5. Linux の概念や基本操作に関する知識 II

1. 科目の概要

Linux の高度な活用法として、ファイルシステムの概念と操作、データのバックアップとリストア、シェルスクリプトによる操作、C 言語によるプログラミング、ネットワーキングといった様々な利用方法を解説する。

2. 習得ポイント

本科目の学習により習得することが期待されるポイントは以下の通り。

習得ポイント	説 明	シラバスの対応コマ
II-5-1. ディレクトリ構成の実際	一時的に利用するファイルを置く/tmp、設定ファイル等が格納される/etc、ユーザのデータとして取り扱われるファイルを置く/home、プログラムが配置される/binなど、UNIXやLinuxで一般的なディレクトリ構成を紹介する。	9,10
II-5-2. パーティションとマウント	ディスクのパーティショニングという概念を示し、物理パーティション、論理ディスク(LVM)などの要素技術を紹介する。またそれぞれのパーティションをマウントして利用する考え方を説明し、マウント/アンマウントの手順を示す。	9,10
II-5-3. ファイルシステムの構築	ファイルシステムとは何か説明し、Linuxで取り扱うことができる代表的なファイルシステムの種類を挙げる。またファイルシステムを具体的に構築するコマンドを示し、ファイルシステム構築の手順について述べる。	9,10
II-5-4. データのバックアップ方法	バックアップの必要性、バックアップの範囲、バックアップ対象のメディア等、データのバックアップに必要な知識を説明する。またデータのバックアップに利用するツールを紹介し、バックアップの方法と具体的な手順を示す。	11,12
II-5-5. バックアップしたデータのリストア	データのリストアが必要になるタイミングやそれぞれのバックアップ方法に対応したリストアの手順について解説する。またデータのリストア時に留意すべきポイントを示し、具体的なリストア方法を説明する。	11,12
II-5-6. シェルスクリプトの基本	シェルの役割を再認識し、シェル変数や環境変数といったシェルの高度な設定方法を理解させる。またシェルスクリプトの基本的な機能を紹介する。具体的には、変数の扱いや基本的な制御構造について解説する。	13,14
II-5-7. Linuxシステムで利用されるシェルスクリプト	Linuxシステムでは様々なシェルスクリプトが活用されている。各アプリケーションやサービスの起動スクリプト、設定に利用されるシェルスクリプトなど代表的なスクリプトを示し、実際の動作がどのように行われるかについて説明する。	13,14
II-5-8. C言語による開発	Linuxで活用される多くのプログラムはC言語によって開発されている。ここではC言語によるプログラミングの基本的な開発について触れ、OSSの代表的なCコンパイラであるgccと、開発で必須のツールであるmakeを紹介する。	13,14
II-5-9. TCP/IPネットワークの基本	UNIXやLinuxで標準的に利用されるネットワーク技術であるTCP/IPの基本について説明する。IPアドレス、ネットマスク、デフォルトゲートウェイの設定、ネットワークアドレス、ブロードキャストアドレスなど基本的な項目を解説する。	15
II-5-10. ネットワークの制御と管理	「TCP/IPネットワークの基本」で紹介した様々な基本的項目について、Linuxシステムにおける設定手順や設定に関連するファイル、ディレクトリ等を説明する。また適切に設定されたかどうかを確認するための手順と確認に必要なツールの使い方について解説する。	15

【学習ガイダンスの使い方】

1. 「習得ポイント」により、当該科目で習得することが期待される概念・知識の全体像を把握する。
2. 「シラバス」、「IT 知識体系との対応関係」、「OSS モデルカリキュラム固有知識」をもとに、必要に応じて、従来の IT 教育プログラム等との相違を把握した上で、具体的な講義計画を考案する。
3. 習得ポイント毎の「学習の要点」と「解説」を参考にして、講義で使用する教材等を準備する。

3. IT 知識体系との対応関係

「5. Linux の概念や基本操作に関する知識Ⅱ」と IT 知識体系との対応関係は以下の通り。

科目名	基本レベル（Ⅰ）								応用レベル（Ⅱ）						
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
5. Linuxの概念や基本操作に関するスキル	<Linux概要>	<ファイル操作>		<ユーザの権限と管理>		<システム管理>			<ファイルシステム>	<ファイルシステム>	<データ保全とバックアップ>	<データ保全とバックアップ>	<シェルスクリプトと開発環境>	<シェルスクリプトと開発環境>	<ネットワークの基本>

[シラバス：http://www.ipa.go.jp/software/open/ossc/download/Model_Curriculum_05_05.pdf]

<IT 知識体系上の関連部分>

分野		科目名	1	2	3	4	5	6	7	8	9	10	11	12	13
組織間連事項と情報システム	1	IT-IAS 情報保証と情報セキュリティ	IT-IAS1.基礎的な問題	IT-IAS2.情報セキュリティの仕組み(対策)	IT-IAS3.運用上の問題	IT-IAS4.ポリシー	IT-IAS5.攻撃	IT-IAS6.情報セキュリティ分野	IT-IAS7.フォレンジック(情報証拠)	IT-IAS8.情報の状態	IT-IAS9.情報セキュリティサービス	IT-IAS10.脅威分析モデル	IT-IAS11.脆弱性		
	2	IT-SP 社会的な観点とプロフェッショナルとしての課題	IT-SP1.プロフェッショナルとしてのコミュニケーション	IT-SP2.コンピュータの歴史	IT-SP3.コンピュータを取り巻く社会環境	IT-SP4.チームワーク	IT-SP5.知的財産権	IT-SP6.コンピュータの法的問題	IT-SP7.組織の中のIT	IT-SP8.プロフェッショナルとしての倫理的な問題と責任	IT-SP9.プライバシーと個人の自由				
応用技術	3	IT-IM 情報管理	IT-IM1.情報管理の概念と基礎 [5-11]	IT-IM2.データベース問合わせ言語	IT-IM3.データアーキテクチャ	IT-IM4.データモデリングとデータベース設計	IT-IM5.データと情報の管理	IT-IM6.データベースの応用分野							
	4	IT-WS Webシステムとその技術	IT-WS1.Web技術	IT-WS2.情報アーキテクチャ	IT-WS3.デジタルメディア	IT-WS4.Web開発	IT-WS5.脆弱性	IT-WS6.ソーシャルソフトウェア							
ソフトウェアの方法と技術	5	IT-PF プログラミング基礎	IT-PF1.基本データ構造	IT-PF2.プログラミングの基本的構成要素	IT-PF3.オブジェクト指向プログラミング	IT-PF4.アルゴリズムと問題解決	IT-PF5.イベント駆動プログラミング	IT-PF6.再帰							
	6	IT-IPT 技術を統合するためのプログラミング	IT-IPT1.システム間通信	IT-IPT2.データ割り当てと交換	IT-IPT3.統合的コーディング	IT-IPT4.スクリプティング手法	IT-IPT5.ソフトウェアセキュリティの実現 [5-13,14]	IT-IPT6.種々の問題	IT-IPT7.プログラミング言語の概要						
	7	CE-SWE ソフトウェア工学	CE-SWE0.歴史と概要	CE-SWE1.ソフトウェアプロセス	CE-SWE2.ソフトウェアの要求と仕様	CE-SWE3.ソフトウェアの設計	CE-SWE4.ソフトウェアのテストと検証	CE-SWE5.ソフトウェアの保守	CE-SWE6.ソフトウェア開発・保守ツールと環境	CE-SWE7.ソフトウェアプロジェクト管理	CE-SWE8.言語翻訳	CE-SWE9.ソフトウェアのフォールトトレランス	CE-SWE10.ソフトウェアの構成管理	CE-SWE11.ソフトウェアの標準化	
	8	IT-SIA システムインテグレーションとアーキテクチャ	IT-SIA1.要求仕様	IT-SIA2.調達/手配	IT-SIA3.インテグレーション	IT-SIA4.プロジェクト管理	IT-SIA5.テストと品質保証	IT-SIA6.組織の特性	IT-SIA7.アーキテクチャ						
システム基盤	9	IT-NET ネットワーク	IT-NET1.ネットワークの基礎	IT-NET2.ルーティングとスイッチング	IT-NET3.物理層	IT-NET4.セキュリティ	IT-NET5.アプリケーション分野	IT-NET6.ネットワーク管理 [5-11]							
	10	CE-NWK テレコミュニケーション	CE-NWK0.歴史と概要	CE-NWK1.通信ネットワークのアーキテクチャ	CE-NWK2.通信ネットワークのプロトコル	CE-NWK3.LANとWAN	CE-NWK4.クライアントサーバコンピューティング	CE-NWK5.データのセキュリティと整合性	CE-NWK6.ワイヤレスコンピューティングとモバイルコンピューティング [5-13,14]	CE-NWK7.データ通信	CE-NWK8.組込み機器向けネットワーク	CE-NWK9.通信技術とネットワーク概要	CE-NWK10.性能評価	CE-NWK11.ネットワーク管理	CE-NWK12.圧縮と伸張
			CE-NWK13.クラスタシステム	CE-NWK14.インターネットアプリケーション	CE-NWK15.次世代インターネット	CE-NWK16.放送									
	11	IT-PT プラットフォーム技術	IT-PT1.オペレーティングシステム	IT-PT2.アーキテクチャと機構	IT-PT3.コンピュータインフラストラクチャ	IT-PT4.デプロイメントソフトウェア	IT-PT5.ファームウェア	IT-PT6.ハードウェア							
	12	CE-OPS オペレーティングシステム	CE-OPS0.歴史と概要 [5-1]	CE-OPS1.並行性	CE-OPS2.スケジューリングとディスパッチ	CE-OPS3.メモリ管理	CE-OPS4.セキュリティと保護	CE-OPS5.ファイル管理	CE-OPS6.リアルタイムOS [5-9]	CE-OPS7.OSの概要	CE-OPS8.設計の原則	CE-OPS9.デバイス管理	CE-OPS10.システム性能評価		
コンピュータとアーキテクチャ	13	CE-CA0 コンピュータのアーキテクチャと構成	CE-CA00.歴史と概要	CE-CA01.コンピュータアーキテクチャの基礎	CE-CA02.メモリシステムの構成とアーキテクチャ	CE-CA03.インタフェースと通信	CE-CA04.デバイスサブシステム	CE-CA05.CPUアーキテクチャ	CE-CA06.性能・コスト評価	CE-CA07.分散・並列処理	CE-CA08.コンピュータによる計算	CE-CA09.性能向上			
複数領域にまたがるもの	14	IT-ITF IT基礎	IT-ITF1.ITの一般的なテーマ	IT-ITF2.組織の問題	IT-ITF3.ITの歴史	IT-ITF4.IT分野(学科)とそれに関連のある分野(学科)	IT-ITF5.応用領域	IT-ITF6.IT分野における数学と統計学の活用							
	15	CE-ESY 組込みシステム	CE-ESY0.歴史と概要	CE-ESY1.低電力コンピューティング	CE-ESY2.高信頼性システムの設計	CE-ESY3.組込み用アーキテクチャ	CE-ESY4.開発環境	CE-ESY5.ライフサイクル	CE-ESY6.要件分析	CE-ESY7.仕様定義	CE-ESY8.構造設計	CE-ESY9.テスト	CE-ESY10.プロジェクト管理	CE-ESY11.並行設計(ハードウェア、ソフトウェア	CE-ESY12.実装
			CE-ESY13.リアルタイムシステム設計	CE-ESY14.組込みマイクロコントローラ	CE-ESY15.組込みプログラム	CE-ESY16.設計手法	CE-ESY17.ツールによるサポート	CE-ESY18.ネットワーク型組込みシステム	CE-ESY19.インタフェースシステムと混合信号システム	CE-ESY20.センサ技術	CE-ESY21.デバイスドライバ	CE-ESY22.メンテナンス	CE-ESY23.専門システム	CE-ESY24.信頼性とフォールトトレランス	

