試験的なクラウド基盤の構築

大下弘*、原　祐一*、雨宮　尚範*、佐々木康俊*、藤原　冨未治*、松岡　孝**
*工学系技術支援室 情報通信技術系
**共通基盤技術支援室 情報通信技術系

はじめに

サーバーの管理形態としてクラウドが最近脚光をあびている。本研修では、クラウドの現状を調べるとともに、そのベースとなっている仮想化サーバーを立ち上げ運用した。そして、試験的なクラウド基盤を構築することを目標とした。そのために、昨年度の環境整備費で購入したサーバーを使用してシステム構築を行い、動作検証を行った。

1. OS のインストール及びストレージの設定

1) 本研修が実施できる環境を設定するために、まず、研修用サーバー機に OS のインストールとストレージシステムを構築する。

本研修で参考にした「Linux サーバーの作り方」では商用版の Red Hat Enterprise Linux(以下 RHEL)6 をインストールし、設定しているが、研修実施環境を構築するための機材をそろえるのにも予算枠は不足していた。RHEL の評価版を利用することも含め検討したが、試用期間が 30 日であることと昨年度の研修ではオープンソースで互換の Scientific Linux を利用した。しかし、今年度はすでに 6.3 にバージョンアップされていた、もう一つの互換ディストリビューションの CentOS をインストールし、設定することにした。

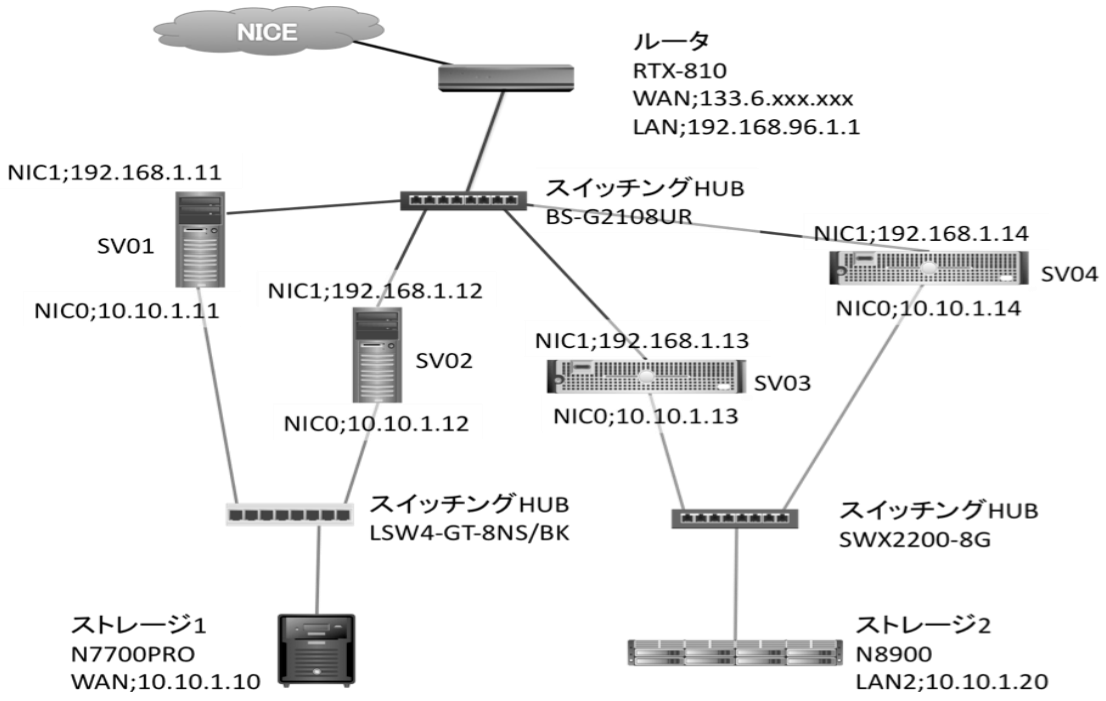


図 1. 仮想サーバーとストレージサーバーの試験環境構成図

2)インストール時には、仮想化ソフトとして、CentOS 同梱版のサーバー仮想化環境(KVM)を利用することにした。本研修には昨年度購入した機材も活用し、今年度さらにもう一組テストできる環境を整えた。その試験環境構成図を図 1 に示す。4 台の仮想サーバーのホスト名は SV01～SV04 とした。ネットワーク環境としては、YAMAHA 製ルータ RTX-810 を利用し、プライーベートネットワークの中に試験環境を構築し、NICE には直接接続しないようにした。また、SV01 と SV02、SV03 と SV04 はそれぞれ同一のハードウェア構成とし、ストレージサーバーを個別に配置し、設定した。表 1 に仮想サーバーの購入時の主な構成部品を示す。その後メモリーの追加等適宜行っている。

表 1. 各仮想サーバーの構成.

