仮想化技術の概説と効用

- VMware製品の実装例 -

ヴイエムウェア株式会社
システムエンジニア
竹洞陽一郎
ytakehora@vmware.com

vmware

ヴイエムウェア社について

企業概要:

- スタンフォード大学内の研究所にて研究されていたテクノロジ
- インテルCPU用のメインフレームクラスの仮想化技術を開発
- 1998年にヴイエムウェア社設立

主な研究開発分野:

- インテルアーキテクチャ上で複数のオペレーティングシステムを動作させる仮想マシン技術
- 従業員の50%以上を研究開発(R&D)分野にアサイン

方向性:

- VMware Workstation製品を1999年にリリース、GSX Serverを2001にリリース、ESX Server を2001にリリース
- 社内テスト、ベータプログラムの徹底 – 品質を最重点項目

会社概要:

- 本社は、カリフォルニア・パロアルト
- 従業員　900人以上
- 健全な財務基盤
- フォーチュン100企業の80%以上がVMware製品のユーザー
- 300万人以上の登録ユーザー
- 100以上の国々に、10,000社以上の企業ユーザー

vmware

仮想化の流れ～何故、今、仮想化なのか？

vmware

何故、今、仮想化技術が注目されているのか?

- 昨今、仮想化技術についてのニュースリリースが相次ぐ
  - CPU
    - Intel～Virtual Technology
    - IBM、東芝、ソニーグループ～Cell
  - UNIX
    - HP～VSE
    - Sun Microsystems～N1 Grid System
    - IBM～IBM Virtualization Engine
  - Intelアーキテクチャ向けOS
    - Microsoft～Virtual PC、Virtual Serverの販売開始
    - オープンソース～Linux上での仮想化技術 Xen

vmware

増大し続けるIAサーバ

- 市場競争の波に晒されて、低価格化、高性能化が進むIAサーバ
  - プロジェクトや部門毎にIAサーバの導入が進む
    - 場所の問題～増殖し続けるIAサーバの設置スペースの確保
    - リソース使用率の問題～ピーク時を考慮してリソースのサイジングをしなければいけないので、台数は減らせない
    - 安定性・安全性の問題～ミッションクリティカルな業務への適用も広がる中、OSの安定性やセキュリティの強度を考慮してシステムを構築しなければならない

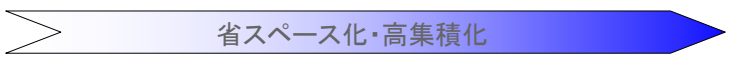
場所の問題

- タワー型サーバからラックマウント型サーバ、そしてブレード型サーバへと、サーバの省スペース化、高集積化によりサーバスペースを節約
  - ブレード型サーバは1サーバ1アプリケーションで稼動するのが最適
  - 高負荷がかかる、サーバリソースを豊富に必要とするシステムには適用できない
  - ブレード型サーバに物理的に集約しても、それぞれのサーバのCPU使用率は低いまま

タワー型サーバ

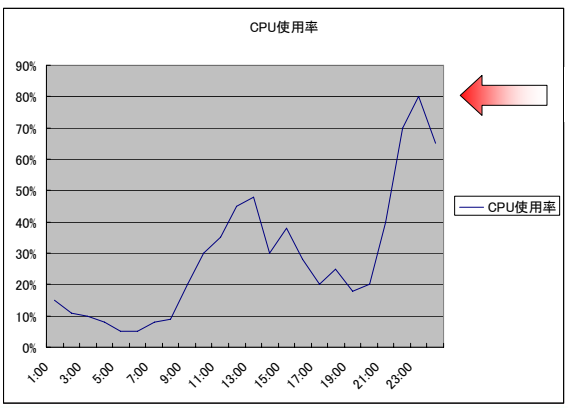
ラックマウント型サーバ

ブレード型サーバ



リソース使用率の問題

- ピーク時を想定してサーバのサイジングが行われる
  - サーバの平均稼働率は、サーバのキャパシティに比べて遥かに低い
  - ビジネスの状況次第で、想定していたピーク時の負荷を大きく上回ったり下回ったりする



