修士論文

仮想化技術を用いたクラウドサービス構築・運用のBCP

平成24年2月8日提出

指導教員 浅見 徹 教授

48106411 **呉 和賢**

情報理工学系研究科 電子情報工学専攻

概要

2011 年 3 月 11 日に宮城県沖で発生した東日本大震災は，地震・津波の直接的な被害が大きかった東北地方のみならず，東日本全域に被害を及ぼした．関東地方では, 福島第一原子力発電所が地震・津波によるダメージで運転停止に追い込まれたことや，火力発電所の障害などにより東京電力の供給可能電力が著しく低下したとともに，震災直後に計画輪番停電も実施され, 不安定な電力供給状況を前提とした対策を実施する必要性が認識された．我々はこの電力逼迫と供給能力不安定な状況に対しての具体的な対応策として，計算機仮想化技術を用いて，低電力消費量でかつ不安定な電源供給状態においても 24 時間稼働が可能なサーバシステムを設計し，実運用するに至った．我々が設計・構築した elab クラウドは，対応の迅速性が最優先課題であったため，非常時に限られたハードウェアで作られたものでありながら，構築時の工夫やその後のサーバ増設等によって，システム稼働開始後およそ一年弱安定して稼働を続けることに成功している．また，サーバシステムの運用においてはリスクマネジメントをどのように行うかが重要なポイントである．我々はこの一年間の elab クラウドの運用を通じ，仮想化技術がインシデント時のトラブルシュートに大いに有用性を発揮することを経験すると共に，VM の仮想ディスクを複製する技術がその有用性をより高めるということも議論した．本稿では，この一年間の運用経験を，elab クラウドの構成・運用ポリシーと共にベスト・カレント・プラクティスとしてまとめた．そして，2011 年 4 月の移行作業や 2012 年 1 月のメールサーバのデータ復旧などの実作業の報告，及びこれらから得られた知見として，リアクティブな対応における仮想化技術の恩恵について，また VM ライブバックアップやホットクローニングといったプロアクティブな技術の必要性及び初期検討についての報告も行っている．

目次

図目次

表目次

第 1 章

はじめに

2011 年 3 月 11 日に発生した東日本大震災は，地震・津波の直接的な被害が大きかった東北地方のみならず，東日本全域に被害を及ぼした．関東地方では, 福島第一原子力発電所が地震・津波によるダメージで運転停止に追い込まれたことなどにより東京電力の供給可能電力が著しく低下したとともに，震災直後に計画輪番停電も実施され, 不安定な電力供給状況を前提とした対策を実施する必要性が認識された．我々はこの電力逼迫と供給能力不安定な状況に対しての具体的な対応策として，計算機仮想化技術を用いて，低電力消費量でかつ不安定な電源供給状態においても 24 時間稼働が可能なサーバシステムを設計し，実運用するに至った．我々が設計・構築した elab クラウドは，対応の迅速性が最優先課題であったため，非常時に限られたハードウェアで作られたものでありながら，構築時の工夫やその後のサーバ増設等によって，システム稼働開始後およそ一年弱安定して稼働を続けることに成功している．

この elab クラウドは節電を主目的として構築したものだが，我々は一年間の運用を通じて，節電とは異なるメリットが elab クラウドによりもたらされていることを発見した．特に，運用時のリスクマネジメントにおけるリアクティブな側面，すなわちトラブルシュートにおいて仮想化技術が大いなる有用性を発揮することを発見した．同時に，VM の仮想ディスクを複製する技術がプロアクティブな運用において有用であるとの結論にも至った．本稿では，この一年間の運用経験を，2011 年 4 月の移行作業や 2012 年 1 月のメールサーバのデータ復旧などの実作業の報告も交えつつ，得られた知見をベスト・カレント・プラクティスとしてまとめ，運用における一例を示したマニュアルとして提供する．また，安定運用を保ちつつライブスナップショットやホットクローニングと言った VM の仮想ディスク複製技術を組み込むための初期検討についても報告する．

以下，本報告書の構成について述べる．二章でクラウドサービスの基盤技術である，仮想化技術ついて紹介する．三章では elab クラウドの現在の構成とその特徴について説明をおこない，現状 elab クラウドが抱えている問題点についても整理する．

四章では，elab クラウドを用いたリアクティブな事例対応について二つ紹介する．一つ目は震災後に仮想化を行った諸サービス，及びその移行作業について述べ，二つ目は 2012 年 1 月の電気系共用サーバ上でのメール消失問題およびその復旧についてである．五章では，今後の elab クラウドを考えて行ってきたストレージ周りの諸技術，およびハイパーバイザとストレージ技術の相性についての検証をまとめる．最後に六章でまとめを行う．

第 2 章

仮想化基盤技術

仮想化とは，CPU，メモリ，ストレージ，ネットワーク等の，計算機を構成するリソースを抽象化する技術である．旧来，計算機へ提供可能なこれらのリソースは物理的な物に限られていたが，仮想化技術によって，実際には単一である物理リソースを，複数の論理リソースとして計算機に提供する，などが可能になる．よって，かつて一台の物理マシンでは同時に一つの OS(Operarion System) しか動作させることができなかったが，仮想化技術を用いることで複数の OS を同じ物理マシンの上で動作させることが可能になった．物理マシン上で動作する，仮想化を実現するソフトウェアをハイパーバイザ（Hypervisor，以下 HV と略す）またはホスト OS と呼び，HV 上で動作する OS を仮想マシン（Virtual machine，以下 VM と記す）またはゲスト OS と呼ぶ．また以下，HV のインストールされた物理サーバを「HV サーバ」と記す．本節では，無償で導入できる代表的なハイパーバイザである Xen[1]，KVM[2]，VMWare ESXi[3] と，同様に無償で導入できる，仮想化環境用の統合管理 API である libvirt[4] について紹介する．

2.1 Xen

Xen はオープンソースのハイパーバイザで，Linux カーネル上で動作する．Xen は元々 1990 年代後期にケンブリッジで研究されていた Xenoserver project が発端になっており，Pratt らによって開発された [5]. 図 2.1 に Xen のアーキテクチャを示す．物理マシン上で動作する OS は図中の Dom0 に該当し，それ以外の DomU（U=1，2・・・）という OS は皆 VM である．これらの OS とハードウェアの間に Xen がハイパーバイザとして存在し，ハードウェアリソースを仮想化して Dom0 を含む OS へと提供する形となる．開発当初は準仮想化のみをサポートしていたが，ver3.02 以降では CPU の仮想化支援機能（Intel VT および AMD-V）や QEMU[6] のエミュレーション機能の支援を受けて，完全仮想化もサポートするようになった．オープンソースでありながら歴史が長く，実績が多い．開発は続いているものの，リリース版の安定性は非常に高い．セキュリティについては，Dom0 として動作する Linux の設定に依存するため，容易に柔軟かつ十分な設定が可能である．Xen の開発拠点は 2007 年に Cytrix Systems 社によって買収され，そちらから商用の Xen Server（無償），Citrix Essentials for XenServer（有償） もリリースされている．

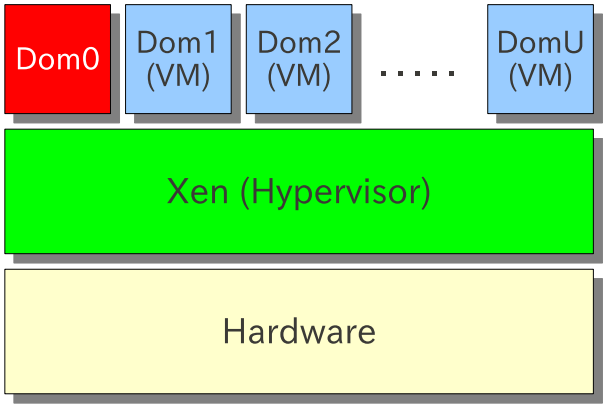

図 2.1　Xen アーキテクチャ概要

2.2　KVM

KVM(Kernel-based Virtual Machine) はオープンソースのハイパーバイザで，Linux カーネル 2.6.20 以降にマージされている．KVM は 2007 年に Qumranet 社の Kivity 等によって発表された [7]．図 2.2 に KVM のアーキテクチャを示す．KVM はハードウェア上で動作する Linux のカーネルモジュールとして動作するハイパーバイザであり，主として CPU の仮想化支援機能及び QEMU のシステムエミュレーション機能を前提として，完全仮想化をサポートする．Linux， Windows 向けに準仮想化用のドライバも開発されている．比較的最近の仮想化実現技術であり，開発活動が非常に盛んである．バージョンのアップデートに伴う機能の拡充も多いが，異なるバージョン間で VM が違う振る舞いをすることもあり，若干不安定である．セキュリティに関しては Xen 同様に，Linux に依存する形となる．

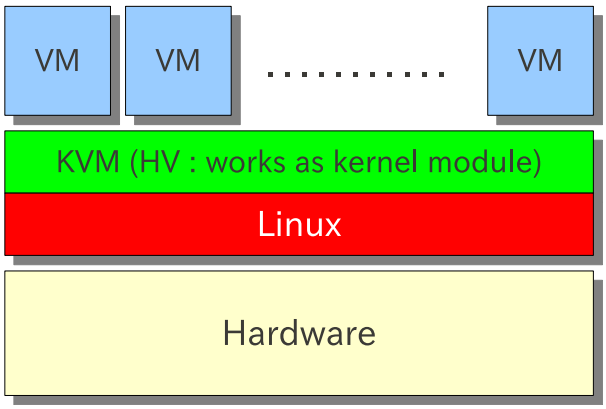

図 2.2　KVM アーキテクチャ概要

2.3 VMWare ESXi

VMware ESXi は，VMware 社から無償で提供されている，ハイパーバイザである．特定の OS の上で動くのでは無く，VMware ESXi 自身が筐体へとインストールされ，その上で各種 OS を VM としてインストール・起動・管理が可能である．VMware ESXi 自身のインストールや設定，新規 VM の作成が非常に容易なのが特徴としてあげられる一方，VMware ESXi の提供する SSH ログイン機能は非常に貧弱であり，HV の取扱いには Windows 上でのみ動作する VMware vSphere Client を用いるのが前提となっている部分があり，柔軟性が非常に低く，セキュリティ的にも不安があるのが欠点である．多機能・サポート付きの VMware ESX という有償版が存在する．