	SV01 SV02	SV03 SV04
CPU	AMD PhenomII X4905eBOX	AMD Opteron 8-Core 6128BOX 2 基
Memory	DDR3 PC3-10600 4GBx2	DDR3 PC3-10600 4GBx8
HDD	Seagate (7200rpm 1TB)	Seagate (7200rpm 500GB)
Power Unit	SPKR5-550P(550W 80+B P)	770W 80PLUS Gold Single Power Supply
Mother Bord	Biostar TA890GXB HD	AMD SR5690+SP5100

3) 試験環境のネットワーク構成は図 1 に示したとおり、ルータ直下の仮想サーバーのネットワーク SV01～SV04 にはそれぞれ IP アドレス 192.168.1.11～192.168.1.14 を割振り、ストレージサーバーとの接続には同様に 10.10.1.11～10.10.1.14 を設定した。各ストレージサーバーの IP アドレスはストレージ 1 に 10.10.1.10、ストレージ 2 に 10.10.1.20 を設定した。仮想サーバーの OS とストレージサーバーのファームウエアは最新のバージョンにそれぞれアップデートした。またクラウド基盤試験用にテキストに基づき、ネットワークカードインターフェース冗長化のためのボンディング設定、仮想 LAN(VLAN)の設定、KVM と仮想サーバーとの通信用に仮想的なブリッジ設定をした。

4) ストレージサーバーのディスク領域設定はストレージ 1(N7700PRO)は 50%を共有用とし 50%を iSCSI にし、ストレージ 2(N8900)は 8TB あるので共有用に 1TB とし、残りを iSCSI 用に設定した。各仮想サーバー4 台に iSCSI イニシエータパッケジのインストール設定後、「iSCSI ディスクを GFS でフォーマットする」
http://anoh.s10.xrea.com/blog.php/2012/01/22/1 を参考にし、ストレージサーバーは iSCSI に gfs2 ファイルシステムを clvm ボリュームで作成した。このため、各仮想サーバーには ccs cman gfs2-utils lvm-cluster パッケージを追加インストールし設定し、ストレージサーバーの iSCSI に gfs2 ボリュームを作成し各仮想サーバーにマウントした。

2. プロビジョニングサーバー

プロビジョニングとは、必要なリソースを事前に準備しておき、必要に応じてリソースを利用できることを言います。そして、プロビジョニングを実現する方法は、２通りある。

表 2. プロビジョニングサーバーの２方式

クローニング方式	１台目の VM に通常の方法でＯＳをインストールし、２台目以降は、１台目のディスクイメージをコピーする。
自動インストール方式	VM のインストールの設定が書かれたファイルを事前に用意し、このファイルを用いて OS インストールを自動化する。

今回の研修では、OS のインストール時、ソフトウェアの選択や仮想ディスクの構成を柔軟に変更できる「自動インストール方式」を評価することとした。

プロビジョニングサーバー（自動インストール方式）は、PXE サーバーと KickStart サーバーからなる。

表 3.　プロビジョニングサーバーの構成

PXE サーバー、KickStart サーバー、それぞれの役割	
PXE サーバー	DHCP によって、下記の 2 点が行われる。 ・VM に対して PXE ブートするＩＰアドレスを付与すること。 ・PXE ブートに必要なアクセス情報を提供すること。 TFTP によって、ブートイメージなど必要な設定ファイルを提供し、インストーラの起動準備まで行う。
KickStart サーバー	インストーラが質問する回答を記述した設定ファイルを参照することで、ＯＳの自動インストールを行う。 ※設定ファイルは、HTTP で提供することで、VM からインストールパッケージを参照できるようにする。

