可用性とパフォーマンスを重視したLDAP

セミナーに参加いただいた皆様へ
当日は質疑応答の時間をとれず，ご迷惑をおかけしました．
当日聞き逃したこと等はユーザ会や私宛に質問いただければ
回答できる範囲で対応したいと思います．

日本LDAPユーザ会
中満英生
nomo@bluecoara.net

自己紹介

- 氏名： 中満英生
- 会社： F5 ネットワークスジャパン
- 所属： プリセールスエンジニア
- 活動： Solaris , Apache , Postfix , LDAP , その他 OSS
  - 最近活動できてない・・・
- ウェブでの記事
  - http://gihyo.jp/admin/serial/01/ldap

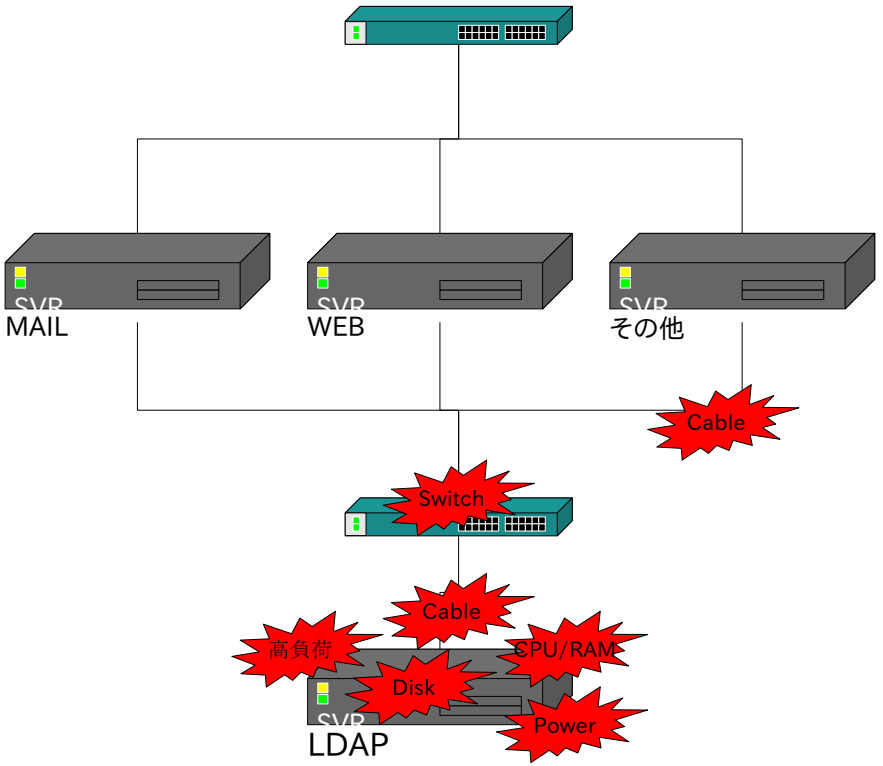
おさらい：LDAP とは

- Lightweight Directory Access Protocol
- 検索を重要視した DB プロトコル
- アドレス帳，UNIX アカウントなど
- メールサーバ，公開鍵，その他独自アプリ

LDAP を導入する上での課題 ～冗長性 (1/2)

- UNIX アカウントを LDAP で管理すれば便利
- でも，LDAP 関連の障害でサーバに入れなくなると困る
  - LDAP サーバ自体の障害
  - ネットワークケーブル，電源，経路障害
  - その他 Single Point Of Failure (SPOF)
  - ・・・不便だが /etc/passwd は信頼できる!

LDAPを導入する上での課題 ～冗長性(2/2)



- 集中管理
  - 運用効率↑
  - 障害時にサービス全体がダウンする可能性
- 分散管理
  - 運用効率↓
  - 障害時には，そのポイントのみサービスダウン

LDAP を導入する上での課題 ～負荷

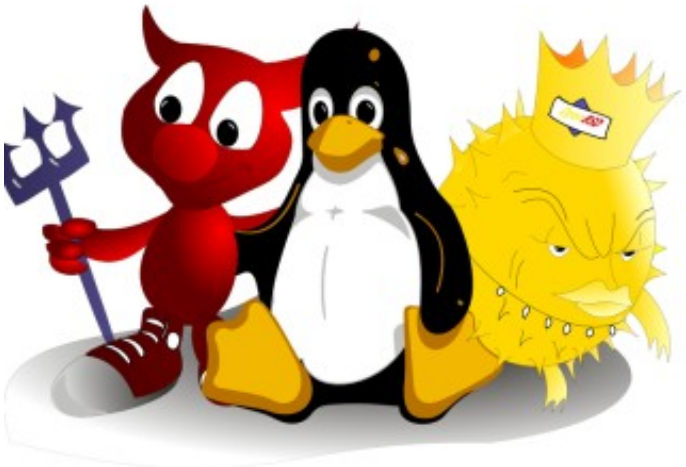
- 検索を得意としたプロトコルであっても, 検索数が膨大だとやっぱりパフォーマンスがでない
- SMTP サーバのデータを管理する場合にありがち
  - 大量の spam メールで LDAP 検索が頻発・・・
  - SMTP サーバ全体のスループットが低下・・・
- 解決法
  - ソフトウェア的なチューニング
  - ハード交換
  - 負荷分散

負荷対策と冗長化対策の基本

- 冗長化対策といえば
  - HA クラスタ (Active/Standby)
  - HA クラスタ (Active/Active)
    - LDAP で言うマルチマスタ
- 負荷対策といえば
  - レプリケーション
  - ロードバランシング
  - DNS ラウンドロビン (安くて簡単・・・だが)

可用性

- 今回は可用性というよりも冗長性のお話
  - 機器自体の信頼性を高めるのではなく, 二重化, 三重化について考える



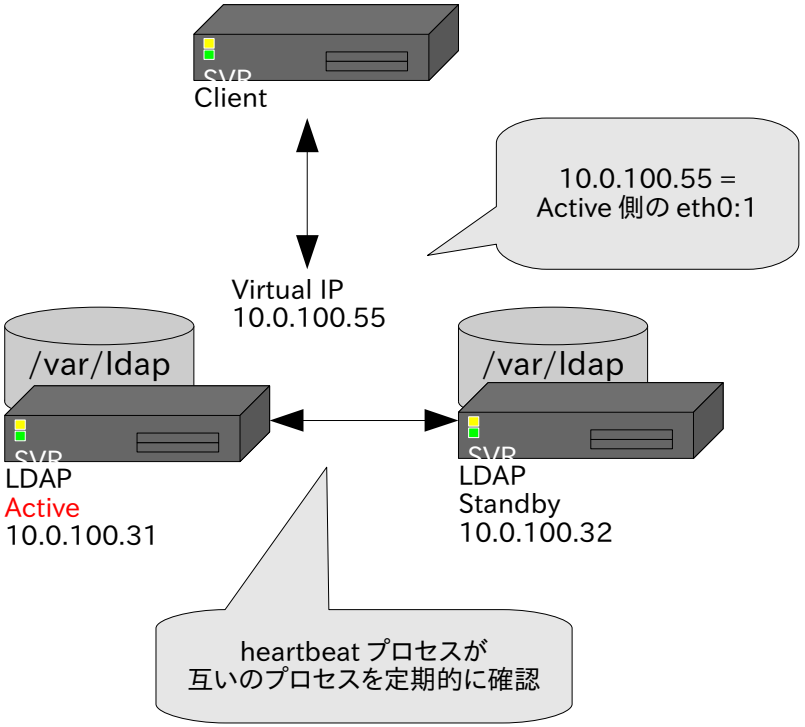
可用性を考える ～小規模

- サービスが一時的にダウンしても, 早く復旧させれば大丈夫
- 必要なもの
  - 定期的なバックアップ (slapcat > backup.ldif)
  - コールドスタンバイ機器 (もし必要であれば)
- 必要ないもの
  - HA クラスタ
  - ロードバランサ
  - レプリケーションなど
- 簡単なので, 今回は取り上げません