最大の難点は，HV サーバ上から直接 VM を触る機能が提供されていないことである．遠隔の別ホストからの ssh，あるいは vSphere Client 経由の VNC 接続が必要であり，いずれも VMWare ESXi サーバ・遠隔ホスト間でのネットワークが安定して提供されている事が前提となっている．

2.4 libvirt

libvirt は，仮想化環境用の統合管理 API であり，オープンソースで活発に開発が行われている．対応しているハイパーバイザの種類が多いのが特徴で，2012 年 2 月現在の最新バージョンでは，ここまでで触れてきた Xen， KVM， VMWare を含む各種ハイパーバイザに対応している．libvirt により，ハイパーバイザの種類をとわずに以下の事が可能になる．

1. VM の各種制御（起動・終了・削除・マイグレーションなど）ができる
   libvirt は virsh というコマンド体系があり，それを用いて各種制御を行う．よく使うコマンドについては付録 6 でまとめる．またマイグレーションについては複雑なので，別途付録 B で紹介する．
2. VM の各種パラメータを xml ファイルに残すことができる
   xml ファイルに記述できるパラメータには以下のようなものがある．
   - VM の名称
   - UUID
   - 割り当てる CPU の個数
   - 割り当てるメモリの容量
   - 起動オプション
   - 仮想ディスクへのパス
   - ネットワークインタフェース設定
   - 各種ドライバ
   - VNC・シリアル・キーボード等のコンソール設定

   これらを libvirt を用いずに管理する場合，Xen や KVM では VM の起動時に同時にオプションをつけて渡すこととなり，大変煩雑となる．

libvirt はその使用において，注意すべき点が二つある．まず一つ目は，libvirt は API に過ぎないため，libvirt 側でコマンドが発行されたからとて，実際に HV 環境で動作するとは限らない．特にマイグレーションについては，VM の仮想ディスクが HV サーバ間で共有されていないとマイグレーションできないため，構成を確認せずにコマンドを発行するとエラーが起きてしまい，次に，開発が活発な分，バージョンによって違いが多い点がある．コマンドのオプションが増えていたり，help が充実していたり，xml の表記法が増えていたりと，色々と違いがある．libvirt は基本的に HV インストール時に，付帯環境として一緒に付いてくることが多いが，その時の OS， HV の組み合わせによって libvirt のバージョンは全く異なったものになってくることが多い．特に HV 環境の移行の際には注意を払う必要がある．

第 3 章

elab クラウドの現状

3.1 構成

3.1.1 HV サーバ

本節では，elab クラウドの構成について述べる．まず，HV サーバの環境について説明する．

筐体：HP Proriant DL120 G6
CPU：Intel(R) Xeon(R) CPU X3430 2.40GHz 4 コア
メモリ：8GB

このようなハードウェアスペックの 1U サーバ 4 台が 4 月当初からあり，うち 3 台を CentOS5.6 + Xen3.4.3，残り 1 台を VMware ESXi の構成で，それぞれ xenhv01， xenhv02，xenhv03，esx01 として運用していた．これら 4 台は東日本大震災以前から江崎研究室に実験用としてあった機材を震災を機に HV サーバに回したものであり，江崎研の基幹サーバ群と電気系共用サーバ群を収容するのでスペック的には限界であった．

そこで，6 月に一台 HV サーバを増設した．スペックは下記である．このサーバも CentOS5.6 + Xen3.4.3 で運用しており，名前を xenhv04 とした．

筐体：HP Proriant DL120 G7
CPU：Intel(R) Xeon(R) CPU E31220 @ 3.10GHz 4 コア
メモリ：16GB

Xen ハイパーバイザのストレージ構成について表 3.1 に示す．これらの筐体はいずれも SATA HDD が 4 つ挿入できる．それに加え，外部ディスクに iscsi ストレージがあり，HV 間で共有されている．

それぞれ，スロット 1 には数百 GB 程度のそれほど容量の大きくない HDD がマウントされており，いずれもこの小さいディスクに OS および HV をインストールしてある．各 HV サーバとも，スロット 2, 3 にマウントされてある 2TB の HDD を VM の仮想ディスクのプールとして利用しており，HV を入れ替えたい時あ

るいは別筐体に VM を移したい時など，HDD を丸ごと抜き，別筐体にマウントすることで容易に移行・変更ができるような物理構成にしてある．iscsi は 5TB という容量の LVM として各 HV からは見えており，VM 毎に論理ボリュームを切り分ける形で用いている．

スロット / HV	SATA1	SATA2	SATA3	SATA4	外部
xenhv01	160GB	2TB	-	-	5TB (iscsi)
xenhv02	160GB	2TB	-	-	
xenhv03	160GB	2TB	2TB	-	
xenhv04	250GB	2TB	2TB	-	

表 3.1　Xen ハイパーバイザのストレージ構成

1 台だけハイパーバイザの種類を分けたのはソフトウェア構成を冗長にするためであり，VMWare ESXi の HV サーバ上では主としてバックアップ・セカンダリ系のサービスを運用している．

メインのハイパーバイザとして Xen を採用した理由については，その安定性が上げられる．我々はセキュリティ的観点やネットワーク設定，運用のしやすさから Unix 系の OS 上で動作するハイパーバイザを望んでおり，候補は Xen と KVM の二択であった．KVM は現在開発が非常に盛んで，バージョンの更新が頻繁である分，安定性に欠ける面がある．KVM のバージョンを上げるために VM を止める必要があったり，KVM のバージョンが異なるハイパーバイザ間でライブマイグレーションを行うと失敗したりと，24 時間安定運用したいサービスを載せるハイパーバイザとしては若干の不安がある．一方で Xen は開発は続いているもののリリースは安定しており，その点で安心した運用を行う事ができる．

HV サーバの OS として Cent OS を採用したことも同様に「安定している」という理由があり，同じ Linux でも Debian や Ubuntu と比べてパッケージ更新に伴う本体再起動の機会は少ない．また，Cent OS 自体が Xen との相性がよく，筐体への Cent OS インストール時のソフトウェア選択で Virtualization を選ぶ事で，容易に Xen 環境を整えることができるのも大きなメリットである．

3.1.2　VM

次に，VM について説明する．現在，elab クラウド上で動いている VM は合計 31 台である．その 31 台について表にまとめたものが表 3.2 である．行に現在乗っている HV サーバを，列に VM の所属を取った．VM の所属について簡単に定義する．

- 電気系サーバ：電気系共用の web，メールサービスが運用されている VM 群.
- 江崎研基幹サーバ：江崎研究室のネットワークのインフラ部分を構成する VM 群.
- 江崎研個人サーバ：江崎研のメンバー一人一人に割り当てられる VM 群．オーナーの名が付く.
- その他：江崎研ネットワークに属するものの，共用・基幹部分ではないサーバ．一部メンバーにより運用されているものであったり，研究プロジェクトの物であったりする.

所属 / HV	電気系サーバ	江崎研基幹サーバ	江崎研個人サーバ	その他
xenhv01	bell, **maxwell**, fleming	**lupin**, ggw, **ns**, **www**	motty, secretary	-
xenhv02	**tesla**	**auth**, **mail**	jo2lxq, gucchi	fiap-mgmt, mozilla
xenhv03	-	-	panda	-
xenhv04	-	-	ace, dk, gashi, kinoshita, kohaku, kure, shou, take, tritm, tsuchy	**ictepc**
esxhv01	-	elab-gw, cc, mail2	-	-

表 3.2　elab クラウド上のアクティブ VM 一覧

表 3.2 中のうち，太い線で示された VM は iscsi ディスクに保存されているものであり，他のものは HV のローカルディスク上で動いている VM である．iscsi ディスクとローカルディスクの違いを簡単に纏める．

ローカルディスク：
ローカルディスク上のイメージファイルに格納されている．特に RAID 等を用いていないため，データは保護されない．i/o は早く，別のディスク上の VM に影響を与えない．

iscsi ディスク：
RAID5 で動いてるため，データが保護されるが，i/o は重い．5T という大きな RAID ディスクを細かく LVM で分けて各 VM のディスクとしている構成上，iscsi 上のある VM が i/o のせいで重くなると，他の iscsi 上のディスクも影響を受けやすい．また，HV 間での VM ライブマイグレーションが可能．ある HV で，他の VM が激しい計算を行って CPU 負荷が高まった時や，その HV を一度再起動したい時など，その VM を無停止で別の HV に移すことができる．

3.1.3　運用ポリシー

■VM 作成ポリシー：　どの VM をどの HV サーバ・どのストレージに格納するか，については，基本的に下記ポリシーに従って決定される．

- xenhv03 はバックアップ用の Xen．とっさにマイグレーションさせたい時，とっさに VM を作りたい時等に使われるため，基本的にはリソースを空けておく．
- esxhv01 は Xen に対するバックアップとしての VMWare sxi なので，主にセカンダリサービス系の VM が格納されている．

- 個人用 VM は基幹サービス VM と分離.
  個人用の VM は用途が人それぞれで，場合によっては走らせるプログラムのミスなどで，急激に計算資源を使用したり，ディスク I/O を大量に発行したりする．そのような際に基幹サーバ VM に影響を与えないよう，基本的に xenhv04 に集められており，xenhv01, 02, 03 で基幹サーバ VM と共存しているものも，ディスク I/O が影響を与えないようにローカルディスクに集められている．
- 基幹サーバは基本的に iscsi 上
  RAID によるデータ保護もあり，かついざという時にマイグレーションで別 HV へ移動させることも可能である．
  例外は bell で，これは重すぎるのでローカルディスク上で動かしている．
- サービスにプライマリとセカンダリがある場合は別 HV， 別ディスクで運用し，冗長性を高める構成にする．表 3.2 における，
  - mail， mail2
  - ns， elab-gw
  - bell， tesla