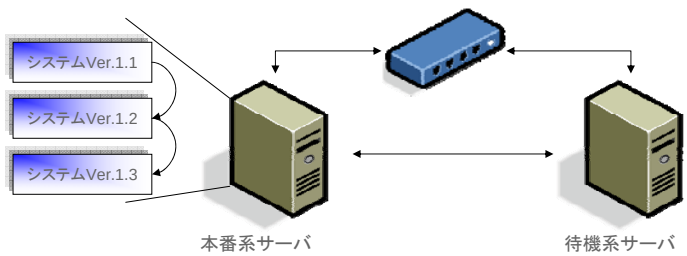
この時の負荷に耐えるようにサイジング

ピーク時を想定して購入したWebサーバ

vmware

安定性、安全性の問題

- システムの安定性・安全性=ビジネスの安定性・安全性
- ミッションクリティカルな業務に関してはHA構成が必須
  - HA構成にすることによって、更に稼動率が低いサーバが増えることに
  - HA構成は手軽な金額で実現しにくい
- システムのバージョンアップ、セキュリティパッチなどのバージョン管理
  - システムを止める事なく、パッチを当てたり、バージョンアップを行いたい
- 一つのサーバにアプリケーションを集約した場合、OSを止めると全てのアプリケーションを止めざるを得ない

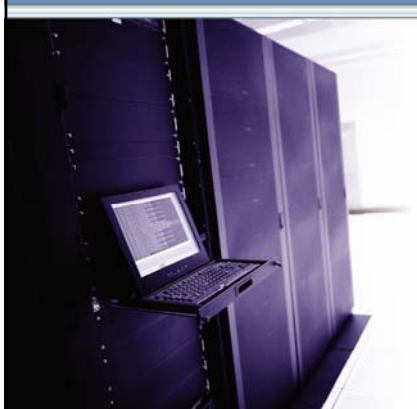


vmware

仮想化によって解決される問題の数々

- 場所の問題
  - 1台の物理サーバ上で、複数台の仮想マシンを稼動させることで、集約可能
  - ブレード型サーバでも、2～3台の仮想マシンを稼動させる事が可能
- リソース使用率の問題
  - 仮想化することにより、システムは物理サーバに縛られなくなる
  - リソースの消費状況を見て、仮想マシンを増やしたり、減らしたりして物理サーバのシステムリソースの消費状況を最適化
  - システムが必要とするリソースを提供できるだけのキャパシティが物理サーバに無い場合に、より強固なサーバへ移動
- 安定性、安全性の問題
  - 仮想化することにより、1台の仮想マシンが落ちても、他の仮想マシンに影響が及ばないため、システムリソースを有効に活用できる
  - HA構成を組む場合に、物理サーバを複数台用意する場合に比べて安く済む

vmware



VMwareの仮想化技術

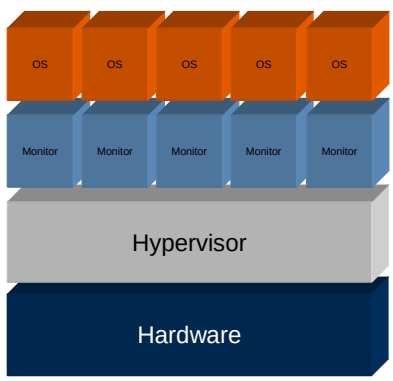
vmware

仮想化技術の原点

- 仮想化技術を最初に実装したのはIBMのOS370
- 1960年代後半にMITがIBMのメインフレーム上で仮想化の技術を実装
- 仮想化の背景
  - メインフレームのOSはシングルユーザだった
  - 当時、メインフレームのCPU処理能力は急速に向上しており、仮想化によって複数のOSを同時に稼動させることにより、高額且つシングルユーザしか使えないリソースの使用率を高めることが目的だった

OS370

- Hypervisor
  - ハードウェアデバイスをマルチタスクで使用し、“Monitor”を生成・稼動させる
  - ハードウェアを直接操作
- Monitor
  - 仮想マシンを管理する親プロセス
  - 個々の仮想マシン毎にMonitorが用意される
  - MonitorのインスタンスはHypervisorによって生成される
- OSのインスタンスはMonitorによって生成された仮想マシン上にインストールされる

