

ディザスタリカバリについての苦労話

(株)創風システム 外山 文規

今日のおはなし

- 弊社のディザスタリカバリ環境の変遷
- DRBDに関するいろいろ
- DRBD Proxyを試してみた

今日のおはなし

- 弊社のディザスタリカバリ環境の変遷
- DRBDに関するいろいろ
- DRBD Proxyを試してみた

ディザスタリカバリってなに？

ディザスタリカバリ(英語: disaster recovery)とは、災害などによる被害からの回復措置、あるいは被害を最小限の抑えるための予防措置のこと。

(by Wikipedia)

参考URL

- 内閣府:「事業継続ガイドライン」

<http://www.bousai.go.jp/kigyoubousai/jigyuu/hajimete.html>

- 経済産業省:「ITサービス継続ガイドライン」

<http://www.meti.go.jp/press/20080903001/20080903001.html>

- 国内の業務継続計画に関する情報:

<http://www.bousai.go.jp/jishin/gyomukeizoku/kokunai-link.html>

大きく見直しが行われた転換点

- 2007年7月16日 中越沖地震発生
 - ・ 柏崎市内 本社社屋(メイン施設)、旧社屋(バックアップ施設)が被災
 - ・ 特に旧社屋は甚大な被害
 - > 地震により事業継続に支障が出ることを露呈
 - > 自社の「規模」でできる地震に対するディザスタリカバリ環境の見直しを実施