  がそれらに該当する．

■HV サーバの権限について　HV サーバにログイン権限があるのは江崎研全体でも一握りのメンバーである．但し，VM の起動・停止・VNC 接続等がそれぞれのユーザの任意にできないのは不便であるし，その度にいちいち管理メンバーに連絡して実行してもらうのは時間的にも労力的にもコストが高いので，libvirt 経由で API を取得して，web ページ経由で触れるようにしてある．また，クラウド全体でリソースに余裕はそれほど無いので，VM の新規作成は管理メンバーに申請する形を取っている．

3.2　課題

3.2.1　安定運用のトレードオフ

■メリット： 意図的に安定側に比重を傾けて構成をした甲斐があり，非常に安定している．4 月の運用スタート以来，クラウドサービスが停止したのは，9 月の法定停電における半日程度である．パッケージも最小構成で HV サーバを作っているので，脆弱性が見つかることも少なく，パッケージのアップデートも少ない．2011 年年末には，あるバージョン以降の Linux のカーネルに，200 日間無停止で運転すると強制的に停止するバグが発見されたが，CentOS のカーネル 2.6.18 は該当の物より古く，安定性を重視しているという構成を裏付ける形となった．

■デメリット： 安定側に倒してしまっている分，他仮想環境技術で提供されているような，挑戦的な機能と縁遠い.
例えば KVM では，デフォルトで提供されているストレージマイグレーションがない．また，Xen3 という少々古い仮想化環境なので動作するドライバが少なかったり，qcow2 を用いて差分ディスクの作成・スナップショットの取得等ができない．但しこれらの技術は機能として使えると便利なのは確かだが，安定して動くか

どうかは多いに議論の余地があり，実際に使うにしても，それぞれ単体で評価検討を暫く実行したい段階である．

また，CentOS5 という若干古い OS なので，新規技術の導入コストが高いのも難点である．これについては，後に 5 で詳しく見ていく．

3.2.2　HV の構成の今後について

3.1.1 で触れたように，xenhv01〜04 に搭載されているメモリ量はそれぞれ 8GB, 8GB, 8GB, 16GB である．そのうち，現在アクティブな 31 台のメモリ使用量の合計を HV ごとに取ると，6.5GB, 6GB, 2GB, 13GB となる．我々の運用では 1VM につきメモリ 1GB を基本としており，また HV 用にもメモリを 1〜2GB は確保させたいところである．この基準で考えると，現在の elab クラウドはメモリにほとんど余裕が無く，新しく VM を立てることがほぼできない．予備の xenhv03 は若干余裕があるが，これも VM を 3〜4 個新しく作ると限界になってしまう．各 VM に割り当てているメモリ量は VM 無停止で変更可能なので，減らすという選択枝も可能なのだが，実は既にメモリ量 512MB で稼働している VM も数個ある．一方，空きディスクには大変余裕がある．表 3.1 にも示されている通り，そもそも SATA スロットが合計で 6 つ空いている．また，iscsi ストレージも物理的にまだ増設可能である．以下，今後の elab クラウドの増設についてのシナリオ一案を示す．

1. 新規 HV サーバの購入．メモリが多く積めるといい．また，同時に既存 HV サーバ用の増設メモリも購入する．
2. 新サーバに一旦，可能な限りの VM を集約する．
3. 既存サーバをラックから降ろし，メモリを増設する．
4. HV サーバを一通り物理的にマウントし直したところで VM の再配置について検討を加える．HV ごとの役割分担を明確にするという案が一案である．例えば，
   - 電気系基幹 HV,
   - 江崎研サービス HV
   - セカンダリサービス VM の動く HV
   - 存在自体がバックアップな HV
   - 個人用 VM + 実験用 VM の動作する HV

   のような分け方が考えられる．

第 4 章

elab クラウド活用事例

本章では，この一年を通じて我々が elab クラウドを多いに活用した事例を二例，紹介する．一つ目は，2011 年 4 月上旬に行った elab クラウドの構築及び電気系共用サーバ群・江崎研基幹サーバ群の Physical to Virtual（以下，P2V と表記する）である．まず P2V の技術的方法について説明したのち，実ホスト名を交えて実際の移行作業について説明する．

二つ目は，2012 年 1 月中旬に発生した，電気系メールサーバでのサーバ管理 TA のオペレーションミスに起因するメールデータ消失とその復旧についてである．オペレーションミスを起こしたのが 1 月 18 日未明，そこからバックアップを 1 月 17 日分へロールバックさせてデータを復旧させるのに約五日を要したが，この作業において仮想化の大きなメリットを実感した．また，これら二件の実例より，クラウドサービスに備わっていると助けとなる機能についても考察を行う．

4.1 P2V 作業について

4.1.1 P2V とは

本節では，物理マシンで動いているサービスの仮想化の一手段として，P2V を紹介する．P2V とは，物理マシンその物を丸ごと仮想化する手法である．他にサービスの仮想マシンへ移行の手段としては，新たに仮想マシンをインストールし，物理マシンから必要な設定ファイルをコピーし再設定すると言う方法があるが，こちらは OS のバージョンの不整合や，コピー忘れなど，特に元の物理マシンの構成が複雑であればあるほど，上手く行かないリスクが大きくなる．その点，P2V を用いれば，安定して運用できているサービスの載ったサーバをそのまま仮想化できるので，再設定のコスト，前述のリスクを小さくする事ができる．

以下，P2V の手法を三つ紹介するが，これらはハイパーバイザとして，Xen，KVM を対象としている．これらのハイパーバイザは，通常の OS 用の物理デバイスをリソースとして VM にそのまま提供できる機能を持っている．

■物理ディスクからの直接起動： 仮想化したい物理マシンからデータディスクを抜きだし，SATA ケーブル，IDE ケーブル，USB-SATA 変換ケーブル等で HV サーバと接続する．この時，ディスクは（HV サーバの構成によるが）例えば/dev/sdb といった形で認識される．この path を VM のディスク位置として定義すると，

その VM は物理マシンほぼそのままとして起動可能である．この手法の最も大きな利点は，移行したいサービスのダウンタイムが 10 分程度で済む所である．一方で欠点は少なくない．まず，HV サーバの SATA，USB 等のスロットの数だけしか VM を作る事が出来ない．また，USB アタッチであった場合は概ねサーバルームに於いて，非常に不安定な形でディスクが置かれている事になる．また，SATA・IDE アタッチなどで，HV サーバのそれらのスロットがホットプラグに対応して居ない場合，一つの物理マシンを P2V を行うために HV サーバの再起動を伴うことになり，その HV サーバ上で動いている VM も一時停止する事になる．最後に，この手法は物理マシンのディスク構成が RAID であった場合には適応不可能である．なお，この欠点は下の手法とも共通している．

■ディスクコピー（物理接続）： 直接起動手法と同様に，仮想化したい物理マシンのディスクを HV サーバと接続する．その後，dd コマンドを用いて物理ディスクをそのまま HV サーバ上にイメージファイルとしてコピーする．コピー終了後，ディスクを撤去し，イメージファイルから VM を起動する．これにより，直接起動手法で存在したスロット数の問題は解決される．欠点としてはまず，物理マシンのディスクの使用量に関わらずディスク全体をコピーするため，ディスクの使用効率の良し悪しがそのまま反映される．また，コピー中はサービスダウンが不可避であり，ダウンタイムはディスクの容量に比例する．我々の経験では，HV サーバの性能や物理ディスクの世代によるが，コピー速度は約 25MBytes/s 程度で安定する．この速度であると，1TByte のディスクのコピーに 10 時間程度かかる計算になる．また，この手法も直接起動手法同様に，物理マシンのディスク構成が RAID であった場合には適応不可能である．

■ディスクコピー（ネットワーク経由）： 移行したい物理マシンからディスクを外さずに，Live CD あるいはシングルユーザモードで起動し，移行したいディスクをリードオンリーでマウントする．そして，ssh で HV サーバとセッションを張りつつ dd コマンドでディスクをイメージファイルとしてコピーする．VM 起動前に，物理マシンをちゃんとシャットダウンしておくことが重要である．移行したい物理マシンがハードウェア RAID 構成であった場合，あるいはディスクインタフェースが特殊であった場合等は物理マシン側でブートし，OS で認識可能な形のディスクをコピーする，というこのような方式を取る事で，HV サーバ側では問題なくイメージファイルから VM として起動可能である．ソフトウェア RAID 構成であった場合は，本方式同様にコピーしてから，VM に残っているソフトウェア RAID 設定を解除することが推奨される．詳細については，本報告書では述べない．

速度については，移行元が十分に新しいハードウェアであり，また経路上のネットワークが十分に早くフィルタもかかっていなければ，ボトルネックはディスクの I/O となるので 4.1.1 の手法と同等のコピー速度が得られる．しかし，移行元の CPU・メモリなどのリソースが貧弱であれば，それだけ dd の実行が遅くなるのでコピー時間は増加する．我々が経験した所では，CPU:500MHz・メモリ:128MB・ディスク:120G の物理マシンをこの手法でコピーしようとした所，1MByte/sec 超のスピードしか出ず，20 時間以上かかる計算になってしまった．その際は，中断して物理接続ディスクコピー手法で改めてコピーした所，20MBytes/sec 程度のスピードが出て，無事一時間前後の短時間で移行を完了した．

■P2V の手法比較： 表 4.1 に，P2V の手法を比較してまとめた．物理ディスク直接起動の場合，元のマシンから HDD を取りだし，移行先のマシンに接続するだけなので，サービス停止時間は短くて済むが，ディス

クコピーの場合はコピーの時間だけサービス停止時間がかかってしまう．その長さはディスクの容量に比例すると共に，ディスクの世代が古かったり，RAID であったり，コピー元の計算能力が低かったりするとどんどん長くなっていく．

手法	サービス停止時間	ボトルネック	RAID 対応	作業の容易さ
物理ディスク直接起動	短 (数十分程度)	物理接続インタフェースの個数	×	○
ディスクコピー（直接接続）	長（数時間から数十時間）	コピー時間	×	◎
ディスクコピー（ネットワーク経由）	長（数時間から数十時間）	コピー時間，ネットワーク，コピー元の計算能力	○	△