4. OSS モデルカリキュラム固有の知識

OSS モデルカリキュラム固有の知識として、UNIX/Linux の具体的な基本操作がある。シェルを通して実行されるファイル操作やネットワーク制御などについて、具体的なコマンドの実行を通して習得する。

科目名	第9回	第10回	第11回	第12回	第13回	第14回	第15回
5.Linux の概念や基本操作に関する知識Ⅱ	(1)UNIX/Linux でのディレクトリ構成		(1)バックアップの意義		(1)シェルの役割		(1)TCP/IP ネットワークの基本項目の確
	(2)マウント(mount/umount)		(2)各パーティション/ディレクトリの重要性の検討		(2)シェル変数と環境変数(特にPATH 環境変数の役割は重要)		(2)ネットワーク制御コマンド
	(3)ディスクパーティション		(3)バックアップメディア		(3)スクリプト機能(Linux 前提であれば、B シェルをとりあえずのデフォルトにする)		(3)動作の確認
	(4)/etc/fstab によるマウント定義		(4)バックアップ用のツール		(4)実例の紹介		
	(5)ファイルシステムの構築(fdisk/mke2fs)		(5)バックアップ方法		(5)C 言語の開発環境		
			(6)リストア方法				

(網掛け部分は IT 知識体系で学習できる知識を示し、それ以外は OSS モデルカリキュラム固有の知識を示している)

スキル区分	OSS モデルカリキュラムの科目	レベル
システム分野	5 Linux の概念や基本操作に関する知識 II	応用
習得ポイント	II -5-1. ディレクトリ構成の実際	
対応するコースウェア	第 9、10 回　ファイルシステム	

II -5-1. ディレクトリ構成の実際

一時的に利用するファイルを置く/tmp、設定ファイル等が格納される/etc、ユーザのデータとして取り扱われるファイルを置く/home、プログラムが配置される/bin など、UNIX や Linux で一般的なディレクトリ構成を紹介する。

【学習の要点】

* Linux ではディストリビューションにより一部異なる箇所はあるが、FHS(Filesystem Hierarchy Standard)によってディレクトリ構成や役割が規定されている。
* 基本的なコマンドがある/bin、様々な設定ファイルがある/etc など、一般的なディレクトリ構成を覚えておくことにより、Linux の利用をより円滑に進めることができる。

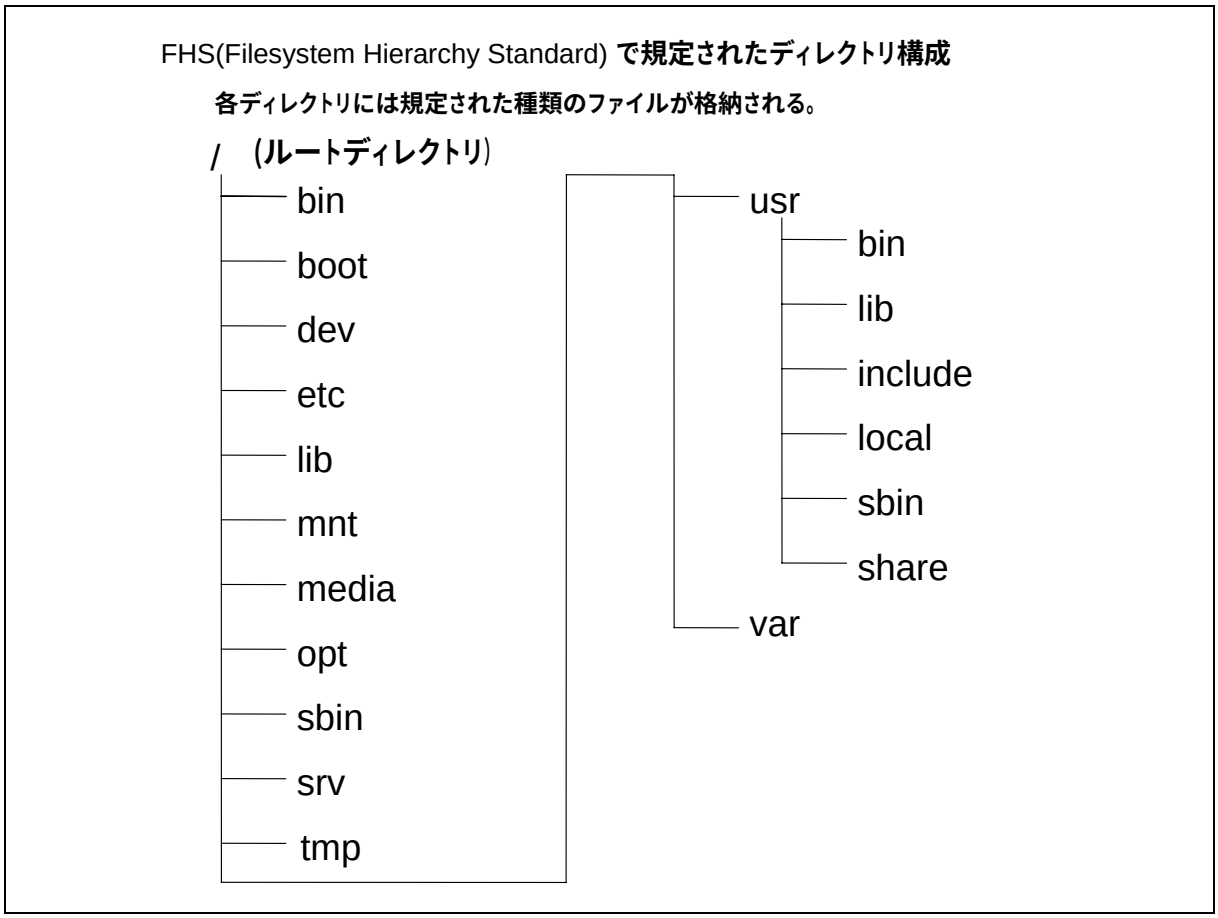

図 II-5-1. ディレクトリ構成の例

【解説】

1) FHS(Filesystem Hierarchy Standard)

FHS とは、各 Linux ディストリビューション間のファイルやディレクトリ構造の違いによる混乱を防ぎ、互換性を保つために Daniel Quinlan 氏らによってまとめられ提唱されているディレクトリ構成を示した、ファイルの標準化仕様書で、最新のバージョンは 2.3 となっている。
ディレクトリ構造、ディレクトリの用途に加え、ディレクトリに配置されるファイルの種類についても規定している。

2) Linux のディレクトリ構成

ディレクトリは、ルートディレクトリ (/) を頂点とする階層化された木構造となっている。
FHS ではルートディレクトリへ配置するディレクトリとして以下のものを規定している。

* /bin : 全てのユーザが使用するコマンドファイルを配置する。
* /boot : システムのブート時に使用されるファイルを配置する。
* /dev : 特殊ファイルまたはデバイスファイルを配置する。
* /etc : 各プログラムの設定ファイルを配置する。
* /lib : 共通ライブラリやカーネルモジュールを配置する。
* /media : リムーバブルメディアのマウントポイントとなるファイルを配置する。
* /mnt : システム管理者が一時的にファイルシステムをマウントするためのファイルを配置する。
* /opt : 追加インストールを行うアプリケーションを配置する。
* /sbin : システムの起動や修復を行う時に使用するコマンドを配置する。
* /srv : システムによって割り当てられた、サービスのためのデータファイルを配置する。
* /tmp : 各プログラムが作成する一時ファイルを配置する。
* /usr : ディレクトリ構成の 2 階層で特に重要であり、以下のフォルダを配置する。
    - /usr/bin : 全てのユーザが使用するユーザコマンドを配置する。
    - /usr/include : プログラム開発に使用するヘッダファイルを配置する。
    - /usr/lib : プログラム開発に使用する共有ライブラリなどを配置する。
    - /usr/local : システム管理者がインストールするアプリケーションを配置する。
    - /usr/sbin : 比較的重要でないシステムコマンドを配置する。
    - /usr/share : ドキュメントやイメージファイルなどの共有ファイルを配置する。
* /var : 管理データやログデータなどの可変データ及びディレクトリを配置する。