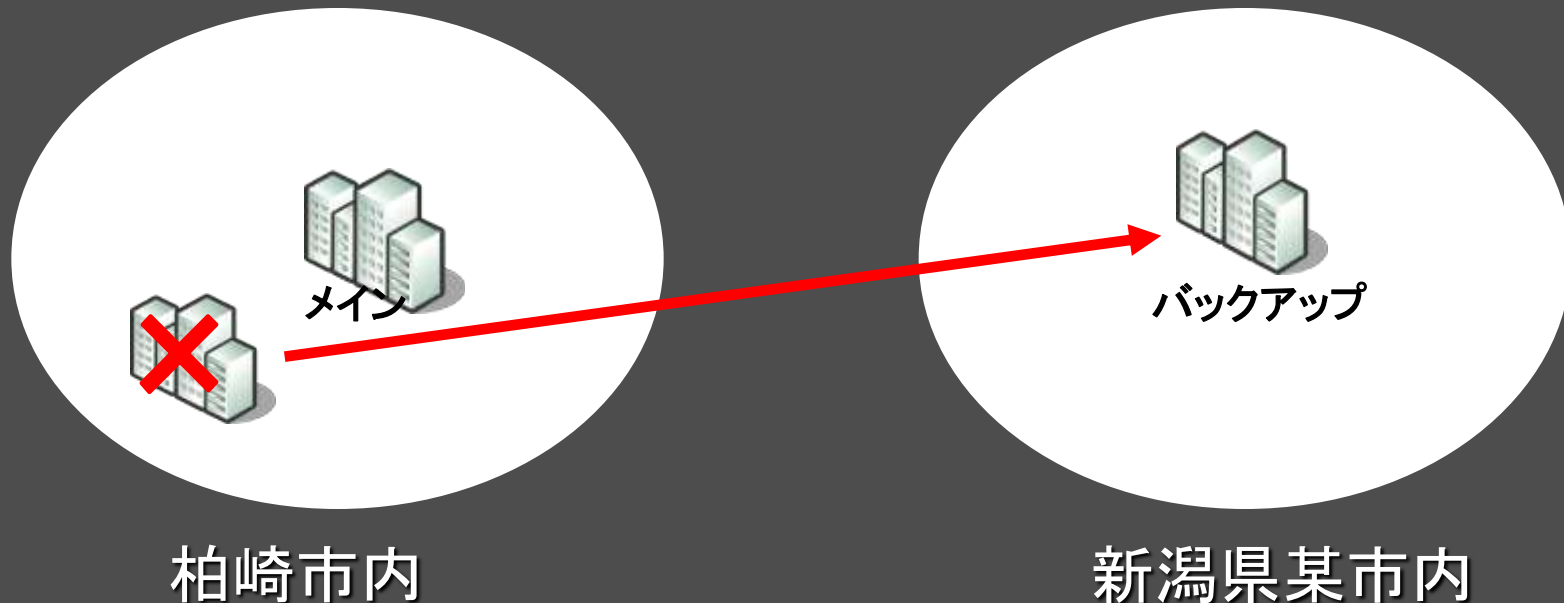






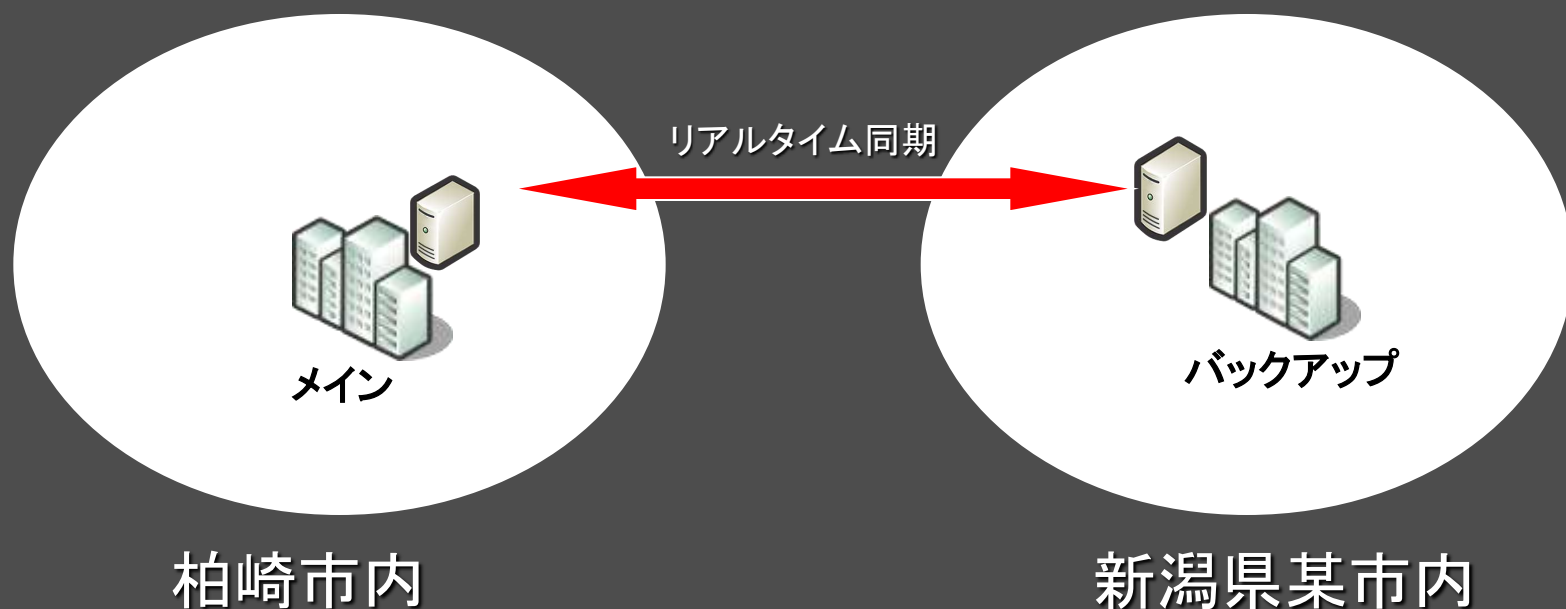
地震に対する事業継続対策

- 2007年 バックアップ施設を移転



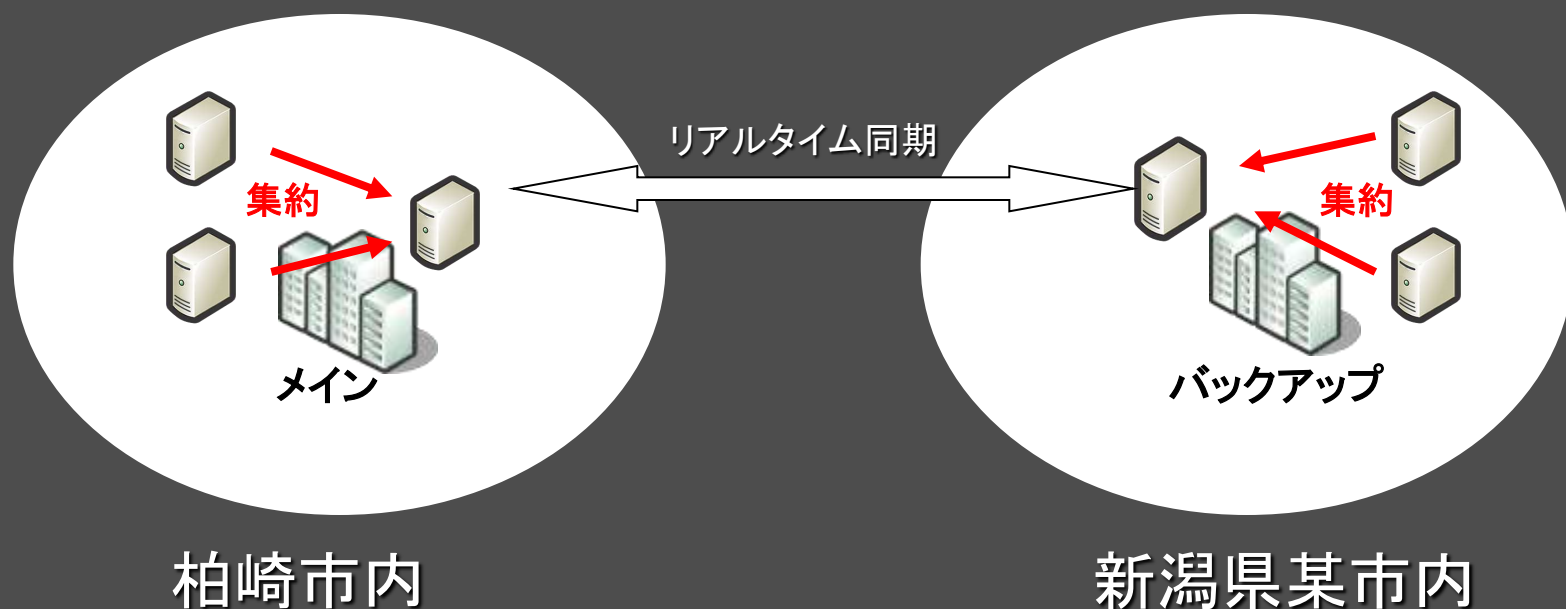
地震に対する事業継続対策

- 2008年 一部のデータをリアルタイム同期



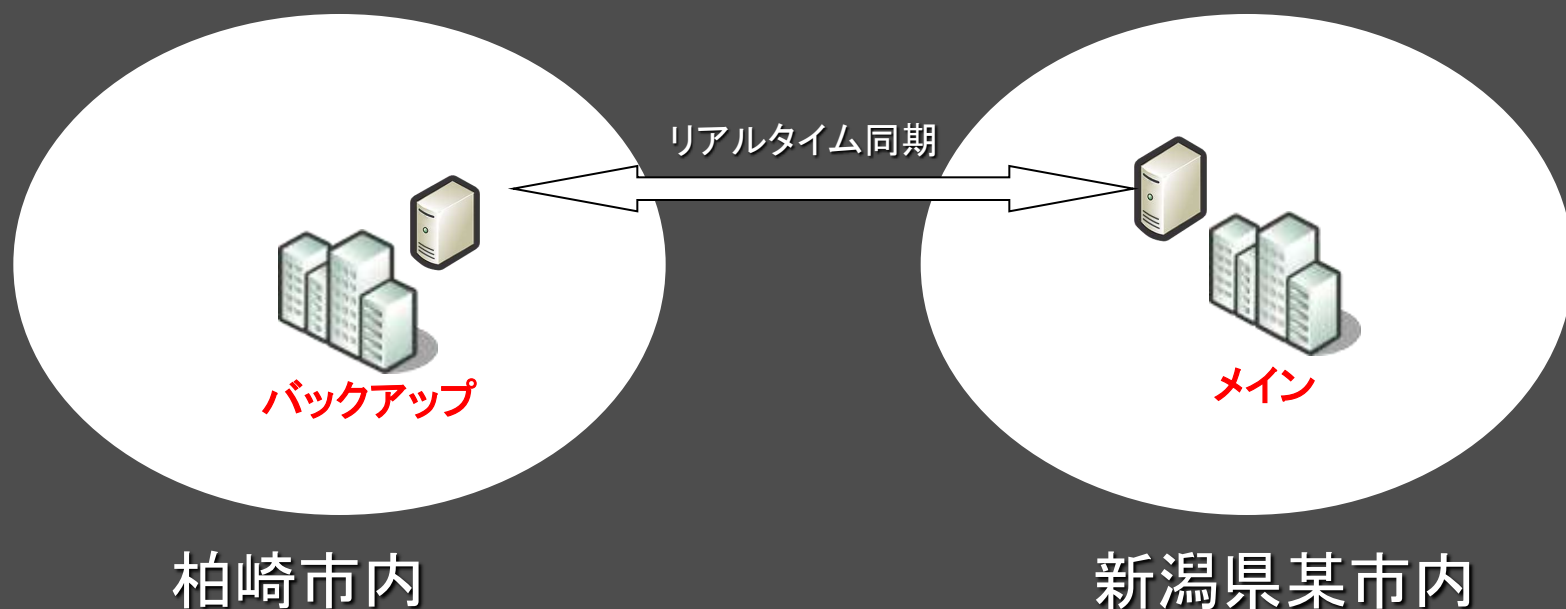
地震に対する事業継続対策

- 2008～2010年 サーバの仮想化推進



地震に対する事業継続対策

- 2010年 主要サービスを提供する施設を交換



リアルタイム同期が一番いい方法？

広帯域のネットワークを必要とします

- NW帯域が狭いとパフォーマンスも低くなる
- 非同期のデータ転送より回線コスト↑
- データのリカバリポイントやシステムの重要度によって同期方法は選びましょう

今日のおはなし

- 弊社のディザスタリカバリ環境の変遷
- DRBDに関するいろいろ
- DRBD Proxyを試してみた

DRBDって何？

- DRBD: Distributed Replicated Block Device
- ネットワーク経由でディスクミラーリングを実現
- 複数のブロックデバイスの複製が可能
- Active/Active構成も可能
- off-site node(3ノード)のレプリケーションも可能
- Linuxベースの環境で動作
- OSS、GPLv2ライセンス

DRBDベンチマーク計測環境

(node01)# mount /dev/drbd0 /share
/shareに対して、bonnie++ 1.03eでベンチマーク
※/dev/drbd0はext3でフォーマット



host: node01.example.com
CPU: Celeron D 347(3.06 GHz)
MEM: 1.5G
HDD: SATA300 7200rpm 8MB
NIC: INTEL 82566DM(1G)
OS: CentOS 5.6 x86_64
DRBD: 8.3.8-1(Primary)

host: node02.example.com
CPU: Celeron D 347(3.06 GHz)
MEM: 1.5G
HDD: SATA300 7200rpm 8MB
NIC: INTEL 82566DM(1G)
OS: CentOS 5.6 x86_64
DRBD: 8.3.8-1(Secondary)

DRBDの設定

■ /etc/drbd.conf

```
