\windows_server_2012_share\
```

サーバ側でスナップショット共有が公開されているかどうかを確認します:

```
# rpcclient $> netshareenum
netname: windows_server_2012_share
remark:
path:   C:\Shares\windows_server_2012_share
password:       (null)

netname: windows_server_2012_share@{1C26544E-8251-445F-BE89-D1E0A3938777}
remark: (null)
path:   \\?\GLOBALROOT\Device\HarddiskVolumeShadowCopy{F6E6507E-F537-11E3-9404-
B8AC6F927453}\Shares\windows_server_2012_share\
password:       (null)
```

スナップショット共有を削除します:

```
# rpcclient $> fss_delete windows_server_2012_share \
13fe880e-e232-493d-87e9-402f21019fb6 1c26544e-8251-445f-be89-d1e0a3938777
13fe880e-e232-493d-87e9-402f21019fb6(1c26544e-8251-445f-be89-d1e0a3938777): \
\\WIN-SERVER\windows_server_2012_share\ shadow-copy deleted
```

サーバ側でスナップショット共有が削除されていることを確認します:

```
# rpcclient $> netshareenum
netname: windows_server_2012_share
remark:
path:   C:\Shares\windows_server_2012_share
password:       (null)
```

21.8.3.4 `DiskShadow.exe` による Windows からのリモートスナップショット管理

Windows 側からも、Linux Samba サーバ内の SMB 共有のスナップショットを管理することができます。Windows Server 2012 には `DiskShadow.exe` ユーティリティが用意されていますので、21.8.3.3項「`rpcclient` による Linux からのリモートスナップショット管理」で説明している `rpcclient` コマンドと同様に、リモートの共有に対するスナップショットを管理することができます。なお、Samba サーバ側では、あらかじめ注意して設定を行っておく必要があります。

下記は Samba サーバの共有のスナップショットを、Windows Server のクライアント側から管理できるようにするための手順例です。なお、下記の例では `EXAMPLE` が Active Directory ドメインを、`fsrvp-server.example.com` が Samba サーバのホスト名を、`/srv/smb` が SMB 共有のパスをそれぞれ表しています。

手順 21.1: より細かい SAMBA サーバの設定

1. YaST を利用して Active Directory ドメインに参加します。詳しくは 21.7項「Active Directory ネットワーク環境での Samba サーバ」をお読みください。

2. Active Directory の DNS 項目が正しく設定されていることを確認します:

```
fsrvp-server:~ # net -U 'Administrator' ads dns register \
fsrvp-server.example.com <IP address>
Successfully registered hostname with DNS
```

3. `/srv/smb` 内に btrfs のサブボリュームを作成します:

```
fsrvp-server:~ # btrfs subvolume create /srv/smb
```

4. `/srv/smb` に対する Snapper の設定ファイルを作成します:

```
fsrvp-server:~ # snapper -c <snapper_config> create-config /srv/smb
```

5. YaST で `/srv/smb` に対する新しい共有を作成します。このとき、［スナップショットの公開］のチェックボックスにチェックが入っていることをご確認ください。また、21.8.3.2項「リモート共有のスナップショット」で説明しているとおり、`/etc/samba/smb.conf` のグローバルセクションに下記の内容が追加されていることも、あわせてご確認ください:

```
[global]
 rpc_daemon:fssd = fork
 registry shares = yes
 include = registry
```

6. あとは `systemctl restart nmb smb` を実行して Samba を再起動します。

7. 続いて Snapper のアクセス権を調整します:

```
fsrvp-server:~ # snapper -c <snapper_config> set-config \
ALLOW_USERS="EXAMPLE\\\\Administrator EXAMPLE\\\\win-client$"
```

   さらに `.snapshots` サブディレクトリに対して、ALLOW_USERS で許可されるユーザからのアクセスを許すように設定します:

```
fsrvp-server:~ # snapper -c <snapper_config> set-config SYNC_ACL=yes
```

重要: パスのエスケープ処理について

'¥' のエスケープ処理にご注意ください。`/etc/snapper/configs/<snapper_config>` 内でエスケープ処理を行うため、コマンドの実行時には 2 回エスケープ処理が必要となります。

"EXAMPLE¥win-client$" は Windows クライアントのコンピュータアカウントを表しています。Windows 側では、最初の FSRVP リクエストを、このアカウントで認証して行います。

8. Windows クライアントのアカウントに対して、必要な権限を許可します:

```
fsrvp-server:~ # net -U 'Administrator' rpc rights grant \
"EXAMPLE\\win-client$" SeBackupPrivilege
Successfully granted rights.
```

上記のコマンドは "EXAMPLE¥Administrator" に対して実行する必要はありません。既に権限が許可されているためです。

手順 21.2: WINDOWS クライアントの設定と `DiskShadow.exe` の実行

1. Windows Server 2012 を起動します (この例では WIN-CLIENT というホスト名とします) 。
2. openSUSE Leap と同じ Active Directory ドメイン (EXAMPLE) に参加します。
3. 再起動します。
4. PowerShell を起動します。
5. `DiskShadow.exe` を起動して、バックアップ手順を開始します:

```
PS C:\Users\Administrator.EXAMPLE> diskshadow.exe
Microsoft DiskShadow バージョン 1.0
Copyright (C) 2012 Microsoft Corporation
On computer:  WIN-CLIENT,  6/17/2014 3:53:54 PM

DISKSHADOW> begin backup
```

6. プログラム終了後やリセット／再起動後にも、シャドウコピーが保持されるように設定します:

```
DISKSHADOW> set context PERSISTENT
```

7. 指定した共有でスナップショットが有効化されているかどうかを調べ、有効化されていればスナップショットを作成します:

```
DISKSHADOW> add volume \\fsrvp-server\sles_snapper
```

```
DISKSHADOW> create
シャドウ ID {de4ddca4-4978-4805-8776-cdf82d190a4a} のエイリアス VSS_SHADOW_1 は環境変数と
して設定されています。
シャドウ セット ID {c58e1452-c554-400e-a266-d11d5c837cb1} のエイリアス VSS_SHADOW_SET は
環境変数として設定されています。

シャドウ コピー セット ID {c58e1452-c554-400e-a266-d11d5c837cb1} を使用してすべてのシャドウ
 コピーを照会しています

 * シャドウ コピー ID = {de4ddca4-4978-4805-8776-cdf82d190a4a}     %VSS_SHADOW_1%
    - シャドウ コピー セット: {c58e1452-c554-400e-a266-d11d5c837cb1}  %VSS_SHADOW_SET%
    - シャドウ コピーのオリジナル カウント数 = 1
    - 元のボリューム名: \\FSRVP-SERVER\SLES_SNAPPER\ \
      [ボリュームはこのマシンにありません]
    - 作成時間: 6/17/2014 3:54:43 PM
    - シャドウ コピー デバイス名:
      \\FSRVP-SERVER\SLES_SNAPPER@{31afd84a-44a7-41be-b9b0-751898756faa}
    - 作成元のコンピューター: FSRVP-SERVER
    - サービス コンピューター: win-client.example.com
    - 露出されていません
    - プロバイダー ID: {89300202-3cec-4981-9171-19f59559e0f2}
    - 属性:  No_Auto_Release Persistent FileShare

一覧表示したシャドウ コピーの数: 1
```

8. 最後にバックアップを完了します:

```
DISKSHADOW> end backup
```

9. スナップショットを作成したら、削除を行ったあと、削除ができたことを確認します:

```
DISKSHADOW> delete shadows volume \\FSRVP-SERVER\SLES_SNAPPER\
プロバイダー {89300202-3cec-4981-9171-19f59559e0f2} からボリューム \\FSRVP-SERVER
\SLES_SNAPPER\ のシャドウ コピー {de4ddca4-4978-4805-8776-cdf82d190a4a} を削除しています
 [Attributes: 0x04000009]...

削除したシャドウ コピーの数: 1

DISKSHADOW> list shadows all

コンピューター上のすべてのシャドウ コピーを照会しています...
システム内にシャドウ コピーが見つかりませんでした。
```

21.9 さらなる情報

- **マニュアルページ:** `samba` パッケージでインストールされるマニュアルページの一覧を表示するには、`apropos samba` と入力して実行します。それぞれのマニュアルページを表示するには、`man マニュアルページの名前` のように入力して実行します。
- **SUSE 固有の README ファイル:** `samba-client` パッケージには、`/usr/share/doc/packages/samba/README.SUSE` (英語) というファイルが含まれています。
- **追加のパッケージドキュメンテーション:** `samba-doc` パッケージを `zypper install samba-doc` と入力して実行し、インストールしてください。
  このドキュメンテーションパッケージは、ファイルを `/usr/share/doc/packages/samba` にインストールします。ここにはマニュアルページの HTML 版のほか、設定例集なども含まれています (例: `smb.conf.SUSE`) 。
- **オンラインのドキュメンテーション:** Samba wiki にはさまざまな User Documentation (ユーザ向けドキュメンテーション) (英語) が含まれています。https://wiki.samba.org/index.php/User_Documentation からアクセスしてください。

22 NFS によるファイル共有

改訂履歴

2025-05-27

Network File System (NFS) とは、サーバ内のファイルにアクセスするためのプロトコルで、これを利用することによってローカルのファイルと同じようなやり方でアクセスすることができるようになる仕組みです。

openSUSE Leap では、スパースファイルやファイルの事前割り当て、サーバ側での複製やコピー、アロケーションデータブロック (ADB) や強制アクセス制御 (MAC) のためのラベル型 NFS (この機能を使用する場合、クライアントとサーバの両方で MAC が必要となります) 等に対応した NFS v4.2 をインストールします。

22.1 概要

Network File System (NFS) はコンピュータ間でのファイル共有を行うためのネットワークプロトコルで、標準化されているだけでなく、十分に実証された幅広いサポートが特長のプロトコルです。

ネットワーク内でのユーザ管理を共通化し、中央でとりまとめる目的で、Network Information Service (NIS) を併用する場合があります。NFS と NIS を併用することで、ネットワーク内でのファイルやディレクトリに対するアクセス制御を集中管理することができるようになります。また、ユーザにとっても NFS と NIS を利用することで、ネットワークの資源とローカルの資源を完全に透過的に扱うことができるようになります。

既定の設定では、NFS はネットワーク側を信用する設計になっていることから、NFS サーバを有効化するにあたっては、信頼できるネットワークに接続するようにしてください。具体的には、NFS サーバが接続するネットワーク内の全てのコンピュータで管理者権限が適切に管理され、かつ物理的なアクセスに対しても保護があることをお確かめください。

また、NFS サーバが接続するネットワークをプライベートなものとするほか、単一のキャビネット内やマシンルーム内に配置して、部外者による不正なアクセスができないよう、十分なセキュリティを確保してください。場合によっては、サブネット全体についても十分な制限が必要となるなど、きめ細かく信頼を構築する必要があることもあります。このような場合の要件に適合させるため、NFS では Kerberos インフラストラクチャなどにも対応しています。なお、Kerberos を使用するには NFSv4 が必要となりますが、既定で有効化されています。詳しくは『セキュリティ強化ガイド』、第6章「Kerberos を利用したネットワーク認証」をお読みください。

YaST モジュールでは、下記の用語を使用します。

エクスポート

NFS サーバが 公開する ディレクトリを意味します。これにより、クライアントは自身のシステム内に、このディレクトリ以下の内容を取り込むことができるようになります。

NFS クライアント

NFS クライアントは、NFS (Network File System) プロトコルを介して、NFS サーバが提供する NFS サービスに接続するシステムを指します。Linux カーネルには TCP/IP プロトコルが統合されているため、クライアント側で追加のソフトウエアをインストールする必要はありません。

NFS サーバ

NFS サーバは、クライアントに対して NFS サービスを提供するシステムを意味します。サーバ側では下記のデーモンを動作させてサービスを提供しています: `nfsd` (ワーカー), `idmapd` (NFSv4 向けの ID と名前のマッピングシステム、特定の用途でのみ必要となります), `statd` (ファイルロック (施錠)), and `mountd` (マウント要求)

NFSv3

NFSv3 はバージョン 3 の実装を意味します。状態遷移機能のない、クライアント認証に対応した「古い」実装です。

NFSv4

NFSv4 はバージョン 4 の新しい実装で、Kerberos を介して機密を保持することのできるユーザ認証機能に対応しています。また、NFSv4 ではポートを 1 つだけしか使用しない設計になっているため、NFSv3 よりもファイアウオール環境には適した仕組みになっています。
プロトコルは https://datatracker.ietf.org/doc/rfc7531/ で規定されています。

pNFS

Parallel NFS の略で、NFSv4 のプロトコル拡張です。pNFS に対応したクライアントであれば、NFS サーバ内の任意のデータに直接アクセスすることができます。

重要: DNS 側の要件

原理上、全てのエクスポートは IP アドレスのみでアクセスすることができます。ただし、タイムアウトの問題を避けるために、DNS システムをご用意ください。具体的には、`mountd` デーモンがログの記録時に逆引き参照を行いますので、そのために必要となります。

22.2 NFS サーバのインストール

NFS サーバは、既定の設定ではインストールされません。NFS サーバをインストールするには、YaST を起動して［ソフトウエア］›［ソフトウエア管理］を選択し、［パターン］内にある［サーバ機能］の［ファイルサーバ］を選択してください。あとは［了解］を押して必要なパッケージをインストールしてください。

NIS と同様に、NFS もクライアント／サーバ型のシステムになっています。ですが、1 台のマシンでネットワークに対してファイルシステムを提供 (エクスポート) しながら、他のホストにあるファイルシステムをマウントする (インポート) することができます。

22.3 NFS サーバの設定

NFS サーバの設定は、YaST もしくは手作業で実施することができます。認証については、NFS と Kerberos を組み合わせることができます。

22.3.1 YaST を利用したファイルシステムの公開

YaST を利用することで、お使いのコンピュータを NFS サーバに仕立て上げることができます。NFS サーバでは、特定のコンピュータやグループ内の全メンバーに対して、ファイルやディレクトリをエクスポート (公開) することができます。そのため、NFS を利用することで、それぞれのホストにインストールすることなく、特定のアプリケーションを利用できるようにすることもできます。

このようなサーバを構築するには、下記の手順で行います:

手順 22.1: NFS サーバの設定

1. YaST を起動し、［ネットワークサービス］›［NFS サーバ］を選択します (図22.1「NFS サーバ設定ツール」をご覧ください) 。なお、必要に応じて、追加のソフトウエアをインストールするよう求められることがあります。

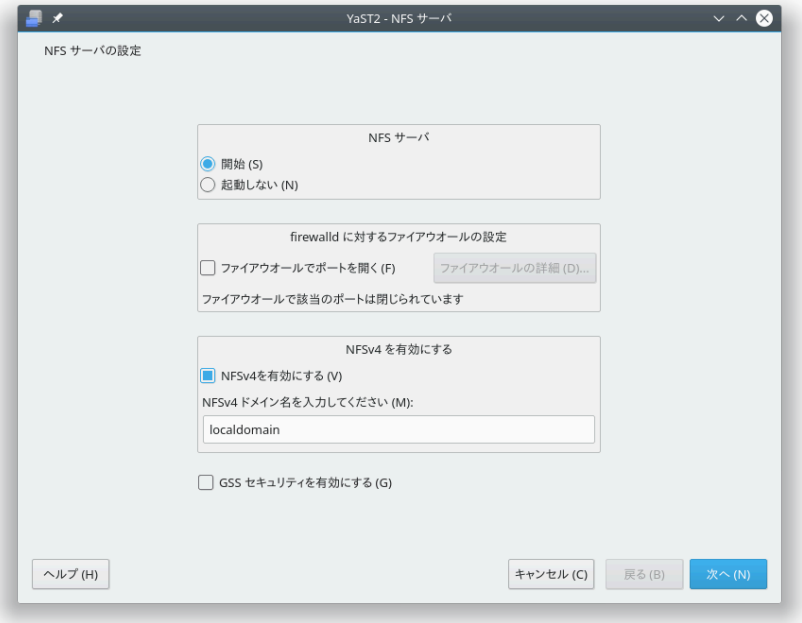


図 22.1: NFS サーバ設定ツール

2. まずは［開始］のラジオボタンを押します。

3. お使いのシステムで `firewalld` が動作している場合は、NFS 向けの設定を別途実施する必要があります (詳しくは 22.5項「ファイアウオールを介した NFS サーバと NFS クライアントの通信」をお読みください) 。現時点の YaST では `firewalld` への対応が完全ではありませんので、「ファイアウオールを設定できません」のメッセージを無視して続行してください。

4. また、必要であれば［NFSv4 を有効にする］をチェックしてください。NFSv4 を無効化した場合、YaST は NFSv3 のみに対応するようになります。なお、NFSv2 の有効化については、注記: NFSv2 をお読みください。

   - NFSv4 を有効化した場合は、追加で NFSv4 ドメイン名を入力します。このパラメータは `idmapd` デーモンが使用するパラメータで、Kerberos の設定時やクライアントが数字のユーザ名では処理できないような場合に利用します。`idmapd` を動作させない場合や、特に要件がない場合は、`localdomain` (既定値) のままでかまいません。`idmapd` について、詳しくは `/etc/idmapd.conf` をお読みください。

重要: NFSv4 ドメイン名について

ドメイン名は全ての NFSv4 クライアント側でも設定する必要があります。同じドメイン名を設定したクライアントでないと、アクセスが拒否されてしまうためです。なお、既定のドメイン名は、サーバ／クライアントとも `localdomain` になっています。

5. サーバに対してのアクセスをさらに保護したい場合は、［GSS セキュリティを有効にする］を選択します。この仕組みを使用するには、お使いのドメイン内に Kerberos がインストールされ、サーバとクライアントの両方で Kerberos が設定されている必要があります。設定が終わったら［次へ］を押すことで、次の設定ダイアログに進むことができます。

6. お使いのディレクトリをエクスポートするには、ダイアログの上半分で［ディレクトリの追加］を押します。

7. 許可するホストの設定を行っていない場合は、クライアントの情報を入力するためのポップアップが自動的に表示されます。ホストをワイルドカードで設定することができます (通常は既定の設定のままでかまいません) 。
   ホストのワイルドカードでは、下記の 4 種類の設定を行うことができます。1 種類目は単一のホスト名を指すためのホスト名や IP アドレスの設定、2 種類目はネットグループによる設定、3 種類目はワイルドカード (たとえば `*` を指定すると、全てのマシンからアクセスできるようになります) 、そして 4 種類目は IP ネットワークです。これらの設定について、詳しくは `exports` のマニュアルページをお読みください。
   btrfs の場合、セキュリティ上の理由から、`no_subtree_check` の既定値を `subtree_check` に変更しておくことを強くお勧めします。これは、同じ btrfs ファイルシステムに対して異なるセキュリティ設定のエクスポートが存在する場合、一方のエクスポートに対してアクセス権を持つクライアントは、残り全てのエクスポートに対してもアクセス権を持つことになってしまうためです。これは、前述の設定変更を行うことで防ぐことができます。なお、それぞれのエクスポートを異なるファイルシステムに分ける方式でもかまいません。

8. 最後に［完了］を押すと、設定を完了することができます。

22.3.2 手作業によるファイルシステムの公開

NFS のエクスポートサービスで使用される設定ファイルは、`/etc/exports` と `/etc/sysconfig/nfs` の 2 種類です。これらのファイルに加えて、Kerberos が有効化された NFSv4 サーバの設定や、クライアントが数字のユーザ名を扱うことができない場合は、`/etc/idmapd.conf` も必要となります。

サービスを開始もしくは再起動するには、`systemctl restart nfs-server` を実行します。これにより、NFS サーバが必要とする RPC port mapper も再起動することができます。

NFS サーバをシステムの起動時に開始するようにするには。`sudo systemctl enable nfs-server` を実行します。

注記: NFSv4

NFSv4 は openSUSE Leap で利用できる最新バージョンの NFS プロトコルです。現在の NFSv4 でディレクトリをエクスポートする際の設定は、NFSv3 と同じです。

Leap 以前の openSUSE では `/etc/exports` 内での bind マウントが必須でした。現在もなお、この構成に対応してはいますが、廃止予定になっています。

`/etc/exports`

`/etc/exports` ファイルには、エクスポートするディレクトリの設定一覧が含まれています。各行には共有するディレクトリのほか、共有方法の指定も含まれています。`/etc/exports` では、下記のような書式で各行を設定します:

```
/共有/ディレクトリ　ホスト(オプションリスト)
```

たとえば、下記のようになります:

```
/nfs_exports/public *(rw,sync,root_squash,wdelay)
/nfs_exports/dept1 *.dept1.example.com(rw,sync,root_squash,wdelay)
/nfs_exports/team1 192.168.1.0/24(rw,sync,root_squash,wdelay)
/nfs_exports/tux 192.168.1.2(rw,sync,root_squash)
```

上記の例では、`ホスト` の欄にそれぞれ下記を指定しています:

- `*` : ネットワーク内の全てのクライアントに対して共有するよう指定しています
- `*.dept1.example.com` : *.dept1.example.com のドメインに属するクライアントにのみ共有するよう指定しています
- `192.168.1.0/24` : 192.168.1.0/24 の IP アドレス範囲内にのみ共有するよう指定しています
- `192.168.1.2` : 192.168.1.2 の IP アドレスを持つマシンにのみ共有するよう指定しています

btrfs ファイルシステムに対して共有を作成しようとしている場合は、セキュリティ上の理由から、それぞれのエクスポートに対して `subtree_check` を設定しておくことを強くお勧めします。

上記の例に加えて、`/etc/netgroup` ファイルで定義できるネットグループを利用した制限を設定 (`@my-hosts`) することもできます。指定可能な全てのオプションとその意味について、詳しくは `/etc/exports` のマニュアルページ (`man exports`) をお読みください。
なお、NFS サーバが動作している間に `/etc/exports` ファイルを変更した場合は、`sudo systemctl restart nfs-server` を実行して、変更内容を反映させるために再起動を行う必要があります。

`/etc/sysconfig/nfs`

`/etc/sysconfig/nfs` ファイルには、NFSv4 サーバデーモンの動作に関わるいくつかのパラメータが含まれています。特に `NFS4_SUPPORT` を `yes` (既定値) に設定しておくことが重要です。`NFS4_SUPPORT` は NFS サーバが NFSv4 のエクスポートとクライアントに対応するかどうかを決定するパラメータです。
なお、NFS サーバが動作している間に `/etc/sysconfig/nfs` ファイルを変更した場合は、`sudo systemctl restart nfs-server` を実行して、変更内容を反映させるために再起動を行う必要があります。

ヒント: マウントオプション

Leap 以前の openSUSE では、`/etc/exports` 内で `--bind` マウントのオプションが必須でした。現在もなお、この構成に対応してはいますが、廃止予定になっています。また、現在の NFSv4 でディレクトリをエクスポートする際の設定は、NFSv3 と同じです。

注記: NFSv2

NFS クライアント側で NFSv2 を必要としている場合は、サーバ側でも `/etc/sysconfig/nfs` に下記の設定を行う必要があります:

```
NFSD_OPTIONS="-V2"
MOUNTD_OPTIONS="-V2"
```

サービスを再起動したら、下記のコマンドを実行すると、バージョン 2 に対応しているかどうかがわかります:

```
> cat /proc/fs/nfsd/versions
+2 +3 +4 +4.1 +4.2
```

`/etc/idmapd.conf`

`idmapd` デーモンは、Kerberos 認証を使用している場合や、クライアント側で数字のユーザ名を扱うことができない場合にのみ必要となるデーモンです。Linux カーネル 2.6.39 以降では、Linux クライアントは数字のユーザ名にも対応しています。`idmapd` デーモンは、NFSv4 のサーバ宛のリクエストやクライアント宛の応答に対して、名前と ID とのマッピングを行います。必要であれば、`idmapd` を NFSv4 サーバ内で動作させる必要があります。クライアント側での名前と ID のマッピングは、`nfs-client` パッケージで提供されている `nfsidmap` が行います。

なお、ユーザ名と ID (uid) が、NFS でファイルを共有する可能性のある全てのマシン間で統一されていることをご確認ください。これは NIS や LDAP など、任意のドメイン認証の仕組みで実施することができます。

また、`/etc/idmapd.conf` 内の `Domain` パラメータは設定必須となります。この値は NFSv4 のクライアントとサーバの両方で同じ値になっていなければなりません。よく分からない場合は、サーバとクライアントの両方で、既定値である `localdomain` をお使いください。それ以外の値を設定する場合、通常はホストの FQDN 名からホスト名を抜いたものを指定します。設定ファイルの例は下記のようになります:

```
[General]
Verbosity = 0
Pipefs-Directory = /var/lib/nfs/rpc_pipefs
Domain = localdomain

[Mapping]
Nobody-User = nobody
Nobody-Group = nobody
```

`idmapd` デーモンを開始するには、`systemctl start nfs-idmapd` を実行します。デーモンが動作している間に `/etc/idmapd.conf` ファイルを変更した場合は、`systemctl restart nfs-idmapd` を実行して設定を適用してください。

詳しくは `idmapd` と `idmapd.conf` のマニュアルページをお読みください (`man idmapd` および `man idmapd.conf` で読むことができます) 。

22.3.3 Kerberos を併用する NFS

NFS で Kerberos: 認証を使用するには、Kerberos サーバの構築をしなければなりません。なお、YaST の NFS サーバモジュールでは Kerberos サーバの設定を行うことはできません。また、サーバとクライアントの双方を同一の Kerberos のレルム内に配置するため、NFS サーバを構築する前に下記の手順を実施してください。

手順 22.2: サーバの設定

1. まずは NFS サーバに対して、nfs/server.fqdn の Kerberos プリンシパル、もしくはサービスプリンシパル名が存在することを確認します。
2. 次にシステムの keytab 内に、nfs/server.fqdn のプリンシパルキーが存在することを確認します。
3. エクスポートしたディレクトリに対して Kerberos を有効化します。
4. `systemctl enable gssproxy.service` を実行して、gssproxy サービスを有効化します。
5. `systemctl start gssproxy.service` を実行して、gssproxy サービスを開始します。

手順 22.3: クライアントの設定

1. まずは NFS クライアントに対して、nfs/client.fqdn の Kerberos プリンシパル、もしくはサービスプリンシパル名 (SPN) が存在することを確認します。
2. 次にシステムの keytab 内に、nfs/client.fqdn のプリンシパルキーが存在することを確認します。
3. NFS マウントに対して Kerberos のセキュリティを有効化します。
4. `systemctl start rpc-gssd.service` を実行して、rpc-gssd サービスを開始します。

Kerberos 認証を使用するには、サーバ側で `idmapd` デーモンを動作させる必要もあります。詳しくは `/etc/idmapd.conf` をお読みください。

Kerberos が有効化された NFS について、さらに詳しくは 22.7項「さらなる情報」にあるリンク先をお読みください。

22.4 クライアントの設定

お使いのホストを NFS クライアントとして設定する場合は、追加のソフトウエアをインストールする必要はありません。必要な全てのパッケージが既定でインストールされるためです。

22.4.1 YaST を利用したファイルシステムの取り込み

認可済みのユーザであれば、YaST の NFS クライアントモジュールを利用することで、NFS サーバ内でエクスポートされたディレクトリをマウントし、ローカルのファイルツリーに組み込むことができます。具体的には、下記の手順で実施します:

手順 22.4: NFS ディレクトリのインポート

1. まずは YaST の NFS クライアントモジュールを起動します。
2. [NFS 共有] タブ内にある [追加] を押します。NFS サーバのホスト名と取り込むディレクトリ、そしてローカル側のマウント先ディレクトリをそれぞれ指定します。
3. NFSv4 を使用している場合は、[NFS 設定] タブ内にある [NFSv4 を有効にする] を選択します。これに加えて、[NFSv4 ドメイン名] に、NFSv4 サーバと同じ値を設定しなければなりません。既定のドメイン名は `localdomain` です。
4. NFS 向けに Kerberos 認証を使用するには、GSS セキュリティを有効化しなければなりません。この場合は、[GSS セキュリティを有効にする] を選択します。
5. お使いのシステムで `firewalld` が動作している場合は、NFS 向けの設定を別途実施する必要があります (詳しくは 22.5項「ファイアウォールを介した NFS サーバと NFS クライアントの通信」をお読みください) 。現時点の YaST では `firewalld` への対応が完全ではありませんので、「ファイアウォールを設定できません」のメッセージを無視して続行してください。
6. 最後に [OK] を押すと、設定を保存することができます。

設定した内容は `/etc/fstab` 内に書き込まれ、マウント処理が行われます。後から YaST の設定を起動した場合は、このファイル内にある既存の設定が読み込まれます。

ヒント: ルートファイルシステムとしての NFS について

ディスクレス (自分自身にはディスクが接続されていない) システムでは一般に、NFS の共有をネットワーク経由でマウントしてルートファイルシステムにすることがありますが、この場合は NFS 共有にアクセスするネットワークデバイスの設定に注意する必要があります。

システムをシャットダウンしたり再起動したりする場合、既定の処理順序では、ネットワークの接続を切ってからルートパーティションのマウントを解除します。ルートファイルシステムに NFS を使用している場合は、ルートファイルシステムのマウントを解除しようとしても、その時点では既にネットワークが切断されているため、正しく終了することができなくなってしまいます。この問題を解決するには、13.4.1.2.5項「ネットワークデバイスの有効化」で説明している手順でネットワークデバイスの設定タブを開いて、[デバイスの有効化] の値を [NFSroot] に設定してください。

22.4.2 手作業によるファイルシステムの取り込み

NFS サーバを動作させる場合、ファイルシステムを手動で取り込む際にあらかじめ設定しておくべきことは、RPC portmapper の有効化です。`nfs` サービスを開始すれば、RPC portmapper についても適切に起動が行われますので、`root` で `systemctl start nfs` を実行してください。あとは `mount` コマンドを利用して、ローカルのファイルシステムをマウントする場合と同様の手順を実施するだけです:

```
> sudo mount ホスト名:リモートのパス ローカルのパス
```

たとえば `nfs.example.com` というマシンのホームディレクトリをマウントしたい場合は、下記のように実行します:

```
> sudo mount nfs.example.com:/home /home
```

NFS クライアントがサーバに対して確立する TCP 接続数を設定するには、`mount` コマンドの `nconnect` オプションを使用します。設定値は 1 から 16 までの範囲で、オプションを指定しない場合の既定値は 1 です。

なお、`nconnect` の設定は、特定の NFS サーバに対する初めてのマウント処理にのみ適用されます。また、同じクライアント内で同一の NFS サーバに複数のマウントを行うと、既存の接続は共有され、新しい接続は行われません。そのため、`nconnect` の設定を変更したい場合は、あらかじめ特定の NFS サーバに対するマウントを すべて 解除しておく必要があります。これで `nconnect` の新しい設定値を適用できるようになります。

現在使用中の `nconnect` の設定値を確認したい場合は、`mount` コマンドの出力を参照するか、もしくは `/proc/mounts` ファイルをお読みください。マウントオプションに何も値が書かれていない場合は、マウント時にオプションを指定しておらず、既定値である 1 が使用されていることになります。

注記: nconnect で指定されているものとは異なる可能性がある件について

初回のマウント後にも接続を開いたり閉じたりすることができるため、実際の接続数は `nconnect` の設定値とは異なる場合があります。

22.4.2.1 automount サービスの使用

aufofs デーモンは、リモートのファイルシステムを自動的にマウントする際に使用するデーモンです。下記の内容を `/etc/auto.master` ファイル内に追加してください:

```
/nfsmounts /etc/auto.nfs
```

上記のように設定すると、`/nfsmounts` ディレクトリが NFS マウントのルートディレクトリとして設定され、`auto.nfs` の内容に応じて自動的にマウントされるようになります。`auto.nfs` はそのファイル名でなければならないわけではありません。要件に応じて自由に変更してかまいません。また、`auto.nfs` には、下記のようにして全ての NFS マウントの項目を記述することができます:

```
localdata -fstype=nfs server1:/data
nfs4mount -fstype=nfs4 server2:/
```

`root` で `systemctl start autofs` と実行すると、設定を反映させることができます。この例では、`/nfsmounts/localdata` ディレクトリが `server1` の `/data` ディレクトリを、`/nfsmounts/nfs4mount` が `server2` のルートディレクトリをそれぞれマウントするように設定しています。

autofs の動作中に `/etc/auto.master` を変更した場合は、変更内容を反映させるために automounter の再起動を行わなければなりません。`systemctl restart autofs` を実行して再起動してください。

22.4.2.2 /etc/fstab の手作業による編集

NFSv3 の場合、`/etc/fstab` ファイル内では下記のように設定します:

```
nfs.example.com:/data /local/path nfs rw,noauto 0 0
```

NFSv4 でマウントする場合は、3 列目の `nfs` を `nfs4` として設定します:

```
nfs.example.com:/data /local/pathv4 nfs4 rw,noauto 0 0
```

`noauto` オプションを指定することで、システムの起動時には自動的にマウントされないようにしています。このように設定している状況で、手作業でファイルシステムをマウントするには、下記のようにマウントポイントを指定するだけでかまいません:

```
> sudo mount /local/path
```

注記: 起動時点でのマウントについて

`noauto` オプションを指定しない場合、システムの起動時に動作する初期化スクリプトが自動的にマウント処理を行います。また、ネットワークが動作する前にマウント処理を実施しないようにするため、`_netdev` オプションを設定してもかまいません。

22.4.3 Parallel NFS (pNFS)

NFS は 1980 年代に開発された古いプロトコルです。そのため、NFS は小さなファイルを共有するには十分なプロトコルですが、巨大なファイルや多数のクライアントを扱うような環境では、NFS がボトルネックとなって、システムの性能に顕著な悪影響を及ぼしてしまいます。これは昔に比べてファイルが容易に巨大化しやすい環境になったためで、イーサネットの速度がそれに追随できていないことによるものです。

通常の NFS サーバに対してクライアントがファイルを要求すると、サーバはファイルのメタデータを読み込んだあと、全てのデータを収集してネットワーク経由でファイルを転送します。しかしながら、小さなファイルであっても大きなファイルであっても、性能のボトルネックが発生します:

- 小さなファイルの場合、かかる時間の多くはメタデータの収集に費やされています。
- 大きなファイルの場合、かかる時間の多くはサーバからクライアントへのファイル転送に費やされています。

pNFS (parallel NFS) では、このような制限を、ファイルシステムのメタデータとデータの位置をそれぞれ分離することで克服しています:

- 非データトラフィックを処理する メタデータ もしくは コントロールサーバ
- データを保持する 1 つもしくは複数の ストレージサーバ

メタデータとストレージサーバは単一の論理 NFS サーバを構成します。クライアントがファイルに対して読み込みまたは書き込みの要求を行うと、メタデータサーバが NFSv4 クライアントに対して、ファイルチャンクにアクセスする際に使用するストレージサーバを指定します。クライアントはサーバ内のデータに直接アクセスすることができます。

openSUSE Leap では、pNFS のクライアント側のみに対応しています。

22.4.3.1 YaST による pNFS クライアントの設定

基本的には 手順22.4「NFS ディレクトリのインポート」 に示されている手順に従って作業を行いますが、[pNFS (v4.2)] のチェックボックスにチェックを入れることと、必要であれば [NFSv4 共有] を選択する点が異なります。YaST でこれを行った場合、必要な手順全てを実施し、必要なオプション類を `/etc/exports` に書き込みます。

22.4.3.2 手作業による pNFS クライアントの設定

起動については 22.4.2項「手作業によるファイルシステムの取り込み」をお読みください。ほとんどの設定は NFSv4 サーバ側で行います。pNFS の場合、`mount` に指定する `nfsvers` オプションと、メタデータサーバを指定する `MDS_サーバ` が異なります。

```
> sudo mount -t nfs4 -o nfsvers=4.2 MDS_サーバ マウントポイント
```

デバッグ機能を有効にするには、下記の `/proc` ファイルシステム内の値を変更してください:

```
> sudo echo 32767 > /proc/sys/sunrpc/nfsd_debug
> sudo echo 32767 > /proc/sys/sunrpc/nfs_debug
```

22.5 ファイアウオールを介した NFS サーバと NFS クライアントの通信

NFS サーバと NFS クライアントとの通信は、Remote Procedure Calls (RPC) と言う仕組みを利用します。マウントデーモンやファイル施錠 (ロック) サービスなどの RPC サービスは、いずれも Linux の NFS 実装の一部として提供されているものです。サーバとクライアントの間にファイアウオールが存在する場合、これらのサービスとファイアウオールは、通信を妨害されることの無いように設定しておく必要があります。

NFS 4 サーバは NFS バージョン 3 との後方互換性がありますが、バージョンによってファイアウオールの設定内容が異なります。また、NFS 3 で共有をマウントするクライアントが存在する場合、ファイアウオールでは NFS 4 と NFS 3 の両方の設定を実施する必要があります。

22.5.1 NFS 4. x

NFS 4 では、サーバ側で TCP ポート 2049 を開くだけでかまいません。ファイアウオールでポートを開くには、NFS サーバ側の firewalld で、`nfs` サービスを有効化します:

```
> sudo firewall-cmd --permanent --add-service=nfs --zone=ゾーン名
firewall-cmd --reload
```

ここで、`ゾーン名` には NFS サーバ側で使用しているゾーン名を指定します。

NFSv4 の場合、クライアント側でのファイアウオール設定は不要です。特に指定を行わない限り、クライアント側は利用可能な最新の NFS バージョンを利用してマウントを実施しますので、クライアント側が NFSv4 に対応していれば、共有は自動的にバージョン 4.2 でマウントすることになります。

22.5.2 NFS 3

NFS 3 では下記のようなサービスが必要となります:

- `portmapper`
- `nfsd`
- `mountd`
- `lockd`
- `statd`

これらのサービスは `rpcbind` で制御され、既定では動的にポートが割り当てられます。ファイアウォール内に位置する NFS サーバの場合、まずは使用するサービスに対して固定のポートを使用するように設定する必要があります。ポートを固定したのち、ファイアウォールでポートを開いてください。

`portmapper`

openSUSE Leap の場合、`portmapper` は既定で固定のポートを使用するように設定されています。

ポート	111
プロトコル	TCP, UDP
動作	クライアント, サーバ

```
> sudo firewall-cmd --add-service=rpc-bind --permanent --zone=ゾーン名
```

`nfsd`

openSUSE Leap の場合、`nfsd` は既定で固定のポートを使用するように設定されています。

ポート	2049
プロトコル	TCP, UDP
動作	サーバ

```
> sudo firewall-cmd --add-service=nfs3 --permanent --zone=ゾーン名
```

`mountd`

openSUSE Leap の場合、`mountd` は既定で固定のポートを使用するように設定されています。

ポート	`20048`
プロトコル	TCP, UDP
動作	サーバ

```
> sudo firewall-cmd --add-service=mountd --permanent --zone=ゾーン名
```

`lockd`

`lockd` に対して固定のポートを使用するように設定するには、下記のようにします:

1. サーバ側の `/etc/sysconfig/nfs` ファイルの下記の箇所を編集します:

   ```
   LOCKD_TCPPORT=NNNNN
   LOCKD_UDPPORT=NNNN
   ```

   なお、`NNNNN` の箇所には未使用のポート番号を指定します。なお、両方のプロトコルで同じポートを使用するようにしてください。

2. あとは NFS サーバを再起動します:

   ```
   > sudo systemctl restart nfs-server
   ```

ポート	`NNNNN`
プロトコル	TCP, UDP
動作	クライアント, サーバ

```
> sudo firewall-cmd --add-port=NNNNN/{tcp,udp} --permanent --zone=ゾーン名
```

`statd`

`statd` に対して固定のポートを使用するように設定するには、下記のようにします:

1. サーバ側の `/etc/sysconfig/nfs` ファイルの下記の箇所を編集します:

   ```
   STATD_PORT=NNNNN
   ```

なお、NNNNN の箇所には未使用のポート番号を指定します。

2. あとは NFS サーバを再起動します:

```
> sudo systemctl restart nfs-server
```

ポート	NNNNN
プロトコル	TCP, UDP
動作	クライアント, サーバ

```
> sudo firewall-cmd --add-port=NNNNN/{tcp,udp} --permanent --zone=ゾーン名
```

重要: firewalld の設定再読み込みについて

firewalld の設定を変更したあとは、変更を反映させるために下記のコマンドを実行し、デーモンに再読み込みを行わせる必要があります:

```
> sudo firewall-cmd --reload
```

注記: ファイアウオールのゾーンについて

ゾーン名 の箇所には、マシン内で現在使用しているファイアウオールゾーンを指定してください。どのゾーンを使用しているのかは、マシンごとに異なる可能性があることに注意してください。

22.6 NFSv4 を介したアクセス制御リスト (ACL) の管理

Linux では、ユーザやグループ、その他 (ugo) に対して、それぞれ読み込みと書き込み、実行 (rwx) のフラグを設定するパーミッション機能が標準化されているものの、それより細かい制御を行うアクセス制御リスト (ACL) については、統一された標準が存在していないのが現状です。その標準の候補として存在するのが ドラフト POSIX ACL と NFSv4 ACL です。NFSv4 ACL は NFSv4 ネットワークファイルシステムの一部として存在するもので、Linux (POSIX) システムと Microsoft Windows (WIN32) システムとの妥当な互換性を提供しようとするものです。

NFSv4 ACL はドラフト POSIX ACL を正確に実装するには不十分な存在であるため、NFSv4 クライアントからのアクセスで ACL にマップするような機能は提供されていません (たとえば `setfacl` など) 。

NFSv4 を使用する場合、ドラフト POSIX ACL は擬似的にであっても使用することはできず、NFSv4 ACL を直接使用する必要があります。つまり、`setfacl` は NFSv3 に対しては動作するものの、NFSv4 に対しては動作しないことになります。NFSv4 ファイルシステムで NFSv4 ACL を使用するには、openSUSE Leap では `nfs4-acl-tools` というパッケージをインストールする必要があります。このパッケージには、下記のようなものが含まれています:

- `nfs4-getfacl`
- `nfs4-setfacl`
- `nfs4-editacl`

これらのコマンドは `getfacl` や `setfacl` に似た動作をするもので、いずれも NFSv4 ACL を調査したり修正したりする際に使用することができます。なお、これらのコマンドは NFSv4 ACL を完全サポートした NFS サーバのファイルシステムに対してのみ動作します。サーバ側で何らかの制限がある場合、これらのプログラムでアクセス制御項目 (ACE) を操作できない場合があります。

また、NFS の共有を公開しているサーバで直接 NFS ボリュームをマウントすることもできません。

追加の情報

詳しくは https://wiki.linux-nfs.org/wiki/index.php/ACLs#Introduction_to_NFSv4_ACLs (英語) にある Introduction to NFSv4 ACLs (NFSv4 ACL の紹介) をお読みください。

22.7 さらなる情報

`exports` , `nfs` , `mount` の各マニュアルページに加え、NFS サーバと NFS クライアントを設定するための情報が `/usr/share/doc/packages/nfsidmap/README` に用意されています。また、下記の Web サイトには、さらなるオンライン文書が用意されています:

- ネットワークセキュリティに関する一般的な情報については、**『セキュリティ強化ガイド』、第24章「マスカレードとファイアウォール」**をお読みください。
- NFS 共有を自動的にマウントする必要がある場合は、23.4項「NFS 共有の自動マウント」をお読みください。

- AutoYaST を利用して NFS を設定する方法については、『AutoYaST ガイド』、第4章「設定およびインストールのオプション」、4.20項「NFS クライアントおよびサーバ」をお読みください。
- Kerberos で NFS 共有の機密性を向上したい場合は、『セキュリティ強化ガイド』、第6章「Kerberos を利用したネットワーク認証」、6.6項「Kerberos と NFS」をお読みください。
- 詳細な技術文書をお読みになりたい場合は、SourceForge (https://nfs.sourceforge.net/) (英語) をお読みください。

22.8 NFS のトラブルシューティングにおける情報収集

22.8.1 一般的なトラブルシューティング

NFS で発生する問題は、表示されるエラーメッセージのほか、`/var/log/messages` に記録されるログメッセージを読むことで原因を探り当てることができる場合があります。ですが、表示されるメッセージや `/var/log/messages` 内のログメッセージだけでは不十分で、原因を探り当てられないことも多くあります。このような場合、問題を再現させてネットワークのパケットを採取することで、原因に辿り着きやすくすることができます。

まずは問題点を明確にしてください。様々な方法でシステムのテストを実施して問題を再現させ、どこで問題が発生しているのかを明確にしましょう。あとは問題を切り分けていって、最も簡単な再現手順を作り上げてから、下記に示す再現手順を実施してください。

手順 22.5: 問題の再現

1. まずはネットワークパケットの採取を行います。Linux では `tcpdump` コマンド (パッケージ名 `tcpdump`) を使用することができます。
   `tcpdump` コマンドは、たとえば下記のように入力して実行します:

   ```
   tcpdump -s0 -i eth0 -w /tmp/nfs-demo.cap host x.x.x.x
   ```

   それぞれのオプションの意味は下記のとおりです:

   s0
   : ネットワークパケットの採取時、長いパケットを省略せずに採取するようにしています。

   eth0
   : こちらは実際のローカルインターフェイス名に置き換えてください。ここに `any` を指定すると、全てのインターフェイスで送受信されるパケットを採取できますが、不要なデータが多数含まれることになり、分析の際の邪魔になってしまうため、あまりお勧めしません。

w
: 採取したパケットの保存先ファイル名を指定します。

x.x.x.x
: NFS 接続の対向側 IP アドレスを指定します。たとえば NFS のクライアント側で `tcpdump` を実行してパケットを採取する場合、ここに指定するのは NFS サーバとなります。

注記

場合によっては、NFS クライアントもしくは NFS サーバのいずれかでパケットを採取すれば原因が判明することもありますが、ネットワークの通信状況が原因のうちの 1 つと疑われる場合は、クライアントとサーバの両方で採取することをお勧めします。

起動した `tcpdump` コマンドは動作させたままの状態で、次の手順に進んでください。

2. (必要であれば) 直面している問題が `nfs mount` の実行時に発生しているものである場合は、`nfs mount` コマンドに対して冗長出力を指定するオプション (`-vvv`) を追加して試してみてもかまいません。

3. (必要であれば) 問題の再現時に `strace` を取得する方法もあります。再現時に `strace` を実行すると、どこでどのシステムコールを使用したのかを正確に記録することができます。この情報は `tcpdump` 内のイベント情報とともに使用します。
   たとえば NFS マウント内で mycommand --param を実行すると問題が発生する場合、`strace` は下記のように入力して実行します:

   ```
   strace -ttf -s128 -o/tmp/nfs-strace.out mycommand --param
   ```

   再現時に `strace` を採取しない場合は、問題を再現させた時刻を覚えておいて、`/var/log/messages` 内のログのうち、その時刻の前後に存在するログを確認してもかまいません。

4. 問題を再現させることができたら、`tcpdump` を動作させている端末内で `CTRL` – `c` を押して、`tcpdump` を停止してください。なお、`strace` は通常、コマンドの終了後に自動的に終了しますが、停止しない場合は同様の手順で停止してください。

5. 採取したパケットや `strace` のデータは、知識と経験のある管理者であれば分析することができます。上記の手順を実施した場合、それぞれ `/tmp/nfs-demo.cap` と `/tmp/nfs-strace.out` というファイルに保存されています。

22.8.2 高度な NFS デバッグ

重要: 高度なデバッグは熟練者向けの仕組みである件について

本章で説明している内容は、NFS のコードへの理解がある高度な NFS 管理者向けの説明です。そのため、まずは 22.8.1項「一般的なトラブルシューティング」で示している説明に従って、問題点を絞り込むところから始めるようにしてください。その後、必要であれば、より詳しく状況を調べるために本章に示すデバッグ手順を実施してください。

NFS 関連の情報を収集するにあたっては、様々な分野のデバッグ用コードが存在しています。しかしながら、デバッグメッセージは非常に分かりにくいものであるほか、出力量も非常に大きいため、性能への影響もあり得ます。場合によっては、デバッグ機能によって問題の再現を妨害してしまうこともあります。ほとんどの場合においてデバッグ機能は不要であり、NFS のコードを熟知した人間にのみ有用であることに注意してください。

22.8.2.1 `rpcdebug` によるデバッグの有効化

`rpcdebug` は NFS クライアントと NFS サーバの両方でそれぞれデバッグフラグを設定することができます。ご利用の openSUSE Leap で `rpcdebug` が利用できない場合は、NFS クライアントであれば `nfs-client` パッケージを、NFS サーバであれば `nfs-kernel-server` パッケージをそれぞれインストールしてください。

デバッグフラグを設定するには、下記のように入力して実行します:

```
rpcdebug -m モジュール名 -s flags
```

デバッグフラグの設定を解除するには、下記のように入力して実行します:

```
rpcdebug -m モジュール名 -c flags
```

ここで、`モジュール名` には下記のいずれかを指定します:

nfsd
: NFS サーバコードのデバッグ向け

nfs
: NFS クライアントコードのデバッグ向け

nlm
: NFS ロック (施錠) マネージャのデバッグ向け。NFS クライアントと NFS サーバのいずれかで指定します。また、NFS v2/v3 でのみ使用されます。

rpc

Remote Procedure Call (リモートプロシージャコール) のデバッグ向け。NFS クライアントと NFS サーバのいずれかで指定します。

`rpcdebug` コマンドの詳しい使い方については、下記のマニュアルページをお読みください:

```
man 8 rpcdebug
```

22.8.2.2 NFS が依存する各種サービスに対するデバッグの有効化

NFS の動作は他の関連サービスに依存して変化します。これにはたとえば NFS マウントデーモン (`rpc.mountd`) などがあります。関連サービスに対して設定するオプションは、`/etc/sysconfig/nfs` で指定します。

たとえば `/etc/sysconfig/nfs` には下記のようなパラメータがあります:

```
MOUNTD_OPTIONS=""
```

デバッグモードを有効化するには、`-d` オプションに続いて下記のいずれかの値を指定します: `all`, `auth`, `call`, `general`, `parse`

たとえば `rpc.mountd` の全てのデバッグ機能を有効化するには、下記のように指定します:

```
MOUNTD_OPTIONS="-d all"
```

指定可能なオプションについて知りたい場合は、下記のマニュアルページをお読みください:

```
man 8 rpc.mountd
```

また、`/etc/sysconfig/nfs` を変更した場合は、サービスを再起動する必要があります:

```
systemctl restart nfs-server  # NFS サーバ関連の変更の場合
systemctl restart nfs  # NFS クライアント関連の変更の場合
```

23 autofs によるオンデマンド型のマウント

改訂履歴

2022-02-11

`autofs` は指定したディレクトリに対して、必要となった時に自動的なマウントを実施するプログラムです。効率性を高めるためにカーネルモジュールをベースとした作りになっていて、ローカルのディレクトリだけでなく、ネットワーク共有を設定することもできます。これらの自動的なマウントポイントは、そこに対してアクセスが行われた際にマウントが行われ、一定時間アクセスがないと自動的にマウントが解除されるようになっています。このような「オンデマンド」型の仕組みにより、`/etc/fstab` でマウントを固定で指定するよりもずっと柔軟で、ネットワーク帯域にも優しく性能を高めることもできるようになっています。ちなみに、`autofs` は制御スクリプトを、`automount` は実際の自動マウント処理を行うコマンド (デーモン) を意味しています。

23.1 インストール

`autofs` は openSUSE Leap の既定ではインストールされません。自動マウント機能を使用する場合は、まず下記のように実行して、必要なパッケージをインストールしてください:

```
> sudo zypper install autofs
```

23.2 設定

`autofs` の設定を行うには、`vim` などのテキストエディタを利用し、手作業で設定ファイルを編集する必要があります。`autofs` の設定は、大きく分けて 2 種類のファイルがあります。一方は マスター マップファイル、もう一方は固有のマップファイルです。

23.2.1 マスターマップファイル

`autofs` における既定のマスターマップファイルは `/etc/auto.master` です。`/etc/sysconfig/autofs` 内の `DEFAULT_MASTER_MAP_NAME` オプションの値を変更することで、任意の場所にファイルを作成することができます。ここでは openSUSE Leap における既定のマップファイルを使用して作業を行います:

```
#
# Sample auto.master file
# This is an automounter map and it has the following format
# key [ -mount-options-separated-by-comma ] location
# For details of the format look at autofs(5).❶
#
#/misc  /etc/auto.misc❷
#/net -hosts
#
# Include /etc/auto.master.d/*.autofs❸
#
#+dir:/etc/auto.master.d
#
# Include central master map if it can be found using
# nsswitch sources.
#
# Note that if there are entries for /net or /misc (as
# above) in the included master map any keys that are the
# same will not be seen as the first read key seen takes
# precedence.
#
+auto.master❹
```

❶ `autofs` のマニュアルページ (`man 5 autofs`) には、automounter マップに関する書式について、より詳しい説明が多数提供されています。

❷ 既定では (#) 印が付けられてコメントアウトされていますが、シンプルな automounter マッピングの書式に関するシンプルな例が示されています。

❸ マスターマップを複数のファイルに分割するには、この行のコメントマーク (#) を外してください。これにより、`/etc/auto.master.d/` ディレクトリ内に必要なマッピングファイルを配置できるようになります (ただし、`.autofs` で終わるファイル名にしなければならないことに注意してください) 。

❹ `+auto.master` と指定することで、NIS (NIS について、詳しくは『セキュリティ強化ガイド』、第 3章「NIS の使用」、3.1項「NIS サーバの設定」をお読みください) を使用しているユーザに対しても、マスターマップを利用できるようにしています。

`auto.master` 内の各項目は、下記の書式で記述します:

```
マウントポイント　　マップ名　　オプション
```

マウントポイント

`autofs` ファイルシステムのマウント先となるベースの場所を指定します。たとえば `/home` のようになります。

マップ名

マウントの際に使用するマップソースの名前を指定します。マップファイルの書式について、詳しくは 23.2.2項「マップファイル」をお読みください。

オプション

これらのオプションが指定されていれば、指定されたマップ内の全ての項目に対して、既定値の形で適用されるようになります。

ヒント: さらなる情報

マップの種類，**書式**，**オプション** などの設定値に関する詳しい情報については、[auto.master] のマニュアルページ (`man 5 auto.master`) をお読みください。

たとえば `auto.master` 内で下記のような設定を行うと、`autofs` が `/etc/auto.smb` 内を参照し、`/smb` というマウントポイントを作成します:

```
/smb    /etc/auto.smb
```

23.2.1.1 直接マウント

直接マウントとは、対応するマップファイル内で指定されているパスに対して、マウントポイントを直接作成することを意味します。`auto.master` 内でマウントポイントを指定する代わりに、マウントポイントの箇所に `/-` を指定してください。たとえば下記のような設定を行うと、`autofs` は `auto.smb` 内で指定されているパスにマウントポイントを作成します:

```
/-        /etc/auto.smb
```

ヒント: フルパスを指定しないマップについて

マップファイル側でローカルもしくはネットワークパスをフルパスで指定しない場合、Name Service Switch (NSS) の設定を利用して場所を判断するようになります:

```
/-        auto.smb
```

23.2.2 マップファイル

重要: その他の種類のマップについて

`autofs` で自動マウントを行う場合、ファイル タイプが最もよく使用されますが、その他のタイプも用意されています。マップ仕様ではコマンドの出力や LDAP やデータベースの問い合わせ結果などを指定することができます。マップタイプの詳細について、詳しくはマニュアルページ `man 5 auto.master` をお読みください。

マップファイルではマウント元の場所 (ローカルまたはネットワーク) を指定するほか、ローカル側でのマウント先となるマウントポイントを指定します。マップファイルの一般的な書式はマスターマップに似ています。違いは オプション 指定の位置で、項目の末尾ではなくマウントポイントと場所の間に記述します:

```
マウントポイント　　オプション　　場所
```

なお、マップファイルは実行可能であるとしてマークされていないことをご確認ください。実行可能マークを外すには、`chmod -x マップファイル` を実行します。

マウントポイント

マウントする先のディレクトリを指定します。単一のディレクトリ名 (間接 マウントと呼びます) を指定することができるほか、`auto.master` 内で指定したベースパスからの相対パスを指定したり、マウントポイントのフルパス (直接マウントと呼びます、詳しくは 23.2.1.1項「直接マウント」をお読みください) を指定したりすることもできます。

オプション

対応する項目に対して設定する、マウントオプションのリストをカンマ区切りで指定します。このマップファイルに対する `auto.master` の行内にオプションの指定が存在する場合は、それらも追加されます。

場所

ファイルシステムのマウント元の場所を指定します。通常は NFS もしくは SMB のボリュームで、一般的な表記方法 (`ホスト名:パス`) で記述します。`/dev` の項目や smbfs の共有など、'/' で始まるパスをマウント元として指定したい場合は、冒頭に ':' を付けて記述します (例: `:/dev/sda1`) 。

23.3 操作とデバッグ

本章では、`autofs` サービスの制御方法について、および automounter の操作を調整する際のデバッグ情報の表示方法について説明しています。

23.3.1 autofs サービスの制御

`autofs` サービスは `systemd` で制御を行います。`autofs` に対する `systemctl` コマンドの一般的な書式は下記の通りです:

```
> sudo systemctl サブコマンド autofs
```

`サブコマンド` には下記のいずれかを指定します:

enable
: システムの起動時に automounter デーモンを開始するように設定します。

start
: automounter デーモンを開始します。

stop
: automounter デーモンを停止します。自動的なマウントポイントにはアクセスできなくなります。

status
: `autofs` の現状のほか、関連するログファイルの一部を表示します。

restart
: automounter をいったん停止させ、開始し直します。動作中の全てのデーモンを終了させたあと、新しいデーモンを開始します。

reload
: 現在の `auto.master` マップを確認し、項目に変更点があれば対応するデーモンを再起動し、項目に追加があれば新しい項目のデーモンを起動します。

23.3.2 automounter の問題調査

`autofs` でのディレクトリへのマウントについて、何らかの問題が発生した場合は、`automount` デーモンを手動で起動して、出力されるメッセージを確認することをお勧めします:

1. まずは `autofs` を停止します。

   ```
   > sudo systemctl stop autofs
   ```

2. 一方の端末から、冗長な出力を行うよう `automount` を実行し、そのまま前面で動き続けるようにします。

```
> sudo automount -f -v
```

3. もう 1 つの端末を起動して、自動マウントされるはずのディレクトリにアクセスを行います (たとえば `cd` や `ls` など) 。

4. `automount` を起動したほうの端末に戻って出力を確認し、なぜマウントが失敗しているのか、もしくはなぜマウントが行われなかったのかを調べます。

23.4 NFS 共有の自動マウント

下記の手順では、`autofs` を利用してネットワーク内に存在する NFS 共有を自動マウントする流れを説明しています。下記の手順では、ここまでの説明と NFS に関する知識があることを前提にしています。NFS に関して、詳しくは 第22章「NFS によるファイル共有」をお読みください。

1. まずはマスターマップファイルを編集します:

   ```
   > sudo vim /etc/auto.master
   ```

   `/etc/auto.master` の末尾に、新しい NFS マウントの項目を追加します:

   ```
   /nfs      /etc/auto.nfs      --timeout=10
   ```

   これにより、`autofs` はベースマウントポイントが `/nfs` であり、実際の NFS 共有が `/etc/auto.nfs` 内に記述されている設定になります。なお、10 秒間何も処理を行わない場合、自動的にマウントが解除されます。

2. NFS 共有に対する新しいマップファイルを作成します:

   ```
   > sudo vim /etc/auto.nfs
   ```

   `/etc/auto.nfs` には通常、それぞれの NFS 共有に対して 1 行ずつ項目を記述します。書式は 23.2.2項「マップファイル」に示されているとおりです。マウントポイントと、NFS 共有のネットワークアドレスを下記のように指定します:

   ```
   export      jupiter.com:/home/geeko/doc/export
   ```

   上記の行は、`jupiter.com` というホストにある `/home/geeko/doc/export` というディレクトリを、`/nfs/export` というローカル側のディレクトリに自動マウントするための設定です (`/nfs` は `auto.master` マスターマップに記述されているものからの値を取ります) 。`/nfs/export` ディレクトリは、`autofs` で自動的に作成されます。

3. 上記と同じ NFS 共有を `/etc/fstab` に記述していた場合は、下記のようにしてコメントアウトしてください:

```
#jupiter.com:/home/geeko/doc/export /nfs/export nfs defaults 0 0
```

4. 最後に `autofs` を再読み込みさせて、動作するかどうかを確認します:

```
> sudo systemctl restart autofs
```

```
# ls -l /nfs/export
total 20
drwxr-xr-x  5 1001 users 4096 Jan 14  2017 .images/
drwxr-xr-x 10 1001 users 4096 Aug 16  2017 .profiled/
drwxr-xr-x  3 1001 users 4096 Aug 30  2017 .tmp/
drwxr-xr-x  4 1001 users 4096 Apr 25 08:56 manual/
```

上記のように実行することでリモートの共有内のファイルを表示することができれば、`autofs` は問題なく動作していることになります。

23.5 高度なトピック

本章では、`autofs` に関するより高度なトピックを説明しています。ネットワーク内で提供されている NFS 共有の自動マウントやマップファイル内でのワイルドカードの使用、そして CIFS ファイルシステム固有の情報などを説明しています。

23.5.1 /net マウントポイント

このヘルパーマウントポイントは、多数の NFS 共有を使用するような場合に便利です。`/net` はローカルのネットワーク内にある全ての NFS 共有を、必要に応じて自動マウントすることができます。この項目は `auto.master` ファイル内で既に設定されていますので、コメントアウトを解除して `autofs` を再起動するだけで利用することができます:

```
/net      -hosts
```

```
> sudo systemctl restart autofs
```

たとえば `jupiter` というサーバにある `/export` という共有にアクセスしたい場合は、下記をコマンドラインで実行することで、マウントを行うことができます:

```
> sudo cd /net/jupiter/export
```

"¥n ¥n"

23.5.2 自動マウントサブディレクトリでのワイルドカードの使用

たとえば `/home` などがそれにあたりますが、個別に自動マウントしたいサブディレクトリを含むディレクトリがある場合、`autofs` では、これをより賢く解決することができます。

たとえばホームディレクトリの場合、`auto.master` ファイル内に下記の行を追加します:

```
/home      /etc/auto.home
```

あとは `/etc/auto.home` ファイル内に適切なマッピングを追加していくことで、ユーザのホームディレクトリを自動的にマウントすることができるようになります。今までの設定方法からすると、下記のようにそれぞれ個別のディレクトリを指定していくことになります:

```
wilber      jupiter.com:/home/wilber
penguin      jupiter.com:/home/penguin
tux      jupiter.com:/home/tux
[...]
```

でも、このような設定にしてしまうと、ユーザの一覧を `auto.home` 内でも管理しなければならなくなりますので、非常に面倒です。その代わり、マウントポイントの指定で '*' (アスタリスク) を指定し、ディレクトリ名の代わりに '&' を指定することで、上記の設定と同じ効果をもたらすことができます:

```
*      jupiter:/home/&
```

23.5.3 CIFS ファイルシステムの自動マウント

SMB/CIFS の共有 (SMB/CIFS プロトコルに関する詳細は、第21章「Samba」をお読みください) を自動マウントしたい場合、マップファイル内の書式を変更する必要があります。オプション欄に `-fstype=cifs` と指定して、コロン ':' に続けてホスト名と共有名を指定します:

```
マウントポイント      -fstype=cifs      ://jupiter.com/export
```

24 Apache HTTP サーバ

改訂履歴

2025-03-17

https://www.netcraft.com/ や https://w3techs.com/ の調査によると、Apache HTTP サーバ (Apache) は世界で最も有名な Web サーバのうちの 1 つです。この Web サーバは Apache Software Foundation (https://www.apache.org/) によって開発されているソフトウエアで、ほとんどのオペレーティングシステムで動作します。openSUSE® Leap では Apache バージョン 2.4 が同梱されています。本章では Apache のインストールから設定／運用のほか、SSL や CGI などの追加モジュールの設定方法やトラブルシューティングまでの範囲を説明しています。

24.1 クイックスタート

本章では、Apache を簡単に設定して起動するまでの手順を説明しています。なお、Apache のインストールと設定は、`root` で行わなければなりません。

24.1.1 要件

Apache Web サーバを設定しようとする前に、まずは下記の要件全てを満たしていることをご確認ください:

1. 対象となるマシンのネットワークが正しく設定されている必要があります。ネットワークの設定について、詳しくは 第13章「ネットワークの基礎」をお読みください。
2. 対象となるマシンのシステム時刻が、タイムサーバとの間で正しく同期している必要があります。HTTP プロトコルでは、時刻をベースにした仕組みを利用しているものがあるためです。詳しくは 第18章「NTP を利用した時刻同期」をお読みください。
3. 最新のセキュリティ更新を適用しておく必要があります。何か不安な点があれば、念のため YaST オンライン更新を実行しておいてください。
4. 既定の Web サーバのポート (`80`) をファイアウオールで開く必要があります。ファイアウオールでポートを開くには、`firewalld` で `http` サービスを公開ゾーンで許可する必要があります。詳しくは『セキュリティ強化ガイド』、第24章「マスカレードとファイアウオール」、24.4.1項「コマンドラインからのファイアウオール設定」をお読みください。

24.1.2 インストール

openSUSE Leap では、Apache は既定ではインストールされません。インストールを行い、標準的な「出荷時設定」で開始したい場合は、下記の手順を実施します:

手順 24.1: 既定の設定による APACHE のインストール

1. YaST を起動して、[ソフトウエア] › [ソフトウエア管理] を選択します。
2. [表示] › [パターン] を選択し、[Web および LAMP サーバ] にチェックを入れます。
3. あとは [了解] を押して、インストール処理を完了します。依存関係のパッケージ一覧が表示されますので、こちらでも [続行] を押してください。

24.1.3 起動

Apache はシステムの起動時に自動的に開始することができるほか、手作業で開始することもできます。

Apache をシステムの起動時に自動的に開始したい場合は、`multi-user.target` と `graphical.target` のターゲット内に Apache を追加します。具体的には、下記のコマンドを実行します:

```
> sudo systemctl enable apache2.service
```

openSUSE Leap における `systemd` のターゲットに関する説明や、YaST の [サービスマネージャ] の説明について、詳しくは 10.4項「YaST を利用したサービスの管理」をお読みください。

シェルを利用して Apache を手作業で開始したい場合は、`systemctl start apache2.service` を実行します。

手順 24.2: APACHE が動作しているかどうかの確認

Apache の開始時に何もエラーメッセージが表示されない場合は、Web サーバが問題なく起動して動作していることを表しています。動作確認を行うには、下記の手順で行います:

1. ブラウザを起動して `http://localhost/` を開きます。
   Apache が開始されていて動作していれば、「It works!」というメッセージが表示されるはずです。
2. このようなページが表示されない場合は、24.9項「トラブルシューティング」をご覧ください。

Web サーバが動作していることを確認できたら、あとは必要な文書を追加したり、必要に応じて設定を調整したり、モジュールを追加して機能を追加したりすることができるようになります。

24.2 Apache の設定

openSUSE Leap では 2 種類の設定方法を用意しています:

- Apache の手作業での設定
- YaST による Apache の設定

手作業で設定を行うと、より細かいところまで設定を行うことができますが、YaST GUI ほどの利便性はありません。

重要: 設定変更後の Apache の再読み込みもしくは再起動について

設定を変更した場合は、設定を反映させるために Apache の再読み込みまたは再起動が必要となります。手作業で再読み込みを行いたい場合は、`systemctl reload apache2.service` を実行してください。また、再起動は 24.3項「Apache の開始と停止」に示されているいずれかの方法を実行してください。

YaST で Apache を設定した場合は、［HTTP サービス］が［有効］に設定されていると、再読み込みを自動的に実施するようになっています。詳しくは 24.2.3.2項「HTTP サーバの設定」をお読みください。

24.2.1 Apache の設定ファイル

本章では、Apache の設定ファイルに関する概要説明を行っています。設定にあたって YaST を使用している場合は、これらのファイルを直接編集する必要はありません。しかしながら、あとから手作業で編集するように切り替える場合に備えて、あらかじめいくつかの情報を知っておくと良いでしょう。

Apache の設定ファイルは、下記の 2 箇所に配置されています:

- `/etc/sysconfig/apache2`
- `/etc/apache2/`

24.2.1.1 /etc/sysconfig/apache2

`/etc/sysconfig/apache2` では、Apache 全体のグローバル設定を行います。たとえば読み込むべきモジュールや取り込むべき追加の設定ファイルや、サーバを起動する際のフラグやコマンドラインに追加すべき設定などがあります。このファイル内での全ての設定オプションは、設定ファイル内にコメントで説明しているため、ここでは詳しく説明しません。一般的な Web サーバであれば、`/etc/sysconfig/apache2` を設定するだけで、要件に十分対応することができるはずです。

24.2.1.2 /etc/apache2/

`/etc/apache2/` には Apache に対する全ての設定ファイルが含まれています。下記ではそれぞれのファイルの用途を説明しています。各ファイルにはいくつかの設定オプション (ディレクティブ と呼びます) が用意されていますが、同様にこれらのファイルのコメント内に説明が書かれています。そのため、こちらもここでは詳しく説明しません。

Apache の設定ファイルは、下記のような構成になっています:

```
/etc/apache2/
     |
     |- charset.conv
     |- conf.d/
     |   |
     |   |- *.conf
     |
     |- default-server.conf
     |- errors.conf
     |- global.conf
     |- httpd.conf
     |- listen.conf
     |- loadmodule.conf
     |- magic
     |- mime.types
     |- mod_*.conf
     |- protocols.conf
     |- server-tuning.conf
     |- ssl-global.conf
     |- ssl.*
     |- sysconfig.d
     |   |
     |   |- global.conf
     |   |- include.conf
     |   |- loadmodule.conf . .
     |
     |- uid.conf
     |- vhosts.d
     |   |- *.conf
```

/ETC/APACHE2/ 内の APACHE 設定ファイル

`charset.conv`

様々な言語でどの文字セットを使用するのかを指定しています。このファイルについては編集してはなりません。

`conf.d/*.conf`

他のモジュールによって追加された設定ファイルが保存されます。これらの設定ファイルは、必要に応じて仮想ホストの設定ファイルから取り込むことができます。使用例について、詳しくは `vhosts.d/vhost.template` をお読みください。また、これを行うことで、それぞれの仮想ホストに対して別々のモジュールセットを提供することができるようになります。

`default-server.conf`

適切な既定値が設定された、全ての仮想ホスト向けのグローバル設定です。ここに書かれている値を書き換えるのではなく、仮想ホストの設定ファイル内で値を指定して変更するようにしてください。

`errors.conf`

Apache のエラー応答方法を定義しています。全ての仮想ホストに対してエラーメッセージをカスタマイズしたい場合は、このファイルを編集してください。それぞれの仮想ホストで別々のエラーメッセージを設定したい場合は、それぞれの仮想ホストの設定内で設定を行ってください。

`global.conf`

Web サーバのメインプロセスに対する一般的な設定を記述するファイルです。アクセスパスやエラーログ、ログレベルなどの設定が含まれています。

`httpd.conf`

Apache のメインの設定ファイルです。このファイルを変更することは避けてください。主にステートメントとグローバル設定が含まれています。設定を変更したい場合は、対応する設定ファイル内で設定を行ってください。また、ホスト固有の設定 (たとえばドキュメントルートなど) を行いたい場合は、仮想ホストの設定で行ってください。

`listen.conf`

Apache がどの IP アドレスやポートで待ち受けるかを設定するファイルです。名前ベースの仮想ホストについても、ここで設定を行います。詳しくは 24.2.2.1.1項「名前ベースの仮想ホスト」をお読みください。

`magic`

Apache が未知のファイルに対して MIME タイプを自動判別する際に使用する、mime_magic モジュール向けのデータファイルです。このファイルを変更してはなりません。

`mime.types`

システム側で既知の MIME タイプを一覧で示しているファイルです (`/etc/mime.types` へのリンクになっています) 。このファイルを変更してはなりません。ここに書かれていない MIME タイプを追加したい場合は、 `mod_mime-defaults.conf` ファイルに記述してください。

`mod_*.conf`

既定でインストールされるモジュール向けの設定ファイルです。詳しくは 24.4項「モジュールのインストール／有効化／設定」をお読みください。なお、オプションモジュール向けの設定ファイルは、 `conf.d` ディレクトリ内に存在しています。

`protocols.conf`

HTTP2 でページを提供するための設定ディレクティブが含まれるファイルです。

`server-tuning.conf`

様々な MPM (詳しくは 24.4.4項「マルチプロセッシングモジュール (MPM)」をお読みください) に対する設定ディレクティブと、Apache の性能を制御するための一般的な設定オプションが含まれています。このファイル内の設定を変更するにあたっては、お使いの Web サーバで十分なテストを行ってください。

`ssl-global.conf` および `ssl.*`

SSL の全体設定と SSL の証明書ファイルが含まれています。詳しくは 24.6項「SSL を利用した暗号化機能付き Web サーバの設定」をお読みください。

`sysconfig.d/*.conf`

`/etc/sysconfig/apache2` から自動生成された設定ファイルです。これらのファイルについては変更せず、 `/etc/sysconfig/apache2` を変更して対応してください。また、このディレクトリには他の設定ファイルを配置しないでください。

`uid.conf`

Apache を動作させる際のユーザ ID とグループ ID を指定しています。このファイルについても変更してはなりません。

`vhosts.d/*.conf`

仮想ホストの設定は、この形式で保存する必要があります。このディレクトリ内には、SSL ありの場合と無しの場合に対応する、仮想ホストの設定テンプレート (雛形) が用意されています。また、Apache の設定ファイルとして自動的に取り込むには、ファイル名の末尾が `.conf` で終わっていなければなりません。詳しくは 24.2.2.1項「仮想ホストの設定」をお読みください。

24.2.2 Apache の手作業での設定

Apache を手作業で設定するにあたっては、純粋なテキストファイルで書かれている設定ファイルを `root` で編集する必要があります。

24.2.2.1 仮想ホストの設定

Apache における 仮想ホスト とは、同一の物理マシン内で複数の統一資源識別子 (URI) を賄う仕組みです。たとえば www.example.com や www.example.net などの異なるドメイン名の Web サーバを、1 台の物理マシン内にある 1 つの Web サーバで提供することができます。

仮想ホストの使用は、管理費用の削減 (管理すべき Web サーバを 1 台にまとめることができる) やハードウエア費用の削減 (それぞれのドメインに対して別々の専用サーバを用意する必要が無くなる) などの効果がある実践的な方法です。仮想ホストは名前 (ドメイン名, ホスト名) ベースのほか、IP ベースやポートベースでも設定することができます。

設定済みの仮想ホストを一覧表示したい場合は、`apache2ctl -S` を実行します。これにより、既定のサーバと全ての仮想ホストが一覧表示することができるほか、待ち受ける IP アドレスとポートも表示することができます。これに加えて、それぞれの仮想ホストが指定されている設定ファイル内の場所も表示することができます。

仮想ホストは 24.2.3.1.4項「仮想ホスト」で説明しているとおり YaST で設定することができるほか、設定ファイルを直接編集して設定することもできます。既定では、openSUSE Leap 内の Apache は `/etc/apache2/vhosts.d/` ディレクトリ内にある 1 つのファイルで 1 つの仮想ホストを設定します。また、Apache の設定ファイルとして自動的に取り込むには、ファイル名の末尾が `.conf` で終わっていなければなりません。また、このディレクトリ内には、仮想ホストを設定するためのテンプレート (雛形) も用意されています。SSL 無しの場合は `vhost.template` を、SSL ありの場合は `vhost-ssl.template` をご覧ください。

ヒント: 仮想ホストの作成について

お使いの Web サーバに 1 つのドメインしか存在しないような場合であっても、仮想ホストの設定ファイルを作成しておくことをお勧めします。これにより、ドメイン固有の設定を 1 つのファイルにまとめることができるだけでなく、仮想ホストの設定ファイルを移動したり削除したり名前変更したりするだけで、容易に元の設定に戻すことができるようになるためです。複数のドメインが存在する場合も同じ理由により、それぞれの仮想ホストに対してそれぞれ設定ファイルを個別に作成することをお勧めします。

名前ベースの仮想ホストを使用する場合は、どの仮想ホストの設定にも該当しなかった場合に備えて、その他のドメインの場合 (既定の仮想ホスト) の設定ファイルを作成しておくことをお勧めします。Apache では、設定ファイルの中で最初に読み込まれる仮想ホストが既定の仮想ホ

ストに設定されますが、読み込みの順序はファイル名で判断される仕組みになっていますので、たとえばファイル名の最初に (_) 文字を入れるなど (例: _default_vhost.conf) して、最初に読み込まれるようにすると良いでしょう。

<VirtualHost> から </VirtualHost> までのブロックには、特定のドメイン (仮想ホスト) にのみ適用される設定を記述します。Apache が設定済みの仮想ホストに対する要求を受け取ると、対応するセクション内に書かれた設定を利用して要求を処理します。仮想ホストのブロックには、ほとんど全てのディレクティブを記述することができます。Apache の設定ディレクティブについて、詳しくは https://httpd.apache.org/docs/2.4/mod/quickreference.html をお読みください。

24.2.2.1.1 名前ベースの仮想ホスト

名前ベースの仮想ホストを使用すると、複数の Web サイトを 1 つの IP アドレスで提供できるようになります。Apache では、クライアントが送信する HTTP ヘッダ内の Host フィールドを、それぞれの仮想ホスト設定の ServerName と比較して、一致する仮想ホストを使用するようになっています。該当する ServerName が見つからない場合は、最初に設定された仮想ホストを既定で使用するようになっています。

<VirtualHost> ブロックを作成するにあたって最初にやるべきことは、提供する名前ベースの仮想ホストの名前設定です。それぞれの <VirtualHost> ブロック内に ServerName ディレクティブを指定してドメイン名を指定し、そのドメイン名にアクセスがあった場合に、どのディレクトリからファイルを提供するのかを DocumentRoot ディレクティブで指定します。

例 24.1: 名前ベースの仮想ホスト (VirtualHost) の設定例

