仮想化によるサーバの再構成

尾西 克之

大阪大学 理学研究科 技術部

1. はじめに

理学研究科トップの DNS サーバを始め、管理運用を行うサーバの多くはハードウェアの老朽化が進み、更新時期を迎えている。しかし全てのハードウェアを購入には費用が掛かり又、作業時間も掛かるため、一度に更新するのは不可能である。そこでサーバの仮想化技術を用いてサーバの再構成を行った。

2. 概要

再構成の対象となるサーバを表１に示す。これらを「１サービス=１仮想環境」の基本方針による構成にした。

表１．対象のサーバ

ドメイン	サービス	OS	形状・台数
sci	DNS, Mail	NetBSD	ラック×2
lns	DNS, Mail, Web	FreeBSD	タワー×3
hep	DNS, Mail	FreeBSD	タワー×1

次に仮想化の方法だが、VMWare, Xen, Hyper-V, VirtualBox, KVM など様々なアプリケーションがあるが、今回は FreeBSD の「jail(図 1)」を使用した。

その選択理由は、

(1). 構造がシンプルである。

(2). オーバーヘッドが少なく、軽い。

(3). 既存のハードウェアでも導入できる。

(4). 安定度が非常に高い。

(5). FreeBSD に使い慣れている。

などである。

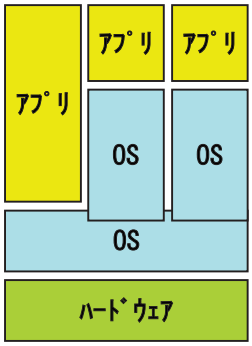

図 1　BSD jail

なお、jail とはパーティショニングによる仮想化の１つで、FreeBSD 独自の機能です。chroot(8)の機能をさらに強化したような機構で、jail 内で動作しているプロセスは許可されたディレクトリ以下の資源にしかアクセスすることができず、Jail より外のプロセスにアクセスすることもできないなどの特徴があります。

3. 構築

3.1. ハードウェア構成

仮想化用サーバのハードウェア構成を表 2 に示す。

ハードウェアは、新規購入２機と NIC、HDD、リムーバルブケースを増設した分を含む３機を利用した。

3.2. システム構成

仮想化用サーバのシステム構成を表 3 に示す。

prisoner(ゲスト)は概要で記したように、サービス毎に構築した。さらに表 2 に記したようにこのシステム構成で sciﾄﾞﾒｲﾝと lnsﾄﾞﾒｲﾝは、マスターサーバとセカンダリサーバを構築した。ただし、Hepﾄﾞﾒｲﾝはマスターサーバのみである。

※ sciﾄﾞﾒｲﾝおよび lnsﾄﾞﾒｲﾝの DNS サーバには、Hepﾄﾞﾒｲﾝの DNS ｷｬｯｼｭをするように設定している。

表 2. 仮想化用サーバのハードウェア構成

マスターサーバ			
Domain 名	sciﾄﾞﾒｲﾝ	lnsﾄﾞﾒｲﾝ	hepﾄﾞﾒｲﾝ
CPU	Pentium Dual-Core 2.4GHz	Intel Core2Duo 3.0GHz	Intel Celeron 1.0GHz
MEM	1GB	1GB	256MB
HDD	320GB(RAID1)	250GB(RAID1)	40GB×2
NIC	1000BASE-T ×2	1000 BASE-T ×4	100 BASE-T ×2

セカンダリサーバ			
Domain 名	sciﾄﾞﾒｲﾝ	lnsﾄﾞﾒｲﾝ	hepﾄﾞﾒｲﾝ
CPU	Intel Xeon 3.0GHz	Intel Celeron 1.1GHz	
MEM	1GB	512MB	
HDD	250GB×3	40GB×3	
NIC	1000BASE-T ×2	100 BASE-T ×4	

表 3. 仮想化用サーバのシステム構成

prisoner(ゲスト)名	sciﾄﾞﾒｲﾝ	lnsﾄﾞﾒｲﾝ	hepﾄﾞﾒｲﾝ
jail① / DNS ｻｰﾊﾞ	Bind	bind	bind
jail② / ﾒｰﾙｻｰﾊﾞ	Postfix	postfix	postfix
jail③ / Web ｻｰﾊﾞ		nginx	

3.3. 仕様

NIC（ﾈｯﾄﾜｰｸｲﾝﾀｰﾌｪｰｽｶｰﾄﾞ）は複数枚使用し、グローバル IP とプライベート IP を設定した。これはメンテナンス時のログイン接続(ssh:22)をプライベートネットワークからに限定し、さらに TCP Wrapper や IPfilter で制限をかけることにより、ハードウェア的にもソフトウェア的にもネットワークのセキュリティ性を高めることにした。