表 4.1　P2V 手法の比較

4.1.2　実際の仮想化作業について

■仮想化前の状態　本節では，我々が実際に仮想化した電気系共用サーバ及び江崎研究室のサーバと，それらが提供していたサービスについて紹介する．また，P2V を行ったサーバについては，そのディスクサイズも付加する．

■電気系共用サーバ

1. bell.ee.t.u-tokyo.ac.jp：250GB
   メールサーバ，及び Web サーバ．電子情報工学科・電気電子工学科，電子電気系事務室，GCOE の web ページなどが載っている．
2. maxwell.ee.t.u-tokyo.ac.jp：37.6GB
   ee.t.u-tokyo.ac.jp 以下のゾーンの DNS のマスターレコードを持つサーバ．
3. tesla.ee.t.u-tokyo.ac.jp：173GB
   二号館の電気系事務室，電気系同窓会室，GCOE オフィスなどに NAT を提供している．bell の提供しているサービスのバックアップも行う．

これら三台は，P2V を行って仮想化した．物理マシンで動いている環境（特に bell）を，できる限りそのまま仮想化したかったという背景がある．

■江崎研究室のサーバ

1. lupin.hongo.wide.ad.jp：173GB
   二号館に無線 LAN によるインターネット接続，SSID:build2-guest を提供している．
2. mozilla.hongo.wide.ad.jp：466GB

mozilla のミラーサイト．

3. mail.hongo.wide.ad.jp
   江崎研究室の基幹メールサーバ．
4. xlab.hongo.wide.da.jp
   江崎研究室のセカンダリメールサーバ．

lupin と mozilla は P2V を行って仮想化したが，mail と xlab は別に新しくサーバを立て，必要なファイルだけコピーするという方法を取った．mail と xlab は物理マシンのディスク構成・ソフトウェア構成などに不満があり，これを機にシステムの再構築を行った．

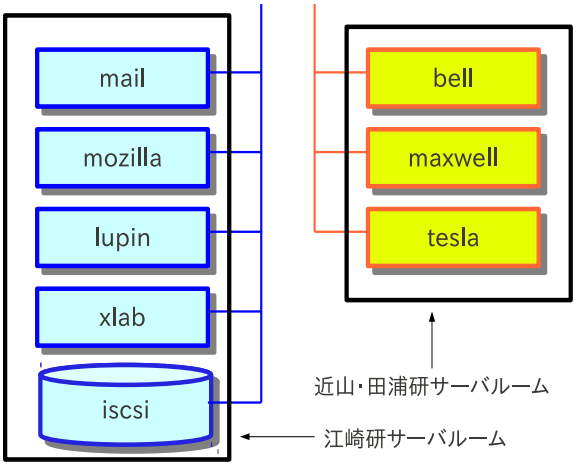

図 4.1　サービス構成図（仮想化前）

4.1.3　実移行作業とその時苦労した点

■仮想化前後の構成　図 4.1 に仮想化前のシステム模式図を示す．青い四角形・線が江崎研究室のサーバ・ネットワークであり，黄色い四角形・線が電気系共用サーバ・ネットワークである．元は二号館内の別々の部屋に置かれていた．図 4.2 に仮想化後のシステム模式図を示す．赤い四角が HV サーバであり，青・黄色については図 4.1 に準ずる．破線で囲まれた四角は VM を意味し，その VM の下にある HV 上で動作していることを意味している．仮想化前は物理的に離れた部屋で提供されていたサービスを一部屋にまとめた事が分かる．

仮想化後の構成における工夫は以下である．

1. 各 HV サーバが複数のネットワークをまとめて取扱い，それぞれの VM へ適したネットワークのみを渡す構成をとっている．これによりネットワーク設定は HV サーバの Linux 上で基本的に完結し，HV，あるいは VM の設定が煩雑にならないようにしている．
2. 重いアクセスを伴う bell と mail を別々の HV に載せ，負荷分散を行っている．
3. tesla が bell の，mail2 が mail のバックアップを行っている．これらが同時に落ちてしまう事の無いよ

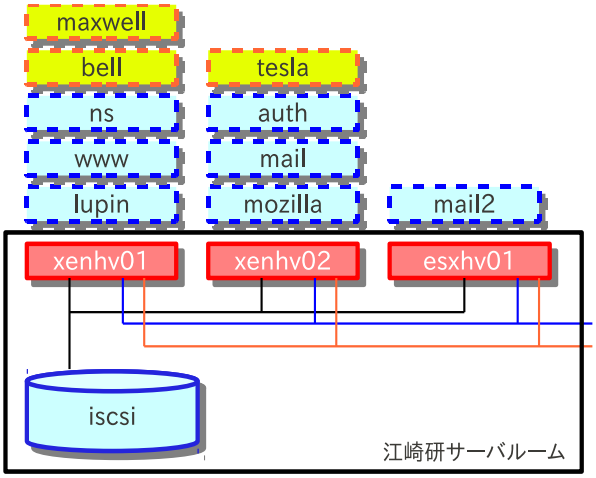

図 4.2　サービス構成図（仮想化後）

うな構成にしている．それぞれが別々の HV サーバ上で動くのは勿論のことであるが，片方が HV サーバのローカルディスク上に，もう片方が iscsi 接続の RAID ディスク上にイメージファイルを置くようにしている．

■P2V と新規インストールについて　作業前は，P2V が十分に動作すると我々は期待していなかった．また，当初は P2V が必須なほど設定・ファイルの込み入ったサーバは bell だけであり，P2V が機能しない場合それ以外のサーバは新規インストール＋必要なファイルのコピー，という手段で仮想化予定だった．

作業を進めていくうちに，P2V はコピー時間以外のコストがほとんどかからない事が判明した．具体的には，VM 起動時に，HV 側で UUID とネットワークインタフェースの MAC アドレスが重複していないことを確認すれば，問題なく起動する．VM 側では，物理マシンだった頃と MAC アドレスが変わってしまっているからネットワークインタフェースの番号付けがおかしくなるが，BSD，Linux どちらの環境でも少しの設定変更で正常に戻す事が可能．我々の設定の都合上，VM 自体が意識して VLAN を扱うことが無くなるので，その面でも DHCP や NAT を提供している VM の設定の変更も必要だが，これも該当設定ファイルのネットワークインタフェース名称変更程度で済む．

一方で，仮想マシン新規インストール＋ファイルコピーも，例えばメールサーバであると小さいファイルが大量にある形になるので，ランダムアクセスでコピーをすると，シーケンシャルアクセスでディスクを丸ごとコピーするのと同等の時間がかかることも判明．結果的に，想定より多くのサーバを P2V を行って仮想化した．

■P2V に要した時間　P2V にあたり，まず全ての物理マシンを HV サーバのローカルディスクへとコピーを行った．P2V を行った全ディスク合計で約 1.1TB であり，これだけで述べ 11 時間近くコピーに時間を要した．

その後，bell，mozilla 以外のサーバは，iscsi 接続の RAID ディスクへとコピーしたのだが，同時に複数の

コピーを行ったり，コピー中に iscsiRAID ディスク上で起動している VM が fsck を始めたりしてしまうと，RAID コントローラのキャッシュがパンクしてフリーズしてしまう事態が多発した．RAID コントローラのライトキャッシュ機能を切ることでフリーズは回避されるようになったが，コピーには非常に時間を要するようになった．大容量の bell と mozilla を除いて 380GB 程度しか無いが，これら全てを RAID ディスク上へ移行するのに 15 時間以上費やした．

4.2　電気系サーバのメールデータ消失とその復旧

4.2.1　概要

以下，時系列で起こったでき事について触れていく．

■1/18

- 1 月 18 日 午前 5 時頃，電気系サーバ管理 TA の一人がオペレーションミスでメールデータの一部を削除してしまう．
- 1 月 18 日 午前 6 時頃，同 TA が定期バックアップスクリプトを kill してしまう，
- 1 月 18 日 午前 7 時頃，postfix を停止し，extundelete という復元ツールを試し始める，
- 1 月 18 日 午後 1 時頃，復元ツールが終了して，ある程度復旧する．（全部は直っていない）
- 1 月 18 日 午後 6 時頃，bell を停止して，VM のスナップショット取得を開始する．
- 1 月 18 日 午後 9 時頃，スナップショット取得完了．復元ツールで直った分のデータのマージ作業開始．
- 1 月 19 日 午前 0 時頃，メールサービスを再開．残りのデータ復旧はひとまず翌日に回す．

■1/19　前日，オペレーションミス直後に停止させてしまい，誤動作していたバックアップツールを正常に戻す目処が立つ．電気系サーバ管理 TA では，rdiff-backup という差分バックアップツールを用い，メールデータ・web データ・設定ファイル等をディレクトリ単位で毎日の差分バックアップを作成している．スクリプトが走っている途中に強制停止したせいで，1/18 日分のデータが中途半端にバックアップされ，差分が適切に作成されなかったのが原因となったようだ．

そこで，容量の小さいディレクトリに対して，1/19 日分の差分バックアップを改めて作成し，そこから 1/17 分のデータへとロールバックが可能であることを確認した．メールデータ本体に対する再差分バックアップ取得＋ロールバックを開始したが，20GB 以上の分量になるので，完了したのは翌日となった．

■1/20　バックアップツールのロールバックを完了し，1/17 時点でのバックアップが取得できた．ここで，復旧に向けての具体的な作業方針が立った．まず，この段階で我々が保持できていたデータについて整理をする．まず，データの概要図を図 4.3 に示す．横に時間軸を取り，縦にユーザを取ったものである．

1. bell のスナップショット＝ rm 後の/var/mail = 図 4.3 の b + c + e + f の領域に該当
2. extundelete による復旧結果：図 4.3 の b + e の領域に該当
3. 1/17 日の/var/mail：図 4.3 の a + b + c の領域に該当
4. 一週間分の SMTP ログファイル：図 4.3 の d + e + f の領域の情報を取得可能