本セッションの研修では、以下の順番でプロビジョニングサーバーの評価を行った。

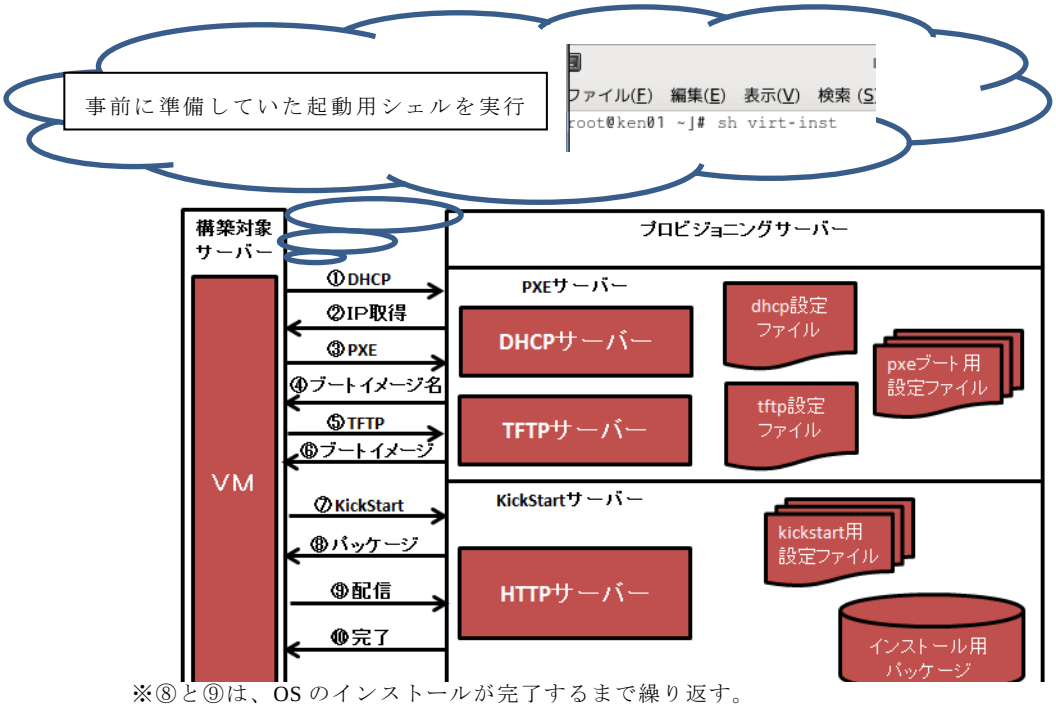
1）プロビジョニングサーバー用ＶＭの構築
2）プロビジョニングサーバー（PXE サーバー）の構築
3）プロビジョニングサーバー（KickStart サーバー）の構築
4）ＶＭの自動インストールの確認

KickStart による VM の自動インストールは、下記２つのＯＳを用いた。

表 4.　VM 自動インストールに用いたＯＳ

・Red Hat Enterprise Linux 30 日間の試用版 ・CentOS 6.3 (32bit)