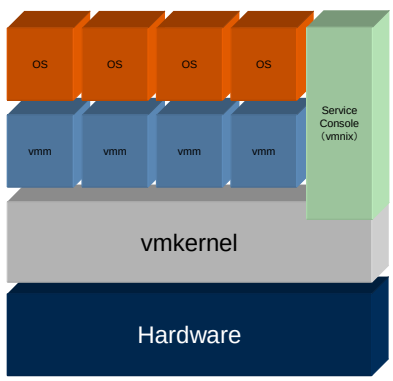


VMware

- X86アーキテクチャも今やメインフレームが1960年代に経験したようなハードウェアリソースを使い切れない状況を迎えている
  - 市場競争に晒されて、高機能化・高速化、低価格化したCPU
  - OS上で1つの“アプリケーション”が稼動している状況
- メインフレームの世界で実績が認められている同じ種類の多重化方法をVMwareの技術でx86アーキテクチャ上で実現
  - 1台の物理サーバ上で複数のOSが稼動することができるようにする

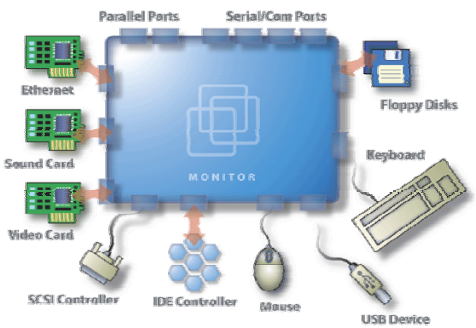
ESX Server

- Vmkernel
  - 各仮想マシンからVMMを経由して出てくる命令をキャプチャして、スケジューリングし、処理する
  - ハードウェアを直接操作
  - 300K LOCの非常にコンパクト且つ堅牢なカーネル
- Service Console
  - 間接的に命令をVmkernelに受け渡してESX Serverを管理するコンソール
- vmm
  - 仮想マシンから出てくるバイナリ命令を監視するモニターが仮想マシン毎に生成される
  - vmmはマシンの全てのオペレーションをコントロールする
    - キーボード/グラフィックス/マウス
    - ネットワークカード
    - SCSIコントローラ



ESXが提供する仮想化ハードウェア

- 仮想化ハードウェアのスペック
  - 440vxチップセット
  - AMD PCIネットワークカード(vlance)
  - LSI LogicもしくはBus Logic SCSIアダプタ
  - VMware独自のネットワークカード (vmxnet)
  - VMware独自のグラフィックスカード
    - パフォーマンスの問題から



vmware

VMware ESX Server

- DOS basics
- Inturrupts
- Memory interrupt remapping
- Processor rings
- Binary Translation
- Translation Cache
- Virtual Machine - OS Type
- Terminal Services / Citrix
- Intel Virtual Technology

vmware

Hosted Virtualization

- Three buckets that all products have
- Why a hosted OS first?
- How the switching works
- Context (World) Switching
- Problems with the Hosted Platform

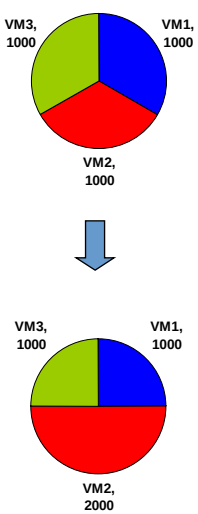
ESX / Bare metal solutions

- ESX Architecture
- ESX ring structure
- Boot file
- Vmkernel Hardware Connections
- SCSI Reservations
- 4 Proc configuration
- Hyper-threading