可用性を考える ～中規模 - 大規模

- サービスが止まると困る
- 必要なもの
  - LDAP サーバやスイッチ, 回線の二重化
  - できれば全ての SPOF を無くす
- UNIX アカウントについては, 念のため定期的に /etc/passwd.ldap や /etc/shadow.ldap のように情報を取得しておいた方が良いかも.
  - 万が一の場合は, root でログインし上記ファイルを従来のファイルにコピー

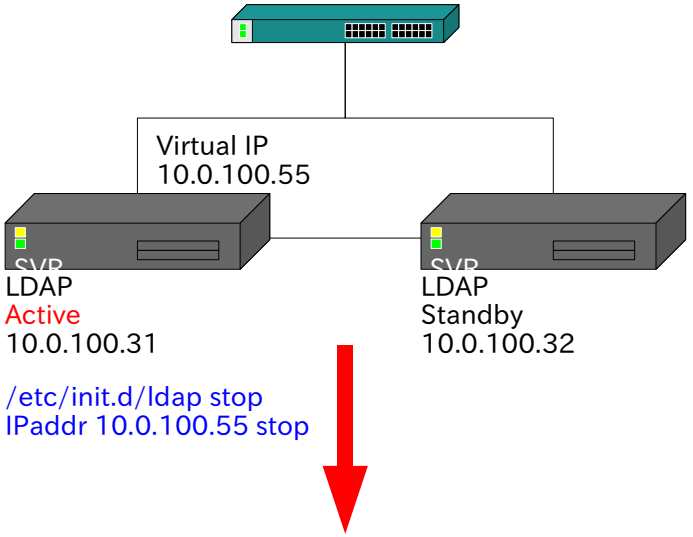
Active/Standby 方式 ～ heartbeat



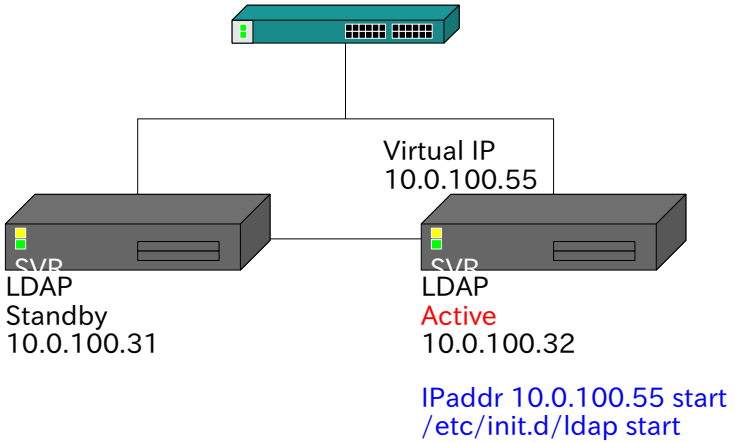
- LDAP に限った話ではないのですが・・・
- http://linux-ha.org/
- Solaris，FreeBSD，OpenBSD などでも動作
- 常に Active 側の eth0:1 に 10.0.100.55 という共有 IP を割り当てる
- Active 側がダウンすると Standby 側の heartbeat プロセスがそれを検知し，10.0.100.55 を自身の eth0:1 に割り当てる
- 以上により，クライアントはどちらかのサーバが生きている限りクライアントは 10.0.100.55 と会話できる仕組み
- 問題点としては，LDAP データベースが互いのファイルシステムに存在しているため，データ更新が発生した場合には両者のつじつまを合わせる必要がある
  - 当然のことながら，NFS で共有して同時に読み書きさせてはダメ

heartbeat 設定例

フェイルオーバー発生直後



フェイルオーバー完了



/etc/ha.d/ha.cf

```
debugfile /var/log/ha-debug
logfile /var/log/ha-log
logfacility local0
keepalive 2
deadtime 10
udpport 694
bcast eth0
auto_failback off
node centos5-osc1
node centos5-osc2
```

実際にはサービス用でなく
専用のインターフェースを指定.
必要に応じて自動フェイルバック設定

/etc/ha.d/haresources

```
centos5-osc1 ¥
IPaddr::10.0.100.55 ¥
ldap
```

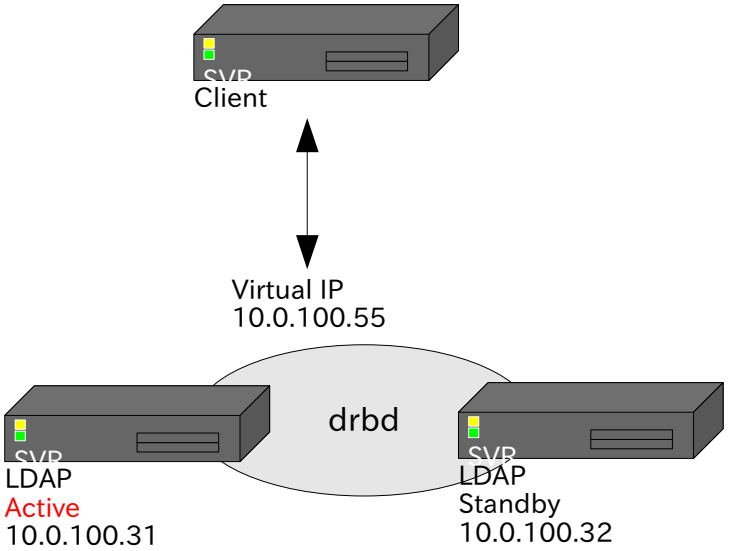
フェイルオーバー時の挙動
Active: IP アドレスを割り当て ldap を実行
Standby: ldap を停止し IP アドレスをリリース

/etc/ha.d/authkeys

```
auth 1
1 crc
```

mon 等を用いて，slapd プロセスが落ちた場合に
フェイルオーバーさせることも可能.
または，フェイルオーバーさせずに crontab 等で何度か
再起動を試みるのもアリ

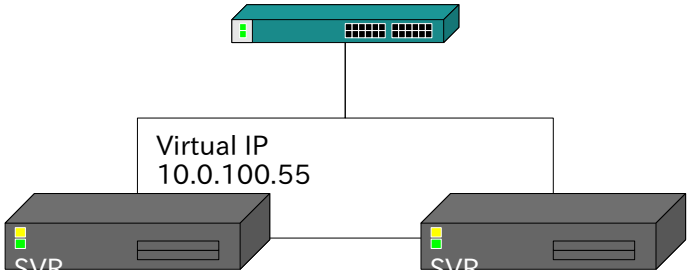
Active/Standby 方式 ～ heartbeat + drbd



- LDAP に限った話ではないのですが・・・
- http://www.drbd.org/
- ネットワーク RAID を実現
- クライアントはネットワーク RAID である /dev/drbd0 をマウントし，LDAP データベースを格納
- /dev/drbd0 に読み書きが発生すると，両サーバのローカルディスクが更新される
- heartbeat と組み合わせることで，HA クラスタなのに高価な共有ディスクが必要ない！
- 欠点としては，ネットワーク越しにミラー作業が行われるため，ローカルディスクへのアクセスと比較すると非常に遅い
- 必然的に LDAP パフォーマンスも遅くなる・・・
- 今のところ Linux のみでサポート

heartbeat+drbd 設定例

フェイルオーバー発生直後



/etc/ha.d/haresources

```
centos5-osc1 ¥
IPaddr::10.0.100.55 ¥
drbddisk::r0 ¥
Filesystem::/dev/drbd0::/mnt/drbd0 ¥
ldap
```