include "drbd.d/global_common.conf";  
include "drbd.d/*.res";
```

■ /etc/drbd.d/global_common.conf

```
global {  
    usage-count yes;  
}  
common {  
    protocol C;  
    handlers {  
        pri-on-incon-degr "/usr/lib/drbd/notify-pri-on-incon-degr.sh; /usr/lib/drbd/notify-emergency-reboot.sh; echo b > /proc/sysrq-trigger  
; reboot -f";  
        pri-lost-after-sb "/usr/lib/drbd/notify-pri-lost-after-sb.sh; /usr/lib/drbd/notify-emergency-reboot.sh; echo b > /proc/sysrq-trigger ;  
reboot -f";  
        local-io-error "/usr/lib/drbd/notify-io-error.sh; /usr/lib/drbd/notify-emergency-shutdown.sh; echo o > /proc/sysrq-trigger ; halt -f";  
    }  
    startup {  
    }  
    disk {  
    }  
    net {  
    }  
    syncer {  
    }  
}
```

DRBDの設定

■ /etc/drbd.d/sda6.res

```
resource r0 {  
    startup {  
        wfc-timeout 120;  
    }  
    on node01.example.com {  
        device /dev/drbd0;  
        disk /dev/sda6;  
        address 10.223.114.100:7777;  
        meta-disk internal;  
    }  
    on node02.example.com {  
        device /dev/drbd0;  
        disk /dev/sda6;  
        address 10.223.114.101:7777;  
        meta-disk internal;  
    }  
}
```