3) 絶対パスと相対パス

ディレクトリを指定する方法として、ルートディレクトリからの絶対パスと現在のディレクトリからの位置を示す相対パスがある。
相対パスでは現在のディレクトリを「.」、上位のディレクトリを「..」で表す。

* 絶対パスの例 : /usr/local
* 相対パスの例: ./local、 ../local

スキル区分	OSS モデルカリキュラムの科目	レベル
システム分野	5 Linux の概念や基本操作に関する知識 II	応用
習得ポイント	II -5-2. パーティションとマウント	
対応する コースウェア	第 9、10 回　ファイルシステム	

II -5-2. パーティションとマウント

ディスクのパーティショニングという概念を示し、物理パーティション、論理ディスク(LVM)などの要素技術を紹介する。またそれぞれのパーティションをマウントして利用する考え方を説明し、マウント/アンマウントの手順を示す。

【学習の要点】

* 物理パーティションは、ディスクの物理単位での領域を示したものであり、ディスクの種類と接続順位によって命名規則がある。
* パーティションの設定（fdisk コマンド）によって、ディスクの領域を複数のパーティションに分割することができる。
* 論理ボリュームマネージャ(LVM)は物理ディスク内に動的に変更できる論理的なディスクを作成する仕組みである。
* mount コマンドでディスクを任意のディレクトリにマウントすることができる。

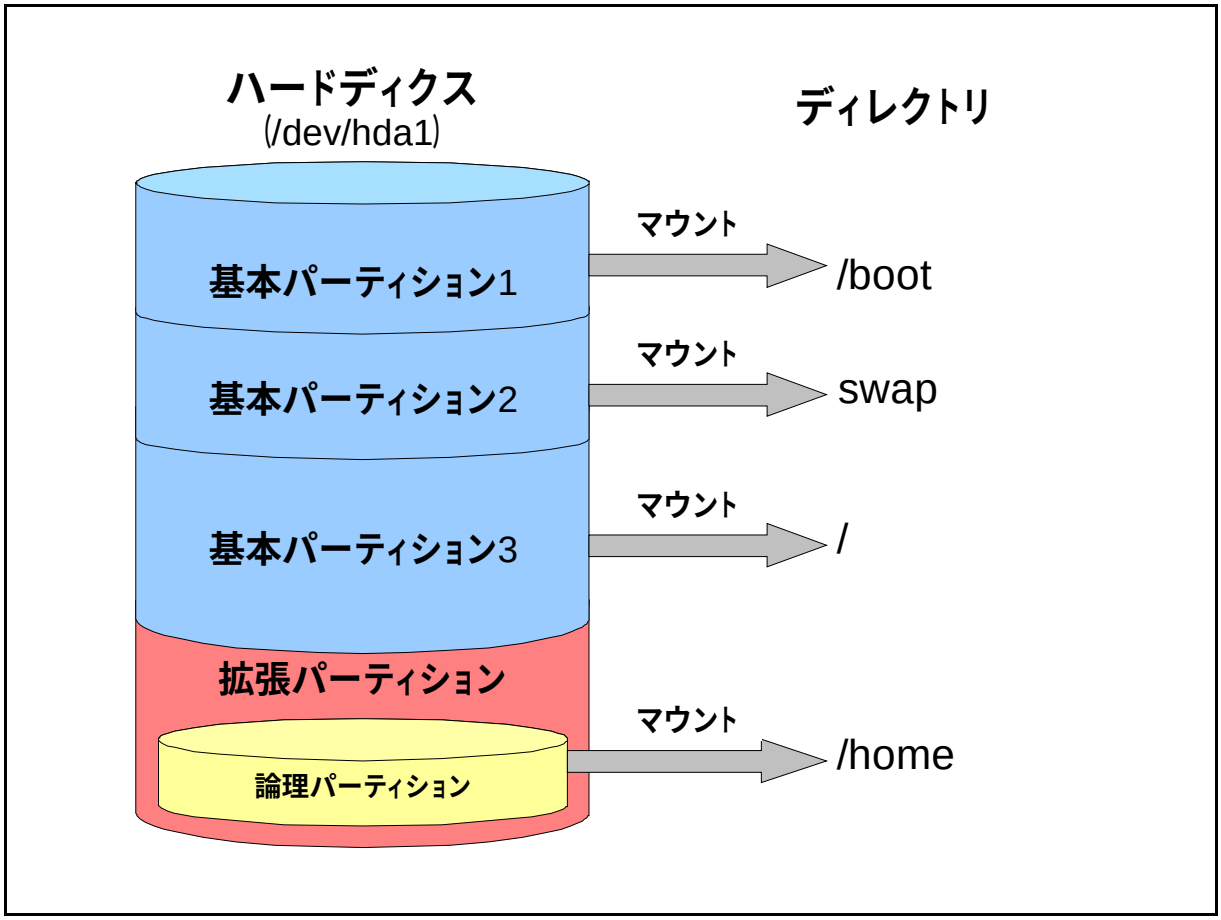

図 II-5-2.　ディスクパーティション

【解説】

1) Linux でのデバイス名の扱い

Linux ではハードディスクや CD-ROM などの記憶領域は何れかのディレクトリにマウントして利用する。Linux において、接続されている機器は接続方法により以下のようなデバイス名が与えられる。

/dev/hda　：IDE 接続されたプライマリマスタ
/dev/hdb　：IDE 接続されたプライマリスレーブ
/dev/sda　：ＳＣＳＩ接続されたＩＤが１番小さいハードディスク
/dev/sdb　：ＳＣＳＩ接続されたＩＤが２番目に小さいハードディスク
/dev/fd0　：フロッピードライブ
/dev/sr0　：CD-ROM ドライブ

2) ディスクパーティション

1つのハードディスクを複数の領域に分割して使用することをパーティションと呼ぶ。パーティションを分割して利用することにより、ハードディスクの一部が物理的に壊れた場合でも影響を受けるのは1つのパーティションとなり、他のパーティションのデータを守ることができる。

ハードディスクのパーティションはハードディスクごとに 4 つのパーティションテーブルで定義される。

Linux で使用されるパーティションとして以下の3つがある。

* 基本パーティション

  1つのハードディスクを分割して使用するためのパーティションで、拡張パーティションと合わせ、4 つまで作成することができる。

* 拡張パーティション

  1つのハードディスクに 1 つだけ作成できるパーティションで、論理パーティションを作成する為に作成する。

* 論理パーティション

  拡張パーティション内に作成するパーティションである。

3) 論理ボリュームネージャ：LVM (logical volume managemer)

ＬＶＭを使用することにより、複数のディスクを1つの大きな論理ドライブとして管理することができる。この論理ドライブ上に作成する仮想パーティションを論理ボリュームと呼ぶ。

論理ボリュームはサイズを動的に拡張･縮小することができる。また、物理ディスクを追加することにより論理ボリュームを拡張することができる。

4) パーティションの操作

パーティションの作成コマンドの例

```
# fdisk /dev/sda
```

記憶領域のマウント／アンマウントの例

```
# mount /dev/sda1 /home
# unmount /home
```

スキル区分	OSS モデルカリキュラムの科目	レベル
システム分野	5 Linux の概念や基本操作に関する知識 II	応用
習得ポイント	II-5-3. ファイルシステムの構築	
対応する コースウェア	第 9、10 回　ファイルシステム	

II-5-3. ファイルシステムの構築

ファイルシステムとは何か説明し、Linux で取り扱うことができる代表的なファイルシステムの種類を挙げる。またファイルシステムを具体的に構築するコマンドを示し、ファイルシステム構築の手順について述べる。

【学習の要点】

* ファイルシステムは、記憶装置に記録されているデータを管理する方法を提供する機能である。
* Linux で利用できるファイルシステムには、ext2、ext3、ReiserFS などがあり、堅牢性、ディスク領域の使用効率、速度等に違いがあり、目的に応じてファイルシステムを選択することができる。
* fdisk コマンド、mke2fs コマンドを使用し、パーティションの作成やフォーマットすることができる。