フェイルオーバー時の挙動
Active: IP アドレスを割り当て ldap を実行
Standby: ldap を停止し IP アドレスをリリース

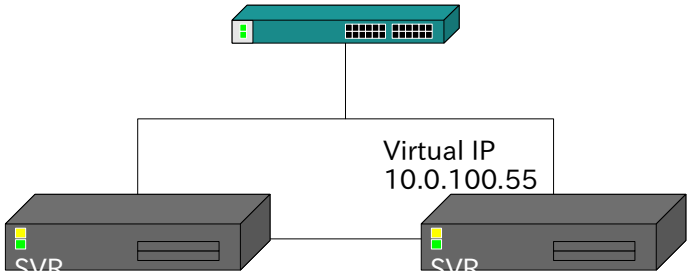
/etc/init.d/ldap stop
Filesystem /dev/drbd0 /mnt/drbd0 stop
drbddisk r0 stop
IPaddr 10.0.100.55 stop

/etc/init.d/drbd.conf

```
resource r0 {
  protocol C;
  syncer {
    rate 40M;
  }
  on centos5-osc1 {
    device  /dev/drbd0;
    disk    /dev/sda4;
    address 10.0.100.31:7788;
    meta-disk  internal;
  }

  on centos5-osc2 {
    device   /dev/drbd0;
    disk     /dev/sda4;
    address  10.0.100.32:7788;
    meta-disk  internal;
  }
}
```

物理デバイスと drbd デバイスのマッピング
/dev/drbd0 への書き込みは両ノードの
/dev/sda4 に反映される

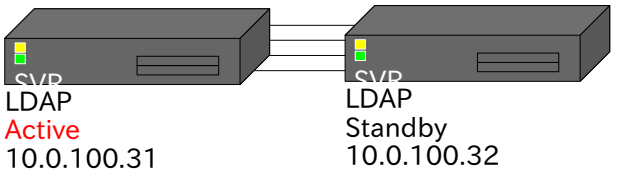
フェイルオーバー完了



IPaddr 10.0.100.55 start
drbddisk r0 start
Filesystem /dev/drbd0 /mnt/drbd0 start
/etc/init.d/ldap start

drbd だと遅い？

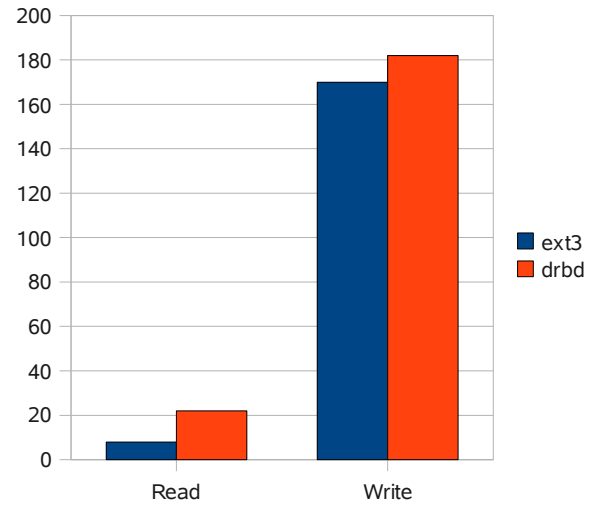
- 低コストで実現できるが，ネットワーク越しに同期を行うため，ローカルディスクや外部ディスクと比較するとかなり遅くなる
- 環境によってはチーミング等によって，ある程度高速化が期待できるが，drbd に CPU リソースが割かれることもあり，限界がある
- ファイルシステムレベルで数倍のパフォーマンスダウン




```
# time dd if=/dev/urandom of=tmp.dat bs=1024k count=100
104857600 bytes (105 MB) copied, 13.8536 seconds, 7.6 MB/s
約 14 秒

# time dd if=/dev/urandom of=tmp.dat bs=1024k count=100
104857600 bytes (105 MB) copied, 25.2878 seconds, 4.1 MB/s
約 25 秒
```

drbd+LDAP だと遅い？

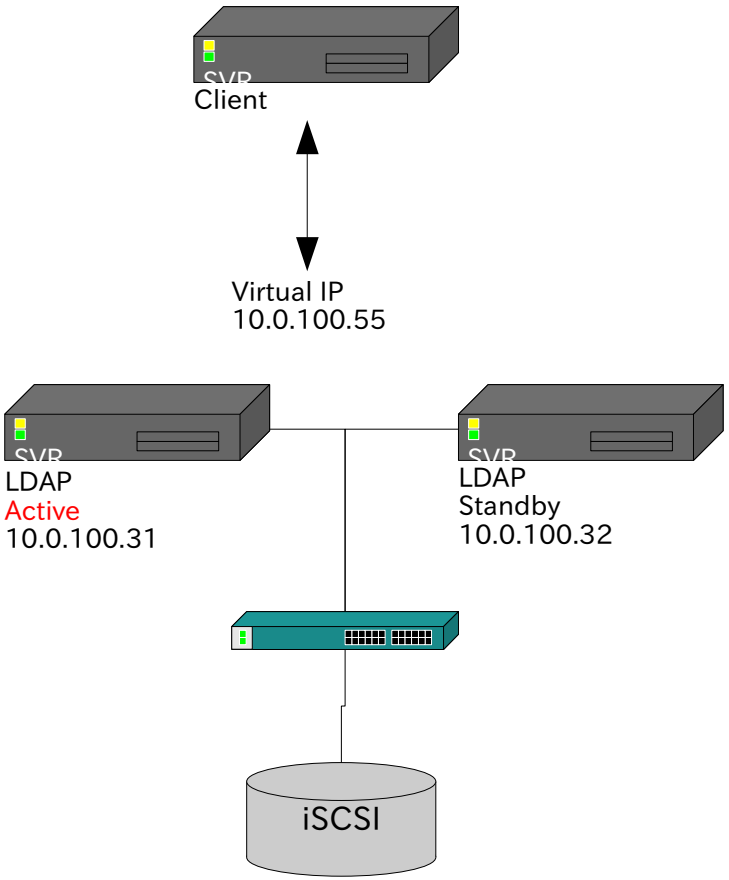
10,000 件のデータ操作



- ファイルシステムのアクセス速度が遅いため，当然 LDAP サービスレベルでも遅くなる
- ただし，それほどパフォーマンスを必要としない環境であれば十分実用的
- nscd などのクライアントキャッシュを用いれば，特にストレスを感じない・・・かも