DRBDのパフォーマンス

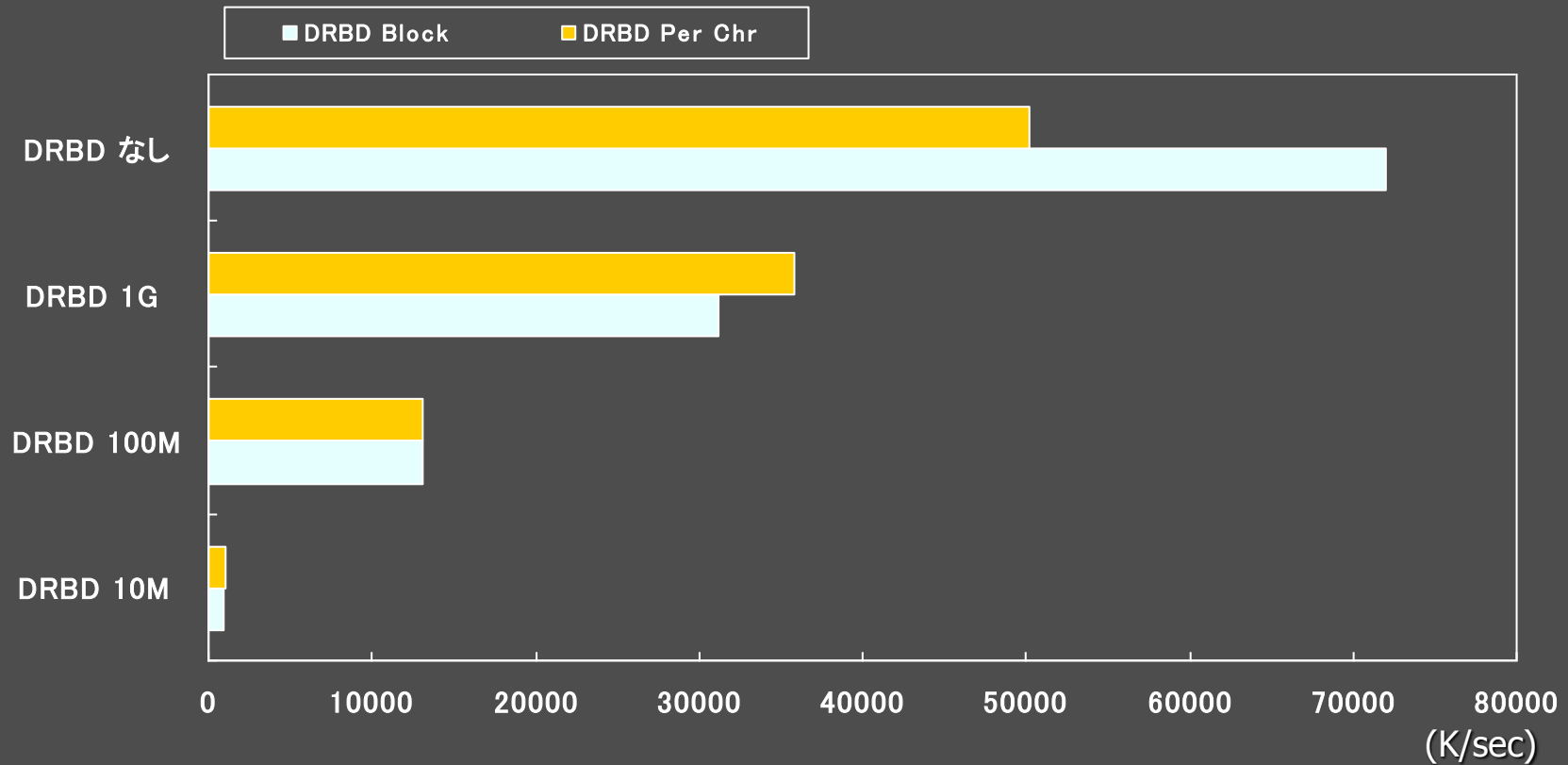
	Sequential Output (Write)			Sequential Input (Read)		Random Seeks (/sec)
	Per Chr (K/sec)	Block (K/sec)	Rewrite (K/sec)	Per Chr (K/sec)	Block (K/sec)	
DRBDなし	50167	71980	28095	47431	74422	192
DRBD (1G)	35853	31210	21784	50180	73895	131
DRBD (100M)	13102	13134	9079	49502	74593	130
DRBD (10M)	1031	948	903	4654	30564	128

※速度は、ethtoolでリンク速度を変更しました。

※3回計測した平均値したものを少数第一位四捨五入

NW帯域別書き込み性能

Sequential Output



DRBDのレプリケーション以外の通信

再同期=RAIDでいうリビルド

再同期のための帯域制御

rateという設定項目があります

- rateという設定項目で最大利用帯域を設定できる
- rateはコマンドで動的に変更もできる
- rateの推奨値は有効帯域幅の30%

例. rateを12Mに変更

```
(node01)# drbdsetup /dev/drbd0 syncer -r 12M
```

```
(node01)# drbdsetup /dev/drbd0 show
```

再同期中のパフォーマンス

	Sequential Output (Write)			Sequential Input (Read)		Random Seeks (/sec)
	Per Chr (K/sec)	Block (K/sec)	Rewrite (K/sec)	Per Chr (K/sec)	Block (K/sec)	
1G 再同期通信なし	35853	31210	21784	50180	73895	131
1G rate 125M(Byte) 再同期通信中	17486	15416	9769	20568	25671	102
1G rate 35M(Byte) 再同期通信中	17323	18139	8836	21254	28503	97
1G rate 12M(Byte) 再同期通信中	28907	20866	15943	31886	43441	106

※ **1** 回計測した平均値したものを少数第一位四捨五入
横着しました(><)

再同期に掛かった時間

負荷なしで約29GBを再同期した結果

	完了までの時間 (sec)	平均転送速度 (K/sec)
1G rate 125M(Byte)	585(約10分)	50088
1G rate 35M(Byte)	819(約14分)	35776
1G rate 12M(Byte)	2349(約40分)	12472

WAN環境でのDRBDってどう？

- 書き込み性能を求められるとちょっとつらい
- 信頼性の高い広帯域のネットワークを推奨
- 再同期についてもちょっと考えておく
- rate設定は適切に
- RAIDとおなじで別途バックアップは必要
- WAN回線でTBレベルのレプリケーションは現実的？