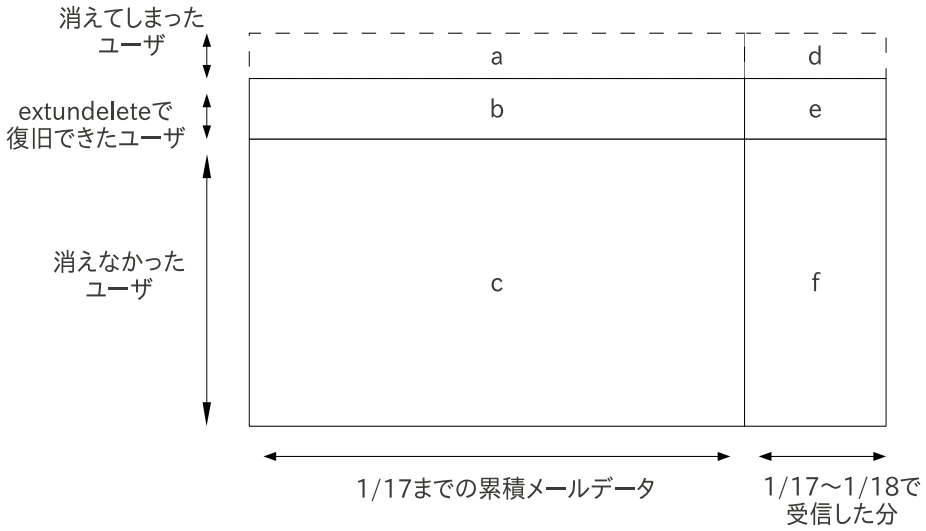

図 4.3　電気系メールサーバのデータ概要

以上を踏まえて，復旧作業の方針は以下となった．

1. マージ作業
   バックアップ分から，消えてしまった部分を特定し，それを現行のメールボックスにマージする作業．消去してしまい，バックアップに含まれる部分は図 4.3 の領域 a に該当するから，上データ群の 1. と 3. を比較し，3. にのみ含まれているデータを，適切なメールボックスの形式として切り出す．マージ作業を行うにあたりメールサーバの再度停止が必須であり，メールサーバの停止には事前のアナウンスが必須となる．ここで，メールサーバの停止及びマージ作業の日時を 1/22 と定めた．
2. アナウンス作業
   バックアップ作成日時より後に受信し，extundelete で復元もできなかったメールについて，メールサービス使用者についてアナウンスを行う作業．消去してしまい，バックアップに含まれていない部分は 4.3 の領域 d に該当する．上データ群の 4. から，バックアップ日時以降の受信メールの
   - 受信元
   - 送信先
   - メッセージ ID（含まれていれば）

   のリストが取得できる．そのリストと，上データ群の 1. を比較し，1. に含まれていないものが消えてしまったメールであり，その各メールの情報についてユーザに通告を行う．

■1/21　終日，マージ作業及びアナウンス作業について試行錯誤を繰り返す．マージ作業については，厳密には，消してしまってから新しくメールを受信してファイルが作成されたユーザが居るため，実際の処理としてはもっと複雑で，各メールファイルの内部まで入って差分を取る必要があった．具体的な方法については，本稿では割愛する．

■1/22　マージ作業の本番を行った．まず，メールサービスを停止し，改めて/var/mail の状態を保存した．そののち，予め切り出しておいた差分をマージし，メールサービスを再開して，問題なく送受信が行われることを確認した．なお，アナウンス作業についても翌週中に完了した．

4.2.2　考察

■偽陽性について　上マージ作業，アナウンス作業については，データ群 1. に含まれていないものが対象であると定義し，実際にそのように作業を行った．

実はこれは正確な定義ではない．上の定義では，具体的には，「pop でメールデータをサーバに残さない設定にしている人で，ダウンロード済みのユーザ」と，「.forward を設定しており，そもそも bell にメールデータを残していないユーザ」の二種類のユーザも，「オペレーションミスで消してしまい，復旧できなかったユーザ」と区別がつかず，同様に対象となってしまった．

これは言わば false positive な検出なのだが，これで発生するデメリットは「既に受信されているメールが再度受信された（マージ作業）or 消えてしまったいう連絡が来た（アナウンス作業)」である．重要なメールが消えている可能性もあり，復旧については可能な限り早く行う必要があったため，この false positive については速度と正確さのトレードオフを TA 内で検討した結果，「人によっては既に受け取っているメールを多重に受信する形になるかもしれません」という形でユーザに通告を行い，速さを優先して作業を進めるという結論に達した．

■仮想化の恩恵

- bell のスナップショットが簡単に取得できた
  4.1.2 で触れたように，bell のディスク容量は 250GB 程度である．仮想化しておいたので，比較的新しい筐体上での通常のファイルコピーという形でスナップショットを取得できた．これは凡そ 3 時間程度であった．
- スナップショットをすぐに VM として立ち上げられる
  3.1.3 で述べたように，xenhv03 は予備用の HV として，日頃からリソースに余裕を持たせて運用していた．そのため，取得したスナップショットを容易に新たな VM として起動させ，消してしまった後の状態を作業環境として再現することが容易にできた．
- バックアップを展開するためのディスクが確保できる
  rdiff-backup は bell の差分バックアップを tesla に作成するツールである．4.2.1 で述べたように，bell の/var/mail は 20G 以上の分量であり，別日時のロールバック分を展開しようと思うと，その分の容量が別途必要になる．今回は xenhv02 上のディスクを 100G ファイルとして確保し，tesla に新たに外部ディスクとして付け足す形で追加した．
- rdiff-backup について
  そもそも，電気系サーバのバックアップとして rdiff-backup を使用し始めたのは，P2V 以降，2011 年 6 月頃からである．それ以前はディスク容量が足りなかったためバックアップしてる領域は今より少なく，また差分バックアップでも無く，一日分を tar で固めただけであった．仮想化によって計算資源・

ディスク資源に余裕ができていたからこそ，差分バックアップが作成できていたのである．

4.3　実作業を通して得られた知見

4.3.1　移行作業についての考察

移行のボトルネックはディスクコピーにかかる時間と, 設定の変更にかかるコストが上げられ, 実際にはこの二つのトレードオフを考えて P2V を行うか新規インストールするかを選択する．

■新規インストールが優れる場合：　設定の移行・変更のコストが小さく, また必要なファイルコピーの量も少なく, ディスクコピーにかかる時間が勿体ない or 十分に小さい場合．一般的に,DNS サーバやバックアップ・セカンダリ系のサーバはこちらに該当すると思われる．内部向けの wiki ページ程度しか載っていない web サーバもこちらで移行するコストは小さい．

■P2V が優れる場合：　設定の移行・変更のコストが大きい, あるいはファイルコピーの量が多い場合は,P2V を用いてディスクを丸ごとコピーした方がよいと考える．メールサーバや, ユーザが多く個々人が自由にファイルをアップロードしているサーバ, 外部向けのコンテンツファイルが大量に載っているサーバは, こちらに該当する．ただし, 物理マシンのソフトウェア構成に問題点がある場合は P2V するとそのまま残るので注意が必要．

4.3.2　ホットクローニングについて

P2V は, 物理マシンの環境をそのまま仮想化できるので, 問題なく動いているサービスの移行方法としては非常に優れている．欠点は, サービスのダウンタイムがディスクサイズに比例する点である．これは, ディスクのコピーとサービスの提供が同時にできないことに起因している．一番の問題は, コピー中にコピー元のディスクに書き込みが行われてしまうと, コピー前後の再現性が保証できなくなってしまう点である．そこでこの点をクリアして, 実環境で使用可能な, サービスを極力落とさずに仮想化する，ホットクローニング技術があればよいことになる．現在, ダウンタイムを最小にする方法が, 仮想化したい物理ディスクを HV サーバに接続し, それをそのままディスクと見て VM を起動する方法であることは表 4.1 で述べた．この状態から，HV サーバ上のイメージファイルから起動している状態へストレージマイグレーションが実行できればいいという事になる．

4.3.3　ライブスナップショットについて

4.2 で触れたように，我々は電気系メールサーバのデータ復旧に際し，二度バックアップを取得したが，どちらについてもサービスを停止せざるをえなかった．この時は他に選択枝も無かったが，そもそも重要な VM に対して，ライブスナップショットのようなものが取れていれば…と思うことは多い．それは，差分である必要は必ずしもない．先にも述べたように現在の elab クラウドにおいてストレージ量はボトルネックになっていないため，重要なデータの格納された VM を選んで，一週間限定等，範囲を調整すれば，フルスナップショッ

トでも問題はないと考えられる.