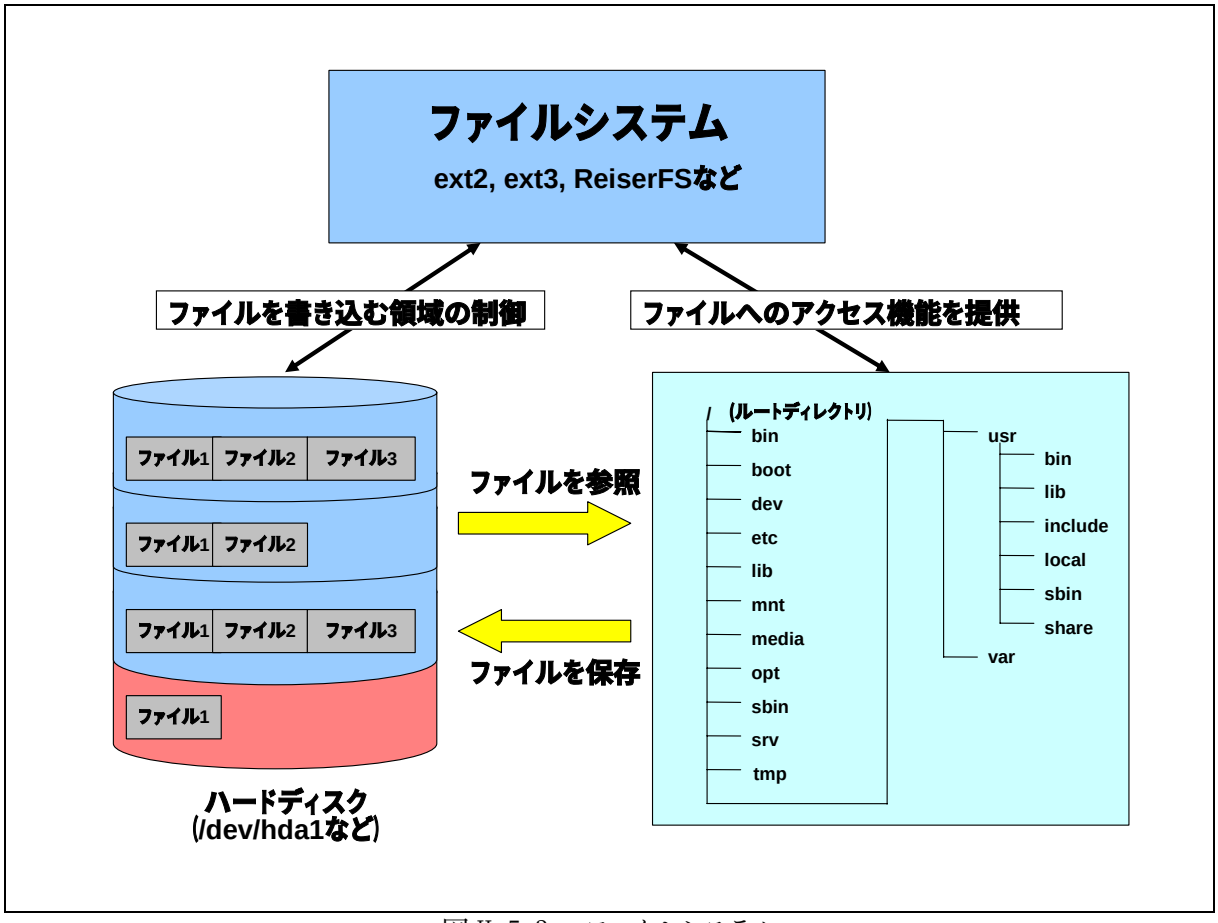

図 II-5-3.　ファイルシステム

【解説】

1) ファイルシステムとは

Linux では文書やプログラムをファイルという単位でハードディスク等へ記録している。
ファイルシステムは、ファイルを書き込む領域の制御や、ファイル名によるファイルの実体へアクセスする機能などを提供する。
Linux で利用できる代表的なファイルシステムは以下のようなものがあり、パーティションごとに利用するファイルシステムを設定することができる。

* ext2

  BSD の FFS(Fast File System)を基に開発されたファイルシステムで、ディスク領域に4Kbyte のブロックを作成し、複数のブロックをブロックグループにまとめて管理を行う。

  ext2 は、ジャーナリングを備えていないため、障害が発生した場合ファイルシステムの復旧に時間がかかる。

* ext3

  ext2 へジャーナル機能を追加したファイルシステムで、現在の Linux では主流となっている。

  ジャーナル機能は、ファイルの書き込み処理ごとにファイルの構成情報を含むメタデータを管理･保存する機能で、書き込み中の障害などからファイルの損失を避けることができる。

* ReiserFS

  ジャーナル機能を持ったファイルシステムの 1 つで、B*-Tree アルゴリズムを使用することで、ファイルの高速な検索を実現している。

  サイズの小さいファイルを多数処理する場合、ext2/ext3 よりもパフォーマンスに優れている。

* XFS

  シリコングラフィックスが IRIX オペレーティングシステムにて巨大な記憶領域を管理するために開発された 64 ビットファイルシステムで、サイズの大きいファイルを処理する場合のパフォーマンスに優れている。

* JFS

  IBM が商用 UNIX、AIX にて実装したジャーナル機能を持ったファイルシステムで、商用利用に耐えられるように高い信頼性、性能、スケーラビリティを実現している。

2) ファイルシステムの構築

fdisk によるファイルシステムの設定

t コマンドにより、対話形式で作成したパーティションのファイルシステムを設定する。

 # fdisk /dev/sda
コマンド(m でヘルプ): **t**
領域番号 (1-4): **2**
16 進数コード (L コマンドでコードリスト表示): **83**

mke2fs コマンドによる ext2/ext3 ファイルシステムの作成

 # mke2fs /dev/sda1
 # mke2fs -j /dev/sda2 (または **mkfs.ext3 /dev/sda2**)

スキル区分	OSS モデルカリキュラムの科目	レベル
システム分野	5 Linux の概念や基本操作に関する知識 II	応用
習得ポイント	II-5-4. データのバックアップ方法	
対応するコースウェア	第 11、12 回　データの保全とバックアップ	

II-5-4. データのバックアップ方法

バックアップの必要性、バックアップの範囲、バックアップ対象のメディア等、データのバックアップに必要な知識を説明する。またデータのバックアップに利用するツールを紹介し、バックアップの方法と具体的な手順を示す。

【学習の要点】

* データの消失、破壊は、機器障害、操作ミス等により発生する。特に Linux をサーバとして利用する場合には、バックアップは必ず実施するべきものである。
* バックアップの方法と範囲は、費用、重要性、速度等を考慮して選択する必要がある。
* tar コマンドは、任意のディレクトリを一つにまとめたアーカイブを作成することができる。
* dump コマンドは、ファイルシステム全体をバックアップすることができ、差分バックアップや増分バックアップを作成することができる。

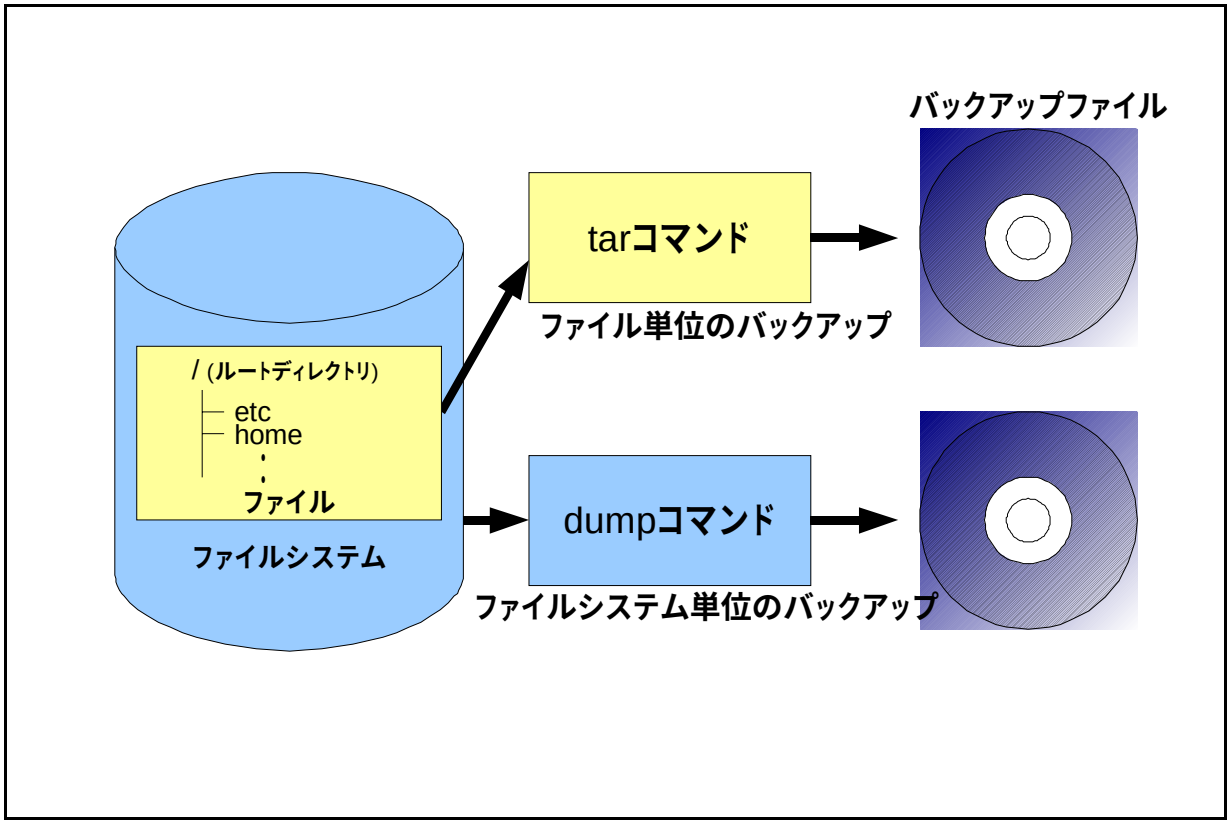

図 II-5-4.　データのバックアップ

【解説】

1) バックアップの意義

ハードディスクの障害や操作ミス、アプリケーションの不具合などにより、作成したデータが破壊や損失してしまうことがある。
事前にデータをバックアップしておくことにより、データが破壊や損失してしまった場合でもバックアップを行った時点のデータに復元することが可能となる。

2) バックアップ方法

データのバックアップを行う際に考慮すべき点は主に以下の項目である。

* バックアップ対象

システム上の全てのデータが破壊、損失した場合、システムの再インストール後にどのデータを復元すればシステムを復旧できるかを考慮してバックアップを行うデータを決定する。

多くの場合、以下のディレクトリのデータについてはバックアップが必要である。

- /etc　　各アプリケーションの設定ファイルが保存されている。
- /var　　サーバアプリケーションで使用するデータやログが保存されている。
- /home　　各ユーザが作成したデータが保存されている。

* バックアップメディア

バックアップしたデータをどのように管理、保管するかを考慮してバックアップデータを保存するメディアを決定する。

- 同じコンピュータ上のハードディスク

  複数のハードディスクが接続されている場合、最も簡単なバックアップ方法であるが、システム全体が破壊、損失した場合は普及することが出来ない。

- 他のコンピュータ上のハードディスク

  ネットワーク等と介して遠隔地のサーバにバックアップすることも可能で、システム全体が破壊、損失した場合でも別のコンピュータへシステムを復元することが可能となる。

- リムーバブルメディア

  CD や DVD などにバックアップデータを書き込むことにより、データを別の場所へ保管することが可能となる。

* バックアップコマンド

バックアップ対応、バックアップメディアを考慮し、使用するコマンドを選定する。

- tar コマンド

  複数のファイルを所有権やアクセス権限などの情報を含めて 1 つのファイルにまとめる機能を持ち、ファイル単位でバックアップを行う際に使用される。

  使用例： tar –cvf backuupfile.tar etc

- dump コマンド

  ファイルシステム単位でデータのバックアップを行う。オプションを指定することによりファイルシステム全体または差分だけをバックアップすることができる。

  使用例： # dump 0uf backupfile.dat /dev/sda2

スキル区分	OSS モデルカリキュラムの科目	レベル
システム分野	5 Linux の概念や基本操作に関する知識 II	応用
習得ポイント	II-5-5. バックアップしたデータのリストア	
対応する コースウェア	第 11、12 回　データの保全とバックアップ	