WAN環境でHA+DRBDを使う 想像してみた

Heartbeatってなに？

- HAクラスタリングを実現するソフトウェア
- サーバ間で生存監視を行う
- 障害(サービス停止等)を検知して自動的にActiveノードを切り替える
- 生存確認はネットワーク、シリアルケーブルなど
- DRBDをサポート
- オープンソースソフトウェア

WAN環境でHA構成をとる場合の問題

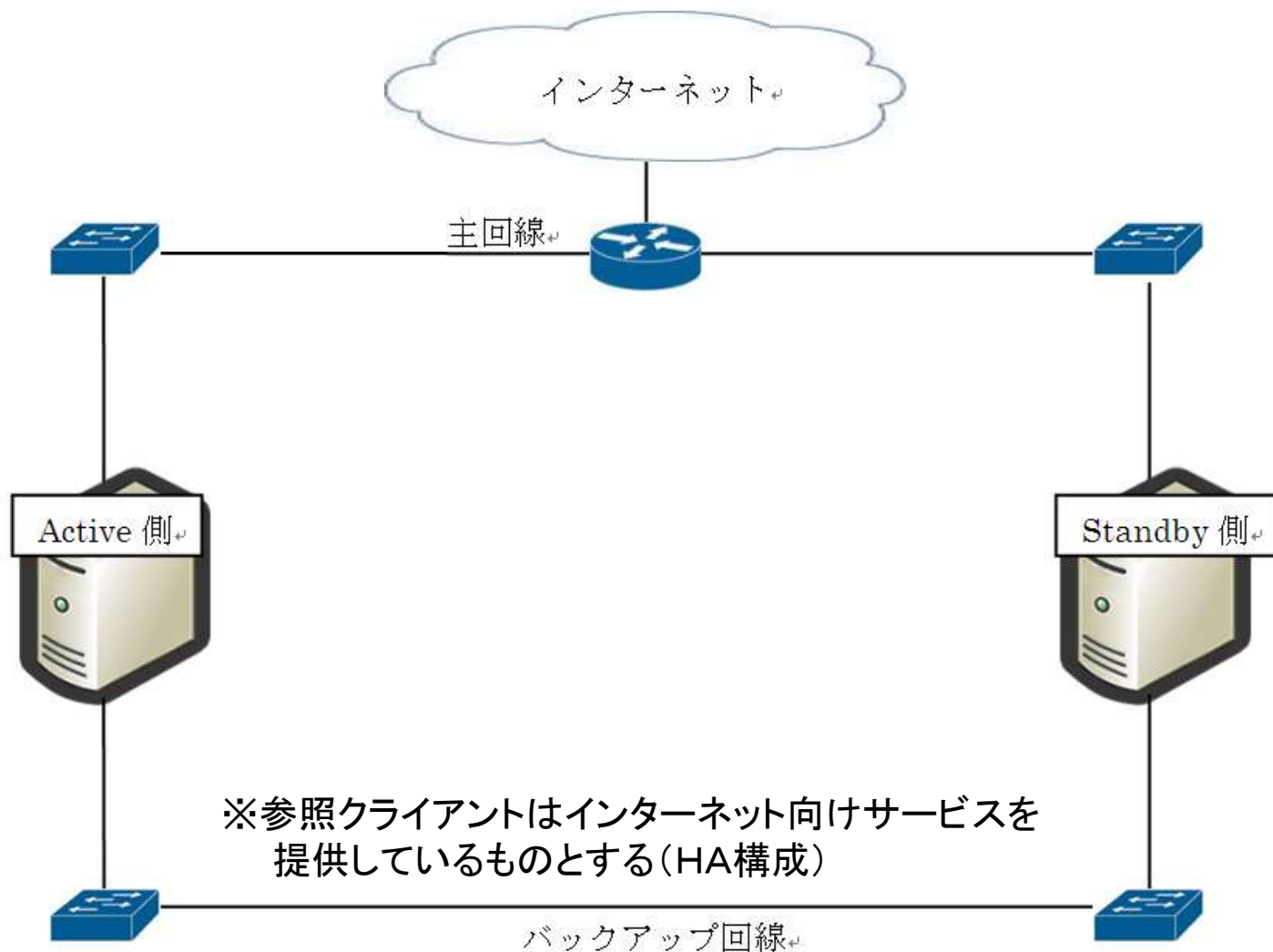
- LAN環境に比べてスプリットブレンが起きやすい
 - >データの不整合発生
- なぜ、起きやすい？
 - >シリアルケーブル使えない
 - >ネットワークダウン⇔相手ノードダウン

それでもWAN環境で使うなら

- スプリットブレインを起こしにくくする対策をとる
 - > 別システムのネットワークを2回線以上用意する
 - > ネットワークが不通の場合に降格する設定も活用する

不測の事態としてスプリットブレインが起きる覚悟はしておく！！

想像したネットワーク



ノード切替条件を想像

条件	動作
相手ノードに障害が発生した場合 (提供サービス停止、ディスク障害、共有IP異常)	Standby側が昇格
主回線がダウン	そのまま
バックアップ回線がダウン	そのまま
メイン施設側全ネットワークダウン	Active側降格、昇格抑止 Standby側昇格 NW復旧時昇格させない
バックアップ施設側全ネットワークダウン	Standby側昇格抑止 NW復旧時昇格させない
全施設全ネットワークダウン	両ノード降格 NW復旧時昇格させない

今日のおはなし

- 弊社のディザスタリカバリ環境の変遷
- DRBDに関するいろいろ
- DRBD Proxyを試してみた

DRBD Proxyってなに？

- その名の通り、Proxyとして動作
- 非同期通信
- 同期するデータのバッファリング
- 同期するデータの圧縮、伸張を行い通信帯域を節約
- DRBDが動作する両ノードにインストールが必要
- 有償ソフトウェア（評価ライセンス有り）