また、一般に同一のネットワーク IP が複数ある場合は、1 枚の NIC に alias などで設定するが、NIC でのボトルネックの軽減などを考慮し、Web サーバの IP は、別の NIC を設定した。

ただしこの場合、arp(Address Resolution Protocol)によるメッセージが出続けるので、それを回避するために、以下の設定を行った。

・「ファイル名：sysctl.conf」に以下を追記。

security.jail.allow_raw_sockets=1

net.link.ether.inet.log_arp_wrong_iface=0

4. まとめ

4.1. 仮想化による利点(メリット)

仮想サーバ化により、以下のメリットを得ることができた。

① リソース有効利用

元々が極めて負荷の少ないサーバ群であったため、CPU 処理能力やメモリといったリソースは 1%未満と、ほとんどアイドリング状態であった。これらを複数の仮想サーバで分配し有効に活用できた。

② 省コスト・省電力

物理サーバを７→５台と運用台数を減らすことができたので、消費電力・設置スペース・管理リソースといった様々な側面でのコスト削減できた。

(lnsﾄﾞﾒｲﾝ、Hepﾄﾞﾒｲﾝの場合、おおよそ消費電力 360Wh(70Wh×5 台) → 140Wh(70Wh×2 台)となった。)

③ 柔軟性

物理的な制約が無くなり、「仮想化されたハードウェア＝仮想マシン上で稼働する」ことになり、多台数の一元管理ができた。またサーバの起動時間も高速化され、再起動が必要な一部のメンテナンス作業時やダウンタイムも最小化できた。更に①に記したとおり、リソースにもまだまだ余裕があるので、ハードウェアの追加無しに新たなサーバの追加が可能になった。

4.2. 仮想化による問題点(デメリット)

仮想サーバ化により、以下のデメリットが生じた。

① 耐障害性

サーバ集約前ではサービス毎に物理サーバを構築することにより、セキュリティ性を含めた耐障害性を考慮したシステム構成であった。しかしこれらを１つの物理サーバ（ハードウェア）に統合したことにより、１台の物理サーバのダウンすることで複数の仮想サーバがダウンにつながる可能性が生じた。

② オーバーヘッド

物理メモリは仮想サーバが増える毎に消費します。加えてホスト部分もその対象となるため、たとえば lnsﾄﾞﾒｲﾝのサーバの場合、マシン４台分のメモリ消費となっている。

③ OS のアップデート

仮想サーバの OS 部は全ての仮想サーバで共有しているので、セキュリティインシデントに対応するために OS のアップデート作業を行う場合は、全ての仮想サーバに対して同時に行う必要がある。

4.3. 問題点への対策

① 耐障害性への対策

【セキュリティへの対策】

仮想マシン間はそれぞれが完全に隔離されていることから、各仮想サーバでは使用するサービスのポート以外は閉じ、外部からのアクセスを最小限とした。またメンテナンス作業もホスト OS からのみ可能とした。更にホスト OS 自体にも内部ネットワークからしかアクセスできないように設定した。

【システムクラッシュへの対策】

ハードウェア構成で述べたように、新規購入機には RAID1 ユニットを導入し、既存機にはリムーバブルケースを

増設、HDD を２～３台として、スクリプトを用いて HDD を自動的にクローン化できるようにした。これによりシステムがクラッシュした場合も直ぐさま復旧でき、ダウンタイムによる影響を最小限にしている。

② **オーバーヘッドへの対策**

サーバの仮想化を jail で行うことでオーバーヘッドは最小限に抑えているが、加えて各サーバで不要なプロセスが稼働しないように、出来る限りシンプルな構成にした。

③ **0S のアップデートへの対策**

システムクラッシュへの対策として HDD のクローン化したことにより、OS のアップデートに失敗しても元の状態に戻すことができた。

5. 今後の課題

今回のサーバの仮想化は、既存のハードウェアの利用も含めたものだったことや既存のサーバの OS が全て BSD 系だったので、仮想化には jail を選択した。現在、Linux KVM による Debian 系サーバの仮想化に取り組んでいる。

参考文献

[1] http://free-vv.dyndns.org/d/?q=node/406

[2] http://bsdsystem.blog40.fc2.com/blog-entry-1.html

[3] http://d.hatena.ne.jp/mteramoto/20090705/p5

[4] http://gihyo.jp/admin/clip/01/fdt/200812/02