II-5-5. バックアップしたデータのリストア

データのリストアが必要になるタイミングやそれぞれのバックアップ方法に対応したリストアの手順について解説する。またデータのリストア時に留意すべきポイントを示し、具体的なリストア方法を説明する。

【学習の要点】

* システムトラブルなどにより、システム内のデータが消失、破壊した場合、それを復旧するため、事前にバックアップされたデータのリストアを行う。
* tar コマンドにてバックアップしたデータは、tar コマンドにてリストアを行う。その際に復旧に必要なデータのみリストアすることができる。
* dump コマンドによって作成したバックアップデータは、restore コマンドにてリストアを行う。

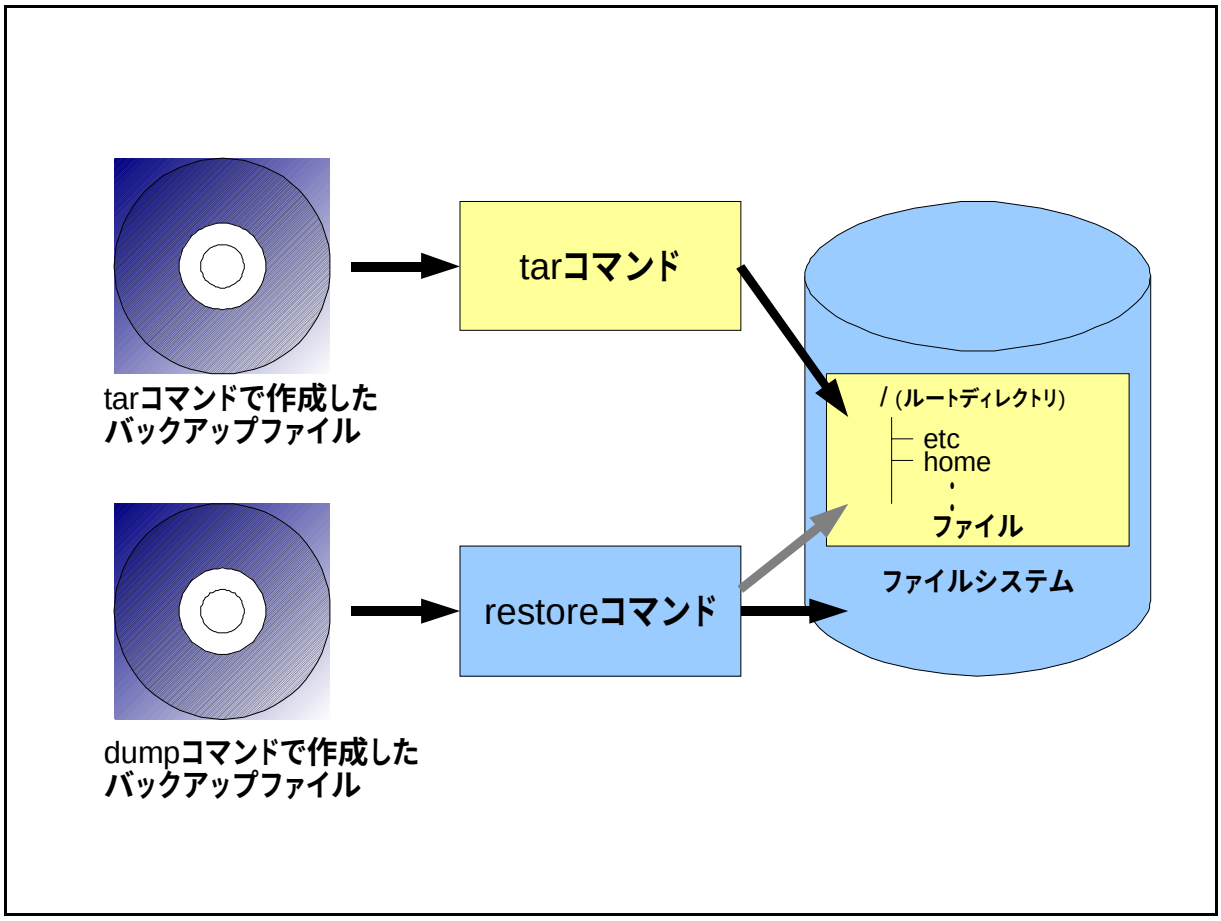

図 II-5-5.　バックアップデータのリストア

【解説】

1) リストアの目的

リストアは事前に作成されたバックアップデータを使用して、システムを復旧することを目的とする。
システム障害やユーザの操作ミスなどにより、データが破壊または損失した時、バックアップデータをリストアすることで、システムの復元を行う。
また、既存のシステムの複製を作成するときなどにバックアップデータのリストアを使用する。

2) リストア方法

システムの障害状況やバックアップデータの作成方法により、どのようにデータをリストアするのか決定する。
システム障害により全てのデータの復旧が必要な場合は、システム全体をバックアップしたデータからリストアを行う。ユーザの操作ミスなどにより、一部のデータが破壊、損失した場合は、ファイル単位でバックアップしたデータからリストアするか、全体をバックアップしたデータから必要なデータのみをリストアする。
また、差分データのバックアップを行っている場合、全体をバックアップしたデータをリストアした後で、差分バックアップを作成した順番にリストアを行う。

* tar コマンド

  tar コマンドでバックアップしたデータは、tar コマンドを使用してリストアを行う。

  バックアップしている全てのデータをリストアする他、引数の指定により、バックアップしているファイルを指定してリストアすることができる。

  tar コマンドにてバックアップファイルを作成した場合、作成時に指定するファイルのパスを絶対パスで指定するとバックアップファイル内でも絶対パスで保存される。また、相対パスで指定した場合は相対パスで保存される。そのため、リストアする場合には tar コマンドを実行するディレクトリにも注意が必要となる。

  使用例：# **tar -xvf backupfile.tar**

* restore コマンド

  dump コマンドでバックアップしたデータは、restore コマンドを使用してリストアを行う。

  オプションを指定することによりファイルシステム全体または差分だけをリストアすることができる。

  restore コマンドはオプションによりリストア方法を指定する他、対話式コマンドにてリストアを行うファイルを指定することも可能である。

  restore コマンドでリストアしたデータは、コマンドを実行したディレクトリへ保存されるため、データのリストアを行うディレクトリへ移動してから実行する必要がある。

  使用例：# **restore 0rf backupfile.dat**

スキル区分	OSS モデルカリキュラムの科目	レベル
システム分野	5 Linux の概念や基本操作に関する知識 II	応用
習得ポイント	II-5-6. シェルスクリプトの基本	
対応するコースウェア	第 13、14 回　シェルスクリプトと開発環境	

II-5-6. シェルスクリプトの基本

シェルの役割を再認識し、シェル変数や環境変数といったシェルの高度な設定方法を理解させる。またシェルスクリプトの基本的な機能を紹介する。具体的には、変数の扱いや基本的な制御構造について解説する。

【学習の要点】

* Linuxではシェルがユーザが入力したコマンドを解釈して内部処理の実行や、外部プログラムへ処理の依頼を行う。
* シェルの種類は色々あるが、Bourne シェルと C シェルに大別できる。Linux では B シェルの機能を拡張した bash (Bourne Again Shell) がデフォルトのシェルになっている。
* シェル変数は現在実行中のシェルだけで有効な変数で、環境変数はシェルから起動されたすべてのプロセスに有効な変数である。
* シェルスクリプトは、複数のコマンドを一つにまとめて実行することや、引数や変数、繰り返しや条件分岐などの制御構文を用いてコマンドを実行することができ、作業の自動化に役立つ。