第 5 章

ハイパーバイザ及びストレージ技術検証

5.1 検証概要

本章では，3.1 にて触れた elab クラウドの現状と 4.3 で得てきたような問題意識を鑑み，今後の elab クラウドの拡充についてどの HV を仮想化環境として用いるか，どのようなストレージ技術を導入するかについて検討を行う．

5.2 検証環境

検証環境として，OS トリプルブートのデスクトップマシン 2 台を用意した．トリプルブートとしたのは，同一ハードウェア環境での HV の差異を比較する目的である．HV として用意した三種の環境は以下である．

- CentOS5.7 + Xen3.4.3：現行の elab クラウドに近い構成
- Debian + Xen4.0.1：Xen でありつつ，パッケージ等が柔軟な Debian
- Debian + KVM：同じく Debian で KVM

デスクトップを二台用意したのは，別物理マシン間でのマイグレーション等のテストも行うからである．また，この二台のトリプルブート構成は同一である．以下，それぞれのハードウェアスペック及びトリプルブート構成について列挙する．

■ハードウェアスペック：

筐体：FMV ESPRIMO D5270
CPU：Intel(R) Core(TM)2 Duo CPU E8500 @ 3.16GHz
メモリ：4GB
HDD：2TB

筐体：FMV ESPRIMO D5290
CPU：Intel(R) Core(TM)2 Duo CPU E8400 @ 3.00GHz
メモリ：4GB
HDD：2TB

■トリプルブート構成：

- /dev/sda1 … /boot：CentOS + Xen3.4.3 (100MB)
- /dev/sda2 … /boot：Debian + Xen4.0.1 (100MB)
- /dev/sda3 … /boot：Debian + KVM (100MB)
- /dev/sda4 … LVM
  - /：CentOS + Xen3.4.3 (100GB)
  - /：Debian + Xen4.0.1 (100GB)
  - /：Debian + KVM (100GB)
  - 残り（1.6TB 強）VM 用ディスクスペース + スワップ領域

5.3 検証内容

検証の対象とするのは，xNBD[8] と ltdd[9] という二つのストレージ技術である．これらは共に Network Block Device(以下，NBD とする) を元として開発されたものであるため，まず NBD についても紹介を行い，ついで xNBD と ltdd についても説明を行う．そののちに評価軸，検証シナリオ，検証に使用する VM について概観する．

5.3.1 Network Block Device

NBD はネットワーク越しに他ホストのファイル or ブロックデバイスを，ローカルにあるブロックデバイスのように扱えるカーネルモジュールであり，ファイル or ブロックデバイスを出す側と使う側のサーバ・クライアントモデルで動作する．NBD を用いることで，iscsi ストレージ同様に VM マイグレーションが実行可能であり，その概要図を図 5.1 に示す．

5.3.2 xNBD

xNBD は，ストレージマイグレーションを実現すべく，産業技術総合研究所にて開発されている NBD の拡張版である．図 5.2 に xNBD を用いたストレージマイグレーションの動作概要を示す．従来の NBD と異なり，移行先に NBD のセッションを中継する xNBD Proxy Server が特徴であり，そこで VM Migration 以降の VM の I/O を Host 2 側でキャッシュとして蓄積しつつ，Host 1 → Host 2 間でストレージをコピーし，I/O と統合することで VM を停止せずにストレージマイグレーションを実現する．Target Server, Proxy

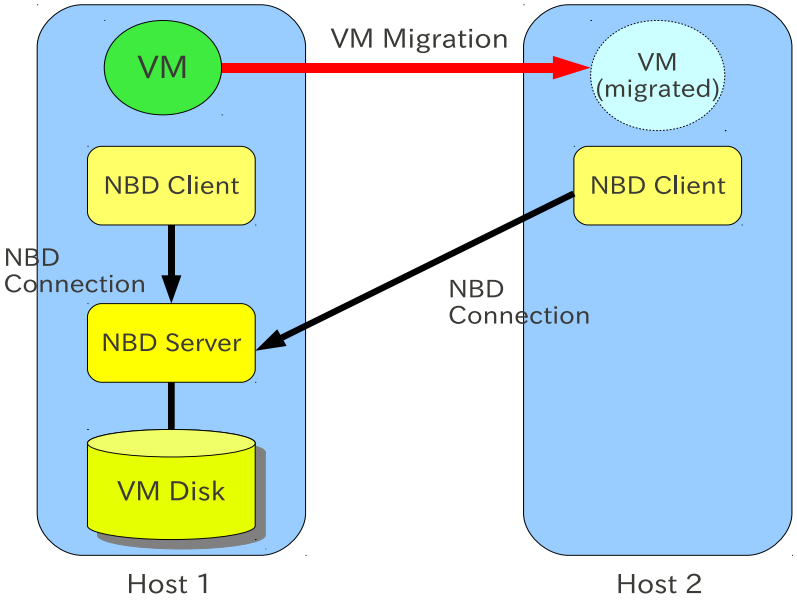

図 5.1　NBD を用いた VM マイグレーション

Server のストレージと接する部分は従来の NBD カーネルモジュールを呼び出す形で使われている．

5.3.3　ltdd

ltdd は，NBD を拡張して作成された VM スナップショット・ホットクローニングを可能にするツールである．4.3 で得た問題意識を元に，我々 elab クラウドチームによって実装・評価を行っている．遠隔ホスト間はまだ動作に不安が残るが，今後も開発・改善を継続していく予定である．
ltdd を用いたライブスナップショット，及びホットクローンの概要をそれぞれ図 5.3，図 5.4 に示す．なお，図では遠隔ホストに対してそれぞれスナップショット，ホットクローンを行っているが，ローカルホストの別ファイル，別ブロックデバイスを移行先として実行することも可能である．NBD，xNBD との大きな違いは，一つのプロセスが同時に二つのストレージを管理している事である．なお NBD クライアントからのセッションを受けとるので，従来の NBD サーバのように用いつつ，定期的に VM のフルスナップショットを取るような運用も可能だと考えられる．

5.3.4　**評価軸**

■対応デバイス：　elab クラウドでは仮想ディスクとしてファイルデバイスとブロックデバイスの混在環境となっているので，そのどちらに対しても平等に扱える技術が必要だった．その結果として，NBD を元にした技術に多く触れることになった．

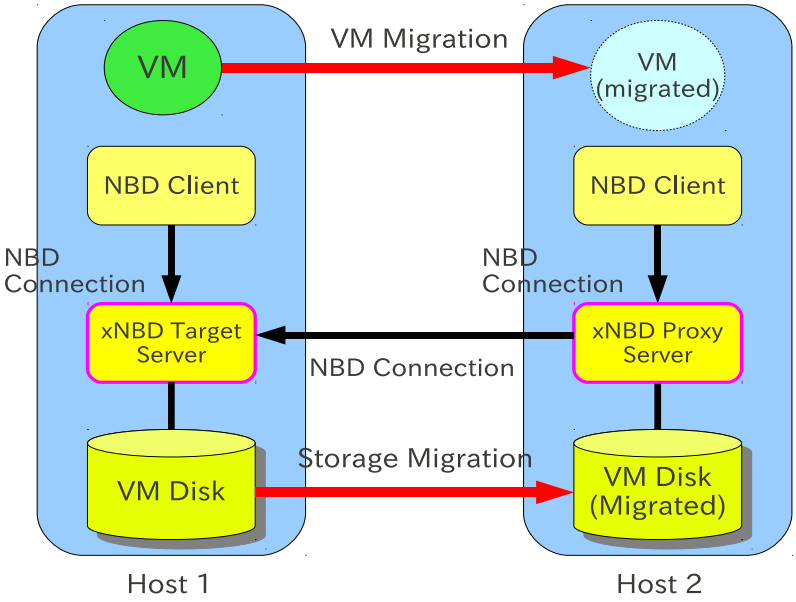

図 5.2　xNBD を用いたストレージマイグレーション

■安定性：　高負荷な環境下でも安定して動くかどうか．その分，パフォーマンスについてはベストエフォートで構わないと我々は考えている．
既に重要なサービスを載せて運用を開始してしまった以上，これらを落とすわけにはいかない．

■運用面：　主に導入・使用・脱却の三点が簡単にできるかどうか．
運用主体が江崎研究室の学生であり，博士課程に進まない限り運用に携わるのは長くて 3 年なので，余り複雑な技術を導入するのは長期的視点から見て問題がある．特に脱却に関しては，P2V 作業の時にソフトウェア RAID を通して感じたことでもあるが，その技術のある・なしが簡単に切り替えられる技術であることが望ましい．

5.3.5　検証シナリオ

1. 異なる HV 環境で同じ仮想ディスクから VM を起動可能か？
2. xNBD, ltdd が動く環境の整理
3. 異なる HV 間で VM マイグレーションが可能か？

以上が，主な検討事項である．なお，VM マイグレーション，及び別ホスト間での I/O を伴う行為（ストレージマイグレーション，ライブスナップショット，ホットクローニング）の検証時には，VM 上で I/O に対する負荷として，ストレージのベンチマークテストである iozone を実行しながら行った．

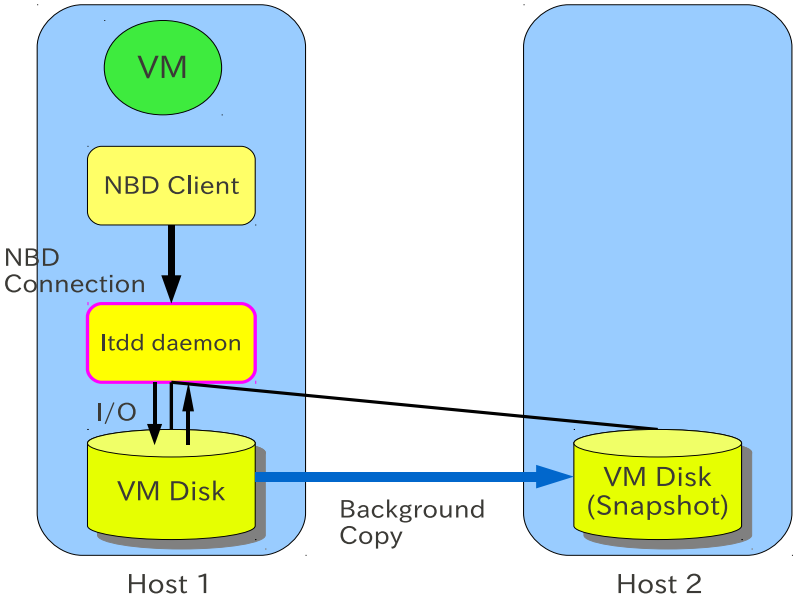

図 5.3　ltdd を用いた VM のライブスナップショット

5.3.6　VM について

検証時に用いた VM のスペックは下記である．

OS：Ubuntu 10.04(x86_64)
CPU：vcpu 1 コア
メモリ：1GB
仮想ディスク容量：8GB

このような VM をテンプレートとして作成して用いた．なお，物理ホストのパーティションとしては，LVM の独自論理ボリュームとして作成しており，3 種類のどの OS + HV の組み合わせからも参照可能である．

5.4　検証結果・考察

5.4.1　異なる HV 環境で同じ仮想ディスクから VM を起動可能か？

基本的には問題なく共用が可能であり，これは P2V に成功した段階からほぼ確実に可能だろう，と予想されていたことの確認でもあった．

但し例外があり，Ubuntu 11 系列を VM として Xen 上で作成・起動することができない．まず，Xen（3.4.3, 4.0.1 の両方）上で Ubuntu を VM としてインストールしようとすると，ハードウェアの認識に失敗し，作成