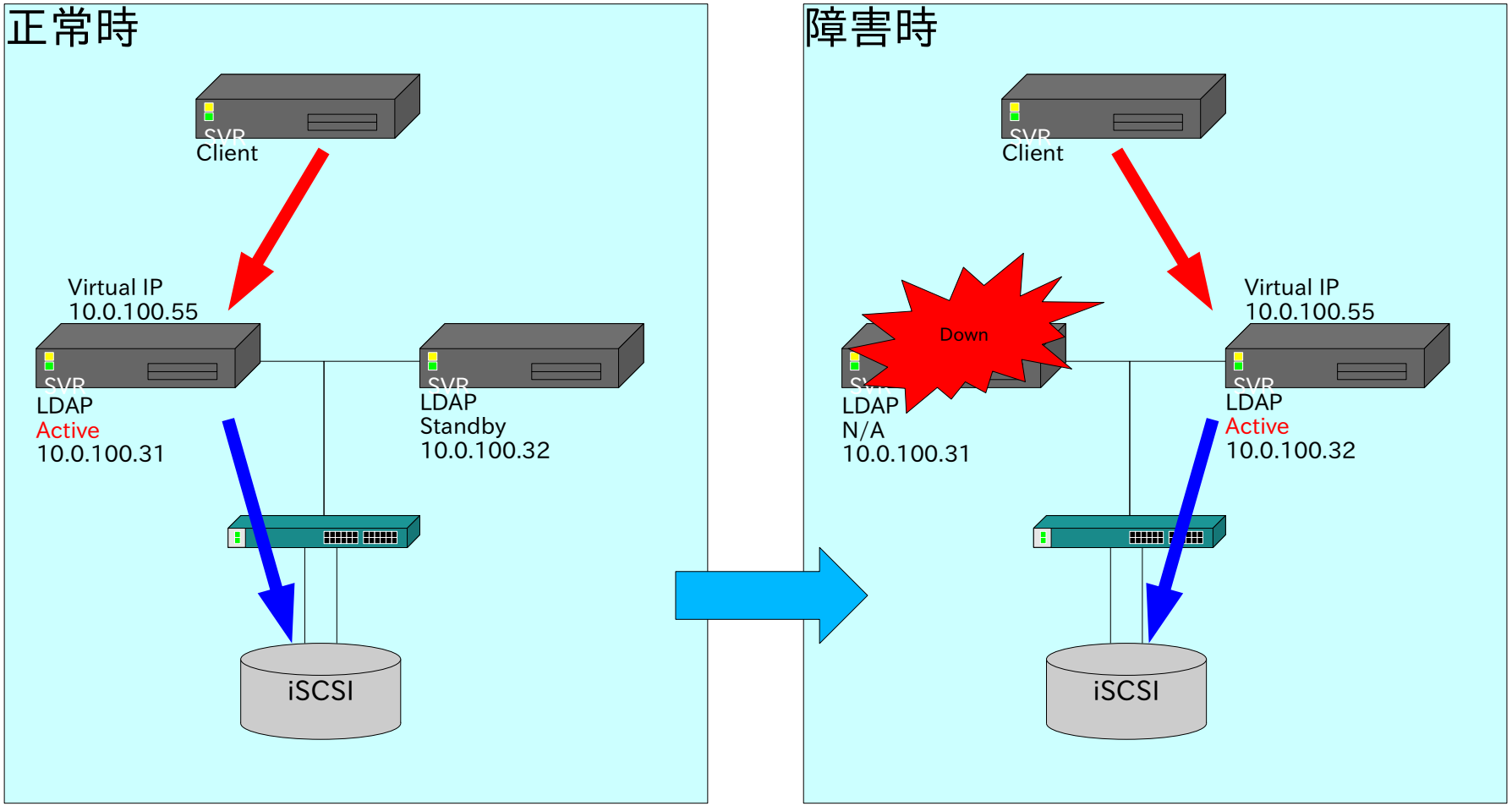
※ グラフのデータは VMWare におけるものなので，
実際の値とは異なる可能性があります

それ，iSCSI で．



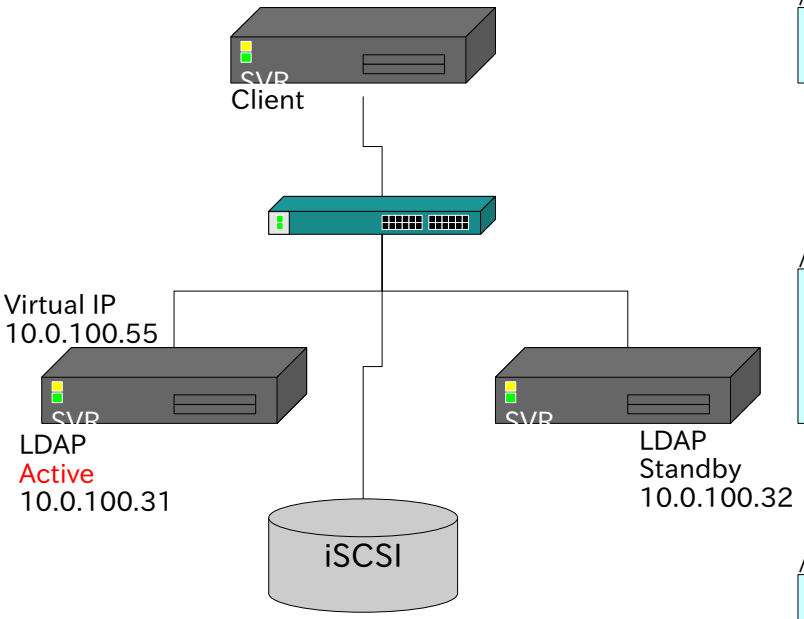
- 共有データ (bdb) は iSCSI デバイスに保存
- SCSI/FC 等の HBA カードは必要なし
- NFS と異なりブロックデバイスなので，Active 機からのみマウントする必要がある (drbd 同様，同時マウント不可能)
- 最近では安価な製品も増えてきている
  - バッファロー TS-I1.0TGL/R5 10 万円弱
  - ただし，電源，コントローラ，インターフェース等の冗長化未対応
- 大規模な環境であればコントローラやインターフェース，スイッチの二重化，トランク等も考慮 (左図では，スイッチが落ちれば全ダウン)
- iSCSI なんて XXX だから使えねーよ!という質問は無しでw

heartbeat + iSCSI



※ iSCSI に限らず SCSI や FC でも同様の構成が可能

heartbeat + iSCSI 検証



/etc/iscsi/initiatorname.iscsi

```
InitiatorName=iqn.2008-02.net.bluecoara:storage.iscsi
```

iSCSI デバイス側で提供されている iqn を登録

/etc/ha.d/haresources

```
centos5-osc1 ¥
IPaddr::10.0.100.55 ¥
Filesystem::/dev/sdb1::/mnt/iscsi ¥
ldap
```

iSCSI デバイスである /dev/sdb1 をマウント

/etc/ietd.conf

```
Target iqn.2008-02.net.bluecoara:storage.iscsi
        IncomingUser
        OutgoingUser
        Lun 0 Path=/dev/sda4,Type=fileio
```

