サーバ機故障時に短時間復旧を行うためのバックアップシステムの検証

藤原 冨未治、佐々木康俊、鬼頭 良彦、若松　進
工学系技術支援室 情報通信技術系

はじめに

サーバ管理業務において、トラブル発生時にサーバをできるだけ短時間で復旧させ、データを回復させることは必須条件となっている。

その方法としては rsync 等で定期的にデータのバックアップをとったり、データを保存するハードディスクを RAID 構成にする等の対策が行われているが、他にも複数のコンピュータを接続し、全体で 1 台のコンピュータのように動作させるクラスタリングという方法がある。クラスタリングには性能の向上を図るために分散処理を行う HPC (High Performance Computing)や負荷分散を行うロードバランサ、1 台が停止してもシステム全体としては処理を継続させる HA (High Availability)技術がある。本技術系研修ではバックアップ方法の習得という目的で HA クラスタリング技術の検証を行った。

1. データ更新が比較的少ないサーバのバックアップ

Web サーバのようにデータの更新は少ないがシステムは 24 時間稼働という条件の場合においては、故障の監視及び自動切換が重要になる。そこで HA クラスタリングソフトの一つである heartbeat を用いてこの問題を解決するための検証を行った。

1) heartbeat のテスト構成

検証に用いたテスト環境の構成図を図 1 に示す。テストはローカルネットワーク内で行い、稼働・待機サーバとクライアント PC はスイッチング HUB を介して同一ネットワークで接続されている。また、稼働サーバと待機サーバにはネットワークボードを追加してサーバ監視の為にクロスケーブルで直接接続している。

コンピュータの仕様を表 1 に示す。待機サーバの仕様が劣るのはサーバリプレイス時に古いサーバを利用して HA クラスタリングが可能であるかの検証も合わせて行うためである。

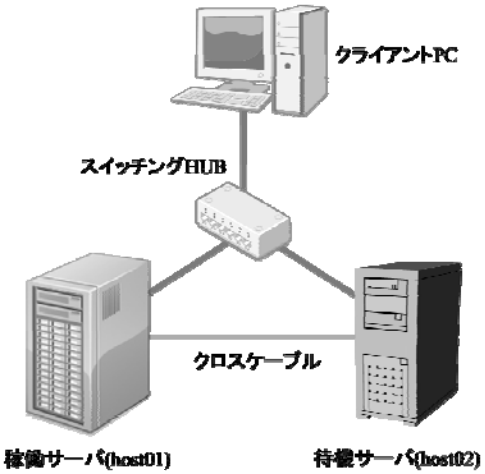

図 1.テスト環境構成図

表 1.サーバの仕様.

	稼働系サーバ	待機系サーバ
CPU	Xeon E5520	Pentium4
Clock	2.26GHz	3.0GHz
Memory	2Gx3	1G
OS	CentOS 5.5	CentOS 5.5

ソフトウェア構成として稼働サーバと待機サーバ双方に CentOS5.5、Web サーバとして Apache、HA クラスタリングとして heartbeat、heartbeat が故障した場合のクラスター自己監視プログラムとして watchdog のインストールを行う。

OS の設定でサーバにファイアウォール機能を使う場合は heartbeat 通信の為に 694udp ポートを開放しておく。

2) heartbeat の設定

設定については ha.cf、authkeys、cib.xml ファイルを編集する。表 2 に heartbeat の設定ファイルの概略、表 3 に ha.cf ファイルの設定項目の概略と本研修で設定した値を示す。

表 2. heartbeat の設定ファイル

設定ファイル		設定内容の概略
ha.cf	クラスタ基本設定ファイル	クラスタ・ノードやハートビート通信に関する設定
authkeys	ハートビート認証設定ファイル	ハートビート通信の認証に関する設定
cib.xml	クラスター詳細設定ファイル	リソース監視やフェイルオーバ・ポリシーに関する設定