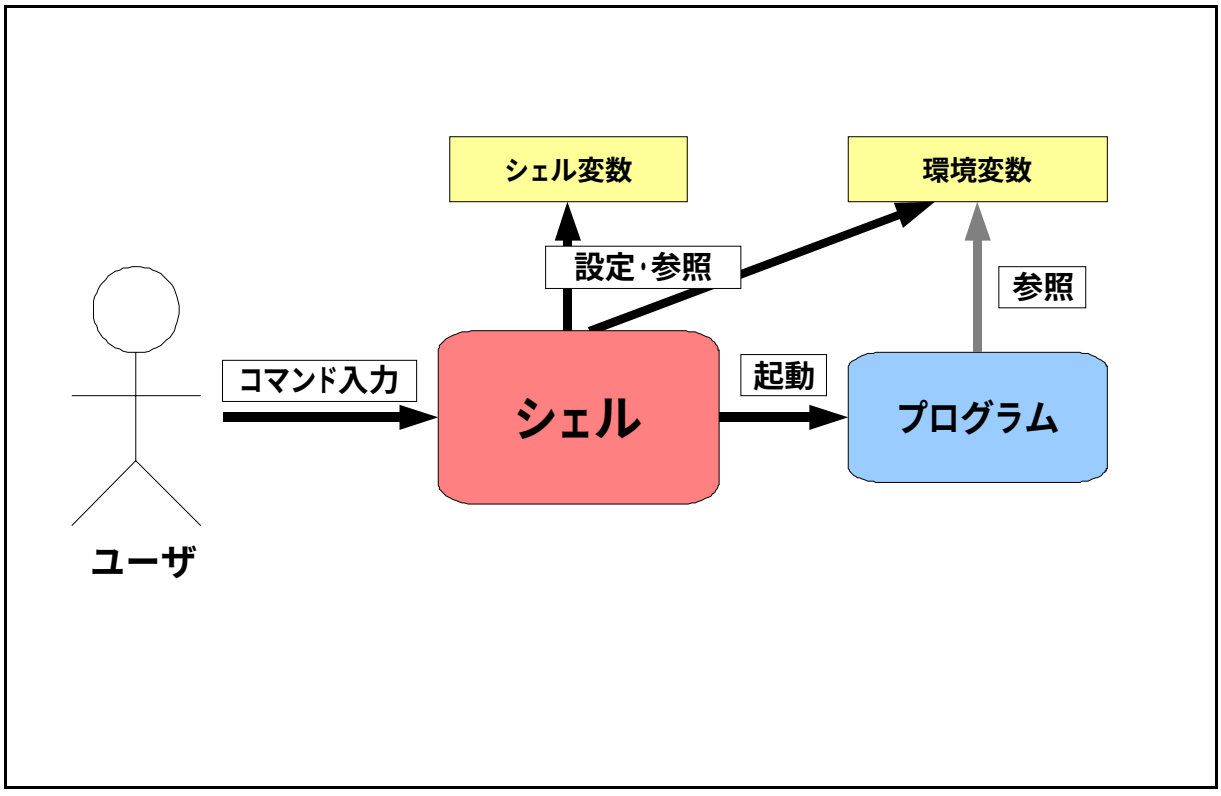

図 II-5-6.　シェルの役割

【解説】

1) シェルの役割

シェルはユーザが入力した文字を読み取り、これを解釈して Linux カーネルへ受け渡す役割をもったプログラムである。
シェルは Bourne シェル（sh 系）と C シェル（csh 系）に大別できるが、Linux では sh 系の bash(Bourne Again Shell)を標準シェルとして使用している。
bash は基本的な機能に加え、コマンド履歴、コマンド置換機能、入力途中のコマンド名やファイル名などの自動補完機能、などをサポートしている。

2) シェル変数と環境変数

シェルでは他のプログラム言語と同様に変数を定義して文字列や設定値などを保持することができる。シェルで使用する変数には、シェル内で定義し、そのシェルでしか利用できない「シェル変数」と、プログラムが起動した際にプログラム間で受け継がれる「環境変数」がある。

* シェル変数

  シェル変数を定義するには「変数名=値」のように変数名と値を等号記号(=)をつないで入力を行う。定義したシェル変数を使用するには「$変数名」のように変数名の頭に「$」をつけるか、「${変数名}」のように中括弧で囲んで記述する。

* 環境変数

  環境変数を定義するには、「export 変数名=値」のように export コマンドを使用する。

設定されている変数は set コマンドにて一覧を表示することができる。

3) シェルスクリプト

複数のコマンドを繰り返し実行する場合など、シェルへの入力内容をあらかじめ記述したファイルを「シェルスクリプト」呼ぶ。
シェルスクリプトではファイルの先頭から順番にコマンドをする他に、シェル変数や分岐やループなどの制御構文を用いることで複雑な処理を実行することが可能となり、作業を単純化したり自動化することができる。

* 分岐構造

  if 文または case 文を使用して、条件が真、または偽によってコマンドを選択して実行をする。

* ループ構造

  for 文または while 文を使用して条件が偽になるまで同じコマンドを実行する。

* シェル関数

  プログラム言語の関数と同様に、まとまった処理をシェル関数として定義しシェルスクリプト内の任意の場所から呼出すことができる。

スキル区分	OSS モデルカリキュラムの科目	レベル
システム分野	5 Linux の概念や基本操作に関する知識 II	応用
習得ポイント	II-5-7. Linux システムで利用されるシェルスクリプト	
対応するコースウェア	第 13、14 回　シェルスクリプトと開発環境	

II-5-7. Linux システムで利用されるシェルスクリプト

Linux システムでは様々なシェルスクリプトが活用されている。各アプリケーションやサービスの起動スクリプト、設定に利用されるシェルスクリプトなど代表的なスクリプトを示し、実際の動作がどのように行われるかについて説明する。

【学習の要点】

* Linux で代表的なシェルスクリプトとしては、/etc/init.d にあるシステムの起動時に使用されるシェルスクリプトがある。
* 各アプリケーションのコマンドにはシェルスクリプトが含まれていることがあり、実行に必要な設定を行うなど、利便性を高める目的などで使用されている。

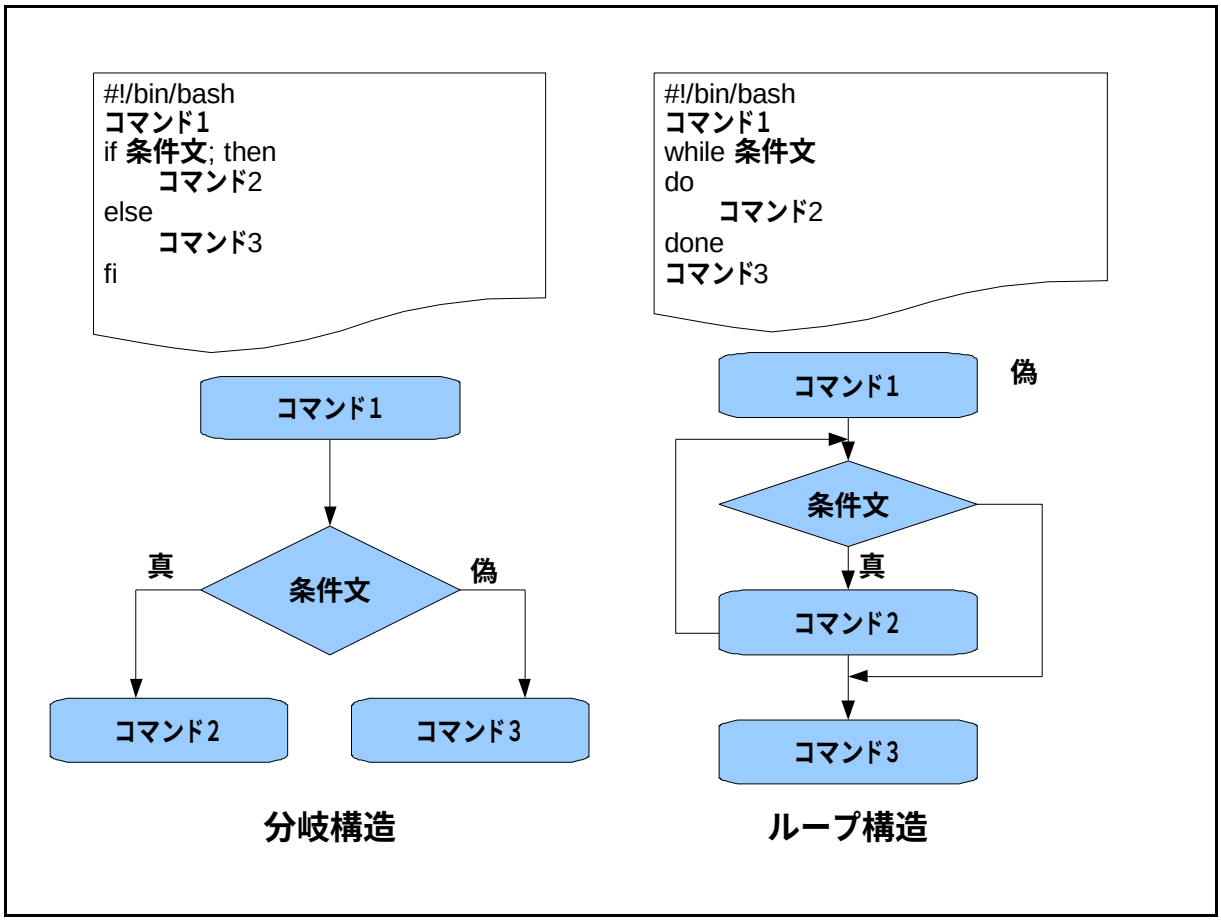

図 II-5-7.　シェルスクリプトの構文

【解説】

1) Linux でのシェルスクリプトの利用

Linux はシェルスクリプトを多用しているシステムで、/etc/init.d に格納されているシステム起動時に実行されるコマンドもシェルスクリプトで記述されている。
また、アプリケーションのインストール、アンインストールや実行時にも設定用のスクリプトが実行されている。

2) シェルスクリプトの作成

シェルスクリプトは実行するコマンドをテキストファイルとして記述していくことにより作成する。
1 行目には先頭に「#!」をつけ、シェルスクリプト内で使用するシェルを記述する。
例えば、/bin/bash を使用する場合には以下のように記述する。

#!/bin/bash

2 行目以降に実行するコマンドを記述する。
シェルスクリプトでは、実行時に指定される引数が以下のような変数として設定される。

$#	:与えられた引数の数
$1、$2…$9,${10}	:与えられた引数。数値は引数の順番を表す
$@	:引数全体
$*	:引数全体
$0	:実行されたコマンド名

シェルスクリプトのファイル名は特に規定されていないが、他のファイルと区別するために通常は「.sh」または「.bash」などの拡張子をつけたファイル名とする。

3) シェルスクリプトの実行

作成したシェルスクリプトを実行するには、以下の方法がある。

* シェルの引数として実行

  bash などの引数として作成したシェルスクリプトのファイル名を与えることにより、シェルがファイルの内容を解析しコマンドの実行を行う。

  $ **sh test.sh**

* 直接コマンドとして実行

  シェルスクリプトに実行権を付与し、直接コマンドとして実行することにより、スクリプトの内容が実行される。

  $ **chmod +x test.sh**

  $ **./test.sh**

スキル区分	OSS モデルカリキュラムの科目	レベル
システム分野	5 Linux の概念や基本操作に関する知識 II	応用
習得ポイント	II-5-8. C 言語による開発	
対応する コースウェア	第 13、14 回　シェルスクリプトと開発環境	

II-5-8. C 言語による開発

Linux で活用される多くのプログラムは C 言語によって開発されている。ここでは C 言語によるプログラミングの基本的な開発について触れ、OSS の代表的な C コンパイラである gcc と、開発で必須のツールである make を紹介する。