/dev/sda4 を iSCSI 用デバイスとして提供

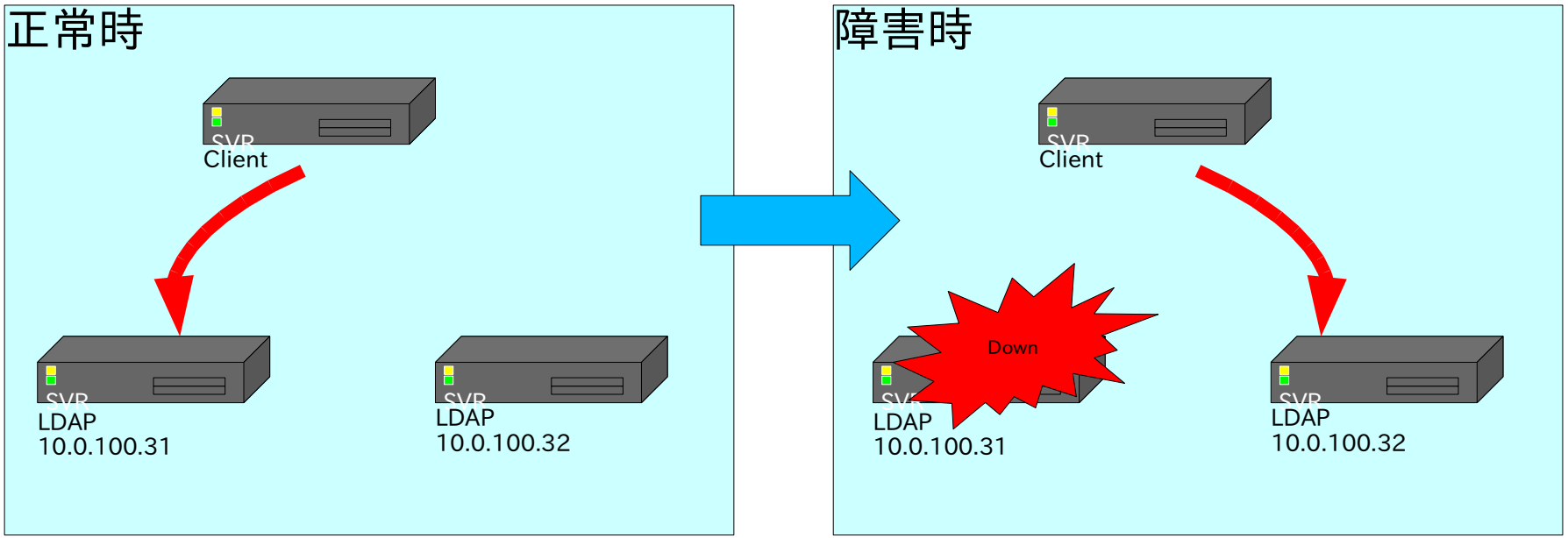
※ 実際にはパフォーマンスを考慮して iSCSI 専用 VLAN 等を活用する
※ 今回のように，Linux サーバを iSCSI デバイスサーバとすることも可能だが，パフォーマンス等を考慮すると現実的には専用デバイスを使用する

クライアント依存型の冗長設計

- 今までの方式は，LDAP サーバ自身が障害時に IP アドレスを変化させることでアクセス先を変更する形
- クライアント側で障害を検知し，切り替える考え方もある
- アプリ側の実装に依存するところが欠点
- 例えば Postfix
  - server_host = host1, host2
  - host1 への接続に失敗したら host2 をトライ
- 内部的には，ldap_open(host1) できなければ ldap_open (host2) のようなイメージ
- アドレス帳等は対応していないことが多いみたい

クライアント切り替え型

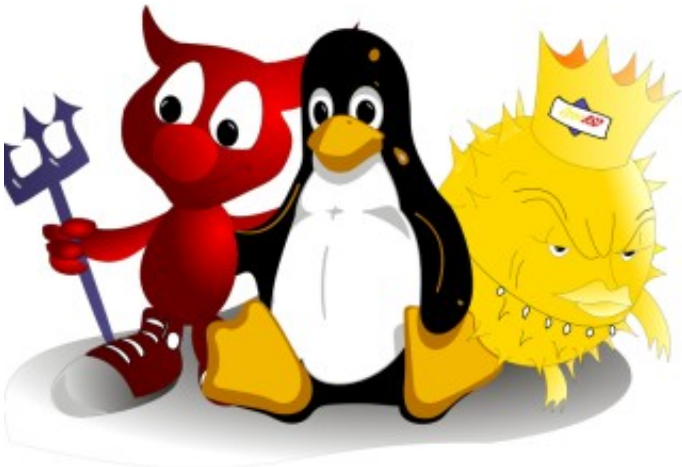
通常は 10.0.100.31 を参照し, そちらに繋がらない場合は
10.0.100.32 に接続を試みる



※ サーバ間でのデータ同期は別途必要になるため注意 (後述)

パフォーマンス

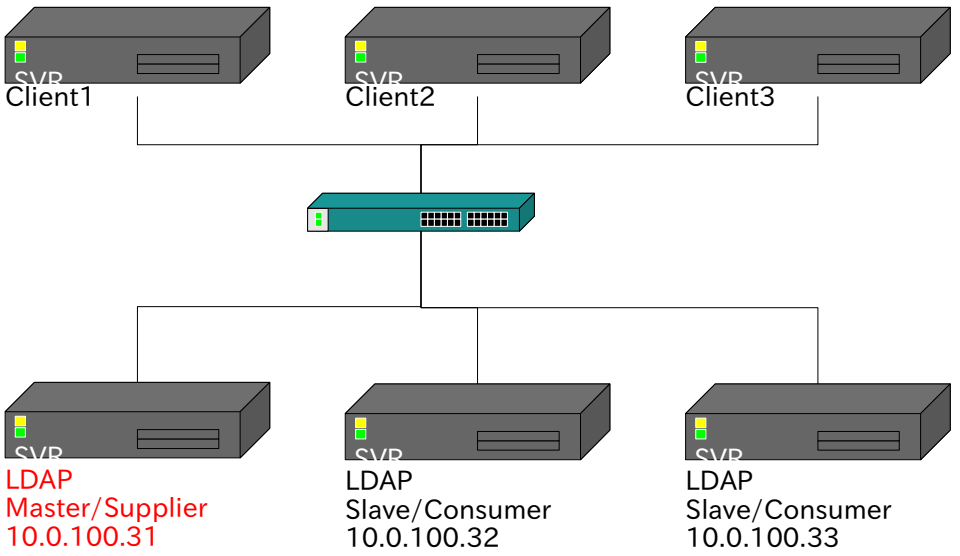
- パフォーマンスにもいろいろ
  - チューニングやサーバアップグレードではなく, 今回は処理を分散させることで (= 負荷分散) 全体のパフォーマンスを向上させる



負荷分散の方法

- ロードバランサを使ってバランシングさせる
- クライアント側から自発的にバランシングさせる
  - A システムでは host = ldap1, ldap2
  - B システムでは host = ldap2, ldap1
- refferals をうまく活用する
  - 例えば 10.0.100.31 の ou=sales,o=local 以下で検索すると,「代わりに 10.0.100.32 で検索してください」という委託情報を返す
  - こちらもクライアントの実装次第であるため汎用的ではない

分散といえばレプリケーションで一発解決!!・・・ではない



- OpenLDAP には slurpd/syncrepl というレプリケーション手段が用意されているが・・・
- クライアントはどれを参照すれば？
- Master がダウンすると Slave が更新されない？
- エントリ追加，更新要求はどこへ？
- スイッチや回線がダウンした場合は？
- サーバがマルチマスタに対応していた場合でも，振り分けについては別途検討する必要がある

LDAP とロードバランサ

- LDAP は基本的に 389/tcp のみを使用する ＝ シンプル
- FTP や SIP のように, 複数のポートをダイナミックに使用するわけではないため, L4 バランシングが可能
- ロードバランサの種類
  - オープンソースでは ultramonkey , crossroad , balance など
  - 商用だと BIG-IP , Citrix , Alteon , ServerIron 等
    - とにかく値段が高い (笑)
    - その分高機能. ASIC 処理やフェイルオーバー時にもセッションが維持される, など

たとえば ipvsadm

- http://dsas.blog.klab.org/archives/50664843.html がわかりやすいので, 細かい説明は省略
- LB の機能自体はコマンド数行で実現可能
- 実際にはサービス監視が重要