```
<VirtualHost *:80>
# 最初に記述された仮想ホストが既定のホストになります
ServerName www.example.com
ServerAlias example.com
DocumentRoot /srv/www/htdocs/domain
</VirtualHost>

<VirtualHost *:80>
ServerName other.example.com
DocumentRoot /srv/www/htdocs/otherdomain
</VirtualHost>
```

名前ベースの仮想ホストを設定する場合、開く側の VirtualHost タグには IP アドレス (もしくは完全修飾ドメイン名) をパラメータとして指定します。ポート番号は指定しても指定していなくてもかまいません。

なお、IP アドレスの代わりにワイルドカード * を指定することもできます。IPv6 アドレスを指定する場合、アドレスは角括弧で括らなければなりません。

例 24.2: 名前ベースの仮想ホスト (VirtualHost) の設定

```
<VirtualHost 192.168.3.100:80>
  ...
</VirtualHost>

<VirtualHost 192.168.3.100>
  ...
</VirtualHost>

<VirtualHost *:80>
  ...
</VirtualHost>

<VirtualHost *>
  ...
</VirtualHost>

<VirtualHost [2002:c0a8:364::]>
  ...
</VirtualHost>
```

24.2.2.1.2 IP ベースの仮想ホスト

IP ベースの仮想ホストを設定するにあたっては、マシンに対して複数の IP アドレスを設定しておかなければなりません。IP ベースの仮想ホストでも、Apache は 1 つのインスタンスで複数のドメインに対応することができます (それぞれ異なる IP アドレスに対応する必要があります) 。

言い換えると、IP ベースの仮想ホストを使用するにあたっては、物理サーバ側に仮想ホストの数と同じだけの IP アドレスを設定する必要があります。ただし、ネットワークカードについては、仮想ホストの数と同じだけ存在する必要はありません。1 つのネットワークカードに複数の IP アドレスを設定すれば十分です。

下記の例では、`192.168.3.100` の IP アドレスを持つマシンで Apache を動作させ、追加で設定した 2 つの IP アドレスを利用して `192.168.3.101` と `192.168.3.102` のドメインを動作させています。それぞれの仮想ホストに対して、別々の `VirtualHost` を指定していることに注意してください。

例 24.3: IP ベースの仮想ホスト (VirtualHost) の設定

```
<VirtualHost 192.168.3.101>
  ...
</VirtualHost>
```

```
<VirtualHost 192.168.3.102>
  ...
</VirtualHost>
```

なお、この例では `VirtualHost` ディレクティブで `192.168.3.100` の IP アドレスを設定していません。`Listen` ディレクティブで `192.168.3.100` の IP アドレスに対するアクセスを受け入れるように設定している場合は、`192.168.3.100` 向けの仮想ホストの設定も必要となります。設定を行わない場合は、既定のサーバ設定 (`/etc/apache2/default-server.conf`) が適用されます。

24.2.2.1.3 基本的な仮想ホストの設定

仮想ホストを設定するにあたっては、少なくともそれぞれの仮想ホストの設定内に下記のディレクティブを設定する必要があります。それ以外のオプションについては `/etc/apache2/vhosts.d/vhost.template` をお読みください。

`ServerName`
: その仮想ホストに対応する完全修飾ドメイン名を指定します。

`DocumentRoot`
: その仮想ホストが提供すべきファイルのあるディレクトリパスを指定します。また、セキュリティ上の理由から、既定ではファイルシステム全体へのアクセスが禁止されていますので、`Directory` コンテナを利用して、ディレクトリに対するアクセスを許可しなければなりません。

`ServerAdmin`
: サーバ管理者の電子メールアドレスを指定します。このアドレスはたとえば、Apache がエラーページを表示する際などに使用されます。

`ErrorLog`
: この仮想ホストに対するエラーログファイルを指定します。各仮想ホストに対して、それぞれ別々のエラーログファイルを用意する必要はありませんが、エラーへの対応を容易にする目的から、それぞれ別々にしておくことをお勧めします。なお、Apache のログファイルの既定ディレクトリは、`/var/log/apache2/` になっています。

`CustomLog`
: この仮想ホストに対するアクセスログファイルを指定します。各仮想ホストに対して、それぞれ別々のアクセスログを用意する必要はありませんが、それぞれの仮想ホストに対するアクセス解析を容易にする目的から、それぞれ別々にしておくことをお勧めします。なお、Apache のログファイルの既定ディレクトリは、`/var/log/apache2/` になっています。

上述のとおり、ファイルシステム全体に対するアクセスは、セキュリティ上の理由から、既定で禁止されるようになっています。そのため、Apache がファイルを提供するディレクトリに対しては、明示的に指定して許可を与える必要があります。たとえば `DocumentRoot` の場合は、下記のように設定します:

```
<Directory "/srv/www/www.example.com/htdocs">
  Require all granted
</Directory>
```

注記: Require all granted

Apache の以前のバージョンでは、`Require all granted` と同じ意味を持つ設定は下記のとおりです:

```
Order allow,deny
Allow from all
```

この古い書式は、`mod_access_compat` モジュールを読み込むことによって、現在も使用することができます。

これらを組み合わせると、完全な設定ファイルは下記のようになります:

例 24.4: 基本的な VirtualHost の設定

```
<VirtualHost 192.168.3.100>
  ServerName www.example.com
  DocumentRoot /srv/www/www.example.com/htdocs
  ServerAdmin webmaster@example.com
  ErrorLog /var/log/apache2/www.example.com_log
  CustomLog /var/log/apache2/www.example.com-access_log common
  <Directory "/srv/www/www.example.com/htdocs">
  Require all granted
  </Directory>
</VirtualHost>
```

24.2.3 YaST による Apache の設定

YaST を利用して Web サーバを管理するには、YaST を起動して［ネットワークサービス］›［HTTP サーバ］と選択していきます。初めてモジュールを起動した場合は［HTTP サーバウイザード］が起動し、サーバの管理に関わるいくつかの基本設定を行うことになります。ウイザードを完了すると、［HTTP サーバの設定］ダイアログが表示されるようになり、これ以降は、［HTTP サーバ］モジュールを起動すると、このダイアログが表示されるようになります。詳しくは 24.2.3.2項「HTTP サーバの設定」をお読みください。

24.2.3.1 HTTP サーバウイザード

HTTP サーバウイザードは 5 つのステップから構成されています。ダイアログの最後のステップでは、熟練者向け設定モードに入ることができますので、ここからさらに細かい設定を行うことができます。

24.2.3.1.1 ネットワークデバイスの選択

ここでは、Apache がリクエストを待ち受けるネットワークインターフェイスとポートを指定します。ここでは既存のネットワークインターフェイスと IP アドレスの組み合わせのうち、任意のものを選択することができます。なお、ポートについては、他のサービスが利用していないことを前提に、任意のポート (いわゆる well-known ポートや registered ポート、もしくは動的なポートや独自のポートなど) を設定することができます。既定では全てのネットワークインターフェイス (IP アドレス) に対して、ポート 80 で待ち受けるようになっています。

Web サーバが待ち受けるポートに対して、ファイアウオールでポートを開きたい場合は、［ファイアウオールでポートを開く］を選択します。これは LAN や WAN のほか、インターネットから Web サーバへのアクセスを受け付ける際に必要な設定です。ポートを閉じたままにしておいた場合は Web サーバ自身からしかアクセスできなくなりますので、テスト用途にのみ有用です。また、複数のネットワークインターフェイスを設定している場合は、［ファイアウオールの詳細］を押して、どのネットワークインターフェイスを利用してポートを開くのかを選択することができます。

［次へ］を押すと設定を続けることができます。

24.2.3.1.2 モジュール

［モジュール］設定オプションでは、Web サーバ側で動作させるスクリプト言語の有効化と無効化を行うことができます。他のモジュールを有効化もしくは無効化したい場合は、24.2.3.2.2項「サーバモジュール」をお読みください。［次へ］を押すと次のダイアログに進みます。

24.2.3.1.3 既定のホスト

この設定画面では、既定の Web サーバに関する設定を行います。24.2.2.1項「仮想ホストの設定」でも説明しているとおり、Apache では複数の仮想ホストを単一の物理マシンで賄うことができます。設定ファイル内で最初に設定した仮想ホストが 既定のホスト として設定され、それぞれの仮想ホストでは、その既定のホスト設定を引き継いで動作するようになります。

ホストの設定 (ディレクティブ とも呼びます) を編集するには、表内で対応する項目を選択して［選択］を押します。新しいディレクティブを追加するには、［追加］を押します。ディレクティブを削除するには、削除したい項目を選んで［削除］を押します。

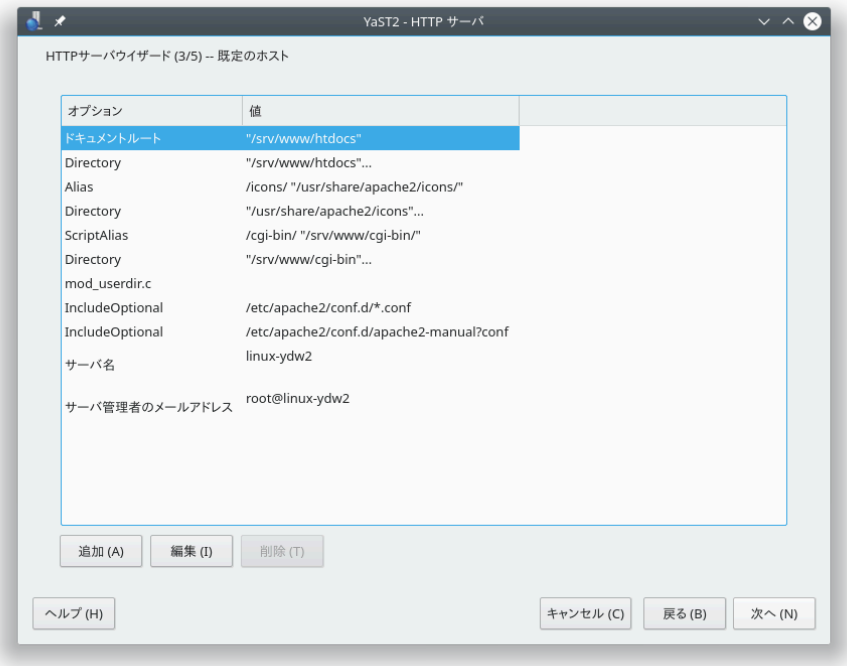


図 24.1: HTTP サーバウイザード: 既定のホスト

ここではサーバに対する既定の設定を行います:

`ドキュメントルート`
: このホストに対するアクセスが届いた場合、Apache が提供すべきファイルのディレクトリを指定します。既定値は `/srv/www/htdocs` です。

`Alias`
: `Alias` ディレクティブを使用することで、URL を物理的なファイルシステムの場所に割り当てることができます。これにより、ファイルシステム内の `ドキュメントルート` 以外の場所を、URL に割り当てることができるようになります。
既定の openSUSE Leap では、`/icons` が `/usr/share/apache2/icons` を指すように `Alias` の設定が行われていて、これによってディレクトリ一覧を表示した場合に、Apache が提供するアイコンを表示するようになっています。

`ScriptAlias`
: `Alias` ディレクティブと同様に、`ScriptAlias` ではファイルシステムと URL との間の対応付けを設定します。ただし、`ScriptAlias` では CGI のディレクトリを設定する点が異なり、そのディレクトリの中ではスクリプトを動作させることができるようになります。

`Directory`

`Directory` 設定を指定することで、特定のディレクトリにのみ適用する設定オプションを定義することができるようになります。
ここでは、それぞれ `/srv/www/htdocs` , `/usr/share/apache2/icons` , `/srv/www/cgi-bin` の各ディレクトリに対するアクセスおよび表示オプションが設定されています。これらの値は特に変更する必要はないはずです。

`Include`

`Include` ディレクティブを指定することで、追加の設定ファイルを指定することができます。既定では 2 種類の `Include` ディレクティブが設定されています。1 つは `/etc/apache2/conf.d/` ディレクトリに対する設定で、こちらは外部モジュールが提供する設定ファイルを取り込むためのものです。このディレクティブを使用することで、そのディレクトリ内に存在していて `.conf` で終わるファイル全てを取り込むことができます。もう 1 つは `/etc/apache2/conf.d/apache2-manual.conf` の指定で、こちらは `apache2-manual` の設定ファイルを取り込むためのものです。

サーバ名

ここでは、Web サーバがアクセスする際にクライアントが使用する既定の URL を指定します。`http://FQDN/` の形式でアクセスできる完全修飾ドメイン名 (FQDN) を指定するか、もしくは IP アドレスで指定してください。なお、ここでは任意の名前を指定することはできません。サーバ名には「既知の」名前を指定しなければなりません。

サーバ管理者のメールアドレス

サーバ管理者の電子メールアドレスを指定します。このアドレスは、たとえば Apache が生成するエラーページなどに記載されます。

［既定のホスト］のステップを完了したら、［次へ］を押して設定を続けてください。

24.2.3.1.4 仮想ホスト

このステップでは、ウイザードは既に設定済みの仮想ホストの一覧を表示します (詳しくは 24.2.2.1項「仮想ホストの設定」をお読みください) 。YaST の HTTP ウイザードを起動するまでに何も手作業で設定していない場合は、仮想ホストが何も表示されません。

仮想ホストを追加するには、［追加］を押してダイアログを表示させ、［サーバ名］, ［サーバコンテンツのルート］(`DocumentRoot`), ［管理者のメールアドレス］など、仮想ホストに関する基本設定を行ってください。また、［サーバの解決］では、ホストの識別方法を指定することができます (名前ベースまたは IP アドレスベース) 。また、［仮想ホスト ID の変更］では、名前もしくは IP アドレスを指定することができます。

［次へ］を押すと、仮想ホストの設定ダイアログの 2 番目に移動することができます。

仮想ホストに対する 2 番目の設定ダイアログでは、CGI スクリプトの有効化を行うかどうかや、これらのスクリプトの配置先ディレクトリをなどを設定することができます。また、SSL の有効化についても設定を行うことができます。SSL を設定する場合は、証明書のパスについても設定を行わなければなりません。SSL と証明書について、詳しくは 24.6.2項「SSL を利用する場合の Apache の設定」をお読みください。また、［ディレクトリインデックス］では、クライアントからディレクトリに対してアクセスがあった場合に、表示するファイルを指定することができます (既定では `index.html` を表示します) 。必要であれば、スペース区切りで複数のファイルを指定することもできます。また、［公開 HTML を有効にする］を選択すると、各ユーザの公開ディレクトリ ((`~ユーザ名/public_html/`) の内容を、`http://www.example.com/~ユーザ名` のような URL で提供することができるようになります。

重要: 仮想ホストの作成について

仮想ホストは完全に自由に作成できるというものではありません。名前ベースの仮想ホストを使用する場合、それぞれのホスト名はネットワーク内で解決可能な名前でなければなりません。IP ベースの仮想ホストを使用する場合は、それぞれの IP アドレスに対して 1 つの仮想ホストを割り当てることができます。

24.2.3.1.5 概要

これがウイザードの最後のステップです。ここでは、Apache サーバの開始方法 (システム起動時に自動的に開始するか、もしくは手作業で開始するか) を指定することができます。また、ここまでに設定してきた内容の概要も表示されます。ここに表示された設定で問題か無ければ、［完了］を押して設定を完了させてください。何らかの項目を変更したい場合は、［戻る］を押して、対応するダイアログまで戻っていってください。また、［HTTP サーバの熟練者向け設定］を押すと、24.2.3.2項「HTTP サーバの設定」で示されているダイアログを表示することができます。

図 24.2: HTTP サーバウイザード: 概要

24.2.3.2 HTTP サーバの設定

[HTTP サーバの設定] ダイアログでは、ウイザードで提示されるものよりも細かい設定を行うことができます (ウイザードは、お使いの Web サーバを初めて設定しようとした場合にのみ表示されます) 。このダイアログは、下記に説明する 4 つのタブから構成されています。ここで設定した内容は即時には反映されず、[完了] を押すことによって反映されるようになっています。また、[中止] を押すと、設定モジュールを終了して変更点を破棄することができます。

24.2.3.2.1 待ち受けるポートとアドレス

[HTTP サービス] の枠内では、Apache を動作させる ([有効]) のか、もしくは停止させる ([無効]) のかを選択することができます。また、[待ち受けるポート] では、サーバが待ち受けるべきアドレスやポートを [追加] , [編集] , [削除] することができます。既定では全てのインターフェイスに対して、ポート 80 で待ち受けるように設定されています。また、外部から Web サーバに対してアクセスができるようにするため、[ファイアウオールでポートを開く] についてもチェックしておくべきでしょう。ポートを閉じたままにしておくと、外部から Web サーバに対してアクセスすることができなくなりますので、テスト用途の場合にのみ役立ちます。また、複数のネットワークインターフェイスを設定している場合は、[ファイアウオールの詳細] を押して、どのインターフェイスのどのポートを開くべきなのかを設定することもできます。

［ログファイル］では、アクセスログファイルやエラーログファイルを表示することができます。これは設定のテストを行う場合に有用な機能です。ログファイルは異なるウインドウ内に表示され、ここから Web サーバを再起動したり再読み込みさせたりすることができます。詳しくは 24.3項「Apache の開始と停止」をお読みください。これらのコマンドはその場で動作しますので、ログメッセージについても最新のものを読むことができます。

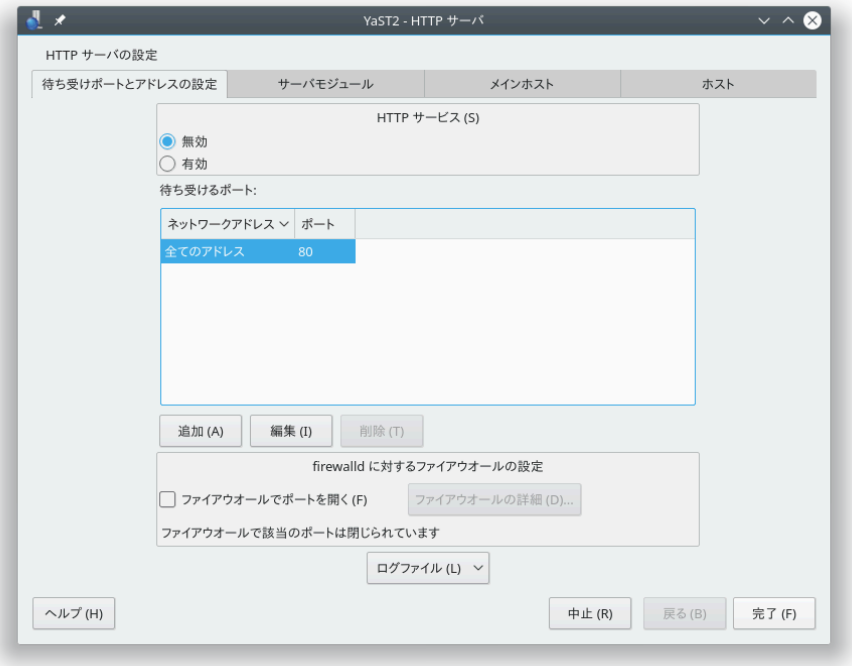


図 24.3: HTTP サーバの設定: 待ち受けポートとアドレスの設定

24.2.3.2.2 サーバモジュール

ここでは［状態の切り替え］を押すことで、Apache2 モジュールの状態を変更 (有効もしくは無効) することができます。既にインストールされているものの、一覧には表示されないモジュールがある場合は、［モジュールの追加］を押してください。モジュールに関する詳細は、24.4項「モジュールのインストール／有効化／設定」をお読みください。

図 24.4: HTTP サーバの設定: サーバモジュール

24.2.3.2.3 メインホストとホスト

これらのダイアログは説明済みのものと同じです。それぞれ 24.2.3.1.3項「既定のホスト」と 24.2.3.1.4項「仮想ホスト」をお読みください。

24.3 Apache の開始と停止

24.2.3項「YaST による Apache の設定」で説明している手順に従って YaST で設定を行った場合、Apache は `multi-user.target` と `graphical.target` 内で、システムの起動時に自動的に開始されるようになります。この動作を変更したい場合は、YaST の［サービスマネージャ］を使用するか、もしくは `systemctl` コマンドラインツールをお使いください (`systemctl enable` もしくは `systemctl disable` を実行します) 。

Apache の開始と停止、もしくは動作中の Apache に対して何らかの処理を行わせたい場合は、下記のようにして `systemctl` もしくは `apachectl` コマンドをご利用ください。

`systemctl` コマンドに関する一般的な情報については、10.2.1項「動作中のシステムにおけるサービスの管理」をお読みください。

`systemctl status apache2.service`

Apache が動作しているかどうかを調べます。

`systemctl start apache2.service`

Apache が動作していない場合、Apache を開始します。

`systemctl stop apache2.service`

親プロセスを終了させることで Apache を終了します。

`systemctl restart apache2.service`

Apache を一旦停止して起動し直します。Apache が動作中でなかった場合は、単純に開始のみを行います。

`systemctl try-restart apache2.service`

動作中の場合にのみ、Apache をいったん停止して起動し直します。

`systemctl reload apache2.service`

全ての起動済み Apache プロセスに対して、現在処理中のリクエストの処理を完了してから終了するように促します。それぞれのプロセスが終了したタイミングで新しいプロセスを起動して入れ替えていき、最終的に Apache が完全に「再起動」できるようにします。

ヒント: 本番環境における Apache の再起動について

このコマンドを実行することで、クライアントから Apache への接続を強制的に終了させることなく、Apache の設定変更を反映させることができるようになります。

`systemctl stop apache2.service`

Web サーバを停止しますが、その際に `GracefulShutdownTimeout` で指定された長さの時間だけ終了を待機します。これにより、既存のリクエスト処理を正常に終了させることができるようになります。

`apachectl configtest`

現在動作中の Web サーバに影響を与えずに、設定ファイルの文法をチェックします。このチェックはサーバの開始時や再読み込み時、再起動時にそれぞれ実行されるように設定されているため、このコマンドを明示的に実行する必要はありません (何らかの設定エラーが検出されると、開始／再読み込み／再起動を行わないようになっています) 。

`apachectl status` および `apachectl fullstatus`

それぞれ現在の Apache サーバに関する情報を、短くもしくは長く表示します。この情報を表示するには、`mod_status` モジュールを有効化する必要があるほか、テキストベースのブラウザ (たとえば `links` や `w3m` など) をインストールしておく必要があります。さらに、`/etc/sysconfig/apache2` ファイル内の `APACHE_SERVER_FLAGS` に対して、`STATUS` を追加しなければなりません。

ヒント: 追加フラグについて

コマンドに対して追加のフラグを指定すると、それらは Web サーバにそのまま渡されます。

24.4 モジュールのインストール／有効化／設定

Apache ソフトウエアはモジュール型の構造になっているため、いくつかの中枢機能を除くと、ほとんどの機能はモジュールとして作られています。HTTP 自身でさえも、モジュールとして作られ処理されています (`http_core`) 。

Apache のモジュールは構築時に Apache バイナリ内に組み込むことができるほか、実行時に動的に読み込むことができます。モジュールを動的に読み込む方法について、詳しくは 24.4.2項「有効化と無効化」をお読みください。

Apache のモジュールは、下記の 4 種類の分野に分かれています:

基本モジュール

基本モジュールは、既定で Apache 内に組み込まれます。openSUSE Leap での Apache では、`mod_so` (他のモジュールを読み込むためのモジュール) と `http_core` のみが組み込まれます。それ以外のモジュールは共有オブジェクトとして提供され、サーバのバイナリファイルそのものに組み込まれずに、実行時に別途読み込む仕組みになっています。

拡張モジュール

拡張として位置づけられているモジュールは、Apache ソフトウエア内に同梱されているものの、通常はサーバのバイナリファイル内に組み込まないものを意味します。openSUSE Leap では、これらについても共有オブジェクトとして提供され、実行時に別途読み込む仕組みになっています。

外部モジュール

外部モジュールとして位置づけられているモジュールは、公式の Apache ソフトウエア内には含まれていないものを意味します。openSUSE Leap では、これらのうちのいくつかを提供しています。

マルチプロセッシングモジュール (MPM)

MPM は Web サーバに対するリクエストを受け付け、処理するためのモジュールです。Web サーバソフトウエアの中枢部分でもあります。

24.4.1 モジュールのインストール

24.1.2項「インストール」の手順に従って既定のインストール作業を行うと、全ての基本モジュールと拡張モジュールのほか、マルチプロセッシングモジュールである Prefork MPM と、外部モジュール `mod_python` がそれぞれインストールされます。

追加の外部モジュールをインストールしたい場合は、YaST を起動して［ソフトウエア］›［ソフトウエア管理］を選択し、［フィルタ］›［検索］を選択して `apache` と入力し、検索を行ってください。これにより、利用可能な全ての外部 Apache モジュールなどを表示することができます。

24.4.2 有効化と無効化

モジュールの有効化と無効化は、手作業で実施することができるほか、YaST でも設定することができます。YaST では、24.2.3.1項「HTTP サーバウイザード」で説明しているとおり、スクリプト言語モジュール (PHP 8 および Python) を有効化もしくは無効化することができます。その他のモジュールの有効化と無効化については、24.2.3.2.2項「サーバモジュール」で説明しています。

モジュールの有効化や無効化を手作業で実施したい場合は、`a2enmod` モジュール名 もしくは `a2dismod` モジュール名 のコマンドをお使いください。`a2enmod -l` と入力して実行すると、現時点で有効化されている全てのモジュールを一覧表示することができます。

重要: 外部モジュールに対する設定ファイルの取り込み

外部モジュールを手作業で有効化した場合は、全ての仮想ホストの設定に対して、外部モジュールの設定ファイルが読み込まれていることをご確認ください。外部モジュールの設定ファイルは `/etc/apache2/conf.d/` 内に存在していて、既定では `/etc/apache2/default-server.conf` から読み込まれるようになっています。より細かい調整を行いたい場合は、`/etc/apache2/default-server.conf` 内にある外部モジュールに対する取り込み設定をコメントアウトして、特定の仮想ホストでのみ読み込むように設定してもかまいません。設定例については `/etc/apache2/vhosts.d/vhost.template` をお読みください。

24.4.3 基本モジュールと拡張モジュール

全ての基本モジュールと拡張モジュールについては、Apache のドキュメンテーション内に詳細説明が用意されています。下記では主要なモジュールに関する概要説明のみを行っています。それぞれのモジュールに関する詳細は、http://httpd.apache.org/docs/2.4/mod/ (https://httpd.apache.org/docs/2.4/mod/) をお読みください。

`mod_actions`

特定の MIME タイプ (たとえば `application/pdf` など) や特定の拡張子を持つファイル (たとえば `.rpm`) 、もしくは特定のリクエストメソッド (たとえば `GET`) が要求された際に、特定のスクリプトを実行する機能を提供します。このモジュールは既定で有効化されます。

`mod_alias`

`Alias` や `Redirect` のディレクティブを提供するモジュールです。これにより、特定の URL に対してディレクトリを割り当てる (`Alias` の名前の通り、別名を定義する) ことができるほか、他の場所に転送を行うことができます。このモジュールは既定で有効化されます。

`mod_auth*`

様々な認証方式を提供する認証モジュールです。`mod_auth_basic` はベーシック認証を、`mod_auth_digest` はダイジェスト認証をそれぞれ提供します。
なお、`mod_auth_basic` と `mod_auth_digest` は、`mod_authn_*` で表される認証プロバイダモジュールと、`mod_authz_*` で表される認可モジュールを組み合わせて使用します。たとえば `mod_authn_file` はテキストファイルに認証情報を保存するモジュールであるほか、`mod_authz_user` はユーザ認可を提供するモジュールです。
この分野の説明について、詳しくは https://httpd.apache.org/docs/2.4/howto/auth.html にある 認証、承認、アクセス制御 をお読みください。

`mod_auth_openidc`

`mod_auth_openidc` は、Apache HTTP サーバで OpenID Connect を使用するための唯一の認証された方式です。詳しくは https://openid.net/developers/certified-openid-connect-implementations/ をお読みください。

`mod_autoindex`

autoindex モジュールは、インデックスファイル (例: `index.html`) が存在しない場合に、ディレクトリ内の一覧を生成する機能を提供します。一覧の外観は様々に設定することができます。このモジュールについても既定で有効化されますが、ディレクトリ一覧の表示機能は `Options` ディレクティブで無効化されています。そのため、この機能が必要である場合は、お使いの仮想ホストの設定内で上書きしてお使いください。このモジュールの既定の設定ファイルは、`/etc/apache2/mod_autoindex-defaults.conf` に配置されています。

`mod_cgi`

`mod_cgi` は CGI スクリプトを実行する際に必要となるモジュールです。このモジュールは既定で有効化されます。

`mod_deflate`

このモジュールを使用すると、Apache が特定のファイルを配信する際、その場でデータ圧縮を行うことができるようになります。

`mod_dir`

`mod_dir` は `DirectoryIndex` ディレクティブを提供するモジュールで、ディレクトリそのものに対してリクエストが届いた際に、どのファイルを自動的に配信するかを指定します。既定値は `index.html` に設定されています。また、このモジュールは、ディレクトリに対するリクエストが届いた場合、末尾がスラッシュで終わっていないと、末尾のスラッシュを追加する (実際にはそのような URL に転送する) 機能も提供します。このモジュールも既定で有効化されます。

`mod_env`

CGI スクリプトや SSI ページに対して渡される、環境変数の制御を行います。環境変数は追加や削除を行うことができるほか、`httpd` プロセスから起動されるシェルからの値を渡すこともできます。このモジュールは既定で有効化されます。

`mod_expires`

`mod_expires` を使用すると、`Expires` ヘッダを送信して、プロキシサーバやブラウザのキャッシュ機能に対して、どれだけの時間で文書の期限が切れるのかを指定することができるようになります。このモジュールは既定で有効化されます。

`mod_http2`

`mod_http2` を利用することで、Apache は HTTP/2 プロトコルに対応するようになります。`VirtualHost` 内で `Protocols h2 http/1.1` と指定することで有効化することができます。

`mod_include`

`mod_include` は Server Side Includes (SSI) と呼ばれる機能を提供するモジュールで、HTML ページを動的に生成するための基本的な技術を提供します。このモジュールは既定で有効化されます。

`mod_info`

`http://localhost/server-info/` にアクセスすることで、サーバの設定概要を表示することができるようになるモジュールです。セキュリティ上の理由から、この URL に対するアクセスは常に制限しておくべきです。既定では `localhost` からのみアクセスすることができます。また、`mod_info` は `/etc/apache2/mod_info.conf` で設定されています。

`mod_log_config`

このモジュールを使用することで、Apache のログファイルに出力される内容を制御することができます。このモジュールは既定で有効化されます。

`mod_mime`

このモジュールは、ファイル名の拡張子を利用することで、ファイルに対する正しい MIME ヘッダ (たとえば HTML 文書であれば `text/html`) を配信する仕組みを提供します。このモジュールは既定で有効化されます。

`mod_negotiation`

コンテンツネゴシエーションを行うために必要なモジュールです。詳しい説明は、http://httpd.apache.org/docs/2.4/content-negotiation.html (https://httpd.apache.org/docs/2.4/content-negotiation.html) をお読みください。このモジュールは既定で有効化されます。

`mod_rewrite`

`mod_alias` と同じような機能を提供するモジュールですが、より多くの機能が追加され、より柔軟な設定ができるようになっています。`mod_rewrite` を利用することで、複数のルールやリクエストヘッダをベースにして URL の転送を設定したりすることができます。

`mod_setenvif`

クライアント側からのリクエストに応じて環境変数を設定するモジュールです。これにはたとえば、クライアントが送信したブラウザ文字列やクライアントの IP アドレスなどが含まれます。このモジュールは既定で有効化されます。

`mod_spelling`

`mod_spelling` は、大文字と小文字の違いなどの入力ミスを自動的に修正しようと試みるモジュールです。

`mod_ssl`

ブラウザとクライアントとの間で、暗号化した接続を実現するモジュールです。詳しくは 24.6項「SSL を利用した暗号化機能付き Web サーバの設定」をお読みください。このモジュールは既定で有効化されます。

`mod_status`

`http://localhost/server-status/` にアクセスすることで、サーバの動作状況と性能情報を表示することができるようになるモジュールです。セキュリティ上の理由から、この URL に対するアクセスは常に制限しておくべきです。既定では `localhost` からのみアクセスすることができます。また、`mod_status` は `/etc/apache2/mod_status.conf` で設定されています。

`mod_suexec`

`mod_suexec` は、CGI スクリプトを異なるユーザやグループで実行することができるモジュールです。このモジュールは既定で有効化されます。

`mod_userdir`

`~ユーザ名/` の形式で、ユーザ別のディレクトリを有効化するモジュールです。このモジュールを利用するには、`UserDir` ディレクティブを設定しなければなりません。このモジュールは既定で有効化されます。

24.4.4 マルチプロセッシングモジュール (MPM)

openSUSE Leap では、Apache で利用することのできるマルチプロセッシングモジュールが 2 種類用意されています:

- prefork MPM
- worker MPM

24.4.4.1 prefork MPM

prefork MPM はスレッドを使用しない、事前 fork 型の Web サーバです。Apache バージョン 1.x に似た動作の Web サーバになります。このバージョンでは、それぞれのリクエストと処理を個別の子プロセスに分離して処理します。これにより、何らかの問題が発生しても、他のプロセスには影響を及ぼすことがなく、Web サーバの機能停止を防ぐことができるようになっています。

このようなプロセスベースの仕組みでは安定性がもたらされるものの、worker MPM よりはより多くのシステム資源を消費してしまいます。また、Unix ベースのオペレーティングシステムでは、prefork MPM を既定の MPM としています。

重要: 本文書内での MPM について

この文書内では、Apache 内で prefork MPM を使用することを前提に説明しています。

24.4.4.2 worker MPM

worker MPM はマルチスレッド型の Web サーバです。スレッドはプロセスよりもずっと「軽量」な仕組みであるため、より少ない資源消費が特長になります。この MPM では、子プロセスを起動する代わりに、それぞれのリクエストに対してスレッドを割り当てて処理を行います。また、場合によっては複数のプロセスを起動することもありますが、子プロセスも同様にマルチスレッドになります。このような仕組みにより、prefork MPM よりずっと少ない資源で Apache を動作させることができます。

worker MPM での最も大きな欠点は、その安定性です。何らかの理由でスレッド内のメモリが破壊されてしまうと、同じプロセスに属する全てのスレッドが影響を受けます。最も悪いケースでは、サーバ自身がクラッシュしてしまいます。特に、負荷の高い環境下で Common Gateway Interface (CGI) を使用した場合、システム資源との通信が失敗することによってサーバ内部エラーが発生することがあります。また、もう 1 つの欠点としては、全ての Apache モジュールがスレッド型の仕組みに対応しているというわけではない (スレッドセーフではない) というものがあります。これにより、worker MPM を利用できない場合もあります。

警告: MPM と PHP の関係について

PHP モジュールによっては、マルチスレッド環境での使用が安全ではない (スレッドセーフではない) ものがあります。そのため、`mod_php` を使用する場合は、Worker MPM を使用しないことを強くお勧めします。

24.4.5 外部モジュール

ここには、openSUSE Leap に同梱されている全ての外部モジュールが一覧で示されています。それぞれのモジュールのドキュメンテーションについては、それぞれのディレクトリをご覧ください。

`mod_apparmor`

`mod_php8` などで処理される CGI スクリプトに対して、AppArmor の制約を適用する Apache 向けのモジュールです。

パッケージ名: `apache2-mod_apparmor`
詳しい情報:『セキュリティ強化ガイド』

`mod_php8`

PHP はサーバサイドスクリプトで複数プラットフォームに対応する HTML 組み込み型のスクリプト言語です。

パッケージ名: `apache2-mod_php8`

設定ファイル: `/etc/apache2/conf.d/php8.conf`

`mod_python`

`mod_python` は Apache HTTP サーバ内に Python の機能を組み込むもので、性能を大きく向上させることができるほか、Web ベースのアプリケーションに柔軟性を与えます。

パッケージ名: `apache2-mod_python`
詳しい情報: `/usr/share/doc/packages/apache2-mod_python`

`mod_security`

`mod_security` は Web アプリケーションファイアウオールで、様々な範囲の攻撃から Web アプリケーションを保護します。HTTP のトラフィック監視やリアルタイム分析に使用することもできます。

パッケージ名: `apache2-mod_security2`
設定ファイル: `/etc/apache2/conf.d/mod_security2.conf`
詳しい情報: `/usr/share/doc/packages/apache2-mod_security2`
ドキュメンテーション: https://github.com/owasp-modsecurity/ModSecurity

24.4.6 コンパイル

知識のあるユーザであれば、Apache に対して独自のモジュールを作成して拡張することができます。Apache 向けのモジュールを開発したり、サードパーティ製のモジュールをコンパイルしたりしたい場合は、`apache2-devel` パッケージをインストールして、提供される開発ツールをお使いください。`apache2-devel` には `apxs2` ツールが含まれていますが、これは Apache 向けのモジュールをコンパイルする際に使用するツールです。

`apxs2` は、ソースコードからモジュールをコンパイルして、インストールする機能を提供します (設定ファイルへの必要な変更を含みます) 。コンパイルを行うと、動的共有オブジェクト (Dynamic Shared Objects; DSO) として作成され、Apache の起動時に読み込みができるようになります。

`apxs2` バイナリは、`/usr/sbin` ディレクトリ内に配置されます:

- `/usr/sbin/apxs2` は、任意の MPM で動作する拡張モジュールをコンパイルするためのツールです。インストールを行うと、そのモジュールは `/usr/lib64/apache2` 内にインストールされます。
- `/usr/sbin/apxs2-prefork` は、prefork MPM で動作する拡張モジュールをコンパイルするためのツールです。インストールを行うと、そのモジュールは `/usr/lib64/apache2-prefork` 内にインストールされます。
- `/usr/sbin/apxs2-worker` は、worker MPM で動作する拡張モジュールをコンパイルするためのツールです。インストールを行うと、そのモジュールは `/usr/lib64/apache2-worker` 内にインストールされます。

ソースコードからモジュールをコンパイルしてインストールし、有効化するには、下記のコマンドを実行します:

```
> sudo cd /モジュールのソースコードのパス
> sudo apxs2 -cia モジュール名.c
```

ここで、`-c` はモジュールのコンパイルを、`-i` はインストールを、`-a` は有効化をそれぞれ行うオプションです。`apxs2` に対するその他のオプションについては、`apxs2(1)` のマニュアルページをお読みください。

24.5 CGI スクリプトの有効化

Apache の汎用ゲートウエイインターフェイス (Common Gateway Interface (CGI)) を利用することで、プログラムやスクリプト (CGI スクリプト) を実行して動的なコンテンツを生成することができます。CGI スクリプトは任意のプログラミング言語を利用することができます。

CGI スクリプトが生成したコンテンツを Apache から配信できるようにするには、`mod_cgi` を有効化する必要があります。また、`mod_alias` もあわせて有効化する必要があります。いずれのモジュールとも既定で有効化されています。モジュールの有効化について、詳しくは 24.4.2項「有効化と無効化」をお読みください。

警告: CGI セキュリティ

サーバ内で CGI スクリプトを実行できるように設定すると、潜在的なセキュリティホールとなる可能性があります。詳しくは 24.8項「セキュリティ問題の回避」をお読みください。

24.5.1 Apache の設定

openSUSE Leap では、CGI スクリプトの実行は `/srv/www/cgi-bin/` ディレクトリ内でのみ許可されるように設定されています。この場所は CGI スクリプトを実行する場所として設定されているディレクトリでもあります。仮想ホストの設定を作成している (詳しくは 24.2.2.1項「仮想ホストの設定」) 場合で、仮想ホスト固有のディレクトリ内で CGI スクリプトを実行したい場合は、そのディレクトリでの CGI スクリプトの実行を許可するよう設定しなければなりません。

例 24.5: VIRTUALHOST CGI 設定

```
ScriptAlias /cgi-bin/ "/srv/www/www.example.com/cgi-bin/" ❶

<Directory "/srv/www/www.example.com/cgi-bin/">
 Options +ExecCGI ❷
 AddHandler cgi-script .cgi .pl ❸
 Require all granted ❹
</Directory>
```

❶ このディレクトリ内にある全てのファイルを、CGI スクリプトとして実行するよう指定しています。

❷ CGI スクリプトの実行を有効化しています。

❸ .pl と .cgi の拡張子を持つファイルを、CGI スクリプトとして扱うよう指定しています。こちらは必要に応じて変更してください。

❹ `Require` ディレクティブは、既定のアクセス状態を制御するためのディレクティブです。この場合、指定したディレクトリへのアクセスは制限無く許可されます。認証と認可について、詳しくは https://httpd.apache.org/docs/2.4/howto/auth.html をお読みください。

24.5.2 サンプルスクリプトの実行

CGI のプログラミングは「通常の」プログラミングとは異なり、冒頭に `Content-type: text/html` のような MIME タイプヘッダなどを送信しなければなりません。このヘッダはクライアントに送信されるもので、どのような種類のコンテンツを送信しているのかを表明しています。また、スクリプトが送信するコンテンツは、クライアント側 (たとえば Web ブラウザ) が解釈できるものでなければなりません。通常は HTML などですが、純粋なテキストや画像などでもかまいません。

`/usr/share/doc/packages/apache2/test-cgi` には、シンプルなテスト用スクリプトが用意されています。これは Apache パッケージの一部として提供されているもので、純粋なテキスト形式で環境変数を出力するものです。このスクリプトを `/srv/www/cgi-bin/` にコピーするか、もしくは仮想ホスト (`/srv/www/www.example.com/cgi-bin/`) などにコピーして、`test.cgi` のような名前に変更してください。また、ファイルの冒頭に `#!/bin/sh` を追加してください。

また、Web サーバからアクセスできるようにするには、所有者が `root` である必要もあります。詳しくは 24.8項「セキュリティ問題の回避」をお読みください。なお、Web サーバは異なるユーザとして動作する仕組みであるため、CGI スクリプトは全てのユーザが実行および読み込みできるものでなければなりません。必要なアクセス権を適用するには、CGI のディレクトリに移動して `chmod 755 test.cgi` を実行してください。

これで `http://localhost/cgi-bin/test.cgi` または `http://www.example.com/cgi-bin/test.cgi` にアクセスすることで、CGI スクリプトを実行することができます。「CGI/1.0 test script report」のような画面が表示されるはずです。

24.5.3 CGI トラブルシューティング

テストプログラムの出力が正しく表示されず、エラーメッセージが表示されてしまった場合は、下記をご確認ください:

CGI トラブルシューティング

- 設定の変更後、設定を反映させるために再読み込みを実施しましたか? 実施していない場合は、`systemctl reload apache2.service` で再読み込みを実施してください。
- 独自の CGI ディレクトリを設定している場合は、その設定が適切に行われていますか? 何か不安な点がある場合は、既定の CGI ディレクトリである `/srv/www/cgi-bin/` にスクリプトをコピーして、`http://localhost/cgi-bin/test.cgi` にアクセスしてみてください。
- ファイルのアクセス権は正しく設定されていますか? CGI のディレクトリに移動して、`ls -l test.cgi` を実行してください。出力の冒頭は下記のようになっているはずです:

  ```
  -rwxr-xr-x  1 root root
  ```

- スクリプトにプログラミングエラーが無いことをご確認ください。`test.cgi` を変更せずにそのまま使用している場合は問題なく動作するはずですが、独自に作成したプログラムを使用している場合は、プログラミングに問題がないことをよく確認する必要があります。

24.6 SSL を利用した暗号化機能付き Web サーバの設定

クレジットカード情報などの機密データを Web サーバとクライアントとの間でやりとりするにあたっては、認証機能付きの暗号化接続を行って機密を保持する必要があります。`mod_ssl` では Secure Sockets Layer (SSL) や Transport Layer Security (TLS) を利用して、クライアントと Web サー

バとの間の HTTP 通信に対して強力な暗号化機能を提供します。つまり、TLS/SSL を利用することで Web サーバとクライアントの間の通信を盗聴できなくすることになります。もちろんデータの整合性についても確認が行われますし、クライアントとサーバはお互いに認証することができます。

このような目的から、サーバは要求された URL に応答する前に、サーバ自身の正しい識別情報を示す SSL 証明書を送信します。これによりクライアント側は、通信している相手が間違いのないものであることを確認できることになります。これに加えて、証明書は暗号化接続を生成することができますので、クライアントとサーバとの間でやり取りされる情報が、純粋なテキストのように漏洩することがなくなります。

`mod_ssl` は TLS/SSL プロトコルを直接実装しているわけではありません。その代わりに、Apache と SSL ライブラリとの仲介インターフェイスとして動作します。openSUSE Leap では OpenSSL ライブラリを使用します。OpenSSL は、Apache をインストールすると自動的にインストールされます。

Apache で `mod_ssl` を利用すると、目に見える違いとしては URL が `http://` ではなく `https://` になります。

24.6.1 SSL 証明書の作成

Web サーバで TLS/SSL を使用するには、まず SSL 証明書を作成する必要があります。この証明書は Web サーバとクライアントとの間で識別を行うのに必要となるものです。なお、証明書の正当性を担保するには、各ユーザが信頼する団体で署名してもらわなければなりません。

作成できる証明書には、3 つの種類があります。1 つ目は「テスト」証明書で、テスト用にのみ使用することができます。2 つ目は自己署名証明書と呼ばれるもので、あなた自身を信頼する特定範囲のユーザのみ利用できるものです。3 つ目は公的な証明書で、証明機関 (Certificate Authority (CA)) と呼ばれる独立した第三者機関が署名する正式な証明書です。

証明書の作成は 2 段階の手順から構成されています。最初に証明機関の機密鍵を作成し、次にその鍵でサーバ証明書に署名を行います。

ヒント: さらなる情報

TLS/SSL の考え方や定義についての詳細は、https://httpd.apache.org/docs/2.4/ssl/ssl_intro.html をお読みください。

24.6.1.1 「テスト」証明書の作成

テスト用の証明書を作成するには、`/usr/bin/gensslcert` を実行します。これを実行すると、下記に示すファイルを作成もしくは上書きします。なお、`gensslcert` のオプションスイッチを指定することで、証明書の詳細を調整することができます。詳しくは `/usr/bin/gensslcert -h` を実行してください。

- `/etc/apache2/ssl.crt/ca.crt`
- `/etc/apache2/ssl.crt/server.crt`
- `/etc/apache2/ssl.key/server.key`
- `/etc/apache2/ssl.csr/server.csr`

`ca.crt` のコピーを `/srv/www/htdocs/CA.crt` に配置して、ダウンロードできるようにしておくとよいでしょう。

重要: テスト用途のみでの使用について

テスト証明書は本番環境で使用すべきではありません。テスト目的にのみお使いください。

24.6.1.2 自己署名証明書の作成

イントラネット環境や特定の範囲のユーザにのみ公開する Web サーバを構築する際は、自分自身が証明機関 (CA) になって証明書に署名を付与し、使用すれば十分な場合があります。ただし、これは公的な証明書ではなく、Web ブラウザが証明書を識別できないことから、ユーザがアクセスすると「安全な接続ではありません」のような警告メッセージが表示されることに注意してください。

重要: 自己署名証明書について

自己署名証明書は、あなた自身を証明機関として信頼してくれる特定の範囲のユーザに対して提供する Web サーバにのみお使いください。たとえば一般向けの販売サイトなどでは、このような証明書を使用しないことをお勧めします。

まずは証明書署名要求 (Certificate Signing Request (CSR)) を生成します。この作業では `openssl` を利用し、`PEM` 形式で作成を行います。また、この手順ではパスフレーズのほか、いくつかの質問に回答する必要があります。パスフレーズは今後入力を求められることになりますので、忘れずに記憶しておいてください。

```
> sudo openssl req -new > new.cert.csr
Generating a 1024 bit RSA private key
```

```
..++++++
.........++++++
writing new private key to 'privkey.pem'
Enter PEM pass phrase:❶
Verifying - Enter PEM pass phrase:❷
-----
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [AU]:❸
State or Province Name (full name) [Some-State]:❹
Locality Name (eg, city) []:❺
Organization Name (eg, company) [Internet Widgits Pty Ltd]:❻
Organizational Unit Name (eg, section) []:❼
Common Name (for example server FQDN, or YOUR name) []:❽
Email Address []:❾

Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []:❿
An optional company name []:⓫
```

❶ パスフレーズを入力します。

❷ 再度パスフレーズを入力します (パスフレーズは忘れてはなりません) 。

❸ 2 文字の国コードを入力します。日本であれば `JP` を入力します。

❹ 通常は州名を入力します。日本の場合は都道府県名を一般に入力します。

❺ 市区町村名を入力します。たとえば `Chiyoda-ku` のように入力します。

❻ 所属する団体名を入力します。

❼ 組織単位を入力します。組織単位がない場合は、何も入力しなくてかまいません。

❽ サーバのドメイン名、もしくはあなた自身のフルネームを入力します。

❾ 電子メールアドレスを入力します。

❿ 必要に応じてチャレンジパスワードを入力します。ここで何らかのパスワードを指定した場合は、Apache Web サーバを再起動するたびに入力を求められます。

⓫ 必要であれば企業名を入力します。不要であれば何も入力しなくてかまいません。

これで証明書を生成できるようになります。再度 `openssl` を使用して、既定の `PEM` 形式で証明書を生成します。

手順 24.3: 証明書の生成

1. 機密鍵を `new.cert.key` にエクスポートします。証明書署名要求 (Certificate Signing Request; CSR) でパスフレーズを入力している場合は、ここで入力を求められます。

```
> sudo openssl rsa -in privkey.pem -out new.cert.key
```

2. 署名要求で入力した情報に従って証明書の公開部分を生成します。`-days` オプションは、証明書の期限が切れるまでの日数を指定します。なお、期限切れになる前でも、証明書を取り消したり新しいものに置き換えたりすることができます。

```
> sudo openssl x509 -in new.cert.csr -out new.cert.cert -req \
-signkey new.cert.key -days 365
```

3. 生成された証明書ファイルを、Apache から読み取ることのできるディレクトリにコピーします。なお、機密鍵である `/etc/apache2/ssl.key/server.key` ファイルは全てのユーザが読み込めるようにしてはなりませんが、公開鍵である `/etc/apache2/ssl.crt/server.crt` については、全てのユーザが読み込めるようにしておく必要があります。

```
> sudo cp new.cert.cert /etc/apache2/ssl.crt/server.crt
> sudo cp new.cert.key /etc/apache2/ssl.key/server.key
```

ヒント: 証明書の公開先について

最後に証明書ファイル `/etc/apache2/ssl.crt/server.crt` をユーザからアクセスできる場所にコピーして、各ユーザの Web ブラウザ内に取り込んで信頼してもらう作業が必要となります。この作業を実施しないと、ブラウザ側では未知の証明機関が発行した証明書であるものとして扱われてしまい、エラーまたは警告が表示されるようになってしまいます。

24.6.1.3 公的に署名された証明書の取得

一般的に利用可能な証明書を発行する機関は様々あります。証明書を公的に信頼されている機関に署名してもらうことで、Web サーバ自身を完全に信頼してもらうことができるようになります。また、一般的な SSL/TLS 対応のブラウザであれば、公的な証明機関の証明書を保持していますので、ここから正当性を確認することができます。一般的に使用される証明機関 (CA) の一覧については、https://en.wikipedia.org/wiki/Certificate_authority#Providers などをご覧ください。

公的に証明された証明書を取得するにあたっては、証明書そのものを CA に送信する必要はありません。その代わりに、証明書署名要求 (Certificate Signing Request; CSR) を送信します。CSR を作成するには、下記のコマンドを実行します:

```
> openssl req -new -newkey rsa:2048 -nodes -keyout newkey.pem -out newreq.pem
```

ここでは識別名 (Distinguished Name) の入力を求められるほか、国コードや組織名などのいくつかの質問に回答する必要があります。ここで入力した内容は証明書内に記載されるほか、証明機関が内容をチェックすることもあります。なお、全ての質問に回答する必要はありません。不要であると判断した項目に対しては、「.」とだけ入力して先に進めてください。

作成した CSR は、スクリプトを実行したディレクトリ内に出力されます。ファイル名は `newreq.pem` という名前になります。

24.6.2 SSL を利用する場合の Apache の設定

Web サーバで TLS/SSL を使用する場合、既定のポートは 443 が割り当てられています。TLS/SSL のない「通常の」Apache はポート 80 で、TLS/SSL を使用する場合の Apache は 443 を使用しますが、両方のサービスを単一の Apache インスタンスで賄うことができます。また、それぞれに対して別々の仮想ホストを設定して、提供する内容やサービスなどを変更したりすることもあります。

重要: ファイアウオールの設定

なお、Apache で SSL (ポート 443) を使用するにあたっては、ファイアウオールでポートを開くことを忘れずに実施してください。この作業は `firewalld` で行います。詳しくは『セキュリティ強化ガイド』、第24章「マスカレードとファイアウオール」、24.4.1項「コマンドラインからのファイアウオール設定」をお読みください。

SSL モジュールは、サーバ全体の設定の中で既定で有効化されています。有効化されていない場合は、`a2enmod ssl` で有効化することができます。また、SSL を有効化する場合は、フラグ「SSL」を設定する必要があります。フラグを設定するには、`a2enflag SSL` (大文字であることに注意してください) を実行します。また、サーバ証明書を暗号化している場合は、`/etc/sysconfig/apache2` 内の `APACHE_TIMEOUT` の値を十分に大きな値にして、Apache 開始時のパスワード入力に十分な時間が確保されるようにしてください。最後にサーバを再起動すると、変更を反映させることができます。再読み込みでは不十分ですので、あらかじめご注意ください。

仮想ホストの設定では、`/etc/apache2/vhosts.d/vhost-ssl.template` の雛形に従って、いくつかの SSL 固有のディレクティブを追加する必要があります。雛形内には詳しく説明がありますので、こちらをお読みください。また、仮想ホストの設定についての一般的な情報は、24.2.2.1項「仮想ホストの設定」をお読みください。

初めて SSL を設定する場合は、雛形を `/etc/apache2/vhosts.d/仮想ホスト名.conf` にコピーして、このファイルを編集していってください。それぞれ下記のディレクティブ部分を調整すれば、サーバを開始することができます:

- `DocumentRoot`
- `ServerName`
- `ServerAdmin`
- `ErrorLog`
- `TransferLog`

24.6.2.1 名前ベースの仮想ホストと SSL の関係

既定では、1 つの IP アドレスしか設定されていないサーバの場合、SSL が有効化された仮想ホストを複数実行することはできません。名前ベースの仮想ホストでは、Apache 側に仮想ホストのホスト名を設定し、Apache がリクエストの内容を読み込んで仮想ホストを判別しますが、SSL 接続の処理は Apache がリクエストを処理するよりも前に行われるため、1 つの IP アドレスに対して SSL の設定を複数持たせることができず、これによって常に既定の仮想ホストのみが使われる結果になってしまいます。これにより、証明書に書かれているサーバ名と、実際にアクセスしているサーバ名が異なる、という警告メッセージをユーザが受け取る結果にもなってしまいます。

そのため、openSUSE Leap では、サーバ名表示 (Server Name Indication (SNI)) と呼ばれる SSL プロトコルの拡張に対応するようになっています。これは、SSL の処理内で仮想ホストのドメイン名を送信するための仕組みで、これによってサーバは、SSL の処理を行う時点で仮想ホストの判別ができるようになります。そのため、サーバ側は早期に適切な仮想ホストに「切り替える」ことができるようなり、ブラウザ側にも正しい証明書を配信できるようになります。

SNI は openSUSE Leap では既定で有効化されています。SSL を利用して名前ベースの仮想ホストを設定するには、24.2.2.1.1項「名前ベースの仮想ホスト」の手順に従って設定を行ってください (ただし、SSL を利用するには、ポート `80` ではなくポート `443` を使用するように設定する必要があります) 。

重要: SNI 対応ブラウザについて

SNI はクライアント側でも対応している必要がありますが、現時点では特定の古いブラウザを除いて、ほとんどのブラウザで対応するようになっています。詳しくは https://ja.wikipedia.org/wiki/Server_Name_Indication#%E5%AF%BE%E5%BF%9C%E7%8A%B6%E6%B3%81 をお読みください。

SNI 対応のブラウザに対する処理を設定するには、`SSLStrictSNIVHostCheck` というディレクティブを設定します。この値をサーバ全体の設定で `on` にすると、全ての仮想ホストに対して SNI 非対応のブラウザからのアクセスを拒否するようになります。`VirtualHost` ディレクティブ内で `on` に設定すると、特定の仮想ホストのみが拒否されるようになります。

サーバ全体の設定で `off` に設定した場合、サーバ側は SNI への対応を全く行わなくなります。この場合、SSL のリクエストが届くと、設定された 最初の 仮想ホスト (ポート: 443) にアクセスがあったものとして扱われます。

24.7 同一サーバ内での複数 Apache インスタンスの起動

同一のサーバ内で複数の Apache インスタンスを動作させると、複数の仮想ホストを動作させるのに比べて、いくつかのメリットが生まれます (詳しくは 24.2.2.1項「仮想ホストの設定」をお読みください) 。

- しばらくの間だけ仮想ホストを無効化したい場合、通常であれば Web サーバの設定を変更して、その変更を反映させるために再起動を行う必要があります。
- いずれかの仮想ホストで何らかの障害が発生した場合、全ての仮想ホストを再起動する必要があります。

既定の Apache インスタンスについては、通常通り開始することができます:

```
> sudo systemctl start apache2.service
```

上記を実行すると、既定の `/etc/sysconfig/apache2` ファイルを読み込みます。このファイルが存在しない場合や、`APACHE_HTTPD_CONF` 変数が設定されていない場合は、`/etc/apache2/httpd.conf` を読み込みます。
もう 1 つの Apache インスタンスを開始したい場合は、下記のように実行します:

```
> sudo systemctl start apache2@インスタンス名
```

たとえば、下記のようになります:

```
> sudo systemctl start apache2@example_web.org
```

既定では、このインスタンスは `/etc/apache2@example_web.org/httpd.conf` というファイル名の設定ファイルを読み込みます。このファイル名の指定は、`/etc/sysconfig/apache2@example_web.org` というファイル名の `APACHE_HTTPD_CONF` の設定で変更することができます。

Apache の追加インスタンスを設定するには、下記の手順を実施します。なお、全てのコマンドを `root` で実行することに注意してください。

手順 24.4: 追加の APACHE インスタンスの設定

1. まずは新しい設定ファイルを、`/etc/sysconfig/apache2` にあるものをベースにして作成します。たとえば `/etc/sysconfig/apache2@example_web.org` を作成します:

   ```
   > sudo cp /etc/sysconfig/apache2 /etc/sysconfig/apache2@example_web.org
   ```

2. `/etc/sysconfig/apache2@example_web.org` ファイルを編集し、下記の変数を修正します:

   ```
   APACHE_HTTPD_CONF
   ```

   上記を、下記のように変更します:

   ```
   APACHE_HTTPD_CONF="/etc/apache2/httpd@example_web.org.conf"
   ```

3. 次に `/etc/apache2/httpd.conf` ファイルをベースにして、`/etc/apache2/httpd@example_web.org.conf` ファイルを作成します。

   ```
   > sudo cp /etc/apache2/httpd.conf /etc/apache2/httpd@example_web.org.conf
   ```

4. `/etc/apache2/httpd@example_web.org.conf` ファイルを編集し、下記を変更します:

   ```
   Include /etc/apache2/listen.conf
   ```

   上記を、下記のように変更します:

   ```
   Include /etc/apache2/listen@example_web.org.conf
   ```

   その他のディレクティブを確認して、必要であれば変更します。おそらくは下記のディレクティブを変更する必要があるものと思われます:

   ```
   Include /etc/apache2/global.conf
   ```

   上記を変更して、新しいインスタンス用に `global@example_web.org.conf` ファイルを作成します。また、下記についても変更しておくことをお勧めします:

   ```
   ErrorLog /var/log/apache2/error_log
   ```

上記を、下記のように変更します:

```
ErrorLog /var/log/apache2/error@example_web.org_log
```

これにより、各インスタンスのログを別々にすることができます。

5. さらに `/etc/apache2/listen.conf` をベースにして、`/etc/apache2/listen@example_web.org.conf` ファイルを作成します。

```
> sudo cp /etc/apache2/listen.conf /etc/apache2/listen@example_web.org.conf
```

6. `/etc/apache2/listen@example_web.org.conf` を編集して、下記を変更します:

```
Listen 80
```

上記を新しいインスタンスのポート番号に変更します。たとえば 82 であれば、下記のようになります:

```
Listen 82
```

新しい Apache インスタンスで SSL/TLS を利用する場合 (詳しくは 24.6項「SSL を利用した暗号化機能付き Web サーバの設定」をお読みください) は、下記の行についても変更する必要があります:

```
Listen 443
```

上記を、たとえば下記のように変更します:

```
Listen 445
```

7. あとは新しい Apache インスタンスを開始します:

```
> sudo systemctl start apache2@example_web.org
```

8. 最後にお使いの Web ブラウザから、サーバが動作していることを確認します。ここまでの設定を行っていれば、`http://server_name:82` でアクセスできるはずです。なお、新しいインスタンスで個別のエラーログを出力するように設定している場合は、下記のようにして確認することができます:

```
> sudo tail -f /var/log/apache2/error@example_web.org_log
```

同一のサーバで複数の Apache インスタンスを動作させるにあたっては、下記の点を考慮する必要があります:

- `/etc/sysconfig/apache2@インスタンス名` ファイルでは、`/etc/sysconfig/apache2` と同じ値を設定することができます。これにより、モジュールの読み込みや MPM の設定を同じにすることができます。
- また、新しい Apache インスタンスは、既定の Apache インスタンスが動作していなくても開始することができます。
- Apache のヘルパーユーティリティである `a2enmod`, `a2dismod`, `apachectl` は、`HTTPD_INSTANCE` 環境変数を指定していない場合、いずれも既定の Apache インスタンスに対して操作を行います。インスタンス名を指定する場合は、たとえば下記のようにして実行します:

```
> sudo export HTTPD_INSTANCE=example_web.org
> sudo a2enmod access_compat
> sudo a2enmod status
> sudo apachectl start
```

上記を実行すると、`/etc/sysconfig/apache2@example_web.org` にある `APACHE_MODULES` の値に対して、`access_compat` と `status` を追加し、`example_web.org` のインスタンスを起動します。

24.8 セキュリティ問題の回避

Web サーバは全世界のインターネットに公開される仕組みであるため、常に管理面の労力を必要とします。ソフトウエアに起因するセキュリティ問題のほか、設定ミスによる情報漏洩なども必然的に発生しうる問題となります。ここでは、これらの問題を解決するためのいくつかのヒントを示しています。

24.8.1 最新版のソフトウエアの使用

Apache ソフトウエアに対して脆弱性が発見されると、SUSE はセキュリティ勧告を発信します。この勧告には詳細の説明のほか、脆弱性の修正手順が含まれています。SUSE のセキュリティ勧告は、下記の場所で公開されます (いずれも基本的には英語のみでの提供となります):

- Web ページ: https://www.suse.com/support/security/
- メーリングリスト: https://lists.opensuse.org/archives/list/security-announce@lists.opensuse.org/
- セキュリティアナウンス: https://www.suse.com/support/update/

24.8.2 DocumentRoot のアクセス権

openSUSE Leap では、`DocumentRoot` で指定されている `/srv/www/htdocs` ディレクトリと、CGI のディレクトリ `/srv/www/cgi-bin` については、`root` のユーザおよびグループが設定されています。これらのアクセス権については、変更すべきではありません。これらのディレクトリが全てのユーザに対して書き込み可能になってしまうと、どのユーザからでもファイルを保存できることになってしまいます。ここに配置されたファイルは、Apache 側では `wwwrun` というユーザの権限で実行されるため、場合によっては予期しないファイルシステムへのアクセスが為されてしまうことがあります。また、仮想ホストに対して `DocumentRoot` や CGI のディレクトリを設定する場合は、`/srv/www` 以下のサブディレクトリを使用するものとし、それらのディレクトリに対して、`root` のユーザおよびグループに所属させるようにしてください。

24.8.3 ファイルシステムのアクセス権

既定では、ファイルシステム全体へのアクセスは `/etc/apache2/httpd.conf` で禁止されるようになっています。これらのディレクティブについては、削除したりしてはなりません。Apache が読み込む必要のあるディレクトリに限定して、アクセスを許可するよう設定してください。詳しくは 24.2.2.1.3項「基本的な仮想ホストの設定」をお読みください。また、指定したディレクトリ内には、パスワードやシステムの設定ファイルなど、機密を確保する必要のあるファイルを配置していないことをご確認ください。誤って配置してしまうと、外部から読み取られてしまう危険性があります。

24.8.4 CGI スクリプト

PHP や SSI などのプログラミング言語で書かれた対話的なスクリプトは、本質的には任意のコマンドを実行することができる仕組みであるため、一般的なセキュリティ問題となりうる存在であります。サーバが実行するスクリプトについては、システム管理者が信頼する情報源から得たスクリプトだけになるようにすべきです。一般ユーザに対してスクリプトを作成させて実行できるようにするのは、適切ではありません。また、全てのスクリプトに対して、監査の仕組みを用意しておくことをお勧めします。

スクリプトの管理をできるだけ簡単に行いたい場合は、CGI スクリプトの実行をサーバ全体で許可するのではなく、特定のディレクトリに限定して許可するようにするとよいでしょう。それぞれ `ScriptAlias` と `Option ExecCGI` のディレクティブで設定することができます。openSUSE Leap の既定の設定では、どこでも CGI スクリプトを実行できるような許可は与えていません。

また、CGI スクリプトは全て同じユーザで実行されるため、場合によってはスクリプト間で競合が発生することがあります。suEXEC モジュールを利用することで、CGI スクリプトを異なるユーザやグループで実行することができます。

24.8.5 ユーザディレクトリ

`mod_userdir` もしくは `mod_rewrite` モジュールを利用してユーザディレクトリを有効化した場合、`.htaccess` ファイルへのアクセスについては、禁止しておくことを強くお勧めします。これは、このファイルがセキュリティ設定を上書きできてしまう存在であるためです。少なくとも `AllowOverRide` を利用して、ユーザ側で設定できる項目を制限してください。openSUSE Leap では、`.htaccess` ファイルへのアクセスは既定で有効化されていますが、`mod_userdir` モジュールを利用した場合、`Option` ディレクティブによる設定の上書きは許可されないようになっています (詳しくは `/etc/apache2/mod_userdir.conf` 設定ファイルをお読みください) 。

24.9 トラブルシューティング

Apache が起動しない場合や Web ページにアクセスできない場合、もしくはユーザが Web サーバに接続できない場合は、問題の原因を突き止めることが重要です。ここでは、一般的なエラーの出力先と、主要なチェック項目について説明しています:

`apache2.service` サブコマンドの出力:

`/usr/sbin/apache2ctl` バイナリで Web サーバを開始したり停止したりするのではなく、`systemctl` コマンドをお使いください (詳しくは 24.3項「Apache の開始と停止」をお読みください) 。なお、`systemctl status apache2.service` と入力して実行すると、エラーに関する詳細な情報を出力することができます。ここから設定エラーを修正するためのヒントが得られます。

ログファイルと饒舌性

致命的なエラーであってもそうでなくても、Apache のログファイルを調査することで原因を調査することができます。既定の設定では、主なエラーログは `/var/log/apache2/error_log` に出力されます。これに加えて `LogLevel` ディレクティブを使用することで、ログファイル内により詳しい情報を出力することができます。

ヒント: シンプルなテストについて

まずは `tail -F /var/log/apache2/エラーログファイル名` を実行して、Apache のログファイルを監視してください。監視を続けた状態で `systemctl restart apache2.service` を実行し、ブラウザで接続してみて出力される内容を確認すると良いでしょう。

ファイアウオールとポート

良くあるミスとして、Apache が待ち受けているポートをファイアウオール側で開いていないという問題があります。YaST で Apache を設定した場合は、この問題に対応するためのオプションが用意されています (詳しくは 24.2.3項「YaST による Apache の設定」をお読みください) 。Apache を手作業で設定している場合は、YaST のファイアウオールモジュールを利用して、HTTP および HTTPS に対するファイアウオールのポートを開いてください。

これらのヒントを利用しても原因がわからない場合は、https://httpd.apache.org/bug_report.html にある Apache のバグデータベースを利用することもできます。これに加えて、Apache ユーザコミュニティのメーリングリストをお使いいただくこともできます。詳しくは https://httpd.apache.org/userslist.html をお読みください。

24.10 さらなる情報

`apache2-doc` パッケージには、様々な言語に対応した Apache の完全マニュアルが用意されています。このパッケージは既定ではインストールされていません。最も手短にインストールするには、`zypper in apache2-doc` コマンドをお使いください。インストールを行うと、Apache のマニュアルを `http://localhost/manual/` からアクセスできるようになります。また、オンライン版のマニュアルにアクセスするには、https://httpd.apache.org/docs/2.4/ をご利用ください。また、SUSE 固有の設定ヒントについては、`/usr/share/doc/packages/apache2/README.*` 内に用意されています。

24.10.1 Apache 2.4

Apache 2.4 での新機能の一覧については、https://httpd.apache.org/docs/2.4/new_features_2_4.html をお読みください。また、2.2 から 2.4 へのアップグレードに関する情報は、https://httpd.apache.org/docs/2.4/upgrading.html にて提供されています。

24.10.2 Apache モジュール

Apache の外部モジュールに関する情報については、概要説明が 24.4.5項「外部モジュール」に用意されています。より詳しくお読みになりたい場合は、下記の場所をご覧ください:

`mod_apparmor`

https://en.opensuse.org/SDB:AppArmor

`mod_php8`

https://www.php.net/manual/en/install.unix.apache2.php

`mod_php8` の設定に関する詳細な情報は、メインの設定ファイルである `/etc/php8/apache2/php.ini` 内にコメントとして記述されています。

`mod_python`

https://modpython.org/

`mod_security`

https://github.com/owasp-modsecurity/ModSecurity

24.10.3 開発

Apache のモジュール開発に関する情報や、Apache Web サーバ自身の開発に参加したい場合は、下記をお読みになることをお勧めします:

Apache 開発者向け情報

https://httpd.apache.org/dev/

Apache 開発者向けドキュメンテーション

https://httpd.apache.org/docs/2.4/developer/

25 YaST による FTP サーバの設定

改訂履歴

2024-05-13

YaST の［FTP サーバ］モジュールを利用することで、お使いのマシンを FTP (File Transfer Protocol) サーバとして動作させることができます。また、匿名ユーザだけでなく、認証を行ってマシンに接続し、FTP プロトコルを利用してファイルをダウンロードさせることもできます。設定にもよりますが、FTP サーバに対してファイルをアップロードすることもできます。なお、YaST では vsftpd (Very Secure FTP Daemon) を利用します。

お使いのシステムに YaST FTP サーバモジュールがインストールされていない場合は、`yast2-ftp-server` パッケージをインストールしてください (コマンドラインで FTP サーバを管理したい場合は、1.4.3.7項「yast ftp-server」をお読みください) 。

YaST を利用して FTP サーバを設定するには、下記の手順で行います:

1. まずは YaST コントロールセンターを開いて［ネットワークサービス］›［FTP サーバ］を選択するか、もしくは `root` で `yast2 ftp-server` を実行します。

2. お使いのシステム内に FTP サーバがインストールされていない場合は、YaST の FTP サーバモジュールが開始した段階でインストールを行うよう求められます。vsftpd を選択して先に進めてください。

3. ［起動］ダイアログでは、FTP サーバの開始方法を設定することができます。詳しくは 25.1項「FTP サーバの開始」をお読みください。
   ［一般］ダイアログでは、FTP のディレクトリや 'ようこそ' メッセージ、ファイル作成時のマスクやその他のパラメータを設定することができます。詳しくは 25.2項「FTP の一般的な設定」をお読みください。
   ［パフォーマンス］ダイアログでは、FTP サーバの負荷制御に関わる設定を行うことができます。詳しくは 25.3項「FTP パフォーマンス設定」をお読みください。
   ［認証］ダイアログでは、FTP サーバを匿名でのみアクセスできるようにするか、もしくはユーザ認証を行ってアクセスできるようにするか、もしくはその両方を許可するかを選択することができます。詳しくは 25.4項「認証」をお読みください。
   ［詳細設定］ダイアログでは、FTP サーバの動作モードや SSL 接続の有効可否、およびファイアウオールの設定を行うことができます。詳しくは 25.5項「詳細設定」をお読みください。

4. ［完了］を押すと、設定を保存することができます。

25.1 FTP サーバの開始

［FTP の起動］ダイアログ内の［サービスの開始］の枠内では、FTP サーバの開始方法を設定することができます。システムの起動時にサーバを自動的に開始するか、もしくは手作業で開始するかの 2 つから選択することができます。このほか、FTP 接続要求があった場合に FTP サーバを開始したい場合は、［ソケットを使用する］を選択することもできます。

FTP サーバの現在の状態は、［FTP の起動］ダイアログ内の［開始／停止］枠内に表示されています。［FTP を今すぐ開始する］を押すと、FTP サーバを即時に開始することができます。また、［FTP を停止する］を押すと、FTP サーバを停止することができます。サーバの設定を変更している場合は、［設定を保存して FTP を再起動する］を押してください。また、［完了］を押して設定を終了しても、設定を保存することができます。

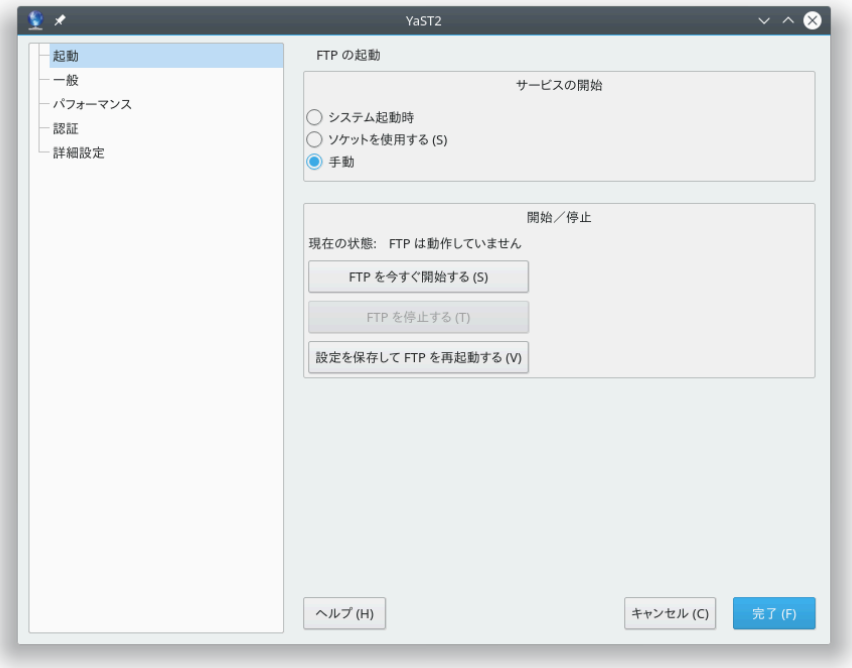


図 25.1: FTP サーバの設定 - 起動

25.2 FTP の一般的な設定

［FTP の一般的な設定］ダイアログ内にある［一般的な設定］の枠内では、FTP サーバの接続時に表示される［'ようこそ' メッセージ］を設定することができます。

また、［全員を chroot ］を選択すると、それぞれのユーザがログインすると、ホームディレクトリ内の chroot jail に閉じこめられることになります。この選択については、特にアップロードやシェルへのアクセスを使用したりするような場合に問題を引き起こす可能性がありますので、注意してお使いください。

［詳細なログ記録］を選択すると、全ての FTP 要求とその応答がログに記録されるようになります。

また、匿名ユーザや認証済みユーザが作成したファイルに対して、umask を利用してアクセス権を制限することができます。［匿名ユーザの umask］では匿名ユーザがファイルを作成した場合のマスク値を、［認証ユーザの umask］では認証済みのユーザがファイルを作成した場合のマスク値を、それぞれ設定することができます。なお、マスク値は 8 進数で指定するものとし、冒頭に 0 を付けてください。umask に関する詳細は、umask のマニュアルページ (`man 1p umask`) をお読みください。

［FTP ディレクトリ］の枠内では、匿名ユーザと認証ユーザのそれぞれが使用するディレクトリを設定することができます。［参照］ボタンを押すことで、ローカルのファイルシステム内からディレクトリを選択することができます。匿名ユーザに対する既定の FTP ディレクトリは `/srv/ftp` です。なお、vsftpd ではこのディレクトリに対して、全てのユーザから書き込むことができるように設定することはできません。その代わりに、匿名ユーザがアップロードすることのできるディレクトリとして、`upload` というディレクトリを作成してください。

25.3 FTP パフォーマンス設定

［パフォーマンス］ダイアログでは、FTP サーバの負荷制御に関わるパラメータを設定することができます。［最大無通信時間］は、クライアントから FTP のコマンドを受け取ってから、次のコマンドを受け取るまでの最大時間を分単位で指定します。ここで指定した値よりも長い時間、何もコマンドを受け取らないと、クライアントとの接続を切断します。［IP アドレスあたりの最大接続数］では、1 つの IP アドレスから接続することのできる最大接続数を指定することができます。また、［最大接続数］では、接続可能な最大数を設定することができます。いずれの場合も、最大接続数を超過するとアクセスが拒否されます。

［認証ユーザの最大通信速度］では、ローカルで認証したユーザの場合の最大通信速度を KB/s 単位で設定することができます。また、［匿名ユーザの最大通信速度］では、匿名ユーザの最大通信速度を KB/s 単位で設定することができます。いずれの設定とも既定値は `0` で、無制限の通信速度を許可するようになっています。

25.4 認証

［認証］ダイアログ内の［匿名ユーザと認証ユーザの許可］の枠内では、FTP サーバにアクセスすることを許可するユーザを設定することができます。ここでは、匿名ユーザのみに許可するか、認証ユーザ (システムにアカウントのあるユーザ) のみに許可するか、その両方に許可するかを選択することができます。

FTP サーバに対してユーザからファイルのアップロードを許可するには、［認証］ダイアログ内の［アップロード］の枠内にある［アップロードの許可］を選択します。ここでは対応するチェックボックスで、匿名ユーザに対してファイルのアップロードやディレクトリの作成を許可するかどうかを選択することができます。

注記: vsftp - 匿名ユーザのアップロード許可

vsftpd サーバを利用していて、匿名ユーザに対してファイルのアップロードやディレクトリの作成を許すようにするには、匿名 FTP ディレクトリ内に全てのユーザから書き込むことのできるサブディレクトリを作成する必要があります。

25.5 詳細設定

FTP サーバにはアクティブモードとパッシブモードが存在します。既定では、サーバはパッシブモードで動作します。アクティブモードに切り替えるには、［詳細設定］ダイアログ内の［パッシブモードを許可する］のチェックを外してください。また、［パッシブモード時の最小ポート番号］と［パッシブモード時の最大ポート番号］の値を調整することで、サーバが使用するポートの範囲を変更することもできます。

また、サーバ・クライアント間の通信を暗号化したい場合は、［SSL を有効にする］および［TLS を有効にする］を選択してください。このとき、暗号化接続に使用する RSA 証明書を指定する必要があります。

重要

なお、既定の `vsftpd` デーモンでは、バージョン 1.2 より前の TLS プロトコルが無効化されています。古いバージョンの TLS プロトコルを使用する FTP クライアントからの接続を受け付ける必要がある場合は、`/etc/vsftpd.conf` ファイルに下記の設定を追加する必要があります:

```
ssl_tlsv1 = YES
ssl_tlsv1_1 = YES
```

設定変更後は `vsftpd` デーモンを再起動して、設定の再読み込みを行います:

```
> sudo systemctl restart vsftpd.service
```

また、お使いのシステムをファイアウオールで保護している場合は、FTP サーバへの接続を許可するため、［ファイアウオールでポートを開く］にチェックを入れてください。

25.6 さらなる情報

FTP サーバに関して、より詳しい情報は、`vsftpd` および `vsftpd.conf` の各マニュアルページをお読みください。

26 squid: キャッシュ機能付きプロキシサーバ

改訂履歴

2025-04-09

squid は Linux プラットフォームや Unix プラットフォームで幅広く使用されている、キャッシュ機能付きプロキシサーバです。Web や FTP サーバなどのインターネット側のオブジェクトに対して、元のサーバよりも端末側に近いマシンが代行してアクセスを行い、それらを保管しておくことができます。また、このようなプロキシを複数階層の構成にして、応答時間の削減やネットワーク負荷の軽減などを行うこともできます。このようにして複数階層の構成にしても、エンドユーザからその構造が見えることはなく、透過的に動作させることができます。

squid はプロキシキャッシュとして動作します。クライアント (一般的には Web ブラウザ) からのオブジェクト要求をサーバに転送します。相手側のサーバから必要なオブジェクトが届くと、それをクライアントに対して配信するとともに、ハードディスク内のキャッシュとして、そのコピーを保存しておきます。このようなキャッシュの仕組みにより、複数のクライアントが同じオブジェクトを要求した場合に、ハードディスク内のキャッシュからオブジェクトを提供できるようになります。これはインターネットに直接アクセスするよりもずっと高速であるだけでなく、ネットワーク負荷の軽減にも繋がります。

squid では、キャッシュ機能のほかに、下記のような幅広い機能を提供します:

- プロキシサーバ間の相互通信による負荷分散
- プロキシにアクセスする全てのクライアントに対して適用できる、厳密なアクセス制御リスト
- 他のアプリケーションを併用することで実現できる、特定の Web ページへのアクセス許可やアクセスの禁止
- Web アクセスの傾向を確認するための Web ページ統計情報機能

squid は汎用のプロキシサーバではありません。通常は HTTP 接続のみを取り扱います。FTP, Gopher, SSL, WAIS などのプロトコルにも対応していますが、news プロトコルやビデオ会議プロトコルなどの他のインターネットプロトコルには対応していません。また squid では、UDP プロトコルへの対応はプロキシキャッシュ間の通信にのみ対応しているため、ほとんどのマルチメディアプログラムにも対応していません。

26.1 プロキシサーバについて

squid はキャッシュ機能付きプロキシサーバとして、様々な用途に使用することができます。たとえばファイアウオールと併用すると、セキュリティを高めることができます。また、プロキシは複数を組み合わせて使用することもできます。また、キャッシュ対象とするオブジェクトの種類を指定したり、どれだけの期間保存しておくのかを指定したりすることもできます。

26.1.1 squid とセキュリティ

squid とファイアウオールを併用することで、外部の攻撃から内部ネットワークを保護することができるようになります。ファイアウオールでは、squid を除いて全てのクライアントからの外部サービスアクセスを拒否するように設定することで、全ての Web 接続をプロキシ経由で行わせることができます。これにより、squid で Web アクセスの全てを制御できるようになります。

ファイアウオールの設定内に DMZ の構成が含まれる場合、プロキシは DMZ 内で動作させるべきです。26.6項「透過型プロキシの設定」では、透過型 プロキシを実装するための手順を説明しています。この透過型プロキシの構成により、クライアント側ではプロキシに関する設定を行う必要がなくなりますので、設定をより簡単に済ませることができます。

26.1.2 複数のキャッシュ

複数の squid インスタンスを動作させて、お互いにオブジェクトを交換できるように設定することもできます。これによりシステム全体の負荷を軽減し、ローカルネットワーク内でオブジェクトの取得を完了させる機会を増やすことができます。また、キャッシュには階層構造を設定することもできます。これにより、オブジェクトへの要求を兄弟関係にあるキャッシュに転送したり、親キャッシュに転送したりすることができるようになります。これにより、ローカルネットワーク内の他のキャッシュからオブジェクトを提供するか、直接情報源から取得するかを効率的に選択することができるようになります。

キャッシュの階層構造を構築するにあたっては、適切なトポロジ選択が重要です。なぜなら、ネットワーク内のトラフィック量を減らすことが、プロキシキャッシュの目的の 1 つであるからです。巨大なネットワークの場合、それぞれのサブネット内にそれぞれプロキシサーバを設置して、それらは親プロキシサーバに接続して、親プロキシサーバからインターネットに直接アクセスするような構成が良いでしょう。

ここで行われる通信は、UDP プロトコル上で動作する ICP (Internet Cache Protocol) で行われます。キャッシュ間のデータ転送は、TCP をベースにした HTTP (HyperText Transmission Protocol) で行われます。

特定のオブジェクトを要求するにあたって、最も適切なサーバを選択する目的で、キャッシュは全ての兄弟関係のプロキシに対して、ICP リクエストを送信します。ICP リクエストを受け取ったプロキシは、ICP レスポンスとしてそれらの要求に応答します。オブジェクトが見つかった場合は `HIT` を、見つからなかった場合は `MISS` を使用して応答します。

複数の `HIT` 応答が届いた場合、プロキシサーバは最も高速に応答したサーバや、最も近いサーバなどの判断基準で、特定のサーバを選択します。逆に、必要な応答が届かなかった場合、要求はそのまま親キャッシュに送信されます。

注記: squid でのオブジェクト重複の回避方法について

ネットワーク内の複数のキャッシュ間でオブジェクトの重複を防ぐため、CARP (Cache Array Routing Protocol) や HTCP (HyperText Cache Protocol) などの他の ICP プロトコルを使用します。ネットワーク内で保持されるオブジェクトが増えれば増えるほど、必要なオブジェクトを見つける機会が増えることになります。

26.1.3 インターネットオブジェクトのキャッシュ

動的に生成されるページや TLS/SSL で暗号化されたコンテンツなど、ネットワーク内にある多くのオブジェクトは静的なものではありません。これらのオブジェクトは、アクセスが行われるたびに変化するものであることから、キャッシュが行われません。

また、キャッシュ内にオブジェクトを保持する期間を判断する目的で、オブジェクトには状態が割り当てられます。Web やプロキシサーバでは、「最終更新日時」や「期限切れ日時」などの仕組みを用意することで、これらのオブジェクトに対して状態を付与できるようになっています。なお、オブジェクトに含まれるその他のヘッダ情報については、そのままキャッシュ内に保持されます。

キャッシュ内のオブジェクトは、ディスク領域の不足などを理由として置き換えられるのが一般的です。この処理を行うにあたっては、LRU (Least Recently Used) などのアルゴリズムを使用します。これにより、最も長い時間使われていないオブジェクトは、キャッシュから消滅することを意味します。

26.2 システム要件

システムに対する要件は、システムが耐えられるべき最大のネットワーク負荷に依存して決まります。ネットワークの負荷は、最大では日々の平均の 4 倍を超えるようなことも珍しくありません。また、心配な場合は、システムの要件よりも少し多めに見積もっておくとよいでしょう。これは、squid は性能の限界まで使用されてしまうと、サービス品質の観点で多大な損失が発生してしまうためです。下記には、システム要件の要素として重要な順序を示しています:

1. メモリサイズ
2. CPU の速度／物理 CPU コア数
3. ディスクキャッシュのサイズ
4. ハードディスクか SSD か、およびその構造

26.2.1 メモリ

squid で必要とされるメモリ量は、キャッシュ内のオブジェクトの数に依存して決まります。メモリはハードディスクや SSD より高速に動作するものであるため、squid プロセスに対しては十分なメモリの割り当てが重要となります。メモリが不足してスワップディスクを使用してしまうと、システムの性能が劇的に下がる結果になります。

squid では、データの取得を高速化する目的で、キャッシュオブジェクトの参照と頻繁に要求されるオブジェクトについては、メインメモリ内に保持するようになっています。これに加えて、処理した全ての IP アドレスの一覧やドメイン名の正確なキャッシュ、そして最もよく要求されるオブジェクトやアクセス制御リスト、バッファなどについても、squid はメモリ内に保持する必要があります。

26.2.2 CPU

squid はプロセッサのコア数の少ない (物理コア数で 4 ～ 8 コア程度) 環境でも十分に動作するようチューニングされています。また、ハイパースレッディングなどの仮想的なコアでは、性能が十分に引き出せない場合があります。

複数の CPU コアを効率的に使用するため、異なるキャッシュデバイスに書き込むワーカースレッド数を多くする必要があります。既定では、マルチコアサポートが無効化されています。

26.2.3 ディスクキャッシュのサイズ

小さなキャッシュの場合は、`HIT` (つまり、オブジェクトが既に取得済みのものである) の確率が小さくなります。なぜなら、小さなキャッシュではすぐにオブジェクトで埋め尽くされてしまうため、頻繁に使用されるものであっても容易に廃棄されてしまうからです。たとえばキャッシュに 1 GB を割り当てていて、ユーザが 1 日あたり 10 MB 程度のアクセスしかしない環境であれば、キャッシュを埋め尽くすのに 100 日程度かかることになります。

キャッシュとして必要なサイズを判断するのに最も簡単な方法は、インターネット接続の最大転送速度から見積もる方法です。たとえば 1 Mbit/s のインターネット接続の場合、最大転送速度は 128 KB/s になります。キャッシュ内を 1 時間で埋め尽くすと仮定すると、必要なサイズは 460 MB になります。また、業務時間を 8 時間と仮定すると、1 日あたり 3.6 GB になります。インターネット接続は一般に、最大まで使用することはありませんので、キャッシュとして使用するのはおおよそ 2 GB 程度で十分ということになります。そのため、この例では squid に対して 2 GB 程度のキャッシュを設定することで、1 日分に相当するデータをキャッシュできることになります。

26.2.4 ハードディスク/SSD の構成

キャッシュ処理においては、速度が重要となります。そのため、この要素に対しては特別に配慮しておくことをお勧めします。ハードディスクや SSD の場合、このパラメータは ランダムシークタイム や ランダムリードパフォーマンス などとして示され、多くはミリ秒で示されます。squid からハードディスクや SSD に書き込んだり、それらから読み込んだりするデータは小さいものであるため、データのスループットよりはシークタイムや読み込み性能のほうがより重要となります。

プロキシとして使用する場合、高速回転のハードディスクや SSD を使用しておくことが最適な選択となります。ハードディスクを使用する場合、複数の小さなハードディスクを使用して、それぞれにキャッシュディレクトリを設定することで、読み込み時間を削減することができます。

RAID システムを使用すると、速度を犠牲にして信頼性を増すことができますが、性能上の理由から、(ソフトウエア) RAID5 やそれに類似の設定は避けておくことをお勧めします。

ほとんどの場合において、ファイルシステムの選択はそれほど重要ではありません。ただし、マウントオプションに `noatime` を指定することで、性能を改善することができます。これは、squid では独自のタイムスタンプを保持しているため、ファイルシステム側でアクセス日時を管理する必要が無いためです。

26.3 squid の基本的な使い方

openSUSE® Leap では `squid` は既定でインストールされませんので、まずはお使いのシステムにインストールを行ってください。

squid は openSUSE Leap 側であらかじめ設定済みであるため、インストール直後からすぐに起動することができます。なお、起動時の問題を回避するため、あらかじめインターネットに接続しておき、少なくとも 1 つ以上のネームサーバを設定しておいてください。また、動的な DNS 設定でダイヤルアップ接続を行っている場合は、起動時に問題が発生する場合があります。これは、squid では、`/var/run/netconfig/resolv.conf` 内に DNS サーバの設定が存在しないと、開始することができないことによるもので、この場合は少なくとも 1 つ以上のネームサーバを固定で設定しておいてください。

26.3.1 squid の開始

squid を起動するには、下記のコマンドを実行します:

```
> sudo systemctl start squid
```

システムの起動時に squid を開始するように設定したい場合は、`systemctl enable squid` のように実行して、サービスを有効化します。

26.3.2 squid が動作しているかどうかの確認

squid が動作しているかどうかを確認するには、下記のコマンドを実行します:

- `systemctl` を使用する場合:

  ```
  > systemctl status squid
  ```

  上記のコマンドの出力に `loaded` と `active (running)` が含まれていれば、squid が動作していることになります。

- また、squid 自身でも確認することができます:

  ```
  > sudo squid -k check | echo $?
  ```

  このコマンドの出力は `0` であるべきですが、追加の警告やメッセージが含まれることもあります。

ローカルシステム内で squid の機能をテストするには、下記のいずれかの方法を使用します:

- テストを行うには、`squidclient` と呼ばれるコマンドラインツールをお使いください。これは Web リクエストを送信して出力を表示することのできるツールで、`wget` や `curl` に似た仕組みです。

なお、`wget` や `curl` 等のツールとは異なり、`squidclient` はローカルの squid (`localhost:3128`) に対して接続を行います。squid 側の設定でポートを変更している場合は、`squidclient` のコマンドラインオプションを指定して、プロキシを指定する必要があります。詳しくは `squidclient --help` をお読みください。

例 26.1: `squidclient` によるリクエスト送信

```
> squidclient http://www.example.org
HTTP/1.1 200 OK
Cache-Control: max-age=604800
Content-Type: text/html
Date: Fri, 22 Jun 2016 12:00:00 GMT
Expires: Fri, 29 Jun 2016 12:00:00 GMT
Last-Modified: Fri, 09 Aug 2013 23:54:35 GMT
Server: ECS (iad/182A)
Vary: Accept-Encoding
X-Cache: HIT
x-ec-custom-error: 1
Content-Length: 1270
X-Cache: MISS from moon❶
X-Cache-Lookup: MISS from moon:3128
Via: 1.1 moon (squid/3.5.16)❷
Connection: close

<!doctype html>
<html>
<head>
    <title>Example domain</title>
[...]
</body>
</html>
```

例26.1「`squidclient` によるリクエスト送信」に示されている出力は、2 つのパートから構成されています:

1. 応答のプロトコルヘッダ: 1 行分の空き (空行) より前の部分がそれにあたります。
2. 応答の本体: 空行より後ろの部分がそれにあたります。

squid を使用していることを確認するため、下記のヘッダ行をご確認ください:

❶ `X-Cache` ヘッダの値には、要求した文書がコンピュータ `moon` 内の squid キャッシュ内に存在していなかった (`MISS`) ことが示されています。
上記の例には、2 つの `X-Cache` ヘッダが存在しています。ですが、最初の `X-Cache` ヘッダについては、無視してかまいません。これは、相手側の Web サーバのキャッシュソフトウエアで生成されたヘッダであるためです。

❷ `Via` の値には、HTTP のバージョンとコンピュータの名前、そして使用している squid のバージョンが示されています。

- 次にブラウザを利用して確認します: プロキシサーバに `localhost` を、プロキシサーバのポートに `3128` を指定してブラウザを設定します。この状態から何らかの Web ページを開いて、ブラウザの インスペクタ や 開発者ツール 内にある［ネットワーク］パネルから、応答ヘッダを確認してください。例26.1「`squidclient` によるリクエスト送信」と同様のヘッダが存在しているはずです。

ローカルシステムや他のシステムから squid 経由でインターネットにアクセスできるようにするには、設定ファイル `/etc/squid/squid.conf` 内にある `http_access deny all` を `http_access allow all` に変更してください。ただし、この設定を実施すると、squid は誰からもアクセスできるようになってしまうことに注意してください。アクセスできるユーザを制限したい場合は、ACL (Access Control List; アクセス制御リスト) を設定して行います。また、設定ファイルを変更した場合は、squid を再読み込みさせるか、再起動しなければなりません。ACL について、詳しくは 26.5.2項「アクセス制御用オプション」をお読みください。

squid が正しく開始できているにもかかわらず、しばらくして squid が停止してしまうような場合は、ネームサーバの設定が正しく行われていないか、`/var/run/netconfig/resolv.conf` ファイルがそもそも存在していないことが考えられます。squid では、開始時の問題を `/var/log/squid/cache.log` ファイル内に記録します。

26.3.3 squid の停止／再読み込み／再起動

squid を再読み込みさせるには、下記のいずれかを実施します:

- `systemctl` を使用する場合:

```
> sudo systemctl reload squid
```

もしくは、下記のように実行してもかまいません:

```
> sudo systemctl restart squid
```

- YaST を使用する場合は、下記のようにします:
  squid モジュールを起動して［設定を保存して squid を再起動する］ボタンを押します。

squid を停止するには、下記のいずれかを実施します:

- `systemctl` を使用する場合:

```
> sudo systemctl stop squid
```

- YaST を使用する場合:
  squid モジュールを起動して［squid を停止する］ボタンを押します。

squid は、その停止時にクライアントとの接続を切断し、データをディスクに書き込む処理を行うため、最大で 30 秒程度の時間がかかります (`/etc/squid/squid.conf` 内の `shutdown_lifetime` オプションで設定することができます) 。

警告: squid の停止について

squid を停止させる際は、`kill` や `killall` コマンドで停止させてはなりません。これを行ってしまうと、キャッシュデータを壊してしまう可能性があります。この場合、squid を再度開始できるようにするには、キャッシュデータを削除しなければならなくなります。

26.3.4 squid の削除

システムから squid を削除しても、キャッシュ構造やログファイルについては削除が行われません。これらを削除するには、`/var/cache/squid` ディレクトリを手作業で削除してください。

26.3.5 ローカル DNS サーバ

たとえ自分自身でドメインを所有していなくても、ローカルで DNS サーバを構築しておくことには意味があります。ローカルの DNS サーバは、キャッシュ専用のネームサーバとして動作させることで、特別な設定を行わなくてもルートネームサーバへの DNS 要求ができるようになります (詳しくは 19.4項「BIND ネームサーバの起動」をお読みください) 。このようなローカルの DNS サーバは、インターネットの接続にあたって動的に DNS サーバの設定を取得しているのか、静的に取得しているのかによって、やるべき作業が異なる点にも注意してください。

動的に DNS 設定を取得している場合

動的に DNS 設定を取得している場合、DNS サーバの情報はインターネットの接続時にプロバイダ側から提供され、それにあわせてローカルの `/var/run/netconfig/resolv.conf` が自動的に調整されるようになっています。この動作は、`/etc/sysconfig/network/config` 内の `NETCONFIG_DNS_POLICY` という sysconfig 変数で変更することができます。YaST sysconfig エディタなどを利用して、`NETCONFIG_DNS_POLICY` を `""` に設定してください。

その後、`/var/run/netconfig/resolv.conf` 内にローカルの DNS サーバを設定します。具体的には、`localhost` を意味する `127.0.0.1` の IP アドレスを設定します。これにより、squid は開始時にローカルのネームサーバを常に見つけられるようになります。
また、プロバイダのネームサーバにアクセスできるようにするため、`/etc/named.conf` 内の `forwarders` 以下に、プロバイダのネームサーバを IP アドレスで指定します。動的に取得する環境の場合は、sysconfig 変数の `NETCONFIG_DNS_POLICY` を `auto` にすることで、自動的に調整させることができるようになります。

静的に DNS 設定を行っている場合

静的な DNS 設定の場合、接続を行っても自動的に DNS の設定が更新されることはありません。そのため、sysconfig 変数についても、特に変更を行う必要はありません。しかしながら、動的に DNS 設定を取得している場合 で説明しているとおり、`/var/run/netconfig/resolv.conf` ファイルでローカルの DNS サーバを指定しておかなければなりません。これに加えて、`/etc/named.conf` ファイル内の `forwarders` 以下に、IP アドレスでプロバイダのネームサーバを指定してください。

ヒント: DNS とファイアウオールについて

ファイアウオールを動作させている場合は、DNS の要求を通すことができるように設定してください。

26.4 YaST squid モジュール

YaST の squid モジュールには、それぞれ下記のタブが用意されています:

[起動]

squid の起動方法や、どのインターフェイスに対してファイアウオールのポートを開くかなどの設定を行います。

[HTTP ポート]

squid がクライアントからの HTTP リクエストを待ち受けるポートを指定します。

[更新パターン]

squid でのキャッシュ内オブジェクトの取り扱い方法を設定します。

[キャッシュ設定]

最大オブジェクトサイズや最小オブジェクトサイズなど、キャッシュメモリに関する設定を行います。

[キャッシュディレクトリ]
: squid が全てのキャッシュスワップファイルを保存するディレクトリを設定します。

[アクセス制御]
: ACL グループを介した squid サーバへのアクセス制御を行います。

[ログとタイムアウト]
: 接続タイムアウトやクライアントの生存時間に加えて、アクセスログ、キャッシュログ、キャッシュ保存ログの保存先を設定します。

[その他]
: 言語や管理者のメールアドレスの設定を行います。

26.5 squid の設定ファイル

squid プロキシサーバの設定は、`/etc/squid/squid.conf` ファイルで行います。squid を初めて起動した場合は、このファイルを修正する必要はありませんが、初期状態では外部からのアクセスが全て拒否されるようになっています。そのため、プロキシは `localhost` 専用のものとなります。また、既定のポートは `3128` です。あらかじめ用意されている `/etc/squid/squid.conf` には、オプションに関する様々な説明や設定例が示されています。

多くの項目は `#` というコメント文字でコメントアウトされていて、その後ろに様々な設定が書かれています。記述されている値は、通常は既定値が書かれているものであるため、コメントアウトを外しても squid の動作は変わりません。また、可能であればコメントアウトされた行は変更せず、その下に行を挿入して必要な設定と値を記述してください。これにより、何らかの設定を間違ってしまっていても、すぐに元に戻すことができるためです。

ヒント: 更新後の設定ファイルの調整について

古いバージョンの squid から更新する場合は、設定ファイルをそのままコピーせず、新しい `/etc/squid/squid.conf` から必要に応じて変更していく方法で実施することをお勧めします。

squid では、バージョンが変わるごとにオプションが追加／削除／変更されます。そのため、古いバージョンに対応した `squid.conf` を新しいバージョンで使用すると、正しく動作しないことがあります。

26.5.1 一般的な設定オプション

下記には、squid の設定オプションのうち主なものを一覧で示しています。下記は全ての設定を網羅しているわけではありません。squid パッケージ内には、全てのオプションを簡易に説明した文書が用意されています。詳しくは `/etc/squid/squid.conf.documented` ファイルをお読みください。

`http_port ポート`

squid がクライアントからの要求を待ち受けるポートを指定します。既定値は `3128` ですが、一般的には `8080` を使用することもあります。

`cache_peer ホスト名 種類 プロキシポート ICP_ポート`

このオプションは、相互に通信を行うキャッシュネットワークを作成するためのオプションです。`cache_peer` で指定する相手は、この squid と同じくネットワークキャッシュを提供するコンピュータで、相互に関係性を持たせるためのものです。関係性の種類は `種類` で指定します。ここには `parent` (親) もしくは `sibling` (兄弟) のいずれかを指定します。

`ホスト名` には対向のプロキシをホスト名または IP アドレスで指定します。`プロキシポート` には一般的なブラウザからアクセスする際のポート (通常は `8080`) を設定します。また、`ICP_ポート` には `7` を指定するか、もしくは ICP ポートが分からない場合や、この相手に対しては使用しない場合、`0` を指定します。

また、squid をプロキシではなく Web ブラウザのように動作させたい場合は、`default` および `no-query` のオプションを追加して、ICP プロトコルの使用を禁止してください。

`cache_mem サイズ`

このオプションには、squid が最もよく使用するオブジェクトに対して使用するメモリ量を設定します。既定値は `8 MB` です。ただし、ここで指定した値は squid 全体のメモリ使用を制限するものではありません。そのため、ここで指定した値を超えてメモリを使用することがあります。

`cache_dir ストレージの種類 キャッシュディレクトリ キャッシュサイズ レベル_1_ディレクトリ レベル_2_ディレクトリ`

`cache_dir` オプションは、ディスクキャッシュのディレクトリを指定するためのオプションです。openSUSE Leap の既定の設定では、squid はディスクキャッシュを作成しません。

ストレージの種類 では、下記のいずれかを指定することができます:

- ディレクトリのストレージ: `ufs` , `aufs` (既定値), `diskd` のいずれかを指定します。3 種類はいずれも `ufs` という形式のストレージを使用しますが、`ufs` を指定すると squid の中枢スレッド内で、`aufs` を指定すると別途のスレッドとして、`diskd` を指定すると別途のプロセスとしてそれぞれストレージが動作するようになります。後者 2 種類については、ディスクの I/O による squid の一時的な処理停止を防ぐことができます。

- データベースのストレージ: `rock` を指定します。このストレージ形式は単一のデータベースファイルから構成される仕組みで、それぞれのオブジェクトが固定長のメモリユニットを 1 つもしくは複数確保することで動作します。

なお、下記はストレージの種類に `ufs` を選択した場合の説明になります。`rock` を選択した場合は異なる設定になります。
キャッシュディレクトリ には、ディスクキャッシュを配置するディレクトリを指定します。既定では `/var/cache/squid` が指定されています。キャッシュサイズ はディレクトリに対する最大のサイズを指定します。既定では 100 MB に設定されています。利用可能なディスク領域のうち、50% から 80% 程度までの範囲で指定してください。
レベル_1_ディレクトリ と レベル_2_ディレクトリ には、キャッシュディレクトリ 内に作成するサブディレクトリの数を設定します。既定では レベル_1_ディレクトリ が 16 に、レベル_2_ディレクトリ が 256 に設定されています。レベル_2_ディレクトリ は、レベル_1_ディレクトリ 内のサブディレクトリとなります。なお、これらの値を増やす場合は注意して設定してください。ディレクトリ数が多すぎると、性能面で問題を引き起こす場合があります。
キャッシュを複数のディスクにまたがって設定したい場合は、`cache_dir` を複数個指定してください。

`cache_access_log` ログファイル,

`cache_log` ログファイル,

`cache_store_log` ログファイル

これら 3 種類のオプションは、squid が様々な動作に対してログ記録を残すためのファイルを指定するためのものです。通常は特に変更する必要はありません。ただし、squid 側の負荷が高い場合は、キャッシュとログファイルを複数のディスクに分割して設定するとよいでしょう。

`client_netmask` ネットマスク

このオプションは、ログファイル内でサブネットマスクを適用することによって、クライアントの IP アドレスをマスクする処理を行います。たとえば IP アドレスの末尾を `0` に設定したい場合は、`255.255.255.0` を指定します。

`ftp_user` 電子メールアドレス

このオプションは、匿名 FTP ログイン時に入力するパスワードを指定します。FTP サーバによっては、電子メールアドレスが正しいかどうかを検証することがありますので、通常は正しい電子メールアドレスを指定します。

`cache_mgr` 電子メールアドレス

何らかの問題で squid がクラッシュしてしまった場合、squid はここで指定した電子メールアドレスに対して、メッセージを送信します。既定値は webmaster です。

`logfile_rotate` 値

`squid -k rotate` を実行すると、`squid` はログファイルをローテーションします (切り替えます) 。ログファイルの末尾には番号が付けられて保存されるほか、古いものから順に上書きされるようになります。既定値は `10` で、この場合は末尾に 0 から 9 までの番号が付くことになります。なお、openSUSE Leap ではログファイルの切り替えは `logrotate` と `/etc/logrotate.d/squid` を使用して自動的に実施しています。

`append_domain` ドメイン名

append_domain は、何もドメインを指定しない場合に自動的に追加するドメイン名を指定します。通常はご利用のドメインを指定しますが、このように設定すると、ブラウザで www とだけ入力すると、ご利用のドメインの Web サーバを表示させることができます。

`forwarded_for` 状態

このオプションを `on` に設定した場合、下記のようなヘッダが追加されます:

```
X-Forwarded-For: 192.168.0.1
```

このオプションを `off` に設定した場合、squid は HTTP リクエスト内にある IP アドレスとシステム名の情報を削除します。

`negative_ttl` 時間,
`negative_dns_ttl` 時間

これらのオプションを設定すると、squid は `404` 応答などを含むいくつかの種類の失敗をキャッシュするようになります。失敗をキャッシュした場合は、元のリソースが利用可能な状態になっても、新しい要求を拒否するようになります。
既定では `negative_ttl` は `0` に、`negative_dns_ttl 1 minutes` (1 分) に設定されています。これらの設定値では、Web リクエストに対する否定応答はキャッシュされないものの、DNS リクエストに対する否定応答は 1 分間だけキャッシュされるようになります。

`never_direct allow ACL_名`

squid がインターネットに対して直接接続することを防ぐには、`never_direct` を指定して他のプロキシへの接続を強制することをお勧めします。この場合、接続先のプロキシは `cache_peer` で指定したものを使用します。`ACL_名` に `all` を指定すると、全ての要求を `親` 宛に転送するようになります。これはたとえば、プロバイダ側でプロキシの使用を求めていて、インターネットへの直接アクセスを拒否しているような場合に必要な設定となります。

26.5.2 アクセス制御用オプション

squid では アクセス制御リスト (ACL; Access Control List) を利用してプロキシサーバへのアクセスを制御します。ACL は上から順に処理されるルールを一覧で指定する形を取るほか、指定する前に定義を行う必要もあります。また、既定の ACL である `all` と `localhost` については、定義せずに利用することができます。しかしながら、ACL の設定だけでは何も効果がありません。対応する `http_access` ルールなどを設定することではじめて効果があります。

`acl` オプションの書式は下記のとおりです:

```
acl ACL_名 種類 データ
```

それぞれの項目の意味は下記のとおりです:

- `ACL_名` には任意の名前を指定します。
- `種類` には様々な値を指定することができます。詳しくは `/etc/squid/squid.conf` ファイル内の `ACCESS CONTROLS` セクションをお読みください。
- `データ` は ACL の種類ごとに異なるデータを指定する箇所で、たとえばホスト名や IP アドレス、URL などを指定します。

YaST の squid モジュールでルールを追加する場合は、モジュールを開いて［アクセス制御］タブを選択します。あとは ACL グループの一覧から［追加］ボタンを押して、ルールの名前と種類、パラメータを指定します。

ACL ルールに関する詳しい説明については、https://www.squid-cache.org にある squid のドキュメンテーションをお読みください。

例 26.2: ACL ルールの設定

```
acl mysurfers srcdomain .example.com ❶
acl teachers src 192.168.1.0/255.255.255.0 ❷
acl students src 192.168.7.0-192.168.9.0/255.255.255.0 ❸
acl lunch time MTWHF 12:00-15:00 ❹
```

❶ この ACL は `mysurfers` という名前で、`.example.com` というドメインからの全ての利用者を意味しています (ドメインの確認は、IP アドレスから逆引きして行います) 。

❷ この ACL は `teachers` という名前で、`192.168.1.` で始まる IP アドレスを持つ全てのコンピュータを意味しています。

❸ この ACL は `students` という名前で、`192.168.7.` , `192.168.8.` , `192.168.9.` のいずれかで始まる IP アドレスを持つ全てのコンピュータを意味しています。

❹ この ACL は `lunch` という名前で、月曜日から金曜日までの正午から午後 3 時までの時間帯を意味しています。

http_access allow `ACL_名`

`http_access` では、プロキシを使用できるユーザと、そのユーザがどのサイトにアクセスできるのかをそれぞれ設定します。ここでは `ACL_名` を指定しますので、対応する ACL をあらかじめ定義しておく必要があります。なお、上述のとおり `localhost` と `all` は最初から定義されているため、`deny` や `allow` で禁止や許可を設定することができます。また `http_access` は必要な数だけ記述することができ、それぞれ上から順に解釈されます。そのため、どちらが最初に現れるのかによって、許可と拒否が入れ替わることもあります。また、末尾には `http_access deny all` を指定しておくと良いでしょう。下記の例は、`localhost` からのアクセスであれば、どのサイトに対しても自由にアクセスすることができますが、その他のホストからのアクセスは拒否する設定です:

```
http_access allow localhost
http_access deny all
```

もう 1 つの例では、下記のようなルールになっています。グループ `teachers` は常にインターネットへのアクセスが許可されます。グループ `students` については、月曜日から金曜日までのランチタイムにのみアクセスが許可されます。

```
http_access deny localhost
http_access allow teachers
http_access allow students lunch time
http_access deny all
```

可読性を高めるため、`http_access` は `/etc/squid/squid.conf` ファイル内の一カ所に集めておくことをお勧めします。

`url_rewrite_program パス`

このオプションを使用すると、URL の書き換えプログラムを指定することができます。

`auth_param basic program` パス

利用者に対してプロキシへのアクセスに際して認証を求める場合、ここでプログラムを指定します。たとえば `/usr/sbin/pam_auth` のように指定します。`pam_auth` では、はじめてアクセスすると、ユーザに対してユーザ名とパスワードの入力を求めます。この設定に加えて ACL を定義することで、インターネットへのアクセス時に正しい認証情報を求めるようにすることができます:

```
acl password proxy_auth REQUIRED

http_access allow password
http_access deny all
```

`acl proxy_auth` では `REQUIRED` と指定していますが、これはユーザ名とパスワードが適合していれば、どのユーザに対してもアクセスを許可する意味になります。`REQUIRED` の代わりに、許可するユーザの一覧を記述してもかまいません。

`ident_lookup_access allow` ACL_名

このオプションを指定すると、`src` の種類の ACL で定義される全てのクライアントに対して、ユーザの識別情報を取得するために ident 要求を実行します。全てのクライアントに対して処理を行いたい場合は、ACL_名 にあらかじめ設定されている ACL 名である `all` を指定してください。

なお、`ident_lookup_access` を指定した全てのクライアントでは、ident デーモンを動作させなければならなくなります。Linux の場合、ident デーモンとして `pidentd` (`pidentd` パッケージ) を使用することができます。ident 要求に対して正常に応答したクライアントだけにアクセスを許可したい場合は、下記のような ACL を設定します:

```
acl identhosts ident REQUIRED

http_access allow identhosts
http_access deny all
```

`acl identhosts ident` オプションに書かれている `REQUIRED` は、正しくユーザ名が設定されていれば全てのユーザを受け入れる意味になります。`REQUIRED` の箇所を許可するユーザの一覧に変更すれば、そのユーザに対してのみ許可することができるようになります。

なお、`ident` オプションを指定すると、squid への要求が発生するたびに ident の参照が発生するため、アクセスに時間がかかるようになりますので、あらかじめご注意ください。

26.6 透過型プロキシの設定

透過型プロキシの場合は、プロキシサーバが Web ブラウザの要求を仲介して処理を行います。そのため、Web ブラウザはプロキシの存在に気付くことがありませんし、名前のとおりユーザに対しても透明な存在になります。

プロキシサーバは一般に、プロキシ内のキャッシュに存在するかどうかに関わらず、Web ブラウザからの要求を受け取ってインターネットへのアクセスを代行する処理を行いますが、それ以外にも透過型プロキシと呼ばれる形態で動作させることもできます。たとえば下記のような用途が考えられます:

- セキュリティ上の理由から、インターネットにアクセスするにあたって全てのクライアントがプロキシを使用することが推奨される場合。
- 全てのクライアントに対して、プロキシの存在に気付くことなく使用させたい場合。
- ネットワーク内のプロキシを移設したいが、既存のクライアント側の設定をいちいち変更させたくない場合。

手順 26.1: SQUID による透過型プロキシの設定 (コマンドライン)

1. まずは `/etc/squid/squid.conf` 内で、`http_port` に `transparent` というパラメータを指定したものを追加します。たとえば下記のようになります:

   ```
   http_port 3128#
   http_port 3128 transparent
   ```

2. squid を再起動します:

   ```
   > sudo systemctl restart squid
   ```

3. あとはファイアウオールを設定して、HTTP のトラフィックを `http_proxy` で指定したポート (上記の例では 3128) に転送するようにします。設定後はファイアウオールの設定を再読み込みします。なお、お使いの LAN インターフェイスが `internal` ゾーン内に存在している必要があります。

   ```
   > sudo firewall-cmd --permanent --zone=internal \
       --add-forward-port=port=80:proto=tcp:toport=3128:toaddr=LAN_IP
   > sudo firewall-cmd --permanent --zone=internal --add-port=3128/tcp
   > sudo firewall-cmd --reload
   ```

   なお、`LAN_IP` の箇所には、お使いの LAN インターフェイスか squid が待ち受けているインターフェイスの IP アドレスを入力してください。

4. 全てが正しく動作していることを確認したい場合は、`/var/log/squid/access.log` にある squid のログファイルを確認してください。

26.7 squid キャッシュマネージャ CGI インターフェイスの使用 (cachemgr.cgi)

squid キャッシュマネージャ CGI インターフェイス (`cachemgr.cgi`) は、動作中の squid プロセスに対して、メモリの使用状況の統計を表示するための CGI ユーティリティです。サーバにログインすることなく、キャッシュを管理したり統計情報を表示したりすることができます。

手順 26.2: `cachemgr.cgi` の設定

1. まずはお使いのシステムで Apache Web サーバを動作させます。Apache の設定方法について、詳しくは 第24章「Apache HTTP サーバ」をお読みください。特に 24.5項「CGI スクリプトの有効化」をよくお読みください。Apache が動作していることを確認するには、下記のコマンドを実行します:

   ```
   > sudo systemctl status apache2
   ```

   `inactive` (動作していない) と表示されてしまった場合は、openSUSE Leap の既定の設定では下記のようにして Apache を開始します:

   ```
   > sudo systemctl start apache2
   ```

2. これで `cachemgr.cgi` を使用する準備が整いました。あとは `ScriptAlias` に対する設定ファイルを作成します。
   `/etc/apache2/conf.d` ディレクトリ内に `cachemgr.conf` というファイルを作成し、下記の内容を記述します:

   ```
   ScriptAlias /squid/cgi-bin/ /usr/lib64/squid/

   <Directory "/usr/lib64/squid/">
   Options +ExecCGI
   AddHandler cgi-script .cgi
   Require host ホスト名
   </Directory>
   ```

   なお `ホスト名` の箇所は、`cachemgr.cgi` へのアクセスを許可するコンピュータのホスト名を指定します。これにより、`cachemgr.cgi` へのアクセスを制限することができます。どのコンピュータからもアクセスできるように設定したい場合は、代わりに `Require all granted` を指定してください。

3.

- なお、squid と Apache Web サーバが同じコンピュータで動作している場合は、`/etc/squid/squid.conf` を変更する必要はありません。ただし、`/etc/squid/squid.conf` 内に下記の内容が含まれていることをご確認ください:

```
http_access allow manager localhost
http_access deny manager
```

これらの行は、マネージャインターフェイスに対して、そのコンピュータ自身 (`localhost`) からのみアクセスを許可する設定になります。

- squid と Apache Web サーバが別々のコンピュータで動作している場合は、squid に対して特定のコンピュータからキャッシュマネージャにアクセスできるように設定する必要があります。この場合は、下記のような ACL になります (`Web_サーバの_IP` には、Web サーバの IP アドレスを指定します):

```
acl webserver src Web_サーバの_IP/255.255.255.255
```

あとは設定ファイル内に下記のルールを記述します。順序を間違えないように注意してください。

```
http_access allow manager localhost
http_access allow manager webserver
http_access deny manager
```

4. また、必要であれば `cachemgr.cgi` に対して 1 つもしくは複数のパスワードを設定することもできます。この設定では、キャッシュをリモートから閉じたり、キャッシュに対する詳細情報を取得したりする際にパスワードを求めたりすることもできます。この場合は、`cache_mgr` や `cachemgr_passwd` のオプションに対して複数のパスワードを設定して、そのパスワードを入力した際に許可するアクションを指定します。
たとえばトップページ (メニュー) と 60 分間の平均値の表示についてはパスワード入力を求めず、オフラインモードへの移行にはパスワード入力 `secretpassword` を求め、それ以外の全ては無効化したい場合、下記のような設定になります:

```
cache_mgr user
cachemgr_passwd none index menu 60min
cachemgr_passwd secretpassword offline_toggle
cachemgr_passwd disable all
```

`cache_mgr` ではユーザ名を指定します。`cachemgr_passwd` では、パスワードと許可するアクションをそれぞれ指定します。
なお、`none` と `disable` は特別な意味を持つキーワードです。`none` はパスワードの入力を求めず、`disable` はその機能を無効化する意味になります。

アクションの一覧については、`cachemgr.cgi` にログインしてご確認ください。また、設定ファイル内でのキーワードについては、それぞれのアクションページ内の `&operation=` 内に書かれています。なお、アクションで `all` を指定すると、全てのアクションを意味することになります。

5. squid と apache の設定を変更した場合は、それぞれ再読み込みを行います:

```
> sudo systemctl reload squid
```

6. 統計情報を表示するには、ここまでで設定した `cachemgr.cgi` のページを表示させてください。たとえば `http://webserver.example.org/squid/cgi-bin/cachemgr.cgi` のような URL になるはずです。
サーバ名の指定のほか、ユーザ名とパスワードを設定していれば、それらも正しく入力してください。あとは［Continue］を押すと、異なる統計情報を表示することができます。

26.8 Calamaris によるキャッシュレポートの生成

Calamaris は ASCII や HTML の出力形式に対応した Perl スクリプトで、squid のアクセスログファイルを読み込んでキャッシュ動作のレポート生成を行います。このツールは openSUSE Leap の既定ではインストールが行われません。必要な場合は `calamaris` パッケージをインストールしてお使いください。なお、Calamaris の Web ページは https://cord.de/calamaris-english (英語) にあります。

まずは `root` でログインして、下記のように入力します:

```
# cat access1.log [access2.log access3.log] | calamaris オプション > レポートファイル
```

複数のログファイルを指定する場合は、それらを時系列で古いものから順に並べて指定してください。上記のように複数のファイルを順に指定してもかまいませんし、`access{1..3}.log` のように指定してもかまいません。

`calamaris` では下記のようなオプションを指定することができます:

`-a`
: 利用可能な全てのレポートを出力します。

`-w`
: HTML 形式で出力します。

`-l`
: レポートヘッダ内にメッセージやロゴを含めます。

上記以外のオプションについては、マニュアルページ内に説明があります。詳しくは `man calamaris` を実行してください。

一般的な使用方法は下記のとおりです:

```
# cat access.log.{10..1} access.log | calamaris -a -w \
> /usr/local/httpd/htdocs/Squid/squidreport.html
```

上記のように実行すると、Web サーバのディレクトリ内にレポートを出力します。レポートを表示するには、Apache を動作させる必要があります。

26.9 さらなる情報

squid の Web ページは https://www.squid-cache.org/ にあります。ここには「Squid User Guide」(squid ユーザガイド) としてマニュアルが用意されているほか、squid の FAQ なども用意されています。

squid のメーリングリストもあります。詳しくは https://www.squid-cache.org/Support/mailing-lists.html をお読みください。

IV モバイルコンピュータ

27 Linux でのモバイルコンピューティング 535

28 NetworkManager の使用 546

29 電源管理 556

27 Linux でのモバイルコンピューティング

改訂履歴

2023-12-22

モバイルコンピューティングとは、主にラップトップや PDA 、携帯電話を利用するコンピュータ環境を意味するほか、それらとの間でのデータ交換をも意味します。また、外付けハードディスクやフラッシュメモリ、デジタルカメラなどのモバイルハードウエアコンポーネントも、ラップトップやデスクトップシステムに接続することがあります。また、モバイルコンピューティングでは、様々なソフトウエアコンポーネントが使用されるほか、アプリケーションによってはモバイル用途に適したものが用意されていることがあります。

27.1 ラップトップ

ラップトップのハードウエアは、通常のデスクトップシステムとは異なる構成になっています。これは様々な環境への接続や容積の制限、そして電源の消費量などの要件が存在するためです。モバイルハードウエアの製造元では、Mini PCI, Mini PCIe などの標準インターフェイスを開発して、ラップトップハードウエアの拡張性を高めることも行っています。このような標準では主に、メモリカードやネットワークインターフェイスカード、そして外付けハードディスクなどのハードウエアを接続するための仕組みです。

27.1.1 電源管理

ラップトップ機の製造にあたって消費電力の低減に努めるシステムコンポーネントを採用することで、電力網への接続を行うことなく利用できる環境を作り上げています。このようなシステム構成は、オペレーティングシステム側での省電力化と同程度に重要なものであり、openSUSE® Leap でも、そのようなラップトップに対して、バッテリー動作時の電源消費を制御する様々な方法を提供しています。下記の一覧は、電源消費を抑える際の貢献度を降順に示しています:

- CPU 速度の制限。
- スリープ中のディスプレイ表示の停止。
- ディスプレイの表示輝度の手動調整。
- ホットプラグに対応していて、使用していないアクセサリ類の取り外し (USB CD-ROM や外付けマウス、Wi-Fi など) 。
- 未使用時のハードディスクの回転停止。

openSUSE Leap での電源管理について、より詳しく知りたい場合は 第29章「電源管理」をお読みください。

27.1.2 利用環境の変更作業

モバイルコンピューティングで使用するにあたっては、まずお使いのシステムを操作環境にあわせて調整する必要があります。環境に依存した様々なサービスや、そのサービスに関わる様々なクライアントを設定しなければなりません。openSUSE Leap では、そのような処理を支援する仕組みを備えています。

図 27.1: 既存の環境に対するモバイル環境の設定

小規模な家庭内ネットワークと職場のネットワークを相互に切り替えて使用するようなラップトップ用途では、下記のようなサービスを設定する必要があります:

ネットワーク
: IP アドレスの管理や名前解決、インターネットへの接続方法や他のネットワークへの接続方式などが含まれます。

印刷
: 接続するネットワークに応じて、利用可能なプリンタの一覧と、印刷サーバを設定しなければなりません。

電子メールとプロキシ
: 印刷と同様に、対応するサーバをそれぞれ設定しなければなりません。

X (グラフィカル環境)
: お使いのラップトップを、一時的にプロジェクタや外付けのモニタに接続するような場合は、異なるディスプレイ設定を利用しなければなりません。

openSUSE Leap では、既存の環境にラップトップを接続するにあたって、いくつかの方法を提供しています:

NetworkManager
: NetworkManager はラップトップ環境のようなモバイル用途で便利な仕組みです。簡単かつ自動的にネットワーク環境を切り替えることができるほか、モバイルブロードバンド (GPRS, EDGE, 3G など) や無線 LAN 、イーサネットなどの様々な種類のネットワークに対応しています。NetworkManager では、無線 LAN の暗号化に対して WEP や WPA-PSK に対応しています。このほか、ダイヤルアップ接続も設定することができます。また、GNOME デスクトップには、NetworkManager に対するフロントエンドも用意されています。詳しくは 28.3項「ネットワーク接続の設定」をお読みください。

表 27.1: NETWORKMANAGER の利用判断

お使いのコンピュータ	NetworkManager の使用
ラップトップである場合	お勧めします
時々異なるネットワークに接続するような場合	お勧めします
ネットワークサービス (DNS, DHCP など) を提供している場合	お勧めできません
固定の IP アドレスのみを使用する場合	お勧めできません

NetworkManager ではネットワークの接続を処理できないような場合は、YaST のツールを利用して設定してください。

ヒント: DNS 設定と様々なネットワーク接続の種類について

ラップトップを携行して様々なネットワーク接続を行き来するような場合、DHCP で全ての DNS アドレスを割り当てる環境であれば、NetworkManager で問題なく動作します。ただし、接続によって固定の DNS アドレスを設定しなければならないものがある場合は、`/etc/sysconfig/network/config` 内に `NETCONFIG_DNS_STATIC_SERVERS` を追加してください。

SLP

Service Location Protocol (SLP) は、ラップトップに対してネットワークへの接続を簡略化するための仕組みです。SLP が無い環境では、ラップトップの管理者がネットワークに対する詳細な情報を取得する必要があります。SLP は、提供されているサービスを全てのクライアントに対して全体周知する仕組みであることから、SLP に対応するアプリケーションであれば、SLP からの情報を受信して自動設定することができるようになります。また、SLP はシステムのインストールに使用することもできます。これにより、必要なインストール元を探す手間を省くことができます。SLP に関する詳細は、第17章「SLP」をお読みください。

27.1.3 ソフトウエアオプション

モバイル用途では様々な作業に対応する様々なソフトウエアが用意されています。たとえばシステムの監視 (主にバッテリーの充電に関する情報の取得) やデータの同期、周辺機器との無線接続やインターネット接続などがあります。下記の章では、openSUSE Leap でそれぞれの作業に対応する主なアプリケーションを紹介しています。

27.1.3.1 システム監視

openSUSE Leap で提供されるシステム監視ツールには、2 種類のものがあります:

電源管理

［電源管理］は GNOME デスクトップの動作のうち、省電力に関わる箇所を調整することのできるアプリケーションです。通常は［コンピュータ］›［コントロールセンター］›［システム］›［電源管理］からアクセスすることができます。

システムモニター

［システムモニター］は、様々なシステムパラメータを一カ所で収集できるアプリケーションです。既定では 3 種類のタブで出力情報を提示します。［プロセス］は現在動作中のプロセスに対して、CPU の負荷やメモリの使用率、プロセス ID や優先度などを表示することができます。収集されたデータの表示方法やフィルタリングはカスタマイズすることができます。プロセスの情報に対して、新しい情報を追加したい場合は、プロセス一覧のヘッダ部分でマウスの左ボタンを押し、どの項目を表示したいのかを選択してください。また、様々なデータページ内のシステムパラメータを監視したり、ネットワークを介して様々なマシンの情報を採取したりすることもできます。［リソース］タブでは、CPU やメモリ、ネットワークに対してグラフを表示することができるほか、［ファイルシステム］では全てのパーティションとその使用状況を表示することができます。

27.1.3.2 データ同期

ネットワークから切り離された環境で作業を行ったり、オフィス内のネットワークに接続して作業を行ったりするモバイルマシンでは、必要なデータを様々なマシンに複製しておきたいという用途があります。これにはたとえば、電子メールのフォルダやディレクトリ、作業中の文書ファイルなどが含まれます。このような用途に対しては、下記のような解決方法があります:

電子メールの同期

オフィスネットワーク内では、IMAP アカウントを利用して電子メールを保存するようにしてください。これにより、Mozilla Thunderbird や Evolution (詳しくは『GNOME ユーザガイド』をお読みください) など、IMAP に対応した電子メールクライアントを利用することで、ラップトップとワークステーションの両方で同じ電子メールを読むことができるようになります。ただし、電子メールクライアント側では、**送信済みメッセージ** にアクセスする際、常に同じフォルダにアクセスするように設定しなければなりません。これにより、全てのメッセージとその状態を同期できるようになります。また、メールの送信に際しては、ローカルの postfix や sendmail ではなく、SMTP サーバを使用するように設定してください。これにより、送信できなかったメールに対して、詳細な情報を取得できるようになります。

ファイルやディレクトリの同期

ラップトップとワークステーションの間でのデータ同期に対しては、いくつかのユーティリティが提供されています。一般的に良く用いられているソフトウエアとしては、`rsync` と呼ばれるものがあります。詳しくはマニュアルページ (`man 1 rsync`) をお読みください。

27.1.3.3 無線通信: Wi-Fi

無線技術の中で最もよく用いられている Wi-Fi は、様々な端末が混在する環境では唯一の無線通信手段であり、場合によっては仕切られた空間同士の通信手段としても使用する場合があります。コンピュータ同士が相互に接続して独立した無線ネットワークを構成する場合があるほか、インターネットへのアクセス手段としても使用することがあります。アクセスポイント と呼ばれるデバイスが存在する場合は、Wi-Fi の有効化されたデバイスに対するベースステーションとなって、インターネットへの接続を仲介することもあります。モバイルユーザは、その場所によって様々なアクセスポイントを切り替えながら使用し、最適な接続を確保するのが一般的です。携帯電話網と同様に、Wi-Fi でも巨大なネットワークを構成し、場所に縛られることなく作業を行うことができるようになっています。

Wi-Fi カードは IEEE が規定する 802.11 標準を利用して通信を行います。当初の規格では最大伝送レートは 2 Mbit/s 程度でしたが、様々な仕様が追加されることによって、伝送レートが大幅に改善しています。これらの追加仕様は変調方式や送信出力、そして伝送レートなどを定義するものです (詳しくは 表27.2「様々な Wi-Fi 標準の概要」 をお読みください) 。これら加えて、多くの製造元では独自の規格や草案レベルの仕様を実装していたりすることもあります。

表 27.2: 様々な WI-FI 標準の概要

名前 (802.11)	周波数 (GHz)	最大伝送レート (Mbit/s)	注記
a	5	54	干渉の起こりにくい規格です
b	2.4	11	現在は一般的には使用されていません
g	2.4	54	11b と後方互換性のある一般的な規格です
n	2.4 か 5, もしくはその両方	300	一般的に使われている規格です
ac	5	最大 865 まで	2015 年以降に一般化された規格です
ad	60	最大 7000 まで	2012 年に公開されましたが、現在はまだ一般的ではありません。また openSUSE Leap でも未対応です

802.11 の古いカードについては、openSUSE® Leap では対応していません。802.11 a/b/g/n のほとんどのカードであれば対応可能です。新しいカードであれば、802.11n 標準に対応していますが、802.11g のカードも対応可能です。

27.1.3.3.1 操作モード

無線ネットワーキングでは、様々な技術や設定によって、高速かつ高品質で、機密の守られるネットワーク環境を構成しています。通常はお使いの Wi-Fi カードを マネージドモード に設定して使用しますが、環境によっては異なる操作モードを使用する場合もあります。無線ネットワークにおいては、下記の 4 種類の操作モードがあります:

マネージドモード (インフラストラクチャモード), アクセスポイント経由 (既定のモード)

マネージドモードの場合は、管理側のデバイス (一般的にはアクセスポイントと呼ばれます) が存在しています。このモード (インフラストラクチャモードや既定のモードと呼ぶ場合もあります) では、ネットワーク内の全ての Wi-Fi 端末は、アクセスポイントを通して通信を行いますので、イーサネットへの接続口としても動作することになります。また、特定の端末のみが接続を許され、様々な認証機構 (WPA など) を使用して接続します。これは最も消費電力の少ないモードであることから、メインモードと呼ぶ場合もあります。

アドホックモード (一対一ネットワーク)

アドホックモードではアクセスポイントを使用しません。それぞれの端末はお互いに直接通信することになります。そのため、マネージドネットワークに比べると、ネットワークの速度は遅くなります。また、アドホックモードでは、伝送可能な範囲と参加できる端末数が大きく制限されます。さらに、WPA 認証にも対応していませんし、カードによってはアドホックモードへの対応が不十分なものもあります。

マスターモード

マスターモードでは、お使いの Wi-Fi カードをアクセスポイントとして動作させることになります。ただし、カード側の対応が必要です。Wi-Fi カードの詳細について、詳しくは https://linux-wless.passys.nl をお読みください。

メッシュモード

ワイヤレスメッシュネットワークは メッシュトポロジ を構成するための仕組みです。無線メッシュネットワークの接続は、全ての無線メッシュ ノード に広がるもので、このネットワークに所属するそれぞれのノードは、接続を共有する他のノードに接続して巨大なエリアを構成します。

27.1.3.3.2 認証

無線ネットワークは、有線のネットワークよりもずっと容易に盗聴や悪用ができてしまう仕組みであることから、認証や暗号化方式に関する様々な標準が存在しています。

古い Wi-Fi カードの場合、WEP (Wired Equivalent Privacy) にのみ対応しています。ただし、既に WEP の機密は破られてしまっているため、Wi-Fi では WPA と呼ばれる拡張を規定しています。これは WEP における脆弱性を排除したと考えられているものです。また、WPA は通常は WPA2 と呼ばれ、これが既定の認証方式として使用されています。

通常はユーザ側で認証方式を選択することはできません。たとえばマネージドモードでカードが動作している場合、認証方法はアクセスポイント側から指定され、NetworkManager では指定された認証方式を表示します。

27.1.3.3.3 暗号化

認可されていないユーザが、無線ネットワーク上を流れるデータパケットを読み取ったり、ネットワークへのアクセス権利を取得したりしてしまうことのないよう、様々な暗号化方式が用意されています:

WEP (IEEE 802.11 内で規定)
: この方式では RC4 と呼ばれる暗号化を使用し、元々は 40 ビット、後に 104 ビットの長さの鍵を使用して暗号化を行っていました。それぞれ初期化ベクトルと呼ばれる 24 ビットの長さが加えられて、64 ビットもしくは 128 ビットの鍵と呼ばれることもあります。しかしながら、現在はこの標準に欠陥が見つかってしまっていて、このシステムで生成された鍵への攻撃が成功してしまっています。とはいえ、ネットワークに何も暗号化を行わないよりは、まだ WEP を使用したほうがよい選択ではあります。
  また、製造元によっては非標準の「ダイナミック WEP」と呼ばれるものを実装していることがあります。これは暗号という観点では WEP と同じであり、脆弱性についても WEP と同じ問題を抱えています。唯一の違いは、鍵管理サービスによって鍵が定期的に変更される点だけです。

TKIP (WPA/IEEE 802.11i 内で規定)
: この鍵管理プロトコルは WPA 標準の中で決められたもので、WEP と同じ暗号化方式を使用しているものの、その脆弱性を克服した仕組みになっています。それぞれのデータパケットに対して新しい鍵を生成する仕組みになり、鍵に対して攻撃を加えても意味のない仕組みになっています。TKIP は WPA-PSK と併用する仕組みです。

CCMP (IEEE 802.11i 内にて規定)
: CCMP は鍵の管理方法を定義しているものです。通常は WPA-EAP の接続で使用するためのものですが、WPA-PSK でも使用することができます。暗号化は AES と呼ばれる仕組みに従って行われ、WEP の標準である RC4 よりは強固な暗号化を実現しています。

27.1.3.4 無線通信: Bluetooth

Bluetooth は全ての無線技術の中で最も広い用途を提供する技術です。従来は IrDA を使用していたコンピュータ (ラップトップ) と PDA ／携帯電話との通信に使用されているばかりか、範囲内にあるコンピュータ間の通信にも使用されます。また、Bluetooth はキーボードやマウスなどの周辺機器の接続にも使用されます。ただし、この無線技術は、ネットワーク内の離れたシステムと接続できるほどの距離には対応しておらず、物理的な壁などが存在するような場合は、今も Wi-Fi が適切な選択肢となっています。

27.1.3.5 無線通信: IrDA

IrDA は短距離向けの無線技術です。通信相手がお互いに見える範囲での通信を目的にしています。壁のような障害物を乗り越えることはできません。IrDA の一般的な用途としては、ラップトップと携帯電話との間でのデータ交換などがあります。近接した環境であれば、IrDA が最適です。それ以上の長距離を必要とする場合は、モバイルネットワークをお使いください。それ以外の IrDA の用途としては、企業内での印刷ジョブの無線送信などもあります。

27.1.4 データセキュリティ

現実的には、ラップトップ内のデータを保護するには、様々な方式で無許可のアクセスを防止する必要があります。データセキュリティという観点では、下記のような保護を考慮する必要があります:

盗難からの保護

できる限りの範囲で、常に物理的な盗難から保護する必要があります。小売店には様々な盗難防止ツール (鎖など) が販売されています。

強固な認証

ユーザ名とパスワードによる標準的な認証に加えて、生体認証を追加することもできます。openSUSE Leap では、指紋認証に対応しています。

システム内のデータの保全

重要なデータはその伝送時にのみ暗号化されていれば良いというものではなく、ハードディスク内でも暗号化しておくことをお勧めします。これにより、盗難に遭った場合の被害を防ぐことができます。なお、openSUSE Leap で暗号化パーティションを作成する方法については、**『セキュリティ強化ガイド』、第13章「パーティションやファイルの暗号化」** をお読みください。また、YaST でユーザを作成する際、ホームディレクトリを暗号化することもできます。

重要: データセキュリティとディスクへのサスペンドについて

暗号化されたパーティションは、ディスクへのサスペンドのイベントが発生した場合も、マウントが解除されません。そのため、ハードウエアごと盗み出してしまえば、ハードディスクの休止状態を解除するだけで、全てのデータを読み出すことができることになります。

ネットワークセキュリティ

全てのデータ転送は、その転送方法にかかわらず常に暗号化しておくべきです。Linux やネットワークでの一般的なセキュリティ問題について、詳しくは『セキュリティ強化ガイド』、第1章「セキュリティと機密保持」をお読みください。

27.2 モバイルハードウエア

openSUSE Leap では FireWire (IEEE 1394) や USB で接続されたモバイルストレージデバイスに対して、それらを自動検出する仕組みが用意されています。ここでの モバイルストレージデバイス とは、FireWire や USB で接続されたハードディスクのほか、フラッシュメモリやデジタルカメラなども意味します。これらのデバイスは、対応するインターフェイスに接続された時点で自動検出され、自動設定が行われます。GNOME のファイルマネージャでは、モバイルハードウエアを柔軟に取り扱うための仕組みが備わっています。これらのメディアを安全に取り外すには、ファイルマネージャの［ボリュームのアンマウント］(GNOME の場合) 機能を利用してください。詳しくは『GNOME ユーザガイド』をお読みください。

外付けハードディスク (USB および FireWire)

外付けハードディスクがシステム側で正しく認識されると、ファイルマネージャ内にアイコンが表示されるようになります。そのアイコンを押すことで、ドライブの内容を表示することができるようになります。また、そこからディレクトリやファイルを作成したり、編集や削除を行ったりすることもできます。ハードディスクの名前を変更したい場合は、コンテキストメニューから対応するメニュー項目を選んで設定してください。ただし、ここでの名前変更はファイルマネージャ内での表示にのみ反映され、`/media` 以下の名前は変更できません。

USB メモリ

これらのデバイスは外付けハードディスクと同様に動作します。ファイルマネージャでの名前の変更についても、外付けハードディスクと同様です。

デジタルカメラ (USB および FireWire)

システム側でデジタルカメラが認識されると、ファイルマネージャ内で外付けドライブとして表示されます。ここから、撮影済みの画像を既定の画像エディタで処理することができるようになります。より高度な処理を行いたい場合は、GIMP などを利用します。GIMP についての詳細は、『GNOME ユーザガイド』、第17章「GIMP: 画像の編集」をお読みください。

27.3 モバイルデバイス (スマートフォンやタブレットなど)

デスクトップシステムやラップトップシステムとモバイルデバイスとの通信に際しては、Bluetooth や Wi-Fi のほか、USB による直接接続も使用することができます。どのような接続形式が利用できるのかは、お使いのモバイルデバイスの型式やお使いの際の要件によって異なります。モバイルデバイスとデスクトップ／ラップトップとの間を USB で接続した場合は、一般的な外付けストレージとして使用することもできます。また Bluetooth や Wi-Fi で接続した場合は、デスクトップマシンやラップトップマシンから、モバイルデバイスを制御したり直接操作したりすることができるようになります。モバイルデバイスとの接続に対しては、いくつかのオープンソースのグラフィカルユーティリティが提供されています (たとえば KDE Connect (https://community.kde.org/KDEConnect) や GSConnect (https://extensions.gnome.org/extension/1319/gsconnect/) などがあります) 。

28 NetworkManager の使用

改訂履歴

2024-05-13

NetworkManager はラップトップなどの可搬型コンピュータにおける現実的なソリューションです。最先端の暗号化方式やネットワーク接続の標準に対応していて、802.1X で保護されたネットワークも利用することができます。802.1X は「IEEE Standard for Local and Metropolitan Area Networks—Port-Based Network Access Control」(ローカルおよび地域エリアネットワーク向け IEEE 標準 — ポートベースのネットワークアクセス制御) と呼ばれるもので、NetworkManager ではネットワークインターフェイスの設定や移動時の無線／有線のネットワーク切り替えなどを、何も不安になることなく実施できる仕組みを備えています。NetworkManager では既知の無線ネットワークが検出されれば、自動的に接続することができるほか、複数のネットワークをまとめて管理することもできます。既定では最も高速なネットワーク接続を使用します。これに加えて、ネットワーク間の切り替えや接続の管理などは、システムトレイ内に常駐しているアプレットから設定することができます。

また、いずれか 1 つのネットワークのみを有効化させることができるだけでなく、複数のネットワークを同時に有効化することができます。この仕組みにより、ラップトップを有線イーサネットから切り離しても、無線接続で継続して使用できるようになります。

28.1 NetworkManager の使用例

NetworkManager では洗練された直感的なユーザインターフェイスを提供しています。これにより、容易にネットワーク環境を切り替えることができるようになっています。ですが、下記のような条件に該当する場合は、NetworkManager をお勧めできません:

- お使いのコンピュータが、ネットワーク内の他のコンピュータに対して何らかのネットワークサービスを提供している場合 (例: DHCP サーバ, DNS サーバなど) 。
- お使いのコンピュータが Xen サーバであるか、もしくは Xen 内の仮想システムである場合。

28.2 NetworkManager の有効化と無効化

デスクトップ型やラップトップ型のコンピュータの場合、NetworkManager は既定で有効化されます。YaST のネットワーク設定モジュールを利用することで、必要に応じて後から有効化／無効化することができます。

1. YaST を起動して、[システム] › [ネットワークの設定] を選択します。

2. ［ネットワークの設定］ダイアログが表示されます。ここから［グローバルオプション］タブを選択します。

3. NetworkManager でネットワーク接続を管理するように設定するには、下記の手順で行います:

    a. ［ネットワークの設定方法］で［NetworkManager サービス］を選択します。

    b. ［OK］を押し、YaST を閉じます。

    c. NetworkManager でのネットワークの設定方法については、28.3項「ネットワーク接続の設定」をお読みください。

4. NetworkManager を無効化し、YaST 内でネットワーク接続を管理するように設定するには、下記の手順で行います:

    a. ［ネットワークの設定方法］で［Wicked サービス］を選択します。

    b. ［OK］を押します。

    c. あとは YaST を利用して、DHCP による自動設定を行うか、もしくは固定の IP アドレスを設定します。
    YaST でのネットワークの設定方法については、13.4項「YaST を利用したネットワークの設定」をお読みください。

28.3 ネットワーク接続の設定

YaST で NetworkManager を有効化したあとは、GNOME 向けに用意された NetworkManager のフロントエンドを利用して、ネットワーク接続を管理します。有線、無線、モバイルブロードバンド、DSL, VPN 接続など、全ての種類のネットワーク接続に対応するタブが表示されます。

GNOME でネットワーク接続のダイアログを開くには、ステータスメニューから設定メニューを開いて、［ネットワーク］の項目を選択します。

注記: オプションの選択肢について

お使いのシステムの設定によっては、特定のネットワーク接続の管理を許可していない場合があります。特に機密を重視するようなシステムでは、いくつかの設定がロック (施錠) されていたり、`root` の権限が必要となったりする場合があります。詳しくはシステム管理者にお尋ねください。

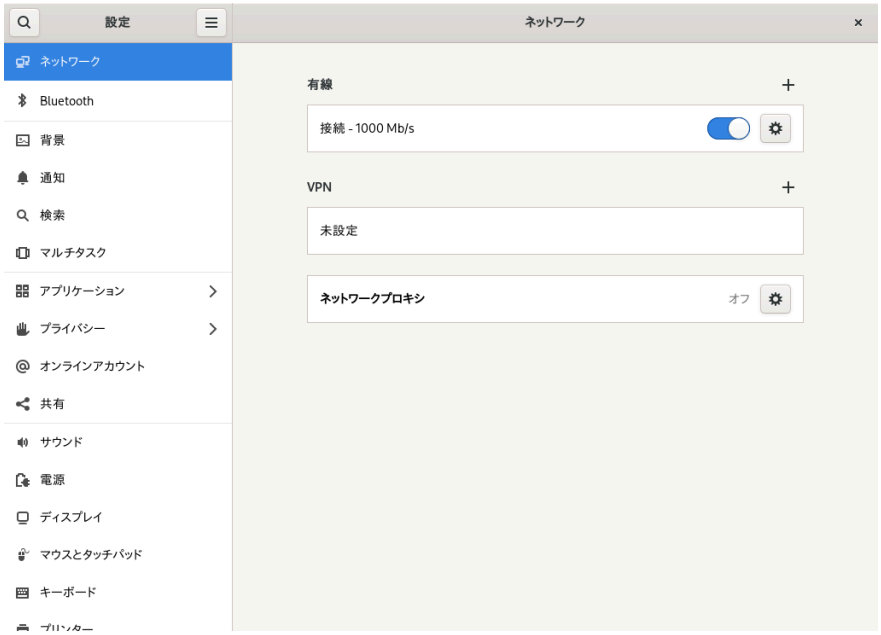


図 28.1: GNOME ネットワーク接続ダイアログ

手順 28.1: 接続の追加と編集

1. ステータスメニューを開いて歯車アイコンを押し、［設定］を表示させた状態で、左側のメニューから［ネットワーク］を選択します。
2. 接続を追加するには、下記の手順を実施します:
    a. 追加したい接続タイプのタブの隣にある［+］アイコンを押します。
    b. 選択した接続タイプに応じて、必要な項目に入力します。
    c. 完了したら［追加］を押します。
    d. 設定を追加すると、ネットワーク内の一覧に新しく作成した設定が表示されますので、そこから接続を行うことができるようになります。
3. 接続を編集するには、下記の手順を実施します:
    a. 編集したい接続タイプのタブの右側にある歯車アイコンを押します。
    b. あとは必要な変更を行って［適用］を押すと、設定を保存することができます。
    c. 作成した接続をシステム接続として設定したい場合は、［Details］タブに移動して、［Make available to other users］のチェックボックスにチェックを入れます。ユーザ接続とシステム接続について、詳しくは 28.4.1項「ユーザ接続とシステム接続」をお読みください。

28.3.1 有線ネットワーク接続の管理

お使いのコンピュータが有線ネットワークに接続されている場合は、NetworkManager アプレットを利用して接続を管理することができます。

1. ステータスメニューを開いて［有線］を選択し、必要に応じて切断や設定変更 (右矢印) などを行います。
2. 設定を変更したい場合は、［有線設定］を押して歯車アイコンを押します。
3. 全てのネットワーク接続を切断したい場合は、［オフにする］を押します。

28.3.2 無線ネットワーク接続の管理

無線ネットワークのうち SSID を公開しているネットワークについては、GNOME の NetworkManager アプレット内の［無線ネットワーク］に表示されます。それぞれのネットワークに対しては、信号の強度も合わせて表示されます。また、暗号化された無線ネットワークの場合は、盾のアイコンが表示されます。

手順 28.2: 表示されているネットワークへの接続

1. 表示されている無線ネットワークに接続したい場合は、ステータスメニューから［Wi-Fi］を選択します。
2. Wi-Fi が有効化されていない場合は、［オンにする］を押します。
3. ［ネットワークを選択］を押します。表示された Wi-Fi ネットワークの中からいずれかを選択して、［接続］を押します。
4. 選択したネットワークが暗号化されている場合は、設定ダイアログが表示されます。ここにはネットワークが使用している暗号化の種類と、ログイン認証を設定するためのテキストボックスが表示されます。

手順 28.3: 表示されていないネットワークへの接続

1. 識別情報 (SSID もしくは ESSID) を公開していないネットワークに接続する場合は、それらは自動では検出されませんので、ステータスメニューを開いて［Wi-Fi］を押します。
2. ［Wi-Fi 設定］を押して、詳細設定メニューを表示します。
3. Wi-Fi が有効化されていることを確認して、［非表示のネットワークに接続］を押します。
4. ダイアログが表示されたら、［ネットワーク名］の欄に SSID もしくは ESSID を入力し、必要に応じて暗号化パラメータを設定します。

明示的に選択した無線ネットワークに対しては、できる限り接続を維持しようとします。その際にネットワークケーブルが接続されると、［可能であれば接続を維持する］に設定したネットワーク側に接続され、無線接続については維持を行います。

28.3.3 Wi-Fi/Bluetooth カードに対するアクセスポイントとしての設定方法

お使いの Wi-Fi や Bluetooth カードがアクセスポイントモードに対応している場合は、NetworkManager を利用して設定を行うことができます。

1. ステータスメニューを開いて［Wi-Fi］を押します。
2. ［Wi-Fi 設定］を押して、詳細設定メニューを表示します。
3. ［アクセスポイントとして使用］を押して、表示された手順に従います。
4. あとは表示されたダイアログ内に認証情報を入力し、リモートのマシンからホットスポットに接続します。

28.3.4 NetworkManager と VPN

NetworkManager は複数の仮想プライベートネットワーク (VPN; Virtual Private Network) に対応しています。openSUSE Leap ではそれぞれの技術に対して、NetworkManager の一般的なサポートを提供する基本パッケージが用意されています。これに加えて、お使いのアプレット向けに用意されたデスクトップ固有パッケージをインストールする必要もあります。

OpenVPN
: この VPN 技術を使用したい場合は、下記をインストールします:

  - `NetworkManager-openvpn`
  - `NetworkManager-openvpn-gnome`

OpenConnect
: この VPN 技術を使用したい場合は、下記をインストールします:

  - `NetworkManager-openconnect`
  - `NetworkManager-openconnect-gnome`

PPTP (point-to-point tunneling protocol)

この VPN 技術を使用したい場合は、下記をインストールします:

- `NetworkManager-pptp`
- `NetworkManager-pptp-gnome`

下記の手順は、NetworkManager を利用してお使いのコンピュータを OpenVPN クライアントとして動作させる手順です。他の種類の VPN についても、下記と同様の手順で行うことができます。

まずは必要となるパッケージ `NetworkManager-openvpn-gnome` がインストールされ、全ての依存関係が満たされていることを確認します。

手順 28.4: NETWORKMANAGER による OPENVPN の設定

1. パネルの右端にあるステータスアイコンを押して［設定］アプリケーションを開きます。そこからレンチとネジ回しの表示されたアイコンを押します。すると、［設定］画面が表示されます。ここから［ネットワーク］を選択します。
2. ［+］アイコンを押します。
3. ［VPN］から［OpenVPN］を選択します。
4. まずは［認証］の種類を選択します。OpenVPN サーバ側の設定によって、［証明書 (TLS)］もしくは［パスワードと証明書 (TLS)］のいずれかを選択します。
5. それぞれのテキストボックス内に必要な情報を入力します。選択した認証方式によって異なりますが、たとえば下記のとおり入力します:

［ゲートウェイ］	VPN サーバの相手側のアドレスまたはホスト名
［ユーザー名］	ユーザ名 (［パスワードと証明書 (TLS)］) を選択した場合のみ)
［パスワード］	ユーザに対するパスワード (［パスワードと証明書 (TLS)］) を選択した場合のみ)
［User 証明書］	`/etc/openvpn/client1.crt`
［CA 証明書］	`/etc/openvpn/ca.crt`
［プライベートキー］	`/etc/openvpn/client1.key`

6. 設定が完了したら［追加］を押します。

7. 接続を有効化するには、［設定］アプリケーション内の［ネットワーク］パネルから、スイッチを切り替えます。それ以外にも、パネルの右端にあるステータスアイコンから、VPN の名前を選択して［接続］を押してもかまいません。

28.4 NetworkManager とセキュリティ

NetworkManager では無線接続を信頼済みと未信頼の 2 種類に分けて管理しています。信頼済みの接続とは、以前に明示的に接続するよう選択したネットワークのことで、それ以外は全て未信頼として扱われます。信頼済みの接続は、名前とアクセスポイントの MAC アドレスで識別します。このように MAC アドレスを識別に加えることで、同じ名前の別のアクセスポイントを、誤って信頼してしまわないようにしています。

NetworkManager は利用可能な無線ネットワークを定期的に検出します。複数の信頼済みネットワークが検出された場合は、最近使用していたものを自動選択します。また NetworkManager では、検出された全てのネットワークが未信頼のものであった場合は、いずれにも自動接続せずに待機します。

名前と MAC アドレスが同じであるにもかかわらず、暗号化の設定が異なった場合も NetworkManager は接続を試行しますが、新しい暗号化設定 (暗号鍵など) を確認するダイアログを表示します。

無線接続をオフラインモードに切り替えた場合は、NetworkManager は SSID や ESSID を空白に設定します。これにより、カード側に対して明示的な切り離しを指示することになります。

28.4.1 ユーザ接続とシステム接続

NetworkManager では 2 種類のネットワーク接続を管理しています。それぞれ **ユーザ** 接続と **システム** 接続と呼ばれています。

ユーザ接続では、各ユーザに対して NetworkManager 内での認証を求めます。これにより、ユーザの認可情報をローカルの GNOME キーリング内に保存させ、接続時に毎回尋ねたりすることがないようにしています。

システム接続は全てのユーザに対して自動的に提供される接続です。接続を作成したユーザが必要な認証情報を入力すると、全てのユーザからその接続を利用できるようになります。このとき、各ユーザが認証情報を知っている必要はありません。なお、ユーザ接続とシステム接続は、チェックボックス［Make available to other users］で制御します。NetworkManager でユーザ接続やシステム接続を作成する方法について、詳しくは 28.3項「ネットワーク接続の設定」をお読みください。

28.4.2 パスワードなどの認証情報の保存

暗号化されたネットワークに接続する際、認証情報を毎回入力したくない場合は、GNOME のキーリングマネージャを利用して、ディスク内に認証情報を保存することができます。認証情報は、マスターパスワードで暗号化されディスク内に保存されます。

28.5 よくある質問

下記には、NetworkManager で特殊なネットワーク設定を行う際に、よくある質問をいくつか並べてあります。

問: 1.接続を特定のデバイスに結びつけるには?

既定では、NetworkManager 内の接続は種類にのみ結びついていて、種類さえ同じであれば全ての物理デバイスに適用されます。たとえば 2 枚のイーサネットカードが接続されたマシンなど、同じ種類の物理デバイスが複数存在するような環境では、特定のデバイスのみを使用するように接続を設定することができます。

GNOME でこれを行うには、まずお使いのデバイスの MAC アドレスをご確認ください (アプレット内の［接続情報］内に書かれているほか、`nm-tool` や `wicked show all` などのコマンドラインからも確認することができます) 。あとはネットワーク接続のダイアログを開いて、変更したいネットワーク接続を選びます。あとは［有線］もしくは［無線］のタブで、デバイスの［MAC アドレス］を入力して保存してください。

問: 2.同じ ESSID のアクセスポイントを複数検出した場合、特定のアクセスポイントを指定して接続するには?

異なる周波数帯 (a/b/g/n) で複数のアクセスポイントが検出された場合、既定では最も強い信号を検出したアクセスポイントに対して接続を行います。この既定値を変更したい場合は、無線の接続を設定する際に［BSSID］の値を指定してください。

BSSID は Basic Service Set Identifier と呼ばれるもので、基本サービスセットを識別するための値です。インフラストラクチャモードでは、BSSID はアクセスポイントの MAC アドレスと同じになります。アドホックモードでは、BSSID は 46 ビットの乱数から生成されるローカル管理 MAC アドレスと同じになります。

28.3項「ネットワーク接続の設定」の手順に従ってネットワーク接続の設定ダイアログを開いて、修正したい無線接続を選んで［編集］を押します。あとは［無線］タブ内で BSSID を指定してください。

問: 3.他のコンピュータと接続を共有するには?

プライマリデバイス (インターネットに接続されているデバイス) 側では、特別な設定は不要です。しかしながら、ローカルハブやローカルマシンに接続されている側については、下記のように設定する必要があります:

1. まずは 28.3項「ネットワーク接続の設定」の手順に従って、ネットワーク接続を管理するためのダイアログを表示します。続いて設定を変更したい接続を選んで [編集] を押し、[IPv4 設定] タブに切り替えます。さらに [メソッド] ドロップダウンリストから [他のコンピューターと共有する] を選択します。これにより、IP トラフィックの転送とデバイス内での DHCP サーバの有効化が行われます。あとは設定を保存して終了してください。

2. DHCP サーバはポート 67 を使用しますので、ファイアウオールでこのポートがブロックされていないことをご確認ください。接続を共有する側のマシンで YaST を開いて、[セキュリティとユーザ] › [ファイアウオール] を選択し、[許可するサービス] のカテゴリに切り替えます。もしも [許可するサービス] 欄に [DHCP サーバ] が表示されていない場合は、[サービス] の一覧から [DHCP サーバ] を選択して [追加] を押します。あとは設定を保存して終了してください。

問: 4.自動アドレス設定 (DHCP, PPP, VPN) の場合、DNS の情報を固定で設定するには?

DHCP サーバが正しくない DNS 情報 (もしくはルーティング情報) を提供するような場合は、それらの設定を上書きする必要があります。まずは 28.3項「ネットワーク接続の設定」の手順に従って、ネットワーク接続を管理するためのダイアログを表示します。そこから設定を変更したい接続を選んで [編集] を押します。[IPv4 設定] タブに切り替えたあと、[メソッド] ドロップダウンボックスで [自動 (DHCP) アドレスのみ] を選択します。あとは [DNS サーバ] の欄と [検索ドメイン] の欄にそれぞれ必要な情報を入力します。また、[ルート] では [自動取得したルートを無視する] を選択して、必要なチェックボックスにチェックを入れて保存してください。

問: 5.ユーザがログインする前に、NetworkManager を利用してパスワードで保護されたネットワークに接続するには?

システム接続 を作成することで、そのような目的を達成することができます。詳しくは 28.4.1項「ユーザ接続とシステム接続」をお読みください。

28.6 トラブルシューティング

接続の問題はよく発生する問題です。アプレットが起動しないとか、VPN のオプションが存在しないなどの問題は NetworkManager 側の問題ですが、いずれの場合も、問題の解決や防止は、使用しているツールに依存して決まります。

NetworkManager デスクトップアプレットが起動しない

ネットワークの設定を NetworkManager で行うように設定すれば、必要なアプレットは自動的に起動するようになります。アプレットが起動しない場合は、28.2項「NetworkManager の有効化と無効化」で説明しているとおり、まずは YaST で NetworkManager が有効化されているかどうかを確認してください。また、NetworkManager-gnome パッケージのインストールについても確認してください。

デスクトップアプレットがインストールされているにもかかわらず起動しない場合は、コマンド `nm-applet` を実行すると開始することができます。

NetworkManager アプレットで必要な VPN オプションが表示されない

NetworkManager, アプレット, NetworkManager の VPN モジュールは、いずれも別々のパッケージとして提供されています。NetworkManager アプレットで必要な VPN の種類が表示されない場合は、まず利用予定の VPN 技術に対応する NetworkManager のサポートパッケージがインストールされていることを確認してください。詳しくは 28.3.4項「NetworkManager と VPN」をお読みください。

ネットワーク接続が利用できない

正しくネットワークの接続を設定していて、ルータなどのネットワーク接続に必要な機材も動作しているにもかかわらず、接続ができない場合は、コンピュータ内のネットワークインターフェイスを再起動すると解決する場合があります。これを行うには、ログインしてから `root` になり、`systemctl restart wickeds` を実行してください。

28.7 さらなる情報

NetworkManager に関する詳しい情報は、下記の Web サイトやディレクトリ内に用意されています:

NetworkManager プロジェクトのページ

https://gitlab.freedesktop.org/NetworkManager/NetworkManager

パッケージのドキュメンテーション

NetworkManager や GNOME アプレットの最新情報については、下記のディレクトリ内に情報が用意されています:

- "¥n `/usr/share/doc/packages/NetworkManager/` ¥n"
- "¥n `/usr/share/doc/packages/NetworkManager-gnome/` ¥n"

29 電源管理

改訂履歴

2024-05-13

電源管理は特にラップトップ型のコンピュータで重要な機能ですが、それ以外のシステムでも様々な利便性を提供します。現在の全てのコンピュータ (ラップトップやデスクトップ、サーバなど) には ACPI (Advanced Configuration and Power Interface) と呼ばれる電源管理機能が用意されていますが、このような技術にはハードウエア側の対応だけでなく、BIOS 側のソフトウエアにも対応が必要となります。また、騒音を減らす目的から、CPU の動作周波数を変更することもできます。

29.1 電源管理機能

電源管理機能はラップトップ型のコンピュータのみで使用する機能ではなく、デスクトップシステムでも重要な仕組みです。ACPI の主な機能と用途は下記の通りです:

スタンバイ
: 対応していません。

サスペンド (メモリに対して)
: このモードは、システムの状態を全てメモリ内に書き込んだあと、メモリ以外のシステムをスリープ状態に移行させます。この状態では、コンピュータの消費電力を大幅に削減することができます。このような仕組みにより、システムやアプリケーションを起動し直したりすることなく、利用をすぐに再開できるようになります。この機能は、ACPI の `S3` と呼ばれる状態 (ステート) に対応しています。

ハイバネーション (ディスクに対するサスペンド)
: この操作モードでは、システムの状態を全てハードディスク内に書き込んで、システムの電源を落とします。必要なデータを全て書き込む必要があることから、少なくとも搭載されているメモリ量と同じサイズのスワップパーティションが必要となります。この状態からの復帰には、30 秒から 90 秒程度の時間が必要となります。復帰処理が終了することで、サスペンド前の状態に戻ることになります。ハードウエアの製造元によっては、IBM Thinkpad の RediSafe のように、ハイバネーションとサスペンドを組み合わせた動作モードを提供しているものもあります。この機能は、ACPI の `S4` と呼ばれる状態 (ステート) に対応しています。Linux では、ディスクへのサスペンドは ACPI ではなく、独自のカーネルルーチンを利用して行います。

注記: `mkswap` でフォーマットした場合のスワップパーティションの UUID 更新について

`mkswap` でのフォーマットは、できる限り避けておくことをお勧めします。それは、`mkswap` でフォーマットし直してしまうと、スワップパーティションの UUID が変更されてしまうためです。その代わりに、YaST を利用してフォーマットし直す (YaST では、UUID が変更されても `/etc/fstab` を自動的に 更新します) か、`mkswap` コマンド実行後に手作業で `/etc/fstab` を更新してください。

バッテリー監視

ACPI ではバッテリーの充電状態などの情報をチェックしています。また、特定の充電量を下回った場合に、何らかの動作を行うこともできます。

自動電源切断

シャットダウンの実施後、コンピュータの電源が自動的に切られます。これは特に、バッテリーがほとんど空になってしまったような場合に重要な機能となります。

プロセッサの速度制御

CPU という観点では、省電力は下記の 3 種類の方法で実現することができます: 周波数と電圧の制御 (PowerNow! や Speedstep として知られた制御方法です) のほか、動作速度の調整やスリープモード (C-ステート) への移行などがあります。コンピュータの動作モードによっては、複数の方式を組み合わせて使用することもできます。

29.2 Advanced Configuration and Power Interface (ACPI)

ACPI はオペレーティングシステムから個別のハードウエアコンポーネントを設定し、制御することを目指して設計された仕組みです。ACPI は Power Management Plug and Play (PnP) と Advanced Power Management (APM) の両方の後継となる仕組みで、バッテリーや AC アダプタ、温度や冷却ファン、システムイベント (たとえば「ラップトップの蓋を閉じた」イベントや、「バッテリーの残り容量が少ない」などのイベント) を配信することができます。

BIOS 側では、個別のコンポーネントやハードウエアへのアクセス方法に関する情報を含むテーブルを提供しています。オペレーティングシステムはこの情報を利用して、割り込みの割り当てやコンポーネントの有効化／無効化などの処理を行います。オペレーティングシステム側では、BIOS 内に保存されたコマンドを実行することから、その機能は BIOS 側の実装に依存することになります。ACPI が検出して読み込むテーブルは、journald 内に記録されます。ジャーナルログメッセージを表示する方法に

ついて、詳しくは 第11章「journalctl：systemd ジャーナルへの問い合わせコマンド」をお読みください。また、ACPI で発生する様々な問題の解決方法については、29.2.2項「トラブルシューティング」をお読みください。

29.2.1 CPU の性能制御

CPU は下記の 3 種類の方法で、消費電力を削減することができます:

- 周波数と電圧の制御
- クロック周波数の低減 (T-ステート)
- プロセッサに対するスリープモードへの移行 (C-ステート)

コンピュータの動作モードによっては、これらの方式を組み合わせて使用することもできます。省電力を実現することによって、システムの発熱を抑えたり、冷却ファンの動作を抑えたりすることにも繋がります。

周波数の制御と動作速度の調整は、プロセッサが処理中の間にのみ影響があります。なぜなら、プロセッサの処理が空いた状態になると、最も省電力の C-ステートが適用されるためです。CPU の処理が多い場合は、周波数制御を行うことで省電力を実現することができます。ただし、多くの場合、プロセッサに対しては間欠的な負荷になりますので、より低い周波数に設定するため、カーネルが提供するオンデマンドガバナーを利用して動的に制御するのが最適です。

動作速度の調整は最後の手段として使われるべきものです。たとえば、システムの負荷が高いにもかかわらず、バッテリーの動作時間を延ばしたい場合などがそれにあたります。動作速度を低くしすぎると、システムによってはうまく動作しなくなってしまうこともあるほか、それ以上速度に低くできなくなってしまう場合もあります。

より詳しい情報については、『システム分析／チューニングガイド』、第12章「電源管理」をお読みください。

29.2.2 トラブルシューティング

ACPI 関連で発生する問題には、2 種類のものがあります。1 つには、カーネル内の ACPI 関連のコードにバグがあって、開発時に見つけられなかったものがあります。この場合は、ダウンロードで解決方法を提供することができます。もう 1 つは BIOS によって引き起こされる問題があります。こちらのほうがより頻繁に発生することになりますが、ACPI 仕様から外れた実装が BIOS 内に存在していて、これによってオペレーティングシステム側をエラーに導いてしまうものがあります。なお、ACPI 実装内に深刻なエラーがあるハードウエアコンポーネントが検出されると、Linux カーネル側でこれらに対する ACPI を使用しないようにするため、ブラックリストを記録することもあります。

ACPI に関して何らかの問題が見つかった場合、最初に試すべきことは BIOS の更新です。コンピュータが起動しないような場合は、下記のいずれかのパラメータを指定することで、問題を回避できる場合もあります:

pci=noacpi
: PCI デバイスを設定する際に ACPI を使用しないようにします。

acpi=ht
: シンプルなリソース設定のみを実施します。その他の目的では ACPI を使用しないようにします。

acpi=off
: ACPI を無効化します。

警告: ACPI を使用しない場合の起動時の問題について

新しいマシン (特に SMP システムや AMD64 システム) では、ハードウエアを正しく設定するために ACPI が必要となります。これらのシステムで ACPI を無効化すると、逆に問題を引き起こす場合があります。

また、マシンに対して USB や FireWire のデバイスが接続されていると、ハードウエアの認識がうまくいかなくなる場合があります。マシンが起動しない場合は、不要なハードウエアを全て取り外してから、起動をお試しください。

まずはシステムの起動後に `dmesg -T | grep -2i acpi` を実行して、システムの起動メッセージをご確認ください。最も重要なテーブルである ACPI テーブル (DSDT (Differentiated System Description Table)) を処理する際にエラーが発生した場合は、改善版に置き換えることもできます。この場合は、BIOS 側に書かれている DSDT が無視されます。詳しい流れについては、29.4項「トラブルシューティング」をお読みください。

カーネル側を設定することで、ACPI デバッグメッセージを出力するように設定することもできます。ACPI デバッグ機能をコンパイルしてインストールすることで、詳細な情報が出力されるようになります。

BIOS やハードウエア関連の問題に直面した場合は、まず製造元に連絡してみることをお勧めします。Linux の場合は十分な支援を受けられない場合もありますが、問題の解決は彼らが行うべきものであるためです。Linux を利用するユーザがそれなりの規模になれば、製造元も問題に真剣に対応することになるでしょう。

29.2.2.1 さらなる情報

- https://tldp.org/HOWTO/ACPI-HOWTO/ (詳細な ACPI HOWTO のほか、DSDT パッチなども含まれています)
- http://www.acpi.info (Advanced Configuration & Power Interface Specification に関する情報)

29.3 ハードディスクの休止

Linux では、ハードディスクが不要であればスリープ状態に設定することができるほか、より経済的で静かなモードで動作させることもできます。新しいラップトップ機であれば、ハードディスクが不要になると自動的に省電力モードに移行しますので、手作業でハードディスクの電源を切る必要はありません。ただし、最大限に省電力を実現したい場合は、`hdparm` コマンドを利用して試してみることをお勧めします。

このコマンドは、様々なハードディスク設定を変更することができます。`-y` オプションを指定すると、ハードディスクを即時にスタンバイモードに移行するよう指示します。`-Y` を指定すると、スリープモードに移行することもできます。また、`hdparm -S X` のように実行すると、指定した時間だけハードディスクへのアクセスが行われなくなると、自動的に回転を停止させることもできます。ここでの `X` は、それぞれ下記のような意味があります: `0` を指定すると、回転を停止せずに動作し続ける意味になります。`1` から `240` までの値を指定すると、その値に 5 [秒] を掛けた時間、`241` から `251` までの値を指定すると、30 分から 30 分の 11 倍までの時間、アクセスがなくなると回転を停止する意味になります。

ハードディスク内部の省電力オプションは、`-B` オプションで制御することができます。`0` (最も省電力) から `255` (最大の性能) までの値で指定します。設定の硬貨はハードディスクによって異なりますので、一概に測定できるものではないことに注意してください。また、ハードディスクの騒音を抑えたい場合は、`-M` オプションを指定してください。こちらは `128` (最も静音) から `254` (最も高速) までの値を指定することができます。

環境によっては、ハードディスクをスリープ状態に置くのは容易ではないことがあります。Linux では様々なプロセスがハードディスクを利用するため、頻繁にスリープが解除されることになります。そのため、Linux がハードディスクにデータを書き込む頻度について、あらかじめ知っておく必要があります。まず全てのデータはメモリ内にバッファされますが、このバッファは `pdflush` デーモンが監視しています。特定の時間期限もしくは一定量を超過すると、バッファの内容をハードディスクに書き込みます。

バッファサイズは動的に設定され、搭載されているメモリ量とシステムの負荷に依存して決まります。既定では pdflush は 5 秒おきにバッファを確認して、必要であると判断すればハードディスクにデータを書き込みます。このほかにも、下記の設定値が提供されています:

`/proc/sys/vm/dirty_writeback_centisecs`
: pdflush スレッドを起動するまでの時間遅延を設定します (100 分の 1 秒単位) 。

`/proc/sys/vm/dirty_expire_centisecs`
: 書き込むべきページが書き込まれるまでの最大の時間間隔を設定します。既定値は `3000` で、30 秒の意味になります。

`/proc/sys/vm/dirty_background_ratio`
: pdflush が書き込みを行うまでの、書き込むべきページの最大割合を指定します。既定値は `5` % です。

`/proc/sys/vm/dirty_ratio`
: 搭載されているメモリ量に対して、この割合以上に書き込むべきページが溜まった場合、書き込みを続ける前にそれらのページを書き込むよう強制する割合を指定します。

警告: データ損失を引き起こす可能性について

`pdflush` デーモンの設定を変更すると、データの整合性が失われてしまう危険性があります。

これらのプロセスとは別に、`Btrfs` , `Ext3` , `Ext4` などのジャーナル機能のあるファイルシステムでは、`pdflush` とは別にメタデータを個別に書き込みます。これによってハードディスクの回転停止が阻害されることがあります。この問題を回避するため、モバイルデバイス向けの特別なカーネル拡張が開発されています。この拡張を利用するには、`laptop-mode-tools` パッケージをインストールのうえ、`/usr/src/linux/Documentation/laptops/laptop-mode.txt` を読んで設定を行ってください。

もう 1 つの重要な要素として、動作中のプログラムの存在にもご注意ください。たとえば有用なエディタでは現在編集中のファイルを定期的に書き込むような処理が存在していたりして、定期的にハードディスクを回転させるようにし向けることができます。ただし、このような機能を無効化してしまうと、何らかの不具合が発生した場合に編集中のファイルを取り戻せなくなってしまうリスクが発生します。

また、これに関連して、メールデーモンである postfix には `POSTFIX_LAPTOP` という変数が用意されています。この値を `yes` にすると、ハードディスクへのアクセスを大きく減らすことができます。

openSUSE Leap では、これらの技術は `laptop-mode-tools` が制御します。

29.4 トラブルシューティング

全てのエラーメッセージや警告などは、システムのジャーナル内に保存されます。ジャーナルへの問い合わせは `journalctl` コマンドを使用します (詳しくは 第11章「`journalctl`：systemd ジャーナルへの問い合わせコマンド」をお読みください) 。下記の章では、最もよく発生する問題を説明しています。

29.4.1 CPU の周波数制御が動作しない

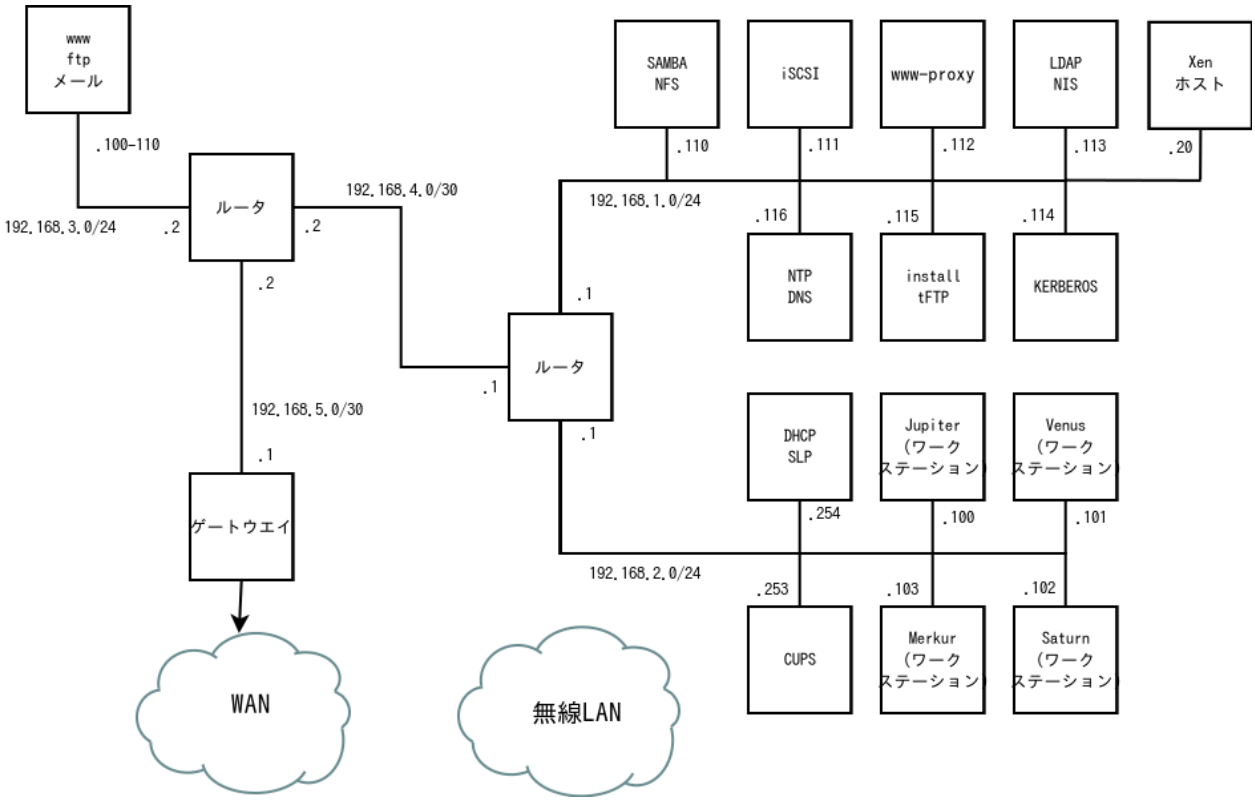
まずはカーネルのソースコードを参照して、お使いのプロセッサに対応しているかどうかをご確認ください。また、CPU の周波数を制御するには、特別なカーネルモジュールやモジュールオプションが必要となる場合もあります。`kernel-source` パッケージをインストールしていれば、`/usr/src/linux/Documentation/cpu-freq/*` 内に情報が用意されています。

A ネットワーク例

改訂履歴

2022-10-10

このネットワーク構成は、openSUSE® Leap のドキュメンテーション内でネットワーク関連の説明を行う際に使用する構成の例です。



B GNU ライセンス

本付録には、GNU Free Documentation License バージョン 1.2 とその日本語訳 (八田真行氏 (mhatta@gnu.org) による翻訳) を収録しています。

GNU Free Documentation License

Copyright (C) 2000, 2001, 2002 Free Software Foundation, Inc. 51 Franklin St, Fifth Floor, Boston, MA 02110-1301 USA. Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.

0. PREAMBLE

The purpose of this License is to make a manual, textbook, or other functional and useful document "free" in the sense of freedom: to assure everyone the effective freedom to copy and redistribute it, with or without modifying it, either commercially or non-commercially. Secondarily, this License preserves for the author and publisher a way to get credit for their work, while not being considered responsible for modifications made by others.

This License is a kind of "copyleft", which means that derivative works of the document must themselves be free in the same sense. It complements the GNU General Public License, which is a copyleft license designed for free software.

We have designed this License to use it for manuals for free software, because free software needs free documentation: a free program should come with manuals providing the same freedoms that the software does. But this License is not limited to software manuals; it can be used for any textual work, regardless of subject matter or whether it is published as a printed book. We recommend this License principally for works whose purpose is instruction or reference.

1. APPLICABILITY AND DEFINITIONS

This License applies to any manual or other work, in any medium, that contains a notice placed by the copyright holder saying it can be distributed under the terms of this License. Such a notice grants a world-wide, royalty-free license, unlimited in duration, to use that work under the conditions stated herein. The "Document", below, refers to any such manual or work. Any member of the public is a licensee, and is addressed as "you". You accept the license if you copy, modify or distribute the work in a way requiring permission under copyright law.

A "Modified Version" of the Document means any work containing the Document or a portion of it, either copied verbatim, or with modifications and/or translated into another language.

A "Secondary Section" is a named appendix or a front-matter section of the Document that deals exclusively with the relationship of the publishers or authors of the Document to the Document's overall subject (or to related matters) and contains nothing that could fall directly within that overall subject. (Thus, if the Document is in part a textbook of mathematics, a Secondary Section may not explain any mathematics.) The relationship could be a matter of historical connection with the subject or with related matters, or of legal, commercial, philosophical, ethical or political position regarding them.

The "Invariant Sections" are certain Secondary Sections whose titles are designated, as being those of Invariant Sections, in the notice that says that the Document is released under this License. If a section does not fit the above definition of Secondary then it is not allowed to be designated as Invariant. The Document may contain zero Invariant Sections. If the Document does not identify any Invariant Sections then there are none.

The "Cover Texts" are certain short passages of text that are listed, as Front-Cover Texts or Back-Cover Texts, in the notice that says that the Document is released under this License. A Front-Cover Text may be at most 5 words, and a Back-Cover Text may be at most 25 words.

A "Transparent" copy of the Document means a machine-readable copy, represented in a format whose specification is available to the general public, that is suitable for revising the document straightforwardly with generic text editors or (for images composed of pixels) generic paint programs or (for drawings) some widely available drawing editor, and that is suitable for input to text formatters or for automatic translation to a variety of formats suitable for input to text formatters. A copy made in an otherwise Transparent file format whose markup, or absence of markup, has been arranged to thwart or discourage subsequent modification by readers is not Transparent. An image format is not Transparent if used for any substantial amount of text. A copy that is not "Transparent" is called "Opaque".

Examples of suitable formats for Transparent copies include plain ASCII without markup, Texinfo input format, LaTeX input format, SGML or XML using a publicly available DTD, and standard-conforming simple HTML, PostScript or PDF designed for human modification. Examples of transparent image formats include PNG, XCF and JPG. Opaque formats include proprietary formats that can be read and edited only by proprietary word processors, SGML or XML for which the DTD and/or processing tools are not generally available, and the machine-generated HTML, PostScript or PDF produced by some word processors for output purposes only.

The "Title Page" means, for a printed book, the title page itself, plus such following pages as are needed to hold, legibly, the material this License requires to appear in the title page. For works in formats which do not have any title page as such, "Title Page" means the text near the most prominent appearance of the work's title, preceding the beginning of the body of the text.

A section "Entitled XYZ" means a named subunit of the Document whose title either is precisely XYZ or contains XYZ in parentheses following text that translates XYZ in another language. (Here XYZ stands for a specific section name mentioned below, such as "Acknowledgements", "Dedications", "Endorsements", or "History".) To "Preserve the Title" of such a section when you modify the Document means that it remains a section "Entitled XYZ" according to this definition.

The Document may include Warranty Disclaimers next to the notice which states that this License applies to the Document. These Warranty Disclaimers are considered to be included by reference in this License, but only as regards disclaiming warranties: any other implication that these Warranty Disclaimers may have is void and has no effect on the meaning of this License.

2. VERBATIM COPYING

You may copy and distribute the Document in any medium, either commercially or non-commercially, provided that this License, the copyright notices, and the license notice saying this License applies to the Document are reproduced in all copies, and that you add no other conditions whatsoever to those of this License. You may not use technical measures to obstruct or control the reading or further copying of the copies you make or distribute. However, you may accept compensation in exchange for copies. If you distribute a large enough number of copies you must also follow the conditions in section 3.

You may also lend copies, under the same conditions stated above, and you may publicly display copies.

3. COPYING IN QUANTITY

If you publish printed copies (or copies in media that commonly have printed covers) of the Document, numbering more than 100, and the Document's license notice requires Cover Texts, you must enclose the copies in covers that carry, clearly and legibly, all these Cover Texts: Front-Cover Texts on the front cover, and Back-Cover Texts on the back cover. Both covers must also clearly and legibly identify you as the publisher of these copies. The front cover must present the full title with all words of the title equally prominent and visible. You may add other material on the covers in addition. Copying with changes limited to the covers, as long as they preserve the title of the Document and satisfy these conditions, can be treated as verbatim copying in other respects.

If the required texts for either cover are too voluminous to fit legibly, you should put the first ones listed (as many as fit reasonably) on the actual cover, and continue the rest onto adjacent pages.

If you publish or distribute Opaque copies of the Document numbering more than 100, you must either include a machine-readable Transparent copy along with each Opaque copy, or state in or with each Opaque copy a computer-network location from which the general network-using public has access to download using public-standard network protocols a complete Transparent copy of the Document, free of added material. If you use the latter option, you must take reasonably prudent steps, when you begin distribution of Opaque copies in quantity, to ensure that this Transparent copy will remain thus accessible at the stated location until at least one year after the last time you distribute an Opaque copy (directly or through your agents or retailers) of that edition to the public.

It is requested, but not required, that you contact the authors of the Document well before redistributing any large number of copies, to give them a chance to provide you with an updated version of the Document.

4. MODIFICATIONS

You may copy and distribute a Modified Version of the Document under the conditions of sections 2 and 3 above, provided that you release the Modified Version under precisely this License, with the Modified Version filling the role of the Document, thus licensing distribution and modification of the Modified Version to whoever possesses a copy of it. In addition, you must do these things in the Modified Version:

A. Use in the Title Page (and on the covers, if any) a title distinct from that of the Document, and from those of previous versions (which should, if there were any, be listed in the History section of the Document). You may use the same title as a previous version if the original publisher of that version gives permission.

B. List on the Title Page, as authors, one or more persons or entities responsible for authorship of the modifications in the Modified Version, together with at least five of the principal authors of the Document (all of its principal authors, if it has fewer than five), unless they release you from this requirement.

C. State on the Title page the name of the publisher of the Modified Version, as the publisher.

D. Preserve all the copyright notices of the Document.

E. Add an appropriate copyright notice for your modifications adjacent to the other copyright notices.

F. Include, immediately after the copyright notices, a license notice giving the public permission to use the Modified Version under the terms of this License, in the form shown in the Addendum below.

G. Preserve in that license notice the full lists of Invariant Sections and required Cover Texts given in the Document's license notice.

H. Include an unaltered copy of this License.

I. Preserve the section Entitled "History", Preserve its Title, and add to it an item stating at least the title, year, new authors, and publisher of the Modified Version as given on the Title Page. If there is no section Entitled "History" in the Document, create one stating the title, year, authors, and publisher of the Document as given on its Title Page, then add an item describing the Modified Version as stated in the previous sentence.

J. Preserve the network location, if any, given in the Document for public access to a Transparent copy of the Document, and likewise the network locations given in the Document for previous versions it was based on. These may be placed in the "History" section. You may omit a network location for a work that was published at least four years before the Document itself, or if the original publisher of the version it refers to gives permission.

K. For any section Entitled "Acknowledgements" or "Dedications", Preserve the Title of the section, and preserve in the section all the substance and tone of each of the contributor acknowledgements and/or dedications given therein.

L. Preserve all the Invariant Sections of the Document, unaltered in their text and in their titles. Section numbers or the equivalent are not considered part of the section titles.

M. Delete any section Entitled "Endorsements". Such a section may not be included in the Modified Version.

N. Do not retitle any existing section to be Entitled "Endorsements" or to conflict in title with any Invariant Section.

O. Preserve any Warranty Disclaimers.

If the Modified Version includes new front-matter sections or appendices that qualify as Secondary Sections and contain no material copied from the Document, you may at your option designate some or all of these sections as invariant. To do this, add their titles to the list of Invariant Sections in the Modified Version's license notice. These titles must be distinct from any other section titles.

You may add a section Entitled "Endorsements", provided it contains nothing but endorsements of your Modified Version by various parties--for example, statements of peer review or that the text has been approved by an organization as the authoritative definition of a standard.

You may add a passage of up to five words as a Front-Cover Text, and a passage of up to 25 words as a Back-Cover Text, to the end of the list of Cover Texts in the Modified Version. Only one passage of Front-Cover Text and one of Back-Cover Text may be added by (or through arrangements made by) any one entity. If the Document already includes a cover text for the same cover, previously added by you or by arrangement made by the same entity you are acting on behalf of, you may not add another; but you may replace the old one, on explicit permission from the previous publisher that added the old one.

The author(s) and publisher(s) of the Document do not by this License give permission to use their names for publicity for or to assert or imply endorsement of any Modified Version.

5. COMBINING DOCUMENTS

You may combine the Document with other documents released under this License, under the terms defined in section 4 above for modified versions, provided that you include in the combination all of the Invariant Sections of all of the original documents, unmodified, and list them all as Invariant Sections of your combined work in its license notice, and that you preserve all their Warranty Disclaimers.

The combined work need only contain one copy of this License, and multiple identical Invariant Sections may be replaced with a single copy. If there are multiple Invariant Sections with the same name but different contents, make the title of each such section unique by adding at the end of it, in parentheses, the name of the original author or publisher of that section if known, or else a unique number. Make the same adjustment to the section titles in the list of Invariant Sections in the license notice of the combined work.

In the combination, you must combine any sections Entitled "History" in the various original documents, forming one section Entitled "History"; likewise combine any sections Entitled "Acknowledgements", and any sections Entitled "Dedications". You must delete all sections Entitled "Endorsements".

6. COLLECTIONS OF DOCUMENTS

You may make a collection consisting of the Document and other documents released under this License, and replace the individual copies of this License in the various documents with a single copy that is included in the collection, provided that you follow the rules of this License for verbatim copying of each of the documents in all other respects.

You may extract a single document from such a collection, and distribute it individually under this License, provided you insert a copy of this License into the extracted document, and follow this License in all other respects regarding verbatim copying of that document.

7. AGGREGATION WITH INDEPENDENT WORKS

A compilation of the Document or its derivatives with other separate and independent documents or works, in or on a volume of a storage or distribution medium, is called an "aggregate" if the copyright resulting from the compilation is not used to limit the legal rights of the compilation's users beyond what the individual works permit. When the Document is included in an aggregate, this License does not apply to the other works in the aggregate which are not themselves derivative works of the Document.

If the Cover Text requirement of section 3 is applicable to these copies of the Document, then if the Document is less than one half of the entire aggregate, the Document's Cover Texts may be placed on covers that bracket the Document within the aggregate, or the electronic equivalent of covers if the Document is in electronic form. Otherwise they must appear on printed covers that bracket the whole aggregate.

8. TRANSLATION

Translation is considered a kind of modification, so you may distribute translations of the Document under the terms of section 4. Replacing Invariant Sections with translations requires special permission from their copyright holders, but you may include translations of some or all Invariant Sections in addition to the original versions of these Invariant Sections. You may include a translation of this License, and all the license notices in the Document, and any Warranty Disclaimers, provided that you also include the original English version of this License and the original versions of those notices and disclaimers. In case of a disagreement between the translation and the original version of this License or a notice or disclaimer, the original version will prevail.

If a section in the Document is Entitled "Acknowledgements", "Dedications", or "History", the requirement (section 4) to Preserve its Title (section 1) will typically require changing the actual title.

9. TERMINATION

You may not copy, modify, sublicense, or distribute the Document except as expressly provided for under this License. Any other attempt to copy, modify, sublicense or distribute the Document is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

10. FUTURE REVISIONS OF THIS LICENSE

The Free Software Foundation may publish new, revised versions of the GNU Free Documentation License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns. See https://www.gnu.org/copyleft/.

Each version of the License is given a distinguishing version number. If the Document specifies that a particular numbered version of this License "or any later version" applies to it, you have the option of following the terms and conditions either of that specified version or of any later version that has been published (not as a draft) by the Free Software Foundation. If the Document does not specify a version number of this License, you may choose any version ever published (not as a draft) by the Free Software Foundation.

ADDENDUM: How to use this License for your documents