ESXで行えるサーバリソース管理

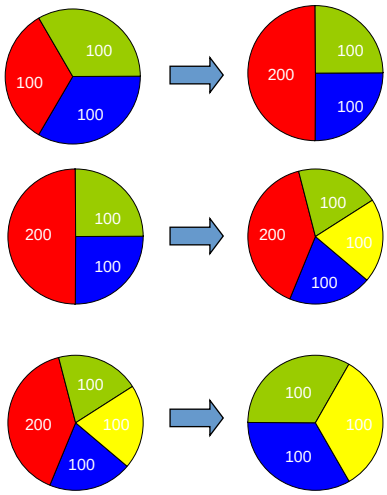
- 以下のサーバリソースを、各仮想マシンに割り当てることが可能
  - CPU
    - Hyperthreadingに対応～論理CPU単位で仮想マシンにアサインできる
    - 仮想SMP機能～SMPを仮想化して仮想マシンに組み込むことが可能（現在2CPUまで）
  - メモリ
    - 最小メモリ量と最大メモリ量を割り当てることが可能
  - NIC
    - トラフィックシェーピング方式で帯域管理が可能
    - 物理NICを束ねて1つのNICに見せる、NIC Teamingの機能を搭載
  - ディスク帯域
    - ディスクの帯域使用割合を設定可能
  - ディスク容量
    - 仮想化ディスクのサイズ変更が可能（OS側でパーティションマジックなどを使って、パーティションを拡大させる作業は必要）
    - 仮想化ディスクの4つのモード
      - Persisitent～コンピュータ上の従来のディスクドライブとまったく同じ様に動作。Persistent モードのディスクに書き込まれたデータはすべて、ゲストOS がデータの書き込みを行った時点でディスクに恒久的に保存される
      - Nonpersistent～ディスクへの変更は仮想マシンを電源オフにすると全て破棄される
      - Undoable～仮想マシンの電源オンから電源オフまでの変更を保存するか破棄するか選択可能
      - Append～変更は継続的にRedoログに追加され、Redoログファイルを削除すれば変更は破棄できる。Commitすれば、恒久的に保存される

リソースの割当方法～シェアという考え方



- VM1～3でディスク帯域幅のシェア値を1000にする
- 160MB/秒のディスク帯域幅がある場合、それぞれのVMは53MB/秒でディスクアクセスを行う
- VM2のディスクアクセスを優先させるために、シェア値を2000に増加
  - 1000：2000：1000 = 1:2:1の比率となり、VM1とVM3は、160MB × 1/4=40MB/秒のディスクアクセス
  - VM2は160MB × 2/4=80MB/秒のディスクアクセス

CPUスケジューリング～シェア値とMin/Maxの設定



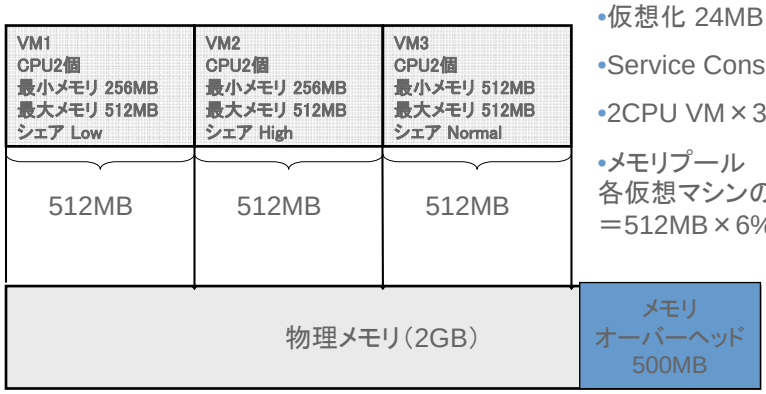
- VM シェアを変更
- ダイナミック リアロケーション

- VM を追加
- シェア相対値は変わらない

- VM を削除
- リソースの浪費はなし

vmware

メモリのシェア割当1

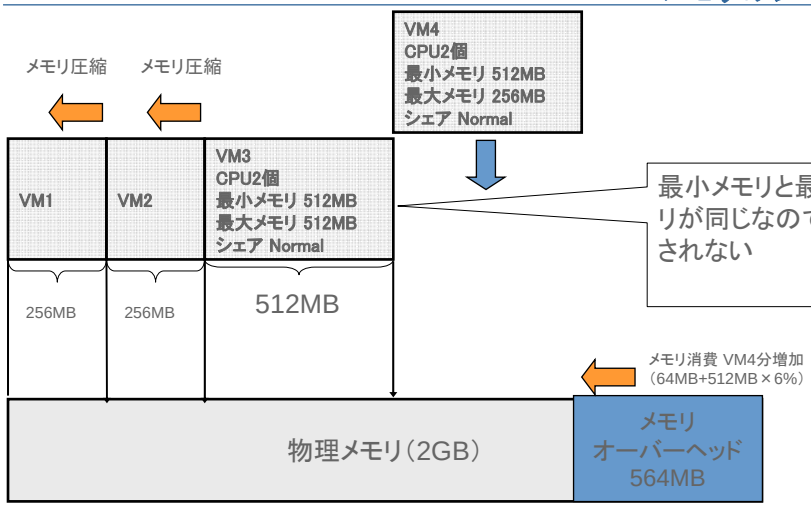


- 仮想化 24MB
- Service Console 192MB
- 2CPU VM×3＝64MB×3
- メモリプール
  各仮想マシンの最大メモリの6%
  ＝512MB×6%×3