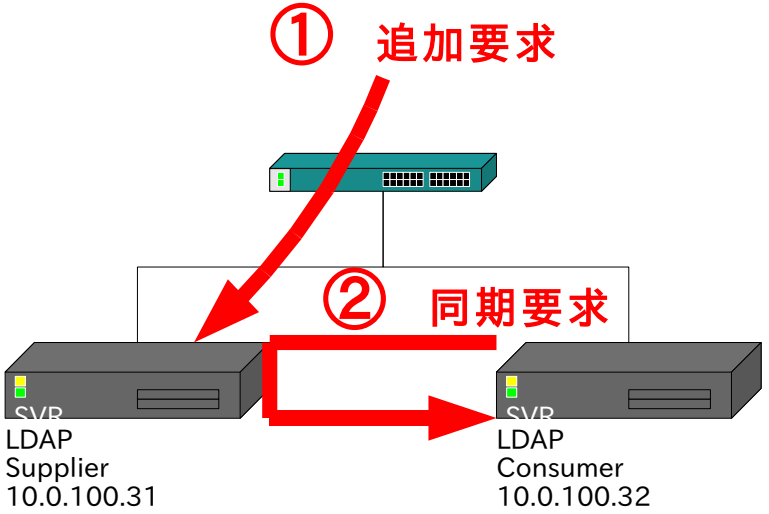
OpenLDAP-2.4 系で slurpd が無くなりました

- The slurpd daemon was the original replication mec hanism inherited from UMich's LDAP and operates in push mode: the master pushes changes to the slave s. It has been replaced for many reasons, in brief:
  - It is not reliable
  - It is extremely sensitive to the ordering of records in the replog
  - It can easily go out of sync, at which point manual intervention is required to resync the slave database with the master directory
  - It isn't very tolerant of unavailable servers. If a slave goes down for a long t ime, the replog may grow to a size that's too large for slurpd to process

代わりに syncrepl

- Why is Syncrepl better?
  - Syncrepl is self-synchronizing; you can start with a database in any state f rom totally empty to fully synced and it will automatically do the right thin g to achieve and maintain synchronization
  - Syncrepl can operate in either direction
  - Data updates can be minimal or maximal

Consumer の追加は簡単



centos5-osc1: slapd.conf

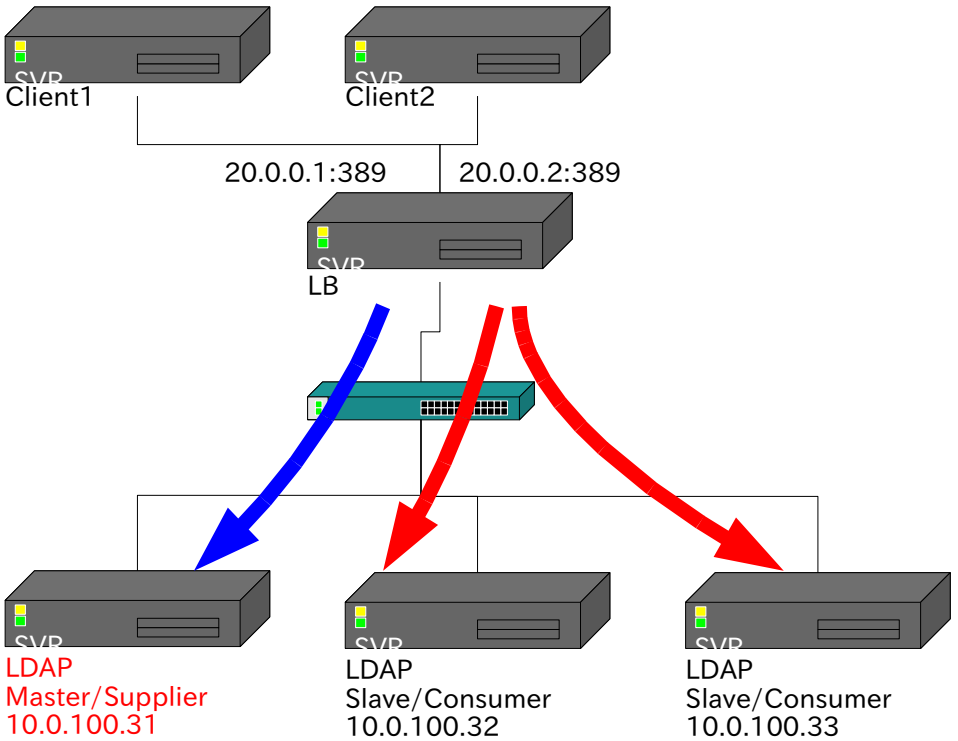
```
index objectclass,entryCSN,entryUUID eq
overlay syncprov
syncprov-checkpoint 100 10
syncprov-sessionlog 100
```

centos5-osc2: slapd.conf

```
index objectclass,entryCSN,entryUUID eq
syncrepl rid=123
        provider=ldap://10.0.100.31:389
        type=refreshAndPersist
        interval=01:00:00:00
        searchbase="dc=example,dc=com"
        schemachecking=off
        bindmethod=simple
        binddn="cn=Manager,dc=example,dc=com"
        credentials=secret
```

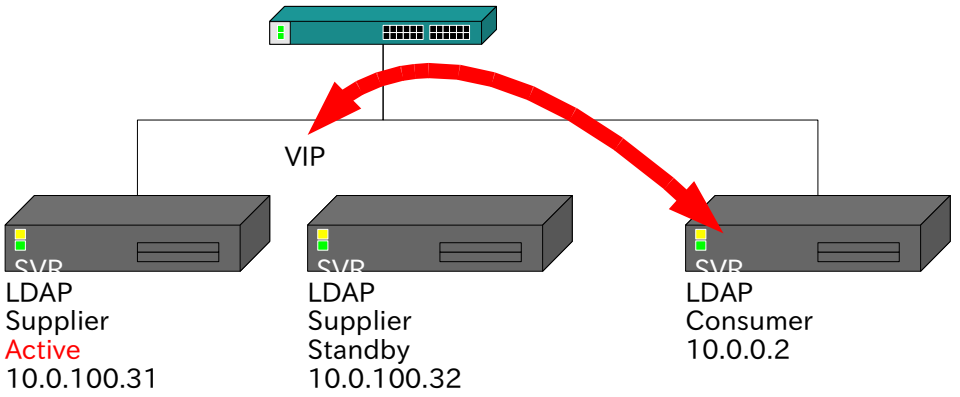
同期時のデータ取得や更新は ldapmodify ではない専用プロトコル

syncrepl: 更新と参照



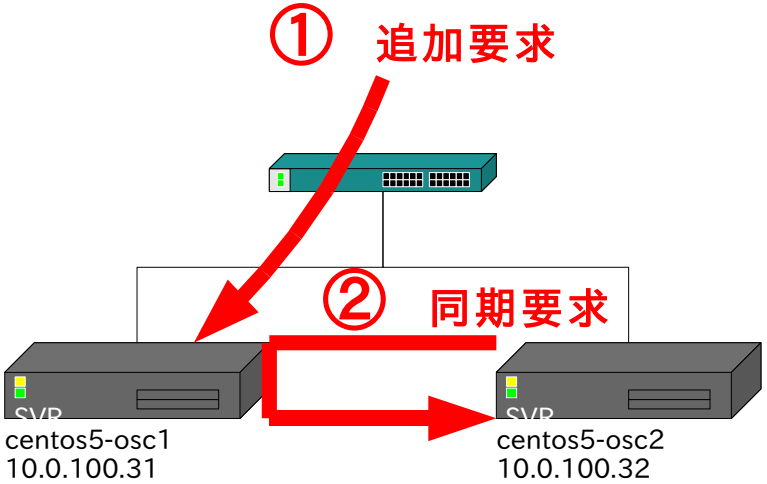
- 追加や更新は 20.0.0.1 へ
- 参照は 20.0.0.2 へ
- Supplier のデータは Consumer に反映
- LDAP に限らず, 一般的なデータベースバランシングでも当てはまる
- LB は DSR 構成でも可