表 3. ha.cf ファイルの設定項目と設定値

ha.cf ファイルの設定項目	設定値	設定項目の概略
debugfile	/var/log/ha-debug	デバッグの出力先ファイルの指定
logfile	/var/log/ha-log	ログの出力先ファイルの指定
keepalive	2	ハートビート通信の間隔（秒）
deadtime	30	ノードの不具合発生と認知する時間（秒）
deadping	40	ping の無反応時に不具合発生と認知する時間（秒）
warntime	10	ハートビート通信で不具合発生時に警告を発し始める時間（秒）
initdead	120	起動時のノード不具合発生認知時間（秒）
udpport	694	ハートビート通信のポート番号
bcast	eth1	ハートビート通信をブロードキャストするネットワークインターフェース
auto_failback	on	ノード復旧後のサービス自動起動
watchdog	/dev/watchdog	ハートビート通信異常時にシステムを再起動
node	host01.xxxx.…	クラスタに参加するノード名(hostname コマンドで表示されるもの)
	host02.xxxx.…	
respawn	root /usr/lib/heartbeat/pingd -m 100 -d 5s -a default_ping_set	

authkeys ファイルの認証方法は sha1、md5、crc の 3 種類が選択でき、sha1 が最も強固で次が md5、crc は認証は行わず誤りの検査だけを行う。研修では稼働サーバと待機サーバを直接クロスケーブルで接続するため crc を選択した。

ha.cf 、authkeys ファイルの設定を行った後、稼働・待機サーバの順で heartbeat を起動し、通信テストを行う。crm_mon コマンドでクラスターノードの状態が表示されるため、各ノードが offline から通信確立後 online になることを確認する。各サーバで heartbeat を起動すると cib.xml が/var/lib/heart/crm ディレクトリーに自動生成されるので、heartbeat を停止して cib.xml ファイルの編集を行う。cib.xml ファイルは四つの構成要素に分

表 4. cib.xml ファイルの構成要素

構成要素	定義の概略
crm_config	クラスタの全般的なオプションを定義
nodes	クラスタを構成するノードを定義
resources	使用するリソースを定義
constraints	リソースの配置、優先度、順序関係を定義

かれており、各構成要素に応じた設定を行う。表 4 に cib.xml ファイルの構成要素をを示す。

研修では仮想 IP アドレスを使った apache を監視・制御をするための編集を行った。次に cib.xml ファイル中の resources に定義した仮想 IP アドレス（192.168.1.80）で使用するための設定例を示す。

```
<primitive id="ipaddr" class="ocf" type="IPaddr" provider="heartbeat">
  <instance_attributes id="ia_ipaddr">
    <attributes>
      <nvpair id="ia_ipaddr_ip" name="ip" value="192.168.1.80"/>        （仮想 IP アドレス）
      <nvpair id="ia_ipaddr_nic" name="nic" value="eth0"/>  （ネットワーク･インターフェース）
      <nvpair id="ia_ipaddr_netmask" name="netmask" value="24"/>    （ネットマスク）
    </attributes>
  </instance_attributes>
</primitive>
```

同様に apache の設定例を示す。

```
<primitive id="apache" class="ocf" type="apache" provider="heartbeat">
  <instance_attributes id="ia_apache">
    <attributes>
      <nvpair id="ia_apache_configfile" name="configfile" value="/etc/httpd/conf/httpd.conf"/>
    </attributes>
  </instance_attributes>
</primitive>
```

仮想 IP アドレスを使用する場合、heartbeat の設定以外に apache の httpd.conf ファイルに仮想 IP アドレスの設定が必要である。

```
Listen 192.168.1.80:80
```

3) hertbeat の動作確認

稼働・待機サーバで heartbeat、apache を起動し、稼働サーバで crm_mon コマンドを使って状態を確認すると次のようにノードの状態と apache が稼働サーバで稼働していることが確認できる。

```
Node: host01.xxxx.xxxx.nagoya-u.ac.jp (**********): online
Node: host02.xxxx.xxxx.nagoya-u.ac.jp (**********): online
  Resource Group: group_apache
    ipaddr        (heartbeat::ocf:IPaddr):        Started host01.xxxx.xxxx.nagoya-u.ac.jp
    apache        (heartbeat::ocf:apache):        Started host01.xxxx.xxxx.nagoya-u.ac.jp
```

このとき稼働サーバ上で ifconfig コマンドを使ってネットワークインターフェースの状態を見ると仮想 IP アドレスで稼働していることが確認できる。

```
inet addr:192.168.1.80  Bcast:192.168.255.255  Mask:255.255.0.0
```