物理メモリに余裕がある内は、各々の仮想マシンに最大メモリ量を割り当てる

vmware

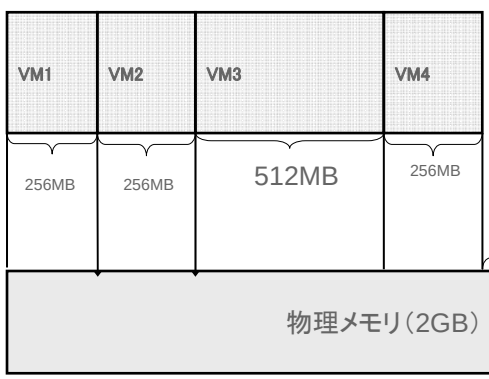
メモリのシェア割当2



新たにVM4を起動しようとすると、メモリが足りないので、VM1とVM2については割り当てメモリが圧縮される

vmware

メモリのシェア割当3

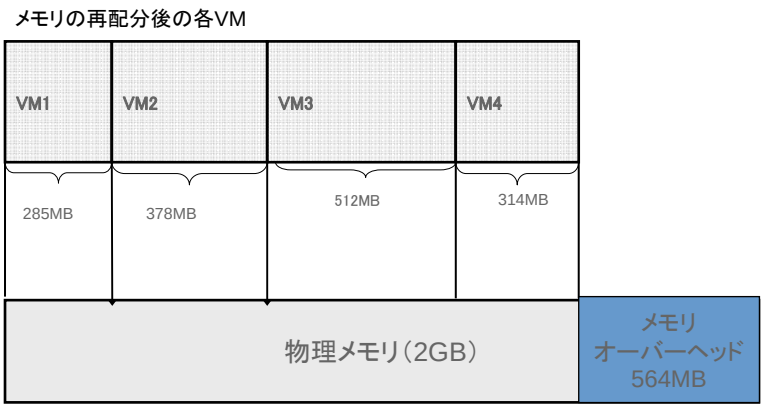


	シェア	比率	割り当てメモリ量
VM1	Low	1	204×1/7 =29MB
VM2	High	4	204×4/7 =116MB
VM4	Nominal	2	204×2/7 =58MB