【学習の要点】

* Linux のプログラムの多くは C 言語によって開発されている。
* C 言語はコンパイラ型言語であるため、開発されたプログラムはコンパイルが必要である。
* C 言語のソースプログラムはテキストファイルであるため vi エディタなどで作成できる。
* OSS の C コンパイラとしては gcc が代表的であり、同コマンドで C 言語プログラムをコンパイルすることができる。
* make コマンドは、プログラムを自動的にコンパイルすることができる。また、ターゲットの指定により、コマンドを自動的に実行することができる。

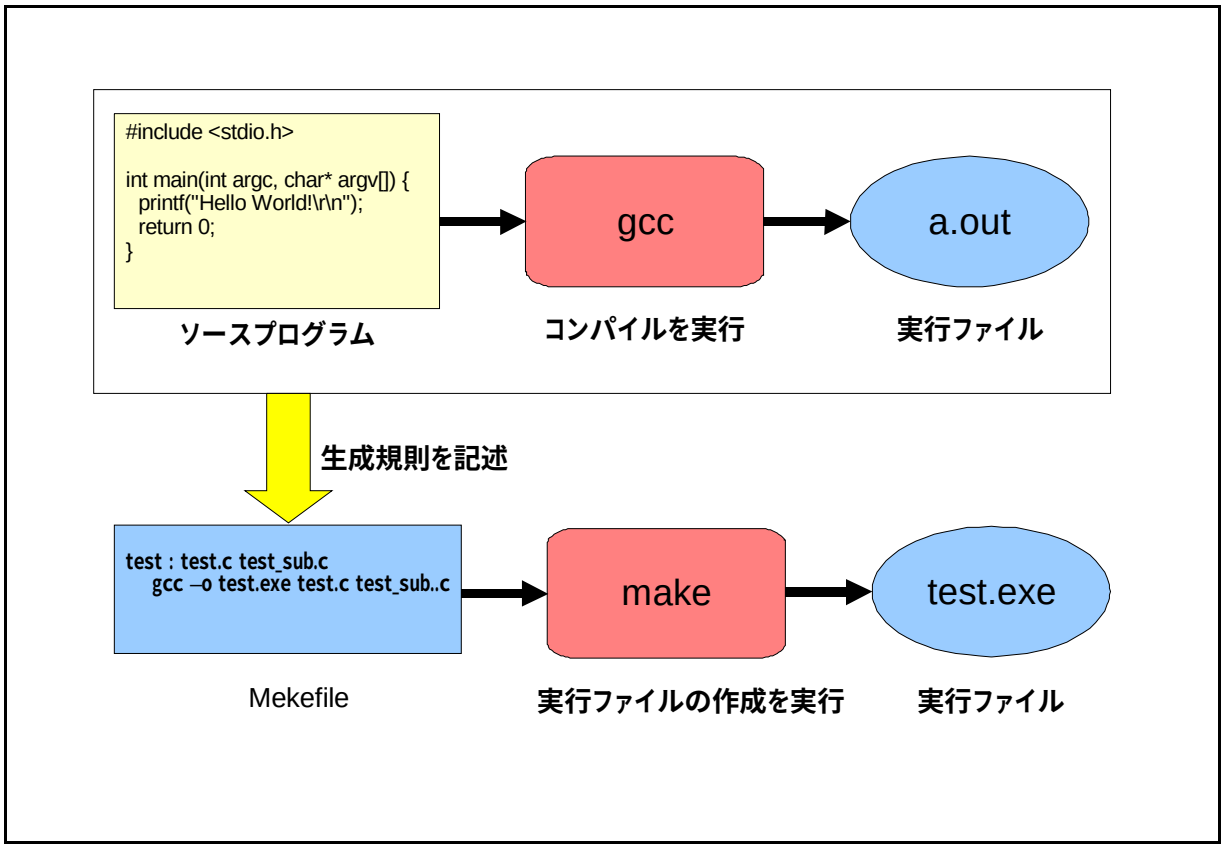

図 II-5-8.　Linux でのプログラム開発

【解説】

1) Linux のプログラム言語

Linux ではカーネルを初め、多くのアプリケーションが C 言語で開発されている。
カーネルやアプリケーションの開発、改変を行うため、プログラム言語のコンパイラとして GCC (GNU Compiler Collection) が提供されている。
また、make コマンドにより、コンパイル作業の自動化を行うことができる。

2) ソースファイルの作成

ソースファイルを作成するにはテキストエディタを使用するが、Linux では vi コマンドが提供されている。vi コマンドでは「コマンドモード」と「テキスト入力モード」を切り替えながらテキストファイルの作成を行う。

3) gcc コマンド

Linux では C 言語のコンパイルを行うため、gcc コマンドが標準で提供されている。
gcc コマンドは ANSI もしくは ISO の標準に準拠しており、Linux のカーネルの開発においても使用されている。
gcc コマンドは、引数として指定された C 言語のソースファイルのコンパイルを行い、実行ファイルを出力する。特に指定が無い場合、実行ファイルのファイル名は「a.out」となる。
ソースファイル「test.c」をコンパイルする場合、以下のようにコマンドを実行する。
$ **gcc test.c**
また、実行ファイルのファイル名を指定する場合は、以下のようにコマンドを実行する。
$ **gcc -o test.exe test.c**

4) make コマンド

make コマンドはプログラムの生成規則を記述したテキストファイルを参照して、効率よくコンパイル作業などを行うために使用する。
複数のソースファイルで構成されるプログラムでソースファイルの依存関係があると、正しい順序でコンパイルをしなくてはいけない。make コマンドでは依存関係から自動的にコンパイルする順番を決定する。また、依存するソースファイルの更新状態を調べ、ソースファイルを変更した時に、依存関係のあるものだけコンパイルを行うことで、コンパイル回数を必要最小限に抑えるようにする。
生成規則を記述したファイルは通常コンパイルを実行するディレクトリに「makefile」または「Makefile」というファイル名で作成する。Makefile のは以下のような構文で記述する。

```
ターゲット名: 依存ファイル名 1 依存ファイル名 2
        コマンド行 1
```

記述例

```
test : test.c test_sub.c
        gcc -o test.exe test.c test_sub..c
```

make コマンドにてコンパイルを行うには以下のようにコマンドを実行する。
$ **make**
また、作成するターゲットを指定する場合は以下のようにコマンドを実行する。
$ **make test**

スキル区分	OSS モデルカリキュラムの科目	レベル
システム分野	5 Linux の概念や基本操作に関する知識 II	応用
習得ポイント	II-5-9. TCP/IP ネットワークの基本	
対応するコースウェア	第 15 回　ネットワークの基本	

II-5-9. TCP/IP ネットワークの基本

UNIX や Linux で標準的に利用されるネットワーク技術である TCP/IP の基本について説明する。IP アドレス、ネットマスク、デフォルトゲートウェイの設定、ネットワークアドレス、ブロードキャストアドレスなど基本的な項目を解説する。

【学習の要点】

* Linux で標準的に利用されているネットワークプロトコルは TCP/IP で、HTTP や SSH などの通信で使用している。
* TCP/IP ではデータを送受信する機器を判別するために、各機器に固有の IP アドレスを付与する必要がある。
* IP アドレスのうち、機器が所属するネットワークを識別する部分をネットワークアドレスと呼ぶ。ネットワークアドレスは IP アドレスとネットマスクにより決定する。
* ブロードキャストアドレスはネットワーク内に全てデータを送信するための IP アドレスを指す。
* デフォルトゲートウェイは、外部のネットワークと通信する際に通過する機器の IP アドレスである。

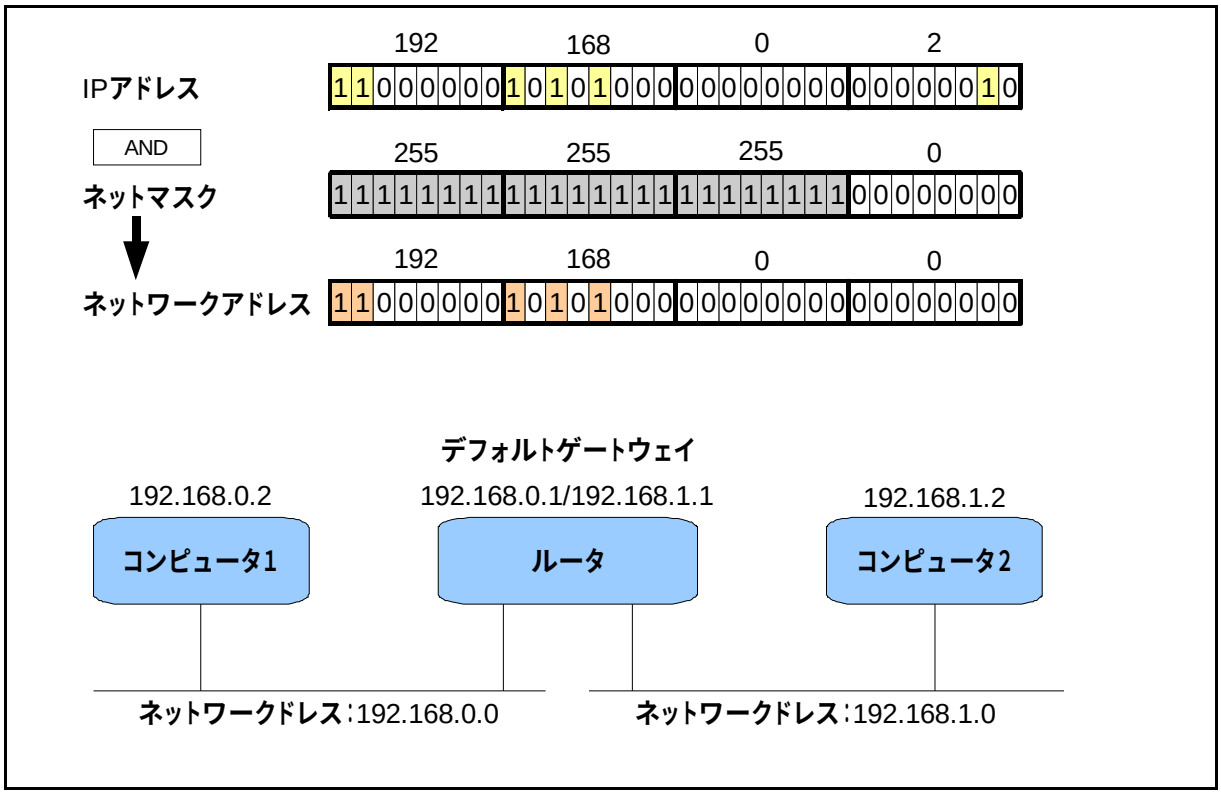

図 II-5-9.　TCP/IP ネットワーク

【解説】

Linux では TCP/IP プロトコルを利用してネットワークに接続し通信を行っている。
TCP/IP は OSI7 層構造モデルのトランスポート層とインターネット層に実装されたプロトコルである。
HTTP や SSH などの信頼性が求められるプロトコルは、TCP/IP の上位層であるアプリケーション層へ実装される。