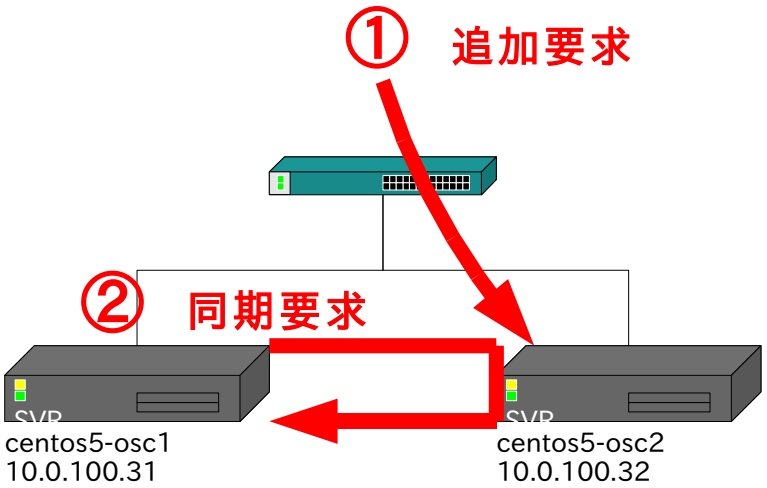
Supplier の冗長化が重要



- Supplier 障害時にも継続的にデータを提供できるように
- Supplier が故障した場合，Consumer を使って検索は可能だが，データの追加，更新が不可能
- 言い換えれば，データの追加，更新が滅多に発生しない場合はシングル構成 + 定期的なバックアップでも良い．
- heartbeat やその他商用ソフト

マルチマスタのサポート (MirrorMode)





centos5-osc1: slapd.conf

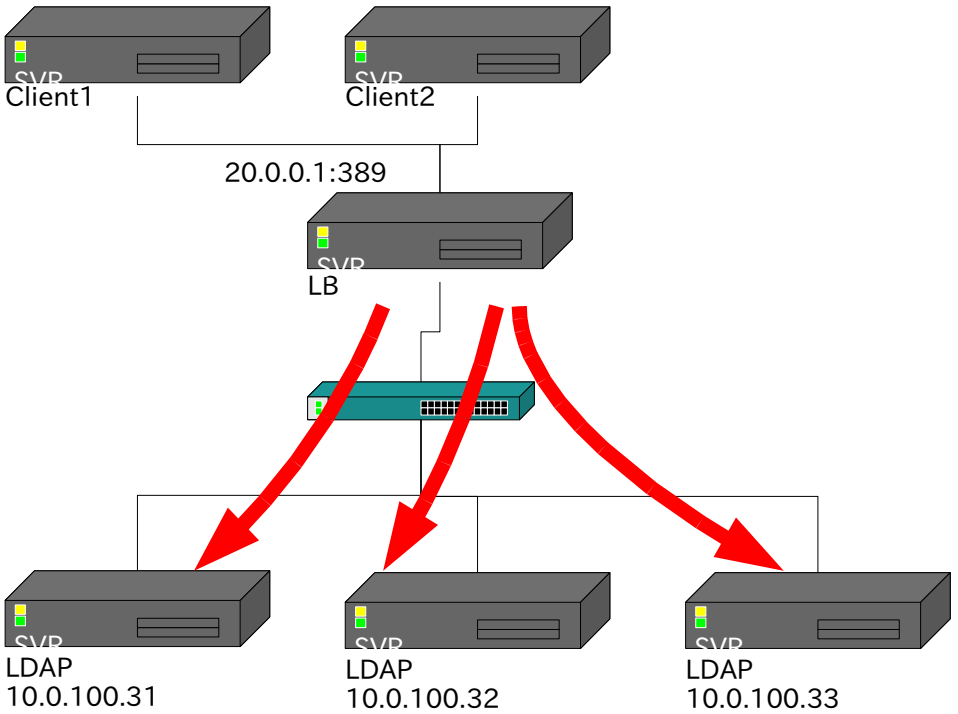
```
index objectclass,entryCSN,entryUUID eq
overlay syncprov
syncprov-checkpoint 100 10
syncprov-sessionlog 100

syncrepl      rid=001
              provider=ldap://10.0.100.31
              bindmethod=simple
              binddn="cn=Manager,dc=example,dc=com"
              credentials=secret
              searchbase="dc=example,dc=com"
              schemachecking=on
              type=refreshAndPersist
              retry="10 +"

syncrepl      rid=002
              provider=ldap://10.0.100.32
              bindmethod=simple
              binddn="cn=Manager,dc=example,dc=com"
              credentials=secret
              searchbase="dc=example,dc=com"
              schemachecking=on
              type=refreshAndPersist
              retry="10 +"

mirrormode on
serverID    1
```

centos5-osc2: slapd.conf

```
index objectclass,entryCSN,entryUUID eq
overlay syncprov
syncprov-checkpoint 100 10
syncprov-sessionlog 100

syncrepl      rid=001
              provider=ldap://10.0.100.31
              bindmethod=simple
              binddn="cn=Manager,dc=example,dc=com"
              credentials=secret
              searchbase="dc=example,dc=com"
              schemachecking=on
              type=refreshAndPersist
              retry="10 +"

syncrepl      rid=002
              provider=ldap://10.0.100.32
              bindmethod=simple
              binddn="cn=Manager,dc=example,dc=com"
              credentials=secret
              searchbase="dc=example,dc=com"
              schemachecking=on
              type=refreshAndPersist
              retry="10 +"

mirrormode on
serverID    2
```