DRBD Proxyのインストール

■ インストール (CentOS5の場合)

(node01)# wget 指定されたURLよりダウンロード

(node01)# rpm -ivh drbd-proxy-1.0.16-1.x86_64.rpm

■ ライセンスキーの配置

発行されたライセンスキーファイルをサーバにコピー

/etc/ に drbd-proxy.license という名前で配置

※同様にnode02側にもDRBD Proxyをインストールする必要があります。

DRBD Proxyの設定

■ /etc/drbd.conf

```
include "drbd.d/global_common.conf";  
include "drbd.d/*.res";
```

■ /etc/drbd.d/global_common.conf

```
global {  
    usage-count yes;  
}  
common {  
    protocol A;  
    handlers {  
        pri-on-incon-degr "/usr/lib/drbd/notify-pri-on-incon-degr.sh; /usr/lib/drbd/notify-emergency-reboot.sh; echo b > /proc/sysrq-trigger  
; reboot -f";  
        pri-lost-after-sb "/usr/lib/drbd/notify-pri-lost-after-sb.sh; /usr/lib/drbd/notify-emergency-reboot.sh; echo b > /proc/sysrq-trigger ;  
reboot -f";  
        local-io-error "/usr/lib/drbd/notify-io-error.sh; /usr/lib/drbd/notify-emergency-shutdown.sh; echo o > /proc/sysrq-trigger ; halt -f";  
    }  
    startup {  
    }  
    disk {  
    }  
    net {  
    }  
    syncer {  
    }  
}
```

DRBD Proxyの設定

■ /etc/drbd.d/sda6.res

```
resource r0 {
  startup {
    wfc-timeout 120;
  }
  proxy {
    compression on;
    memlimit 100M;
  }

  on node01.example.com {
    device /dev/drbd0;
    disk /dev/sda6;
    address 127.0.0.1:7777;
    meta-disk internal;
    proxy on node01.example.com {
      inside 127.0.0.1:7778;
      outside 10.223.114.100:7778;
    }
  }
  on node02.example.com {
    device /dev/drbd0;
    disk /dev/sda6;
    address 127.0.0.1:7777;
    meta-disk internal;
    proxy on node02.example.com {
      inside 127.0.0.1:7778;
      outside 10.223.114.101:7778;
    }
  }
}
```

DRBD Proxyの起動

■ DRBDの再起動

```
(node01)# service drbdproxy start
```

```
(node02)# service drbdproxy start
```

```
(node01)# service drbd start
```

```
(node02)# service drbd start
```

```
(node01)# drbd-overview
```

同期が取れているか確認

DRBD Proxyベンチマーク計測環境

(node01)# mount /dev/drbd0 /share
/shareに対して、bonnie++ 1.03eでベンチマーク
※/dev/drbd0はext3でフォーマット



host: node01.example.com
CPU: Celeron D 347(3.06 GHz)
MEM: 1.5G
HDD: SATA300 7200rpm 8MB
NIC: INTEL 82566DM(1G)
OS: CentOS 5.6 x86_64
DRBD: 8.3.8-1(Active)

host: node02.example.com
CPU: Celeron D 347(3.06 GHz)
MEM: 1.5G
HDD: SATA300 7200rpm 8MB
NIC: INTEL 82566DM(1G)
OS: CentOS 5.6 x86_64
DRBD: 8.3.8-1(Standby)

DRBDベンチマーク環境

- 参考書籍:

オープンソース・ソフトウェアルータ Vyatta入門



VyattaでNW遅延をエミュレーション

ネットワーク遅延を50msに設定

```
$ configuration
# set interfaces bridge br0
# commit
# set interfaces ethernet eth0 bridge-group bridge br0
# set interfaces ethernet eth1 bridge-group bridge br0
# commit
# set traffic-policy network-emulator EMU network-delay 50ms
# set interfaces ethernet eth1 traffic-policy out EMU
# commit
# save
# exit
```

Vyattaで通信遅延をエミュレーション

node01からnode02へPingした結果

(node01)# ping node02

- network-delay なし
平均 0.48ms
- network-delay 10ms
平均 15ms
- network-delay 50ms
平均 53ms
- network-delay 100ms
平均 108ms
- network-delay 200ms
平均 211ms
- network-delay 300ms
平均 301ms

VyattaでNW速度をエミュレーション

ネットワーク速度を75mbpsに設定

```
$ configuration
# set interfaces bridge br0
# commit
# set interfaces ethernet eth0 bridge-group bridge br0
# set interfaces ethernet eth1 bridge-group bridge br0
# commit
# set traffic-policy network-emulator EMU bandwidth 75mbit
# set interfaces ethernet eth1 traffic-policy out EMU
# commit
# save
# exit
```

Vyattaで特定通信速度をエミュレーション

node01からnode02へnetperfで調査した結果

(node01)# netperf -p 10000 -H node02

- bandwidth なし
平均 94(10^6 bits/sec)
- bandwidth 50mbit
平均 43(10^6 bits/sec)
- bandwidth 20mbit
平均 16(10^6 bits/sec)
- bandwidth 10mbit
平均 9.3(10^6 bits/sec)
- bandwidth 5mbit
平均 4.8(10^6 bits/sec)
- bandwidth 1mbit
平均 0.96(10^6 bits/sec)

DRBD NW遅延別結果

	Sequential Output(Write)			Sequential Input(Read)		Random Seeks (/sec)
	Per Chr (K/sec)	Block (K/sec)	Rewrite (K/sec)	Per Chr (K/sec)	Block (K/sec)	
DRBD (Vyatta ブリッジ 100M)	13076	13407	9026	48492	73948	131
DRBD (100M、delay 10ms)	12717	12594	9459	49829	73779	125
DRBD (100M、delay 50ms)	12414	12352	8980	49878	73590	126
DRBD (100M、delay 100ms)	8625	8873	6344	47343	73381	127
DRBD (100M、delay 200ms)	6631	6594	4845	42219	71214	124
DRBD (100M、delay 300ms)	4573	4634	3355	28741	71493	125