プロビジョニングサーバーは下記図 2 の流れで、処理が実行される。



※⑧と⑨は、OS のインストールが完了するまで繰り返す。

図 2.　プロビジョニングサーバーの流れ

図 2 の③の PXE ブートが自動で起動できず、PXE ブートを起動されるためには、

1）起動ウインドを停止

2）起動デバイスの設定で「PXE」をアクティブにする

3）起動ウインドを開始する

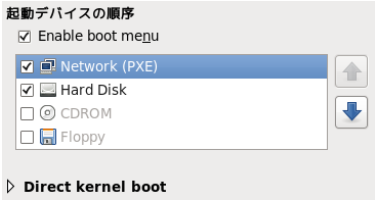


図 3.　PXE のアクティブ画面

という手順が必要となり、完全な自動化まで、実現できなかった。

④PSE ブートし、インストールイメージの選択後は、ＶＭ自動化され、OS（RedHat Linux,CentOS）のインストールに成功した。

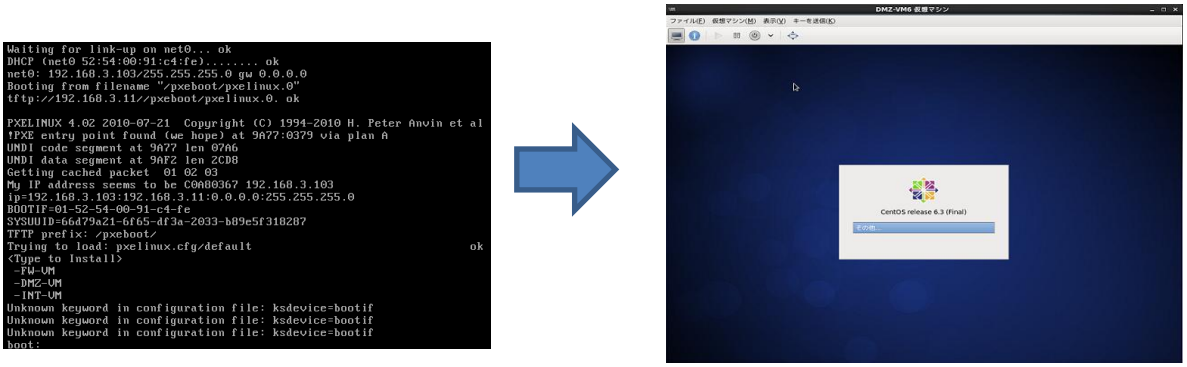


図 4.　PXE ブートから OS インストール完了まで

３．　ハイパーバイザサーバーの冗長化

仮想化によりサーバーを集約した場合、故障発生によりハイパーバイザサーバーが利用できなくなると集約したサーバー全体に影響が及んでしまう。システムの信頼性を高める手段としてはシステムの冗長化がある。ハイパーバイザサーバーの冗長化はサーバーの高可用性クラスタ構成により実現されている。本研修では冗長化構成を行い、動作を検証した。

1) フェイルオーバー

冗長化により実現できる働きは大きく分けて 2 種類ある。1 つ目はハイパーバイザサーバーが故障して仮想マシンが動作しなくなってしまった場合に、別のハイパーバイザサーバーで仮想マシンを再起動するというものである。2 つ目は運用中のハイパーバイザサーバーに関するネットワーク経路などが故障して仮想マシンを利用できなくなってしまった場合に、別の健全なハイパーバイザサーバーに仮想マシンをマイグレーションするというものである。こうしたフェイルオーバー機能により、故障からの復旧を速やかに行うことができる。

2) クラスタ構成の設定

本研修のシステムは、図 5 のようなハードウェア構成である。システムには Pacemaker と Corosync というソフトウェアを利用した。Pacemaker はリソース（ネットワーク経路、仮想マシンなど）の監視と故障発生時の制御を行う。また、Corosync はノード（ハイパーバイザ）間の通信を行う。これらを組み合わせることでクラスタを構築することができる（図 6）。

○システムの設定

- 簡単のため iptables、NetworkManager を停止させ、SELinux を無効にした。

- 共有ストレージを/guest にマウントし、仮想マシンのストレージプールに指定した。
- 相互に名前解決できるようにした。
- root にパスワードなしで ssh 接続できるようにした。

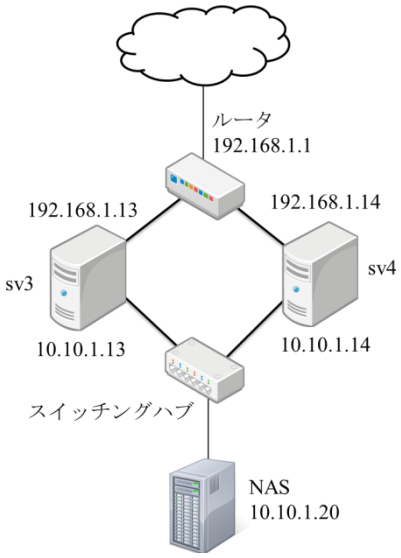


図 5. ハードウェア構成

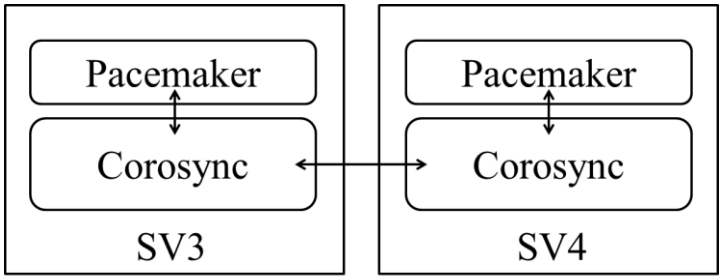


図 6. Pacemaker と Corosync

○Corosync の設定

- corosync.conf.example の bindnetaddr を 10.10.1.0 に変更して設定に使用した。

○Pacemaker の設定

- ネットワーク経路の死活監視をルータ（外側）と NAS（内側）への ping アクセスで行うようにした。
- 仮想マシン vm0 を作成してリソースとして登録した。