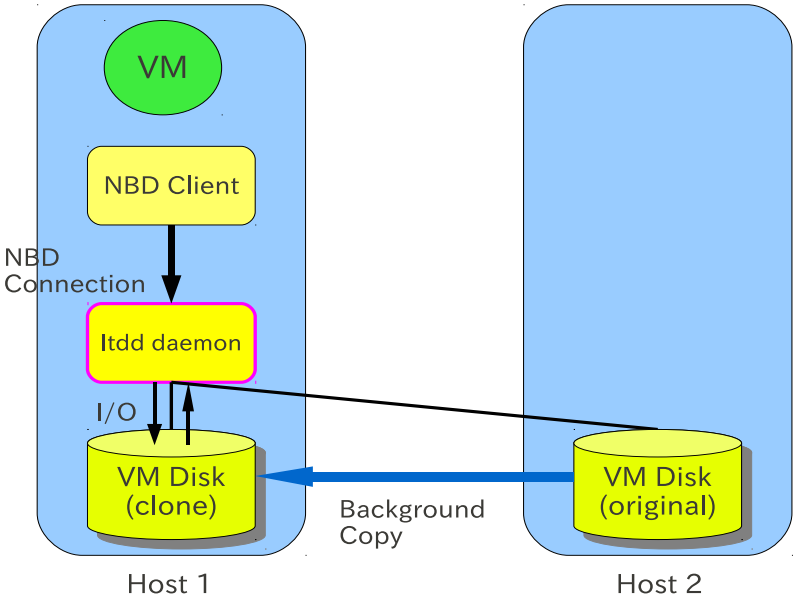

図 5.4　ltdd を用いた VM のホットクローニング

することができない．KVM では問題なく作成することができるのだが，そこで作成した仮想ディスクから VM を Xen で立ち上げようとすると HDD を認識できず，BIOS あるいは grub2 で止まってしまう．elab クラウドの現在の環境では，ubuntu11 を使用できずともそんなに困らないので，深い原因究明は行っていないが，いずれ解決せねばならない問題である．

5.4.2　xNBD, ltdd について

	CentOS		Debian	
	インストール可否	動作	インストール可否	動作
xNBD	×	-	○	○
ltdd	○	△	○	○

表 5.1　xNBD, ltdd の環境別動作結果

表 5.1 に，ltdd，xNBD の環境別のインストール可否及び動作結果について示した．共にブロックデバイスを直接触るツールで在ることから，仮想化環境というよりは物理ホストの OS によって差が出た結果となった．表 5.1 中の○については，問題なく動作ができた部分である．以下，×と△について見てゆく．

■CentOS での xNBD について　xNBD は元々，内部実装の ppoll() システムコールの関係上，linux カーネル 2.6.26 以降での使用が推奨されている．それに対し，CentOS のカーネルバージョンは 2.6.18-274.17.1.el5

であり，一回り古いものである．また，内部のエンディアン変換に htobe64() 等の関数が呼ばれており，これらが glibc にマージされたのは glibc2.9 以降であり，コンパイルにはそれより新しいバージョンが必要になってくる．それに対し，CentOS がパッケージ管理で問題なくインストールできる glibc は 2.5.65 である．

すなわち，xNBD をインストールするには glibc を新たに再コンパイルする必要があり，その上で xNBD が正常に動作するかは現行のカーネルでは保証されていない．そもそもこの検証における軸は安定性及び導入にかかるコストであった．この段階で安定性も不安かつ導入コストが十分に高いことも判明したので，CentOS に対して xNBD をインストールする試みは中断した．

■CentOS での ltdd の動作について　表 5.1 中で△としている部分についてである．低負荷の状態では問題なく動作するのだが，高負荷になると不具合が発生する．壊れ方は場合によって異なり，下記に著者が実際に経験した一例を示す．

1. いつの間にか VM のディスクがリードオンリーマウントになっている
2. VM のディスクが部分的に破損し，変な表示になる
3. HV 側の OS がカーネルパニックを起こして VM 諸共に落ちる

図 5.5 に，カーネルパニックを起こした際の CentOS のモニターを撮影したものを示す．おそらくこれはスタックトレースの末尾でしかなく，断片的な情報ではあるのだが，ブロックシステムへの I/O の最中で落ちたのだろうと推測可能であり，iozone 下でのクローニング，という状況とも一致している．経験的にも 2.6.18linux カーネルの kjournald に負荷がかかりすぎるとカーネルパニックに至るのだろうと我々は考えている．linux にはカーネルパニック時のメモリ状態・カーネル内部でのコールリスト等を保管しておき，デバッグに用いる事が可能な kexec-tool があるのだが，独自にカーネルを入れる必要のある Xen とは相性が悪く，導入には至っていない．

また，現行の ltdd は，遠隔ホスト側の CPU 負荷がだいたい 100% で張り付いてしまい，遠隔へのスナップショット，遠隔へのクローニングは若干動作に不安が残る．遠隔ホストへのサポート機能についてはまだプロトタイプであるので，今後とも継続して改善・評価を続けていきたい．

5.4.3　異なる HV 間での VM マイグレーション

まず，KVM と Xen の間の VM マイグレーションは，API が統一されていないこともあり，不可能である．次に，Xen 3.4.3 + CentOS 5.7 と Xen 4.0.1 + Debian squeeze の間の VM マイグレーションについてであるが，こちらも不可能であった．Xen3.4.3 → Xen4.0.1 という，バージョン違いな同種の HV 間での VM マイグレーションは可能であると推測されるのだが，この場合は CentOS と Debian という環境の差が問題になっている．VM マイグレーションの際には，VM を実際に起動するコマンドのオプション，あるいは設定ファイルがマイグレーション元のホストからマイグレーション先のホストへ渡されるので，両環境でそれらが同値でなければならない．それらはマイグレーションコマンドを撃った際に両ホスト間で評価され，食い違いがあるとマイグレーションは失敗に終わる．例えば仮想ディスクのパスやネットワークインタフェース情報は設定が容易に可能なのだが，実際の仮想エミュレータを呼ぶパスについては，HV サーバ上での仮想化環境インストール段階で調整する必要があり，事前に合わせるのは困難である．

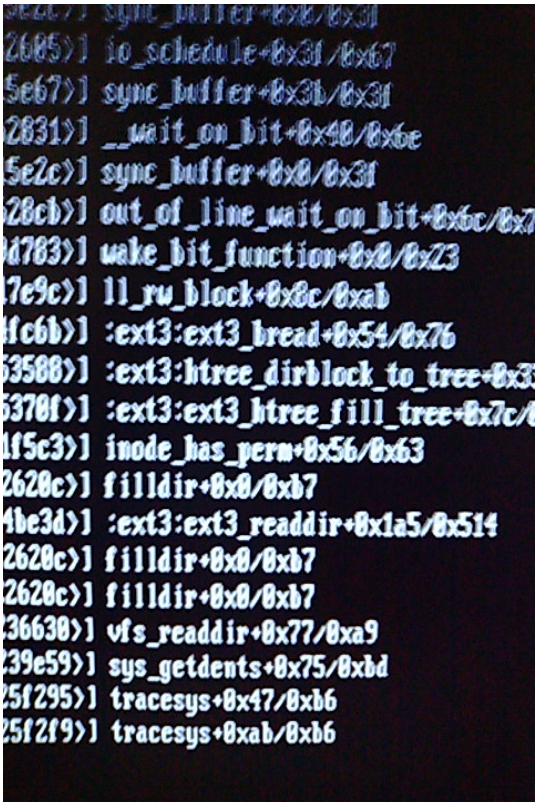

図 5.5　カーネルパニックを起こした CentOS（写真）

インストール後の状態に対して，シンボリックリンクを作成してパスを揃えた上でマイグレーションを試してみたが，CentOS 側が VM を送信できたと判断しているのに対し，Debian 側では受けとった VM をいざ起動する段階で Xen がエラーを吐いて終了していまい，結果的に VM が消えるというより危険な状態になってしまった．こちらについても，より詳細な調査が必要である．

第 6 章

まとめ

本稿では 2011 年四月に構築して以来の，1 年間の運用において培った現状の我々の運用体制をベスト・カレント・プラクティスとしてまとめた．また，電気系共用サーバ及び江崎研インフラサーバ群の P2V 移行作業と，電気系メールサーバのメールデータ復旧という二つの実際の運用実例についても触れた．そして，それらから得られた知見を元に，ホットクローニングやライブスナップショットのような VM の仮想ディスクを無停止で複製する技術の必要性についての議論と，それらの導入を前提とした検証についても報告した．

謝辞

本論文の執筆と修士生活を送るに当たりまして，大変多くの方にご指導・ご助言頂きました．ここに，感謝の意を表します．まずは，大学院生として活動するにあたり，自分の勝手を許して下さった浅見 徹教授と川原 圭博講師に深くお礼申し上げます．浅見研究室の先輩である宮坂 拓也氏と西本 寛氏に感謝致します．お二人は優秀な学生の模範を示して頂くと共に，ご助言も多く頂きました．浅見研究室の同期として楠 慶くんには二年間いい影響を頂きました．感謝すると共に，今後の互いの成功を祈る次第です．浅見研究室の後輩である加藤 拓也くん，佐々木 達哉くん，清水 和人くん，中島 直哉くんに感謝致します．みな大変優秀で，とても刺激を受けました，**僕が先輩諸氏から頂いたほどのお世話ができず，面目ありません．**

学部生時代に担当教員としてお世話になり，大学院に進学してからもネットワークの運用活動を通して，あるいは人生相談等，ご指導・ご鞭撻・叱咤激励を頂いた江崎 浩教授に深くお礼申し上げます．また，親身になって何度も相談に乗って頂いた土本 康生博士にも深くお礼申し上げます．そして，運用・研究を含むあらゆる技術的な面で引っ張って頂いた浅井 大史氏に深くお礼申し上げます．お三方抜きには，大学院生活をここまで全うすることはできませんでした．深く感謝しております．