※3回計測した平均値したものを少数第一位四捨五入

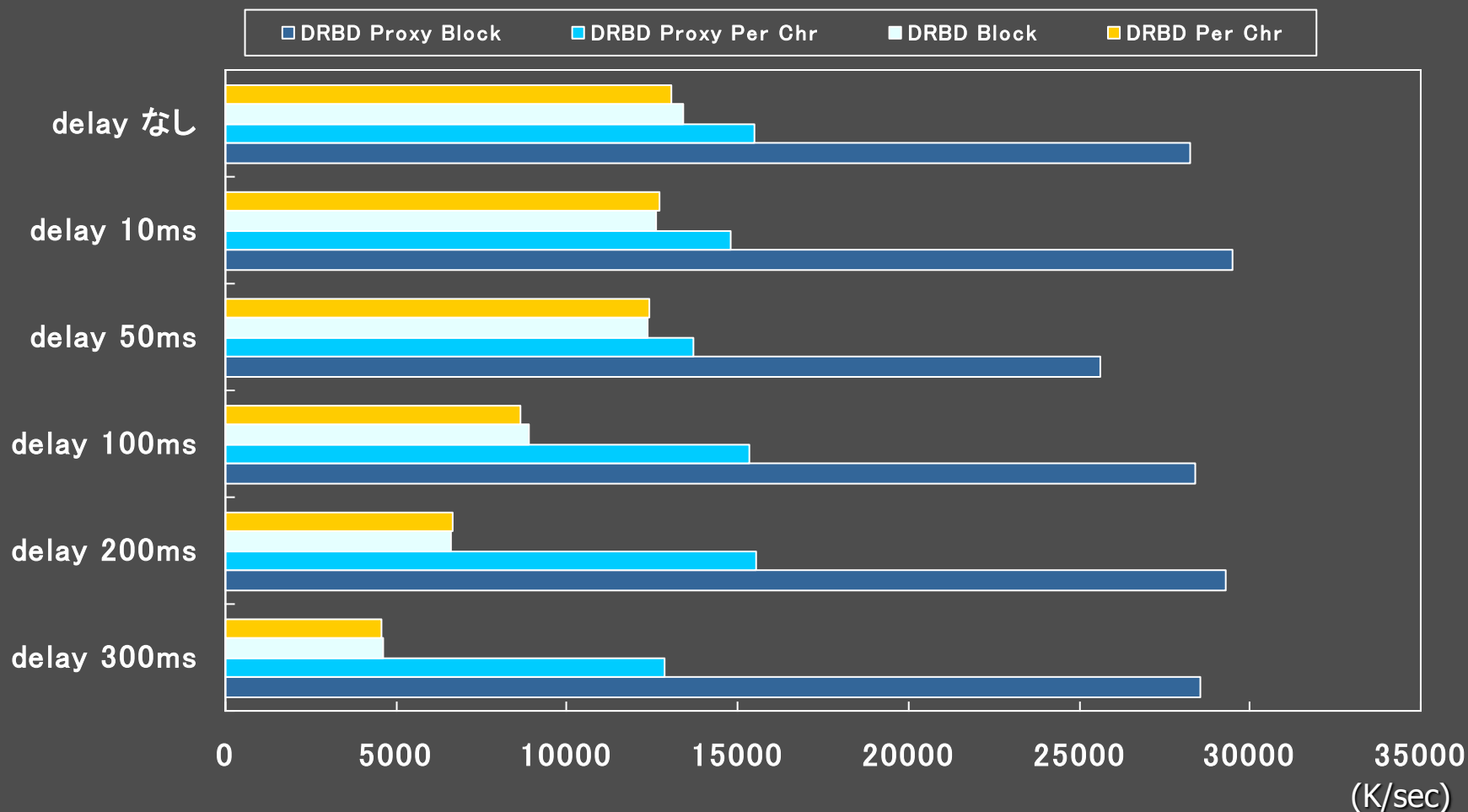
DRBD Proxy NW遅延別結果

	Sequential Output (Write)			Sequential Input (Read)		Random Seeks (/sec)
	Per Chr (K/sec)	Block (K/sec)	Rewrite (K/sec)	Per Chr (K/sec)	Block (K/sec)	
DRBD (Vyatta ブリッジ 100M)	13076	13407	9026	48492	73948	131
DRBD Proxy (100M、delay なし)	15484	28225	16233	44657	73464	120
DRBD Proxy (100M、delay 10ms)	14779	29486	16526	45921	73378	120
DRBD Proxy (100M、delay 50ms)	13712	25601	17062	45498	72965	120
DRBD Proxy (100M、delay 100ms)	15358	28411	16326	44015	73512	120
DRBD Proxy (100M、delay 200ms)	15519	29301	16068	48195	73487	121
DRBD Proxy (100M、delay 300ms)	12872	28563	15340	48994	72925	119

※3回計測した平均値したものを少数第一位四捨五入

NW遅延環境での書き込み性能

Sequential Output



DRBD NW帯域別結果

	Sequential Output (Write)			Sequential Input (Read)		Random Seeks (/sec)
	Per Chr (K/sec)	Block (K/sec)	Rewrite (K/sec)	Per Chr (K/sec)	Block (K/sec)	
DRBD (Vyatta ブリッジ 100M)	13076	13407	9026	48492	73948	131
DRBD (bandwidth 75Mbps)	9776	9899	7327	47851	74478	127
DRBD (bandwidth 50Mbps)	6843	6983	4848	43891	74032	132
DRBD (bandwidth 20Mbps)	2655	2672	1907	27171	74368	127
DRBD (bandwidth 10Mbps)	1380	1353	1033	9456	73712	128
DRBD (bandwidth 5Mbps)	706	704	528	6518	73786	126
DRBD (bandwidth 1Mbps)	6 ベンチマークを実行して暫くして、同期が崩れました。					