3) 動作検証 1（ハイパーバイザサーバーの故障）

設定を行った後、実際に故障を再現してクラスタの動作を確認した。1 つ目の検証（図 7）では sv3 で vm0 を動作させた状態で sv3 を強制的に停止させた。すると、sv3 の停止が認識された後、sv4 で vm0 が自動的に起動した。sv3 を再起動した後に vm0 をマイグレーションさせて復旧を完了することができた。これにより、ハイパーバイザが故障しても自動的にサービスを再開できることが確認できた。

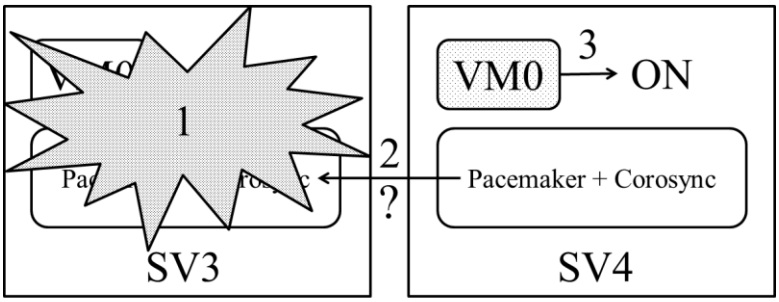


図 7. ハイパーバイザサーバーの故障

4) 動作検証 2（リソースの故障）

2 つ目の検証(図 8)では sv3 で vm0 を動作させた状態で、sv3 の LAN ケーブルを引き抜いて外側ネットワーク経路を断線させた。故障の認識後、vm0 が sv4 に自動的にマイグレーションされた。LAN ケーブルを挿し直し、vm0 を sv3 に手動でマイグレーションすることで復旧を行うことができた。これにより、リソースの故障時にも、サービスを継続できることが確認できた。

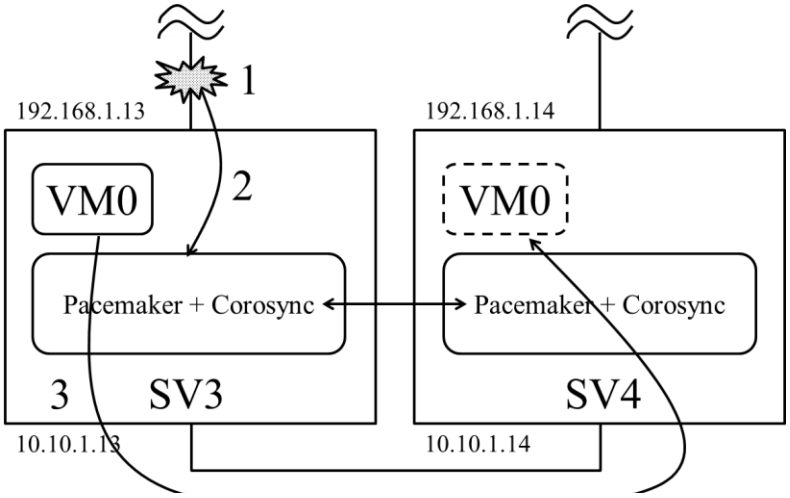


図 8. リソースの故障

4. まとめ

研修計画に沿って、クラウドの現状調査を行い、環境整備費での購入物品及び昨年度購入したサーバーを活用し、以下の事が実現できた。Scientific Linux6.3 及び CentOS 6.3 をインストールして 2 セットのテスト環境を作成した。そして、最初にゲスト OS をインストールしてライブマイグレーションを行えるようにした。次に、ゲスト OS を自動生成するためのプロビジョニングサーバーを構成して、2 種類のゲスト OS を生成することができた。最後に、安定運用のための技術である HA クラスタを実現するため、ハイパーバイザサーバーの冗長化を Pacemaker と Corosync を組み合わせて構築した。そして、片方のサーバーが故障した場合にもう一方のサーバーで、ゲスト OS を自動起動しサービスを再開できることを確認した。

この研修を行うことにより、研修参加者の仮想化技術及びクラウド技術に関する知識が深まった。特に、iSCSI 装置を用いて排他制御のためにクラスタファイルシステムを構成することが困難であること、また、フリーソフトで構成する場合に管理を自動化することが非常に困難であることが認識できた。そのため、試験的なクラウド基盤を安定的に運用するためには、VMware 等の有料ソフトを導入する必要があることがわかった。

5. 参考文献

(1) 知識ゼロから始める Linux サーバーの作り方、日経 Linux、pp162-209、（2012.1.10）