最小メモリでの割り当て状態のVM

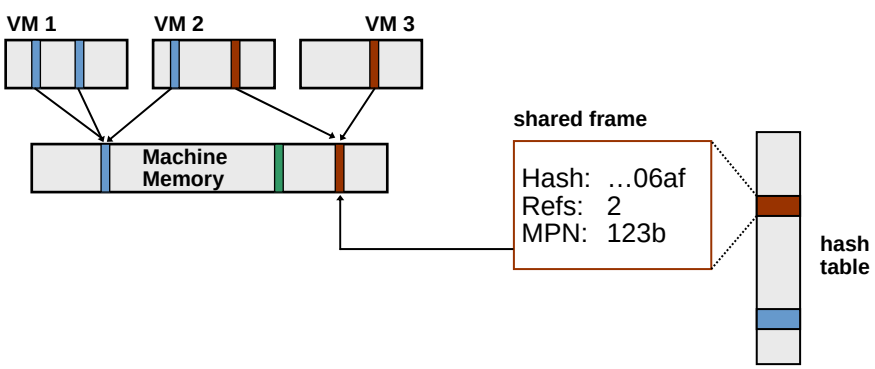
vmware

メモリのシェア割当4



vmware

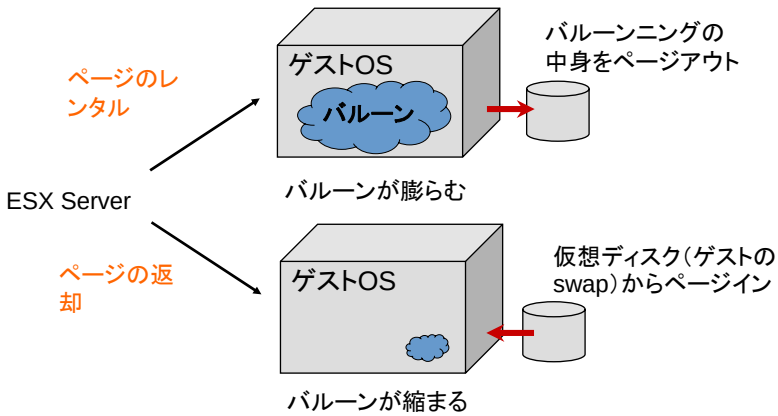
仮想化によるメモリの有効活用1：ページシェアリング



- プログラムコードページやゼロページの共有
- 標準的には10% - 20% のメモリ節約

vmware

仮想化によるメモリの有効活用：バルーンニング

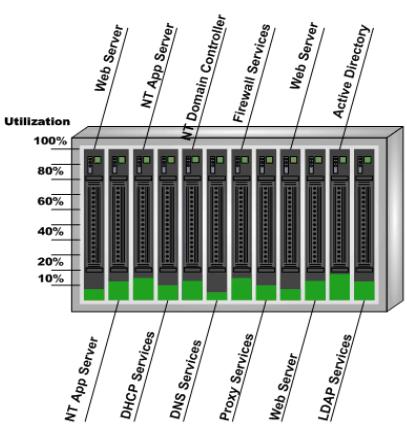


ESXサーバーが引き起こすメモリ・プレッシャー

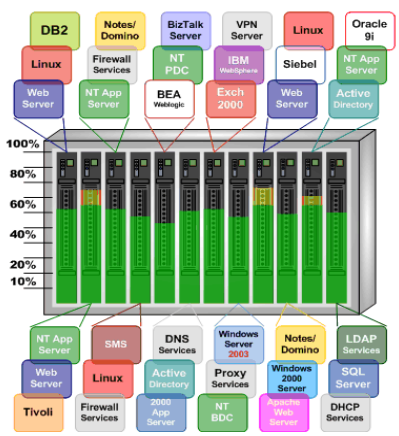
vmware

仮想化で効率的なサーバー利用
1台のサーバ（シャーシー）で4台分のワークロードに対応

Blade Servers without VMware VirtualCenter



Blade Servers with VMware VirtualCenter



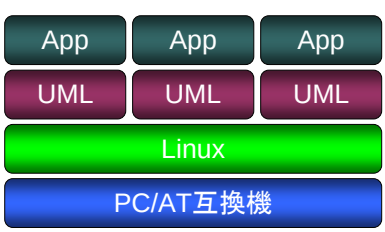
vmware

他の仮想化の実装例との比較

vmware

UML（User Mode Linux)

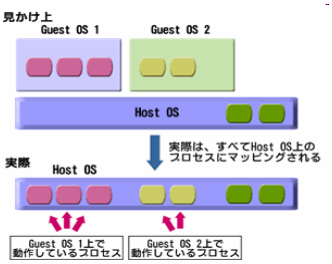
- UMLによるプロセスの独自スケジューリング
- ネットワーク機能のフルサポート
（Universal TUN/TAPドライバ経由）
- ディスクイメージファイル上に構築されたLinux環境
- 差分ファイルへのディスクイメージ変更履歴取得
- hostfs機能による、Host OS環境上のファイル資源へのアクセス
- 仮想化されたシリアル・ドライバ経由でのUMLへのログイン



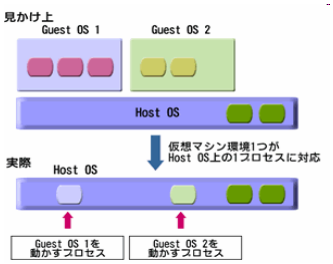
vmware

UMLとVMwareのプロセス処理の違い

- UMLでは、ゲストOS上で稼動しているプロセスは、全てホストOSのプロセスとしてマッピングされる



UMLのプロセスマッピング



ESXのプロセスマッピング