この状態から稼働サーバの故障を想定し、稼働サーバでネットワークケーブルの切断、復旧テストを行い、apache の待機サーバへの自動切換及び稼働サーバへの自動復旧の確認を行った。図 2 に待機サーバへの自動切換のイメージを示す。待機サーバでノードの状態を確認すると自動切換が行われたことが確認できる。

```
Node: host01.xxxx.xxxx.nagoya-u.ac.jp (**********): OFFLINE
Node: host02.xxxx.xxxx.nagoya-u.ac.jp (**********): online
```

Resource Group: group_apache
ipaddr (heartbeat::ocf:IPaddr):
Started host02.xxxx.xxxx.nagoya-u.ac.jp
apache (heartbeat::ocf:apache):
Started host02.xxxx.xxxx.nagoya-u.ac.jp

次にクラスター自己監視ソフト watchdog のテストとして heartbeat の MCP（Master Control Process）を停止させたところ、heartbeat の障害を検知して、稼働サーバが再起動を行うことを確認した。

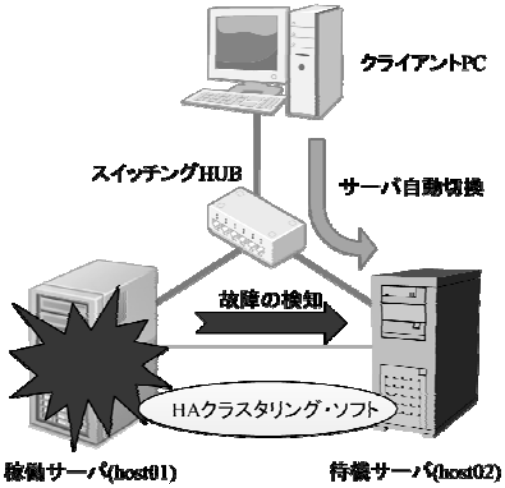

図 2. 故障検知時の自動切換

2. データ更新が頻繁なサーバのバックアップ

メール等のデータが頻繁に更新される場合は heartbeat のように故障時のサーバ切換だけでなく、データ共有等の考慮が必要となる。そこで稼働・待機サーバでデータのミラーリングを行う drbd について検証を行った。

drbd が扱えるのはブロックデバイス(例えば /dev/sda1)となるため、サーバ構築時にミラーリングを行うデバイス容量の統一に考慮した構成にする必要がある。

本研修では drbd 用デバイスとして pysical volume(LVM)10GB を稼働･待機サーバに設定を行った。

1) drbd の設定

稼働･待機サーバに drbd のインストールを行うが drbd は OS のカーネルによってパッケージが異なるため注意が必要である。設定は稼動・待機サーバの/etc/drbd.conf にサーバのホスト名、論理ボリューム名、IP アドレス等を設定する。次に drbd.conf の設定例を示す。

```
global { usage-count yes; }
common { syncer { rate 10M; } }
resource r0 {
    protocol C;
    startup {
        degr-wfc-timeout 120;
    }
    net {
        cram-hmac-alg sha1;
        shared-secret "FooFunFactory";
    }
    on   host01.xxxx.xxxx.nagoya-u.ac.jp {          （ノード 1 (host01) ホスト名）
        device      /dev/drbd0;
        disk        /dev/VolGroup00/lvol0;          （ノード 1 論理ボリューム名）
        address     192.168.1.100:7789;             （ノード 1 IP アドレス）
        meta-disk   internal;
    }
    on   host02.xxxx.xxxx.nagoya-u.ac.jp {          （ノード 2 (host02) ホスト名）
```

```
        device      /dev/drbd0;
        disk        /dev/VolGroup00/lvol0;          （ノード 2 論理ボリューム名）
        address     192.168.1.200:7789;             （ノード 2 IP アドレス）
        meta-disk   internal;
    }
}
```

設定後、drbd のメタデータ作成、起動し、ディスクの初期同期を行って同期の確認を行う。
また、ファイアウォール機能を使用している場合は使用ポートである 7789 を解放しておく。

```
/etc/init.d/drbd   status
    稼動サーバ   0:r0    Connected   Primary/Secondary   UpToDate/UpToDate   C
    待機サーバ   0:r0    Connected   Secondary/ Primary   UpToDate/UpToDate   C
```