※3回計測した平均値したものを少数第一位四捨五入

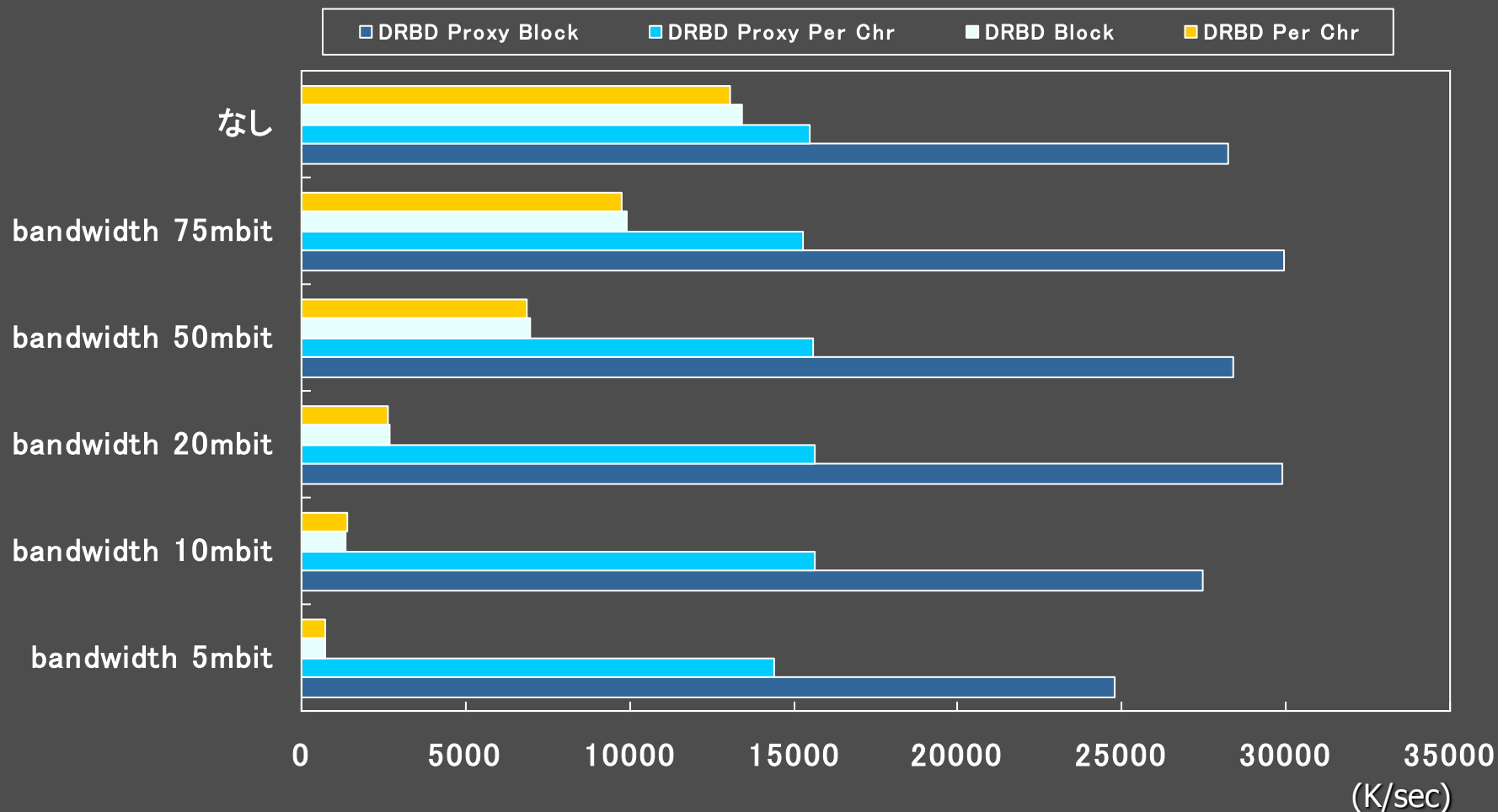
DRBD Proxy NW帯域別結果

	Sequential Output(Write)			Sequential Input(Read)		Random Seeks (/sec)
	Per Chr (K/sec)	Block (K/sec)	Rewrite (K/sec)	Per Chr (K/sec)	Block (K/sec)	
DRBD (Vyatta ブリッジ 100M)	13076	13407	9026	48492	73948	131
DRBD Proxy (bandwidth 75Mbps)	15292	29937	15932	46934	71671	117
DRBD Proxy (bandwidth 50Mbps)	15576	28369	16019	47876	73368	118
DRBD Proxy (bandwidth 20Mbps)	15623	29882	14964	46674	73553	117
DRBD Proxy (bandwidth 10Mbps)	15661	27454	16248	45349	73420	121
DRBD Proxy (bandwidth 5Mbps)	14398	24771	17234	49970	73127	122
DRBD (bandwidth 1Mbps)	ベンチマークを実行して暫くして、同期が崩れました。					

※3回計測した平均値したものを少数第一位四捨五入

NW帯域別の書き込み性能

Sequential Output



DRBD Proxyを試してみても

- 確かに、遅延や狭帯域で効果を発揮
- 圧縮、伸張でCPUをたくさん使います
- CPUがよければもっとパフォーマンスが出たのかも？
- 最低でもフレッツ光程度の帯域は欲しい
- 時間切れで再同期についてチェックし忘れました
- お勧めのネットワークシミュレータってありますか？

参考URL

DRBD:

<http://www.drbd.org/>

株式会社サードウェアさん

<http://www.3ware.co.jp/>

おしまい