江崎研関係者の皆様にも，学部 4 年生の頃から三年間色々とお世話になり，感謝を申し上げます．江崎研究室の先輩として，公私共にお世話になった山本 成一博士，藤田 祥博士，土井 祐介博士，落合 秀也博士，金海 好彦氏，白井 俊宏氏，阪本 裕介氏，肥村 洋輔氏，下忠 健一氏，杉田 毅博氏，川上 雄也氏，に感謝致します．川口 紘典くんと本舘 拓也くんは学部生以来の同期として，研究のみならず苦楽を共にし，またお互いに議論して切磋琢磨できました．感謝致します．Luciano Aparicio 氏，David Jageberg 氏，Jonas Johansson 氏，Sathita Kaveevivitchai 氏，Leela-amornsin Lertluck 氏 Romain Fontugne 氏とは国境をこえた友誼を築け，大変良い経験ができたことを感謝しております．江崎研究室の後輩諸君に感謝致します．石田 渉くん，石橋 尚武くん，大津 恭平くん，小坂 良太くん，正原 竜太くん，李 聖年くん，林 東權くん，木下 僚くん， 美嶋 勇太朗くんの大学院生活が実りある物となることを祈っております．園田 大剛くん，東浦 成良くん，朴 成軍くんの社会での活躍を祈ります．研究室生活を支援頂いた江崎研究室秘書の高橋富美さん，田坂佳苗さん，岩井愛映子さんに感謝致します．皆様のご協力なしに学生の研究室生活は存在しえません．

研究室以外でも，多くの方にお世話になりました．仮想化技術に携わるきっかけを与えて下さった株式会社 IIJ イノベーションインスティテュートの宇夫 陽次朗氏，島 慶一氏に深く感謝申し上げます．仮想化技術に関して多くのご示唆を頂いた関谷 勇司准教授に深く感謝申し上げます．技術のありかたについて議論をさせて頂き，大いにモチベーションとなった奥村 貴史医学博士にも深く感謝致します．電気系共用サーバ管理 TA として，研究室の壁を超えて運用に携わった矢野 友貴氏と河野 瑛くんに感謝致します．運用の大変さを共有できたことは大きな糧になると思います．東京大学 電気自動車クラブ UTECH の諸メンバーに感謝致します．主

活動に参画できなかったのは残念ですが，作成した車が無事走行した時は自分の事のように嬉しかったです．

最後に，24 年間自分を支え続けてきてくれた家族と，大学生活を共に駆け抜けた友人諸氏に，この場を借りて御礼申し上げます．

参考文献

[1] Xen. http://www.xen.org/.

[2] KVM. http://www.linux-kvm.org.

[3] VMware vSphere HypervisorTM(ESXi). http://www.vmware.com/jp/products/vsphere-hypervisor/overview.html.

[4] libvirt. http://libvirt.org/.

[5] P. Barham, B. Dragovic, K. Fraser, S. Hand, T. Harris, A. Ho, R. Neugebauer, I. Pratt, and A. Warfield. Xen and the art of virtualization. In *Proceedings of the nineteenth ACM symposium on Operating systems principles*, pages 164–177. ACM, 2003.

[6] QEMU . http://wiki.qemu.org.

[7] A. Kivity, Y. Kamay, D. Laor, U. Lublin, and A. Liguori. kvm: the Linux virtual machine monitor. In *Proceedings of the Linux Symposium*, volume 1, pages 225–230, 2007.

[8] T. Hirofuchi, H. Nakada, H. Ogawa, S. Itoh, and S. Sekiguchi. A live storage migration mechanism over wan and its performance evaluation. In *Proceedings of the 3rd international workshop on Virtualization technologies in distributed computing*, pages 67–74. ACM, 2009.

[9] ltdd. http://draft.scyphus.co.jp/cloud/ltdd.html.

[10] Network Block Device. http://nbd.sourceforge.net/.

付録 A

virsh の頻出コマンド

■list： VM 一覧を示す.

```
virsh # list --all
 Id Name                 State
----------------------------------
  0 Domain-0             running
  - hoge5901             shut off
  - hoge5903             shut off
  - ubuntu11.10test      shut off
```

■version： libvirt 及びハイパーバイザのバージョン情報を示す.

```
virsh # version
Compiled against library: libvir 0.8.3
Using library: libvir 0.8.3
Using API: Xen 3.0.1
Running hypervisor: Xen 4.0.0
```

■start / shutdown / destroy： それぞれ，VM の起動・停止・強制シャットダウンに相当する．VM 名を指定してコマンドを撃つ.

```
virsh # start hoge5901
virsh # shutdown hoge5901
virsh # destroy hoge5901
```

■edit： VM の設定 xml をデフォルトのエディタ環境（主に vi）で編集する．なお，起動中の VM に対する変更は反映されない.

```
virsh # edit hoge5901
```

■dumpxml / define / undefine :　dumpxml は，VM の設定を xml の形式で出力する．define は xml ファイルを元に，新しい VM を libvirt に登録・抹消する．undefine は libvirt に登録されている VM を抹消する．このうち，dumpxml と define は P2V 作業において頻繁に使用する．HV 環境で既に動いている VM の設定を dumpxml で出力し，その xml ファイルを P2V で移してきた新しい VM に適切に設定し，define を行う．なお，その際に設定する主な項目は 2.4 に準ずる．

```
# virsh dumpxml hoge5901 > fuga.xml
# vim fuga.xml
--
# virsh
virsh # define fuga.xml
--
virsh # undefine hoge5902
```

■console :　console は，HV から直接 VM のシリアルコンソールへ接続するコマンド．VNC を使わずに VM を直接操作できるので，非常に有用であるが，シリアルコンソールなので，OS インストールなどの GUI が必要となる操作では使えない．また，VM 側の OS 上でのシリアル出力と libvirt 側のシリアル設定が適切になされている必要がある．なお，VM 側の GRUB の設定次第では GRUB 画面も取得・操作可能．

```
virsh # console hoge5901
Connected to domain hoge5901
Escape character is ^]

# -- VMのログインプロンプトが表示される
Ubuntu 10.04.3 LTS hoge5901 ttyS0

hoge5901 login:
```

付録 B

VM のマイグレーションについて

本付録では，VM のマイグレーションのシナリオについて例示する．ここでは環境として Debian(squeeze) + Xen4 を用いているが，CentOS 5 + Xen3.4.3 でも同様の動作が可能である．KVM については後述する．以下，登場する物理ホストは二つ，仮想ホストは一つである．

- kure-exp-1：HV，マイグレーション元, 192.168.0.1
- kure-exp-2：HV, マイグレーション先， 192.168.0.2
- hoge5901：VM，マイグレーション対象

マイグレーションの事前準備として，それぞれの HV の Xen の設定をマイグレーションが可能になるように変更する．

```
root@kure-exp-1:/home/kure# vim /etc/xen/xend-config.sxp
(xend-unix-server yes)
(xend-relocation-server yes)
(xend-unix-path /var/lib/xend/xend-socket)
--
# この三行のコメントアウトを外し，yesに

(xend-relocation-hosts-allow '^localhost$ ^localhost\\.localdomain$ ^192\\.168\\.0\\.2$')
--
# マイグレーション相手を xend-relocation-hosts-allow に正規表現で追記する
# kure-exp-2 側では 192.168.0.1 を加える．
```

以下，マイグレーションの流れを説明する．ここでは図 5.1 にあるように NBD を用いた VM マイグレーションを行う．まず，kure-exp-1 と kure-exp-2 の双方から NBD 経由で仮想ディスクにアタッチする．

```
root@kure-exp-1:/home/kure# nbd-server 2000 /dev/VolGroup00/hoge5901.img
root@kure-exp-1:/home/kure# nbd-client localhost 2000 /dev/nbd1
Negotiation: ..size = 10485760KB
bs=1024, sz=10485760
root@kure-exp-1:/home/kure#
--
root@kure-exp-2:/home/kure# nbd-client 192.168.0.1 2000 /dev/nbd1
Negotiation: ..size = 10485760KB
bs=1024, sz=10485760
root@kure-exp-2:/home/kure#
```

kure-exp-1 上で，hoge5901 の仮想ディスクが先ほどアタッチした/dev/nbd1 であることを確認し，VM を起動する．

```
root@kure-exp-1:/home/kure# virsh dumpxml  hoge5901 | grep "source dev"
      <source dev='/dev/nbd1'/>
root@kure-exp-1:/home/kure# virsh start hoge5901
Domain hoge5901 started
```

必要な条件が整っていることを確認できたら，マイグレーションを行う．ここでは Xen なので，virsh 経由ではなく xm という Xen 専用のコンソール経由でコマンドを発行する．migration の前後での VM リスト一覧も示す．

```
root@kure-exp-1:/home/kure# virsh list
 Id Name                 State
----------------------------------
  0 Domain-0             running
  2 hoge5901             idle

root@kure-exp-1:/home/kure# xm migrate hoge5901 192.168.0.2 --live
root@kure-exp-1:/home/kure# virsh list
 Id Name                 State
----------------------------------
  0 Domain-0             running

root@kure-exp-1:/home/kure#
--
# xm migrateの前後でhoge5901がkure-exp-1のVM一覧から消えている

root@kure-exp-2:/home/kure# virsh list
 Id Name                 State
----------------------------------
  0 Domain-0             running
  8 hoge5901             idle

root@kure-exp-2:/home/kure# virsh console hoge5901


Connected to domain hoge5901
Escape character is ^]

Ubuntu 10.04.3 LTS hoge5901 ttyS0

hoge5901 login:
--
%# migration後のkure-exp-2．VM一覧でhoge5901が見えるだけで無く，
%# consoleコマンドでログインプロンプトがでることも確認
```

以上で Xen 環境における VM マイグレーションシナリオの一例を示した．なお,KVM の場合は,xm migrate の代わりに virsh migrate を使用する．それに伴い, 事前のハイパーバイザ間での認証も Xen 同士から libvirt 同士に変更されるため, 設定すべき項目が
/etc/xen/xend-consig.sxp から libvirt の設定に変更される．