2) heartbeat への連携

drbd ディスクを heartbeat に連携するには稼働・待機サーバの IP アドレスやディスクの情報を /root/haresources に設定して、heartbeat を起動し、デバイス情報が読み込まれたデフォルトの cib.xml ファイルを自動生成し、それを編集する。次に稼動サーバの /root/haresources を示す。

```
host01.xxxx.xxxx.nagoya-u.ac.jp ¥
                                        （ノード 1 ホスト名）
  IPaddr2::192.168.1.80/24/eth0/192.168.1.255 ¥
                                        （仮想 IP アドレス）
  drbddisk::r0 ¥                        （DRBD ディスク）
  Filesystem::/dev/drbd0::/mnt
                                        （/dev/drbd0 を/mnt へマウント）
```

図 3 に heartbeat と drbd を利用したクラスタ構成を示す。

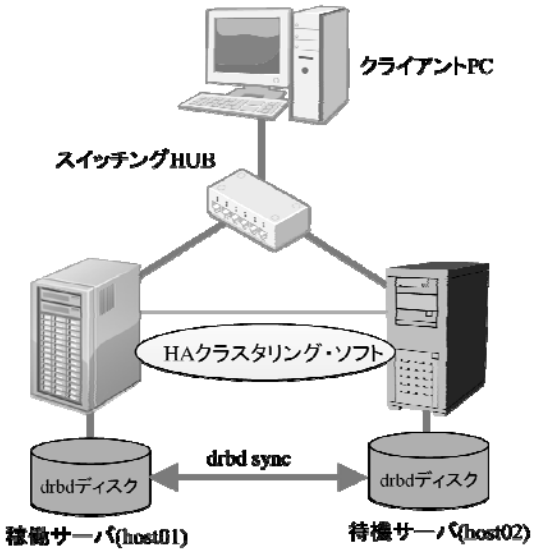

図 3. heartbeat と drbd を利用したクラスタ構成

図 3 の drbd ディスク部分に Web データを設定し、1. 3) と同じように heartbeat の動作確認を行い、待機サーバへの移行及び稼動サーバへの復旧、仮想 IP アドレスの設定、drbd ディスクのマウントを確認し、クライアント PC からも Web の状況を確認する。

3. heartbeat と iSCSI(NAS)の連携

NAS はネットワークに直接接続して使用するファイルサーバ機で、単体で RAID 機能を有しているのでバックアップ機としてよく利用されている。今回 NAS の中でも iSCSI（ネットワーク経由で SCSI 機器として認識）機能を持った NAS を利用し稼働サーバ及び、待機サーバのデータディスクとして認識させることによって heartbeat によるサーバ機の切換のみでデータが共通

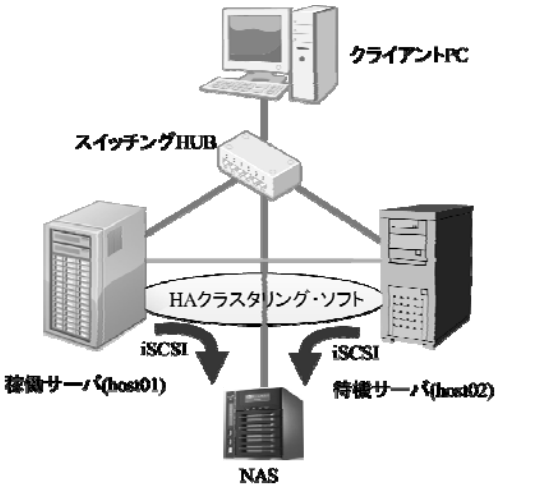

図 4. heartbeat と iSCSI を利用したクラスタ構成

に利用できる利点がある。図４に heartbeat と iSCSI を利用したクラスタ構成を示す。

1) NAS の iSCSI 設定

研修に利用した NAS は Web による設定機能を有しており iSCSI として使用するため、Web によって次の設定を行う。

adminID : admin、　　password : ********、
IP address : 192.168.1.111　　Network : 255.255.255.0、
GatewayIP : 192.168.1.1　　ホスト名: N4200、
ドメイン名: thecus.com、　　リンクアグレゲーション: フェイルオーバー、
RAID レベル:RAID6、　　RAID ID: RAID6