* IP アドレス

  IP アドレスは、コンピュータに搭載されているネットワークインターフェイスを識別するため付与する 32 ビットの数値で、通常 8 ビットごとに分割した 0 から 255 の数字 4 組を「.」（ドット）でつないで表記する。IP アドレスは、サブネットワークを識別するネットワーク部とサブネットワーク内のコンピュータを識別するホスト部で構成される。

  IP アドレスの例： 192.168.0.1

* ネットワークアドレス、ブロードキャストアドレスとサブネットマスク

  コンピュータが所属するサブネットワークはネットワークアドレスによって識別する。ネットワークアドレスは、IP アドレスとサブネットマスクの論理積によって算出する。

  IP アドレスが 192.168.0.2 でサブネットマスクが 255.255.255.0 の場合、ネットワークアドレスは、192.168.0.0 となる。

  ブロードキャストアドレスは IP アドレスのホスト部分のビットを全て1に設定したアドレスで、サブネットワーク内の全てのコンピュータに通知を行うために利用される。

  ネットワークアドレスが、192.168.0.0 でサブネットマスクが 255.255.255.0 の場合、ブロードキャストアドレスは、192.168.0.255 となる。

* デフォルトゲートウェイ

  サブネットワークが異なるコンピュータ間で通信を行う場合、データを中継するルータへ送信する必要がある。このパケットを中継するルータのアドレスを、ゲートウェイアドレスと呼ぶ。ゲートウェウィアドレスは通信先ごとに設定することができるが、デフォルトゲートウェイとして 1 つだけゲートウェイアドレスを設定するのが一般的である。

* DNS（ドメインネームシステム）

  ネットワークインターフェイスは IP アドレスで識別することができるが、ホスト名を付与し IP アドレスとの対応情報を管理することにより人にとって扱いやすいようになっている。

  DNS ではホスト名から IP アドレスを変換したり、逆に IP アドレスからホスト名を変換することができる。DNS を利用して IP アドレスとホスト名を変換することを「名前解決」と呼ぶ。

* ポート番号

  コンピュータで動作しているどのサーバアプリケーションと通信を行うかを指定するためにポート番号が使われる。

  ポート番号は0～65536 までの整数が使われるが、主要なサーバアプリケーションでは1～1023 までのポート番号が割り当てられている。

スキル区分	OSS モデルカリキュラムの科目	レベル
システム分野	5 Linux の概念や基本操作に関する知識 II	応用
習得ポイント	II-5-10. ネットワークの制御と管理	
対応するコースウェア	第 15 回　ネットワークの基本	

II-5-10. ネットワークの制御と管理

「TCP/IP ネットワークの基本」で紹介した様々な基本的項目について、Linux システムにおける設定手順や設定に関連するファイル、ディレクトリ等を説明する。また適切に設定されたかどうかを確認するための手順と確認に必要なツールの使い方について解説する。

【学習の要点】

* ネットワークの基本設定は、sysconfig/network-scripts などの設定ファイルに定義する。インストール時や netconfig 等の半自動で決定してくれる設定ツールも存在する。
* ifconfig（IP アドレスの確認・変更）、route（ルーティングテーブルの確認・変更）、arp（arp キャッシュの初期化）などのネットワーク設定コマンドがある。
* ping（ネットワーク接続の確認）、host（nslookup）（ドメイン名と IP アドレスを対照）、traceroute（リモート接続の経路の表示）などのネットワーク確認コマンドがある。

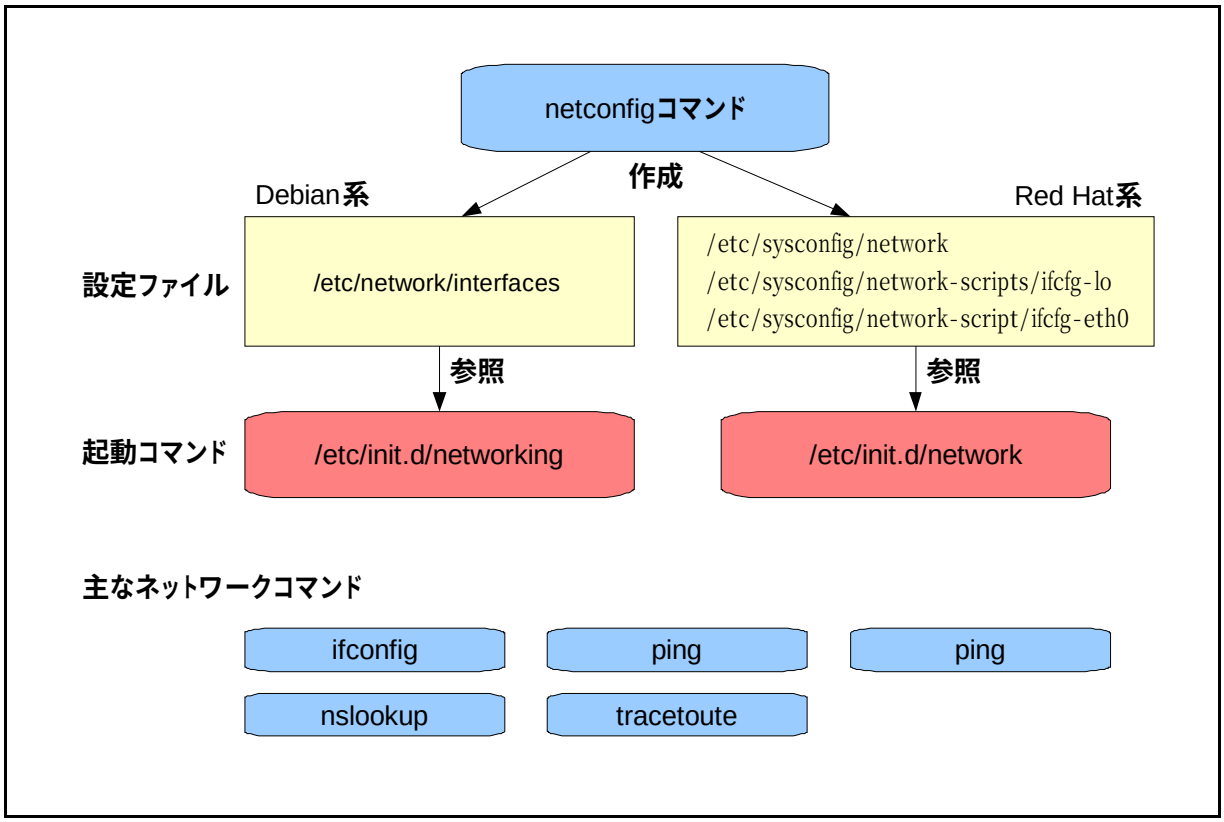

図 II-5-10.　ネットワークの設定

【解説】

1) Linux でのネットワークの設定

Linux ではネットワークに関する設定をテキストファイルに記述しておき、システム起動時に実行されるスクリプトにて設定用のコマンドを実行することにより設定を行う。
設定ファイルはディストリビューションによってネットワークの設定方法が異なっている。代表的なディストリビューションでは以下のように設定を行う。

* Debian 系ディストリビューション

  /etc/network/interfaces にネットワークインターフェイスに関する設定を記述する。

  以下のコマンドを実行することによりネットワークインターフェイスの設定が有効となる。

  # sudo /etc/init.d/networking restart

* Red Hat 系ディストリビューション

  Red Hat 系ディストリビューションでは以下のファイルにネットワークの設定を記述する

  /etc/sysconfig/network :ホスト名やデフォルトゲートウェイの設定

  /etc/sysconfig/network-scripts/ifcfg-lo :ループバックアドレスに関する設定

  /etc/sysconfig/network-script/ifcfg-eth0 :1 つ目のネットワークインターフェイスの設定

  以下のコマンドを実行することによりネットワークインターフェイスの設定が有効となる。

  # sudo /etc/init.d/network restart

設定ファイルの作成は、これらのファイルをテキストエディタにて記述するほか、netconfig コマンドを利用して対話形式で作成することができる。

2) ネットワークコマンド

ネットワークの設定や確認を行うためには以下のようなコマンドを使用する。

* ifconfig コマンド
  ネットワークインターフェイスに設定された IP アドレスやサブネットマスクなどを確認する。
* route コマンド
  デフォルトゲートウェイやルーティングテーブルの設定や確認する。
* arp コマンド
  ネットワークインターフェイスの MAC アドレスと IP アドレスの対応表の設定や確認を行う。
* ping コマンド
  引数として指定したコンピュータとネットワークで接続されているか確認する。
* host (nslookup) コマンド
  ホスト名を指定して IP アドレス等を取得し、DNS での名前解決が正しく行われるか確認する
* traceroute コマンド
  引数として指定したコンピュータまで経由するルータの一覧を表示する。