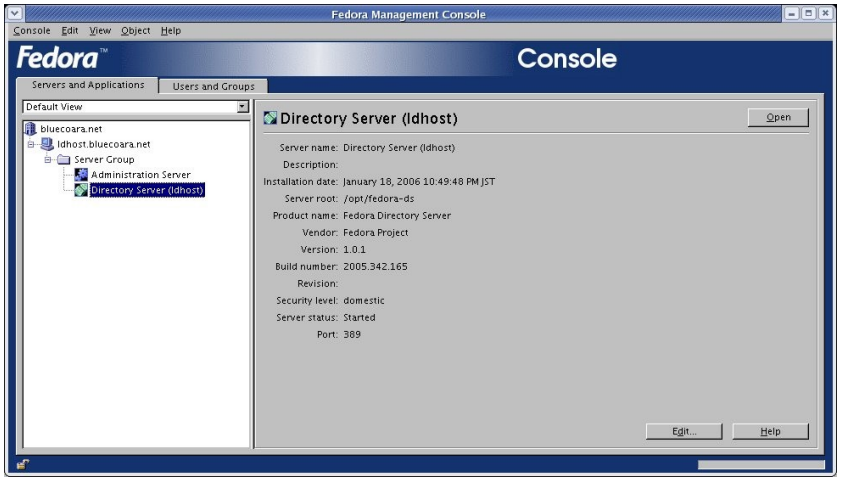
データは非同期で両方 (またはそれ以上) に反映される

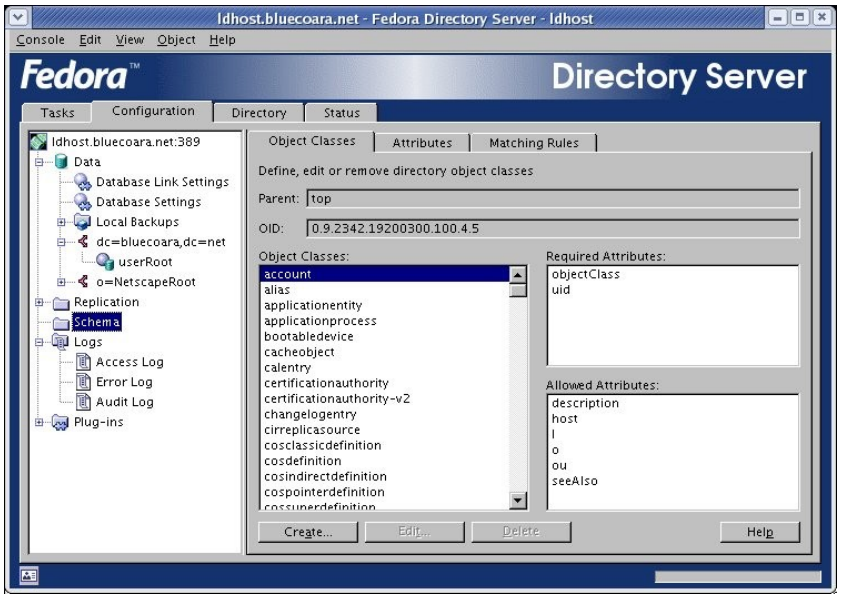
MirrorMode: 更新と参照



- 全てのサーバに対し更新や検索が可能
- 3 台以上でも動作
- ただし, 安定性はまだまだな気がする
- 3 台で検証を行った際, データがうまく他ノードに反映されないどころか, プロセスが落ちてしまった

マルチマスタと言えば，今のところ FDS や SDS



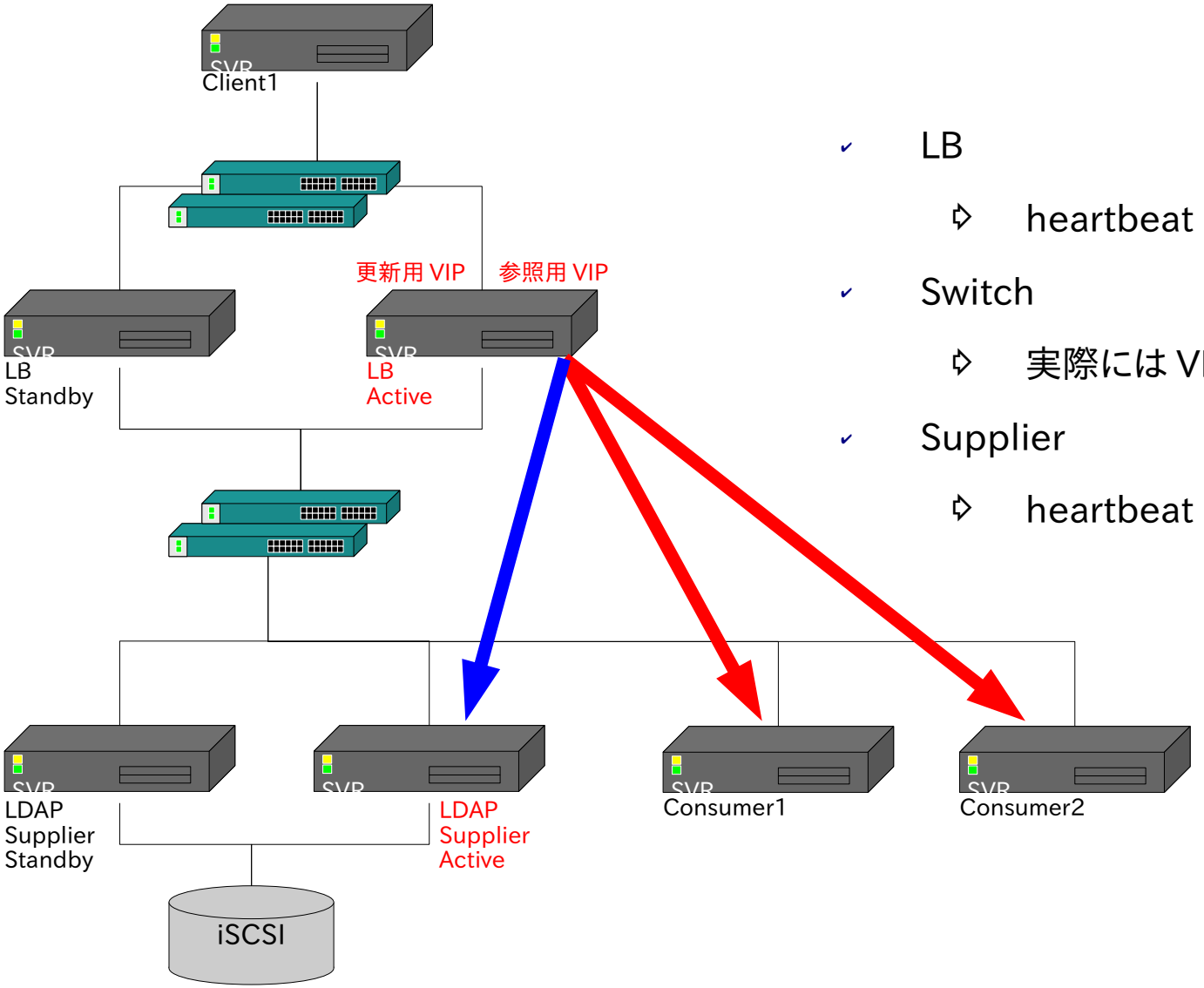


- Sun での実績は十分だし，CTC さんなどがサポートしてくれるので，エンタープライズで使いやすい
- 前職で実際に運用，サポートされてました
  - Sun Directory Server x 2
  - マルチマスタ
  - ロードバランサ

負荷分散： まとめ

- 単に複製を増やすのではなく，Supplier 自身のダウンも考慮する必要がある
- OpenLDAP の場合，syncrepl によって同じデータを持つ Consumer を複製する
  - マルチマスタは信頼性が・・・
- FDS/SDS の場合はマルチマスタも可能
- syncrepl ，FDS 共に，複製は非同期で行われる
  - 検索に比べて更新頻度は少ないため影響は少ない？
- syncbackup の動向
  - コンセプト： フェイルオーバしたときのディレクトリ整合性の信頼をあげるもの

全体図
～冗長性と負荷分散



- LB
  - heartbeat + ultramonkey 等
- Switch
  - 実際には VLAN を用いた最小構成も可
- Supplier
  - heartbeat + 共有ディスク (or drbd)

まとめ

- 「LDAP を使えば便利」と言うのは簡単だが, サービス化するのは案外難しい
- 全体の冗長化には結構な費用が必要なので, 妥協点を見つけることが大切
  - 社内用, サービス用で使い分け
- 最近のハードウェアの進化を考慮すると, LB 無しでの HA クラスタ程度でも十分サービスに耐えれたりする
  - でも, できれば HA クラスタは導入しよう

「そろそろ LDAP にしてみないか？」