NAS の機能として RAID６（障害時にディスク 2 本故障しても大丈夫）があるのでそのテストも行った。

iSCSI のターゲット名は iqn.2010-10.com.thecus:RAID6.iscsi0.vg0.n4200 である。

2) サーバ機の iSCSI 設定

サーバ機の OS から iSCSI ターゲットディスクにアクセスできるようにするためイニシエータをインストールし、 iSCSI ターゲットの iqn の指定を行う。

```
/etc/iscsi/initiatorname
    initiatorname=iqn.2010-10.com.thecus:RAID6.iscsi0.vg0.n4200
```

iSCSI を起動し、イニシエータからターゲットへの接続を行い、SCSI 接続デバイスで認識されていることを確認する。

接続した iSCSI ディスクを Linux LVM（logical volume manager）形式で使用するために LVM パーティションを作成し、PV、VG、LV の作成を行い、マウントのテストを行う。

3) heartbeat と iSCSI の連携

drbd と heartbeat の連携時のように IP アドレスやディスクの情報を/root/haresources-iscsi に設定して、heartbeat を起動し、次のコマンドで iSCSI のデバイス情報を追加した、cib.xml を作成したファイルを編集する。

```
/usr/lib/heartbeat/haresources2cib.py　/root/haresources-iscsi
```

次に/root/haresources-iscsi を示す。

```
稼働サーバ:　host01.xxxx.xxxx.nagoya-u.ac.jp ¥　　（稼働側ホスト名）
　　IPaddr2::192.168.1.80 ¥　　（仮想 IP アドレス）
　　Filesystem::/dev/VolGroup01/lvol0::/mnt/iscsi ¥　　（マウント）
　　httpd
待機サーバ:　host02.xxxx.xxxx.nagoya-u.ac.jp ¥　　（待機側ホスト名）
　　IPaddr2::192.168.1.80 ¥
　　Filesystem::/dev/VolGroup01/lvol0::/mnt/iscsi ¥
　　httpd
```

cib.xml の編集終了後 heartbeat を起動し、heartbeat の状態の確認とマウント状態を確認する。

稼働サーバの heartbeat の状態

```
crm_mon
    Node: host01.xxxx.xxxx.nagoya-u.ac.jp (**********): online
    Node: host02.xxxx.xxxx.nagoya-u.ac.jp (**********): online
```

Resource Group: group_apache

IPaddr2_1	(heartbeat::ocf:IPaddr2):	Started host01.xxxx.xxxx.nagoya-u.ac.jp
Filesystem_2	(heartbeat::ocf:Filesystem):	Started host01.xxxx.xxxx.nagoya-u.ac.jp
httpd_3	(lsb:httpd):	Started host01.xxxx.xxxx.nagoya-u.ac.jp

ディスクのマウント状態

df

/dev/mapper/VolGroup00-LogVol00	***	***	***	/
/dev/sda1	***	***	***	/boot
tmpfs	***	*	***	/dev/shm
/dev/mapper/VolGroup01-lvol0	***	***	***	/mnt/iscsi

heartbeat と iSCSI の動作確認を 1. 3) と同じように行い、待機サーバへの移行及び稼動サーバへの復旧、仮想 IP アドレスの設定、iSCSI ディスクのマウントを確認し、クライアント PC からも Web の動作を確認する。

4. まとめ

サーバ機故障時のバックアップシステムとして HA クラスタシステムの構築手法及びソフトの設定方法、ネットワーク経由のディスクのミラーリングの検証を行うとともに技術の習得ができた。

また、HA クラスタシステムに iSCSI を組み合わせることによりデータ領域をネットワーク上で構成し、ストレージの冗長化を行うことが出来ることが確認できた。

待機サーバに利用した程度の機器であればリプレース後の古いサーバにディスク等を追加するだけで HA クラスタシステムの構築を行えることが確認できたので、今後の依頼業務時にシステムの信頼性向上のための提案として挙げることが可能となった。

5. 参考文献

1) Linux で作るアドバンストシステム構築ガイド, デージーネット著, 秀和システム

2) Linux 高信頼サーバ構築ガイドシングルサーバ編, 笠野英松著, CQ 出版社

3) 実際に作って理解する！ Linux サーバーのクラスタリング, ITpro,
http://itpro.nikkeibp.co.jp/article/COLUMN/20081009/316517/

4) 連載記事 「Heartbeat でかんたんクラスタリング」, @IT,
http://www.atmarkit.co.jp/flinux/index/indexfiles/heartbeatindex.html

5) HA クラスタシステム構築(Heartbeat+DRBD+Apache)
http://centossrv.com/heartbeat-drbd.shtml