```
Copyright (c) YEAR YOUR NAME.
Permission is granted to copy, distribute and/or modify this document
under the terms of the GNU Free Documentation License, Version 1.2
or any later version published by the Free Software Foundation;
with no Invariant Sections, no Front-Cover Texts, and no Back-Cover Texts.
A copy of the license is included in the section entitled “GNU
Free Documentation License”.
```

If you have Invariant Sections, Front-Cover Texts and Back-Cover Texts, replace the “with...Texts.” line with this:

```
with the Invariant Sections being LIST THEIR TITLES, with the
Front-Cover Texts being LIST, and with the Back-Cover Texts being LIST.
```

If you have Invariant Sections without Cover Texts, or some other combination of the three, merge those two alternatives to suit the situation.

If your document contains nontrivial examples of program code, we recommend releasing these examples in parallel under your choice of free software license, such as the GNU General Public License, to permit their use in free software.

GNU フリー文書利用許諾契約書

Copyright (C) 2000, 2001, 2002 Free Software Foundation, Inc. 51 Franklin St, Fifth Floor, Boston, MA 02110-1301 USA. この利用許諾契約書を、一字一句そのままに複製し頒布することは許可する。しかし変更は認めない。

0. はじめに

この利用許諾契約書の目的は、この契約書が適用されるマニュアルや教科書、その他機能本位で実用的な文書を(無料ではなく)自由という意味で「フリー」とすること、すなわち、改変の有無あるいは目的の営利非営利を問わず、文書を複製し再頒布する自由をすべての人々に効果的に保証することです。加えてこの契約書により、著者や出版者が自分たちの著作物に対して相応の敬意と賞賛を得る手段も保護されます。また、他人が行った改変に対して責任を負わずに済むようになります。

この利用許諾契約書は「コピーレフト」的なライセンスの一つであり、この契約書が適用された文書から派生した著作物は、それ自身もまた原本と同じ意味でフリーでなければなりません。この契約書は、フリーソフトウェアのために設計されたコピーレフトなライセンスであるGNU一般公衆使用許諾契約書を補足するものです。

この利用許諾契約書は、フリーソフトウェア用のマニュアルに適用することを目的として書かれました。フリーソフトウェアはフリーな文書を必要としており、フリーなプログラムはそのソフトウェアが保証するのと同じ自由を提供するマニュアルと共に頒布されるべきだからです。しかし、この契約書の適用範囲はソフトウェアのマニュアルに留まりません。対象となる著作物において扱われる主題が何であれ、あるいはそれが印刷された書籍として出版されるか否かに関わらず、この契約書は文字で書かれたいかなる著作物にも適用することが可能です。私たちとしては、主にこの契約書を解説や参照を目的とする著作物に適用することをお勧めします。

1. この利用許諾契約書の適用範囲と用語の定義

著作物がこの利用許諾契約書の定める条件の下で頒布される旨の告知を、著作権者がその中に書いたすべてのマニュアルあるいはその他の著作物は、いかなる媒体上にあってもこの契約書の適用対象となる。そのような告知を置くことで、全世界において、著作権使用料を必要とせず、許可の存続期間を限定されること無く、この契約書の中で述べられている条件の下で当該著作物を利用できるという許可を与えることとする。以下において、「『文書』(Document)」とはそのような告知が記載されたマニュアルないし著作物すべてを指す。公衆の一員ならば誰でも契約の当事者となることができ、この契約書中では「あなた」と表現される。あなたは、著作権法の下で許可を必要とするような方法で著作物を複製や改変、あるいは頒布することにより、この契約書を受諾することになる。

『文書』の「改変版 (Modified Version)」とは、一字一句忠実に複製したか、あるいは改変や他言語への翻訳を行ったかどうかに関わらず、その『文書』の全体あるいは一部分を含む著作物すべてを意味する。

「補遺部分 (Secondary Section)」とは、『文書』中でその旨指定された補遺ないし本文に先だって前付けとして置かれる一部分であり、『文書』の出版者あるいは著者と、『文書』全体の主題(あるいはそれに関連する事柄)との関係のみを論じ、全体としての主題の範疇に直接属する内容を全く含まないものである(たとえば、『文書』の一部が数学の教科書だった場合、補遺部分では数学について何も解説してはならない)。補遺部分で扱われる関係は、その主題あるいは関連する事柄との歴史的なつながりのことかも知れないし、それらに関する法的、商業的、哲学的、倫理的、あるいは政治的立場についてかも知れない。

「変更不可部分 (Invariant Sections)」とは補遺部分の一種で、それらが変更不可部分であることが、『文書』をこの利用許諾契約書の下で発表する旨述べた告知中においてその部分の題名と共に明示されているものである。ある部分が上記のような「補遺」性の定義にそぐわない場合は、その部分を「変更不可」として指定することは認められない。『文書』は、変更不可部分を全く含まなくても良い。『文書』において変更不可部分が全く指定されていなければ、その『文書』に変更不可部分は存在しないということである。

「カバーテキスト(Cover Texts)」とは、『文書』がこの利用許諾契約書の指定する条件の下で発表される旨述べた告知において、「表カバーテキスト」あるいは「裏カバーテキスト」として列挙された短い文章のことを指す。表カバーテキストは最大で5語、裏カバーテキストは最大で25語までとする。

『文書』の「透過的」複製物とは、機械による読み取りが可能な『文書』の複製物のことを指す。透過的な複製物の文書形式は、その仕様が一般の人々に入手可能で、『文書』の内容を一般的なテキストエディタ、または(画素で構成される画像ならば)一般的なペイントプログラム、あるいは(図面ならば)いくつかの広く入手可能な製図エディタで簡単に改訂するのに適しており、なおかつテキストフォーマッタへの入力に適する(あるいはテキストフォーマッタへの入力に適する諸形式への自動的な変換に適する)ものでなければならない。透過的なファイル形式への複製であっても、マークアップ、あるいはマークアップの不在が読者によるそれ以降の改変をわざと邪魔し阻害するように仕組まれたものは透過的であるとは見做されない。ある画像形式が、相当量のテキスト文章を表現するために使われた場合、それは透過的ではない。透過的ではない複製は「非透過的」複製と呼ばれる。

透過的複製に適した形式の例としては、マークアップを含まないプレーンなASCII形式、Texinfo入力形式、LaTeX入力形式、一般に入手可能なDTDを用いたSGMLあるいはXML、または人間による改変を想定して設計された、標準に準拠したシンプルなHTMLやPostScript、PDFなどが挙げられる。透過的な画像形式の例には、PNGやXCF、JPGが含まれる。非透過な形式としては、独占的なワードプロセッサでのみ閲覧編集できる独占的なファイル形式、普通には入手できないDTDまたは処理系を使ったSGMLやXML、ある種のワードプロセッサが生成する、出力のみを目的とした機械生成のHTMLやPostScript、PDFなどが含まれる。

「題扉 (Title Page)」とは、印刷された書籍に於いては、実際の表紙自身のみならず、この利用許諾契約書が表紙に掲載することを義務づける文章や図などを、読みやすい形で載せるのに必要なだけの、表紙に引き続く数ページをも意味する。表紙に類するものが無い形式で発表される著作物においては、「題扉」とは本文の始まりに先だって、その著作物の題名が最も目立つ形で現れる場所の近くに置かれる文章のことを指す。

「XYZと題された (Entitled XYZ)」部分とは、『文書』において「XYZ」と名付けられた一部分であり、その題名は正確に「XYZ」であるか、「XYZ」を他の言語に翻訳した上でその後ろに「XYZ」をそのまま括弧で括ったものを含む記述のどちらかである(ここでの「XYZ」とは、この利用許諾契約書において以下で言 及される特定の部分名を意味している。例えば「謝辞 (Acknowledgements)」、「献辞 (Dedications)」、「推薦の辞 (Endorsements)」、「履歴 (History)」)。あなたが『文書』を改変する場合、そのような部分の「題名を保存する (Preserve the Title)」とは、「XYZと題された」部分として、ここでの定義に従い題名を残すということである。

『文書』は、「保証否認警告 (Warranty Disclaimers)」を、この利用許諾契約書が『文書』に適用されると述べた告知の次に含んでも良い。この種の保証否認警告は、この契約書からの言及という形で利用条件に含まれるものと解されるが、保証の否認に関することについてのみ有効とする。こういった保証否認警告で示しうるその他のいかなる含意も無効であり、この契約書の効能には何ら影響を持たない。

2. 逐語的に忠実な複製

この利用許諾契約書、著作権表示、この契約書が『文書』に適用される旨述べた告知の三つがすべての複製物に複製され、かつあなたがこの契約書で指定されている以外のいかなる条件も追加しない限り、あなたはこの『文書』を、商用であるか否かを問わずいかなる形でも複製頒布することができる。あなたは、あなたが作成あるいは頒布する複製物に対して、閲覧や再複製を技術的な手法によって妨害、規制してはならない。しかしながら、複製と引き換えに代価を得てもかまわない。あなたが相当量の複製物を頒布する際には、本契約書第3項で指定される条件にも従わなければならない。

またあなたは、上記と同じ条件の下で、複製物を貸与したり複製物を公に開示することができる。

3. 大量の複製

もしあなたが、『文書』の印刷された(あるいは通常は印刷された表紙を持つ媒体における)複製物を100部を超えて出版し、また『文書』の利用許諾告知がカバーテキストの掲載を要求している場合には、指定されたすべてのカバーテキストを、表カバーテキストは表表紙に、裏カバーテキストは裏表紙に、はっきりと読みやすい形で載せた表紙の中に複製物本体を綴じ込まなければならない。また、両方の表紙において、それらの複製物の出版者としてのあなたをはっきりとかつ読みやすい形で確認できなければならない。表表紙では『文書』の完全な題名を、題名を構成するすべての語が等しく目立つようにして、視認可能な形で示さなければならない。それらの情報に加えて、表紙に他の文章や図などを加えることは許可される。表紙のみを変更した複製物は、それが『文書』の題名を保存し上記の条件を満たす限り、ほかの点では逐語的に忠実な複製物 として扱われる。

もしどちらかの表紙に要求されるカバーテキストの量が多すぎて読みやすく収めることが不可能ならば、あなたはテキスト先頭の一文(あるいは適切に収まるだけ)を実際の表紙に載せ、続きは隣接したページに載せるべきである。

あなたが『文書』の「非透過的」複製物を100部を超えて出版あるいは頒布する場合、それぞれの非透過な複製物と一緒に機械で読み取り可能な透過的複製物を添付するか、それぞれの非透過な複製物(あるいはそれに付属する文書)中で、公にアクセス可能なコンピュータネットワーク上の所在地を記述しなければならない。その場所には、非透過な複製物と内容的に寸分違わず、余計なものが追加されていない完全な『文書』の透過的複製物が置かれ、またそこから、ネットワークを利用する一般公衆が、一般に標準的と考えられるネットワークプロトコルを使ってダウンロードすることができなければならない。もしあなたが後者の選択肢を選ぶならば、その版の非透過な複製物を公衆に(直接、あるいはあなたの代理人ないし小売業者が)最後に頒布してから最低1年間は、その透過的複製物が指定の場所でアクセス可能であり続けることを保証するよう、非透過な複製物の大量頒布を始める際に十分に慎重な手順を踏まなければならない。

これは要望であり必要条件ではないが、『文書』の著者に、『文書』の更新された版をあなたに提供する機会を与えるため、透過非透過を問わず大量の複製物を再頒布し始める前には彼らにきちんと連絡しておいてほしい。

4. 改変

『文書』の改変版を、この利用許諾契約書と細部まで同一の契約の下で発表する限り、すなわち原本の役割を改変版で置き換えた形での頒布と改変を、その複製物を所有するすべての人々に許可する限り、あなたは改変版を上記第2項および第3項が指定する条件の下で複製および頒布することができる。さらに、あなたは改変版において以下のことを行わなければならない。

A. 題扉に(もしあればその他の表紙にも)、『文書』お よび『文書』のそれ以前の版と見分けがつく題名を載せること(もし以前の 版があれば、『文書』の「履歴 (History)」の部分に列記されているはずである)。もし元の版の出版者から許可を得たならば、以前の版と同じ題名を 使っても良い。

B. 題扉に、改変版における改変を行った1人以上の人物か団体名を列記すること。あわせて元の『文書』の著者として、最低5人(もし5人以下ならばすべて)の主要著者を列記すること。ただし元の著者たちがこの条件を免除した場合は除く。

C. 題扉に、改変版の出版者名を出版者として記載すること。

D. 『文書』にあるすべての著作権表示を残すこと。

E. 他の著作権表示の近くに、あなたの改変に対する適当な著作権表示を追加すること。

F. 著作権表示のすぐ後に、改変版をこの契約書の条件の下で利用することを公衆に対して許可する告知を含めること。その形式はこの契約書の末尾にある付記で示されている。

G. 元の『文書』の利用許諾告知に書かれた、変更不可部分の完全な一覧と、要求されるカバーテキストとを、改変版の利用許諾告 知でもそのまま残すこと。

H. この契約書の、変更されていない複製物を含めること。

I. 「履歴 (History)」と題された部分とその題名を保存し、そこに改変版の、少なくとも題名、出版年、新しく変更した部分の著 者名、出版者名を、題扉に掲載するのと同じように記載した一項を加えること。もし『文書』中に「履歴」と題された部分が存在しない場合には、『文書』の題名、出版年、著者、出版者を題扉に掲載するのと同じように記載した部分を用意し、上記で述べたような、改変版を説明する一項を加えること。

J. 『文書』中に、『文書』の透過的複製物への公共的アクセスのために指定されたネットワーク的所在地が記載されていたならば、それを保存すること。同様に、その『文書』の元になった以前の版で指定されていたネットワーク的所在地も載っていたならば、それも保存すること。これらの情報は「履歴(History)」の部分に置いても良い。ただし、それが『文書』自身より少なくとも4年前に出版された著作物の情報であったり、あるいは改変版が参考にしている版の元々の出版者から許可を得たならば、その情報を削除してもかまわない。

K. 「謝辞 (Acknowledgement)」あるいは「献辞 (Dedication)」等と題されたいかなる部分も、その部分の題名を保存し、その部分の内容(各貢献者への謝意あるいは献呈の意)と語調を保存すること。

L. 『文書』の変更不可部分を、その本文および題名を変更せずに保存すること。章番号やそれに相当するものは部分の題名の一部とは見做さない。

M. 「推薦の辞 (Endorsement)」というような章名が題された部分はすべて削除すること。そのような部分を改変版に含めてはならない。

N. すでに存在する部分を「推薦の辞 (Endorsement)」と題されるように改名したり、題名の点で変更不可部分のどれかと衝突するように改名してはならない。

O. 保証否認警告を保存すること。

もし改変版に、補遺部分としての条件を満たし、かつ『文書』から複製物された文章や図などをいっさい含んでいない、前書き的な章あるいは付録が新しく含まれるならば、あなたは希望によりそれらの部分の一部あるいはすべてを変更不可と宣言することができる。変更不可を宣言するためには、それらの部分の題名を改変版の利用許諾告知中の変更不可部分一覧に追加すれば良い。これらの題名は他の章名とは全く別のものでなければならない。

含まれる内容が、さまざまな集団によるあなたの改変版に対する推薦の辞のみである限り、あなたは、「推薦の辞 (Endorsement)」と題された章を追加することができる。推薦の辞の例としては、ピアレビューの陳述、あるいは文書がある標準の権威ある定義としてその団体に承認されたという声明などがある。

あなたは、5語までの一文を表カバーテキストとして、25語までの文を裏表紙テキストとして、改変版のカバーテキスト一覧の末尾に加えることができる。一個人ないし一団体が直接(あるいは団体内で結ばれた協定によって)加えることができるのは、表カバーテキストおよび裏カバーテキストとしてそれぞれ一文ずつのみである。もし以前すでにその文書において、表裏いずれかの表紙にあなたの(またはあなたが代表する同じ団体内で為された協定に基づく)カバーテキストが含まれていたならば、あなたが新たに追加することはできない。し かしあなたは、その古い文を加えた以前の出版者から明示的な許可を得たならば、古い文を置き換えることができる。

『文書』の著者あるいは出版者は、この利用許諾契約書によって、彼らの名前を利用することを許可しているわけではない。彼らの名前を改変版の宣伝に使ったり、改変版への明示的あるいは黙示的な保証のために使うことを許可するものではない。

5. 文書の結合

あなたは、上記第4項において改変版に関して定義された条件の下で、この利用許諾契約書の下で発表された複数の文書を一つにまとめることができる。その際、原本となる文書にある変更不可部分を全て、改変せずに結合後の著作物中に含め、それらをあなたが統合した著作物の変更不可部分としてその利用許諾告知において列記し、かつ原本にある全ての保証否認警告を保存しなければならない。

結合後の著作物についてはこの契約書の複製物を一つ含んでいればよく、同一内容の変更不可部分が複数ある場合には一つで代用してよい。もし同じ題名だが内容の異なる変更不可部分が複数あるならば、そのような部分のそれぞれの題名の最後に、(もし分かっているならば)その部分の原著者あるいは出版者の名前で、あるいは他と重ならないような番号を括弧で括って記載することで、それぞれ見分けが付くようにしなければならない。結合後の著作物の利用許諾告知における変更不可部分の一覧においても、章の題名に同様の調整をすること。

結合後の著作物においては、あなたはそれぞれの原本の「履歴 (History)」と 題されたあらゆる部分をまとめて、「履歴 (History)」と題された一章にしなければならない。同様に、「謝辞 (Acknowledgements)」あるいは「献辞 (Dedications)」と題されたあらゆる部分もまとめなければならない。あなたは「推薦の辞 (Endowsements)」と題されたあらゆる部分も削除しなければならない。

6. 文書の収集

あなたは、この利用許諾契約書の下で発表された複数の文書で構成される収集著作物を作ることができる。その場合、それぞれの文書が逐語的に忠実に複製されることを保障するために他のすべての点でこの契約書の定める条件に従う限り、さまざまな文書中のこの契約書の個々の複製物を、収集著作物中に複製物を一つ含めることで代用することができる。

あなたは、このような収集著作物から文書を一つ取り出し、それをこの契約書の下で頒布することができる。ただしその際には、この契約書の複製物を抽出された文書に挿入し、またその他すべての点でこの文書の逐語的に忠実な複製に関してこの契約書が定める条件に従わなければならない。

7. 独立した著作物の集積

『文書』あるいはその派生物を、他の別の独立した文書あるいは著作物と一緒にし、一巻の記憶装置あるいは頒布媒体に収めた編集著作物は、編集に起因する著作権が編集著作物に含まれる個々の著作物がその利用者に許可した法的権利を制限するよう行使されない限り、「集積」著作物と呼ばれる。『文書』が集積著作物に含まれる場合、この契約書は、『文書』と共にまとめられた他の独立した著作物には、それら自身が『文書』の派生物で無い限り適用されることにはならない。

このような『文書』の複製物において、この利用許諾契約書の第3項によりカバーテキストの掲載が要求されている場合、『文書』の量が集積著作物全体の2分の1以下であれば、『文書』のカバーテキストは集積著作物中で『文書』そのものの周りを囲む中表紙、あるいは『文書』が電子的形式である場合には表紙の電子的等価物にのみ配置するだけでよい。その場合以外は、カバーテキストは集積著作物全体を取り巻く印刷された表紙に掲載されなければならない。

8. 翻訳

翻訳は改変の一種と見做すので、あなたは『文書』の翻訳をこの利用許諾契約書の第4項の定める条件の下で頒布することができる。変更不可部分を翻訳によって置き換えるには著作権者の特別許可を必要とするが、元の変更不可部分に追加する形で変更不可部分の全てないし一部の翻訳を含めることはかまわない。この契約書や『文書』中の利用許諾告知、保証否認警告すべての英語原本も含める限り、あなたはこの契約書、告知、警告の翻訳を含めることができる。契約書や告知、警告に関して翻訳と英語原本との間に食い違いが生じた場合、英語原本が優先される。

典型的な例として、『文書』のある部分が原文で「Acknowledgements」、「Dedications」、あるいは「History」と題されていた場合、実際の題名を変更するには、題名を保存する(この契約書の第1項)ための条件(同第4項)を満たすことが必要となる。

9. 契約の終了

この利用許諾契約書の下で明確に提示されている場合を除き、あなたは『文書』を複製、改変、サブライセンス、あるいは頒布してはならない。このライセンスで指定されている以外の、『文書』の複製、改変、サブライセンス、頒布に関するすべての企ては無効であり、この契約書によって保証されるあなたの権利を自動的に終結させることとなる。しかし、この契約書の下であなたから複製物ないし諸権利を得た個人や団体に関しては、そういった人々がこの契約書に完全に従ったままである限り、彼らに与えられた許諾は終結しない。

10. 将来における本利用許諾契約書の改訂

フリーソフトウェア財団は、時によってGNUフリー文書利用許諾契約書の新しい改訂版を出版することができる。そのような新版は現在の版と理念においては似たものになるであろうが、新たに生じた問題や懸念を解決するため細部においては違ったものになるだろう。詳しくは https://www.gnu.org/copyleft/ を参照せよ。

GNUフリー文書利用許諾契約書のそれぞれの版には、新旧の区別が付くようなバージョン番号が振られている。もし『文書』において、この契約書のある特定の版か「それ以降のどの版でも」適用して良いと指定されている場合、あなたはフリーソフトウェア財団から発行された(草稿として発表されたものを除く)指定の版かそれ以降の版のうちどれか一つを選び、その条項や条件に従うことができる。もし『文書』がこの契約書のバージョン番号を指定していない場合には、あなたはフリーソフトウェア財団から今までに出版された(草稿として発表されたものを除く)版のうちからどれか一つを選ぶことができる。

ADDENDUM: この利用許諾契約書をあなたの文書に適用するには

この利用許諾契約書をあなたが書いた文書に適用するには、この契約書の複製物一つを文書中に含め、以下に示す著作権表示と利用許諾告知を題扉のすぐ後に置いて下さい:

```
Copyright (c) YEAR YOUR NAME.
Permission is granted to copy, distribute and/or modify this document
under the terms of the GNU Free Documentation License, Version 1.2
or any later version published by the Free Software Foundation;
with no Invariant Sections, no Front-Cover Texts, and no Back-Cover Texts.
A copy of the license is included in the section entitled “GNU
Free Documentation License”.
```

(訳: Copyright (C) 西暦年 あなたの名前. この文書を、フリーソフトウェア財団発行のGNU フリー文書利用許諾契約書(バージョン1.2かそれ以降から一つを選択)が定める条件の下で複製、頒布、あるいは改変することを許可する。変更不可部分、表カバーテキスト、裏カバーテキストは存在しない。この利用許諾契約書の複製物は「GNU フリー文書利用許諾契約書」という章に含まれている。)

もし変更不可部分や表カバーテキスト、裏カバーテキストがあれば、「変更不可部分…は存在しない。」というところを以下で置き換えてください:

```
with the Invariant Sections being LIST THEIR TITLES, with the
Front-Cover Texts being LIST, and with the Back-Cover Texts being LIST.
```

(訳: (章の題名を列記)は変更不可部分であり、(表カバーテキストを列記)は表カバーテキスト、(裏カバーテキストを列記)は裏カバーテキストである。)

変更不可部分はあるがカバーテキストは存在しないなど、その他の三者の組み合せに関しては、状況に合わせて上記二つの選択肢を混ぜてください。

あなたの文書に、他に類を見ない独自のプログラムコードのサンプルが含まれる場合、フリーソフトウェアにおいてそのコードを利用することを許可するために、そういったサンプルに関してはこの利用許諾契約書と同時にGNU一般公衆許諾契約書のようなフリーソフトウェア向けライセンスのうちどれか一つを選択して適用してもよい、というような条件の下で発表することを推奨します。