大規模サイトにおけるユーザ認証

LDAPの概要と認証システムへの適用

Internet Week 2003

樽石将人 (taru@valinux.co.jp)
VA Linux Systems ジャパン 株式会社

概要

オープンソースによるLDAP環境の実現

第一部 - LDAPの概要とOpenLDAPの導入

第二部 - メールサーバとの連携

- postfix
- qpopper

第三部 - 大規模システムへのLDAPの適用

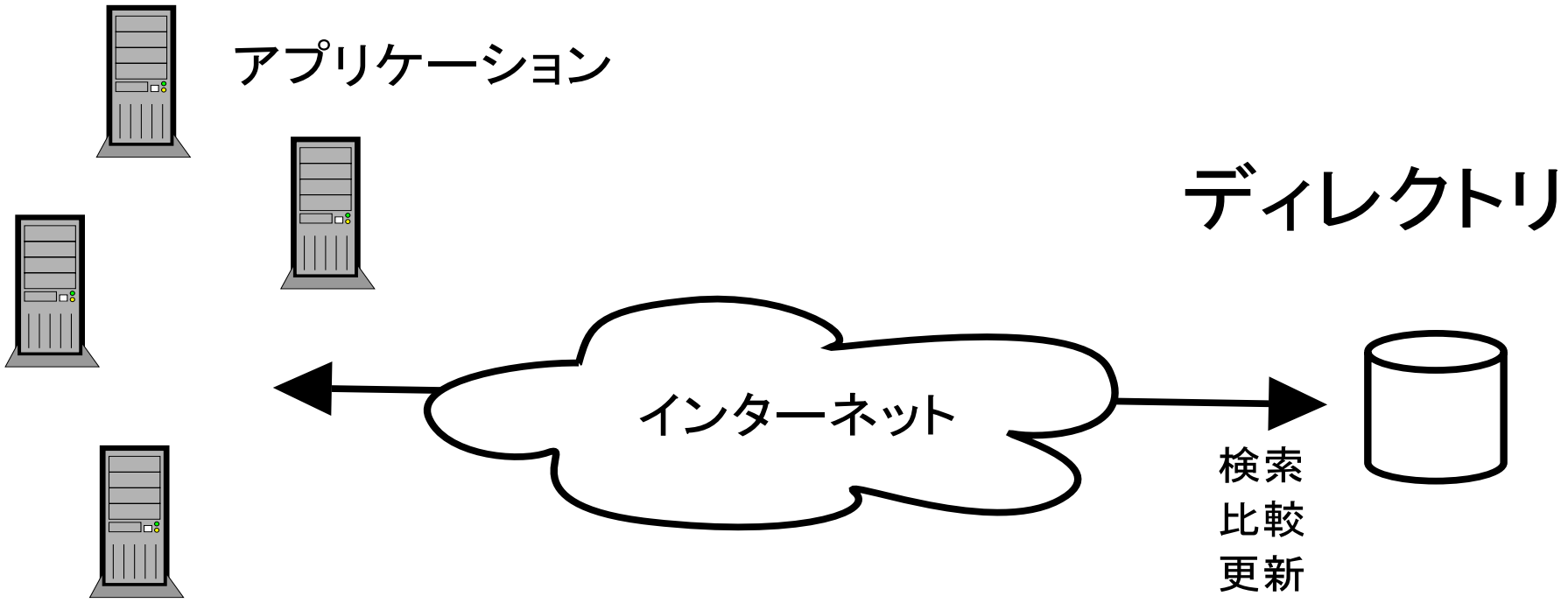
- heartbeat
- mon

第一部 LDAPの概要とOpenLDAPの導入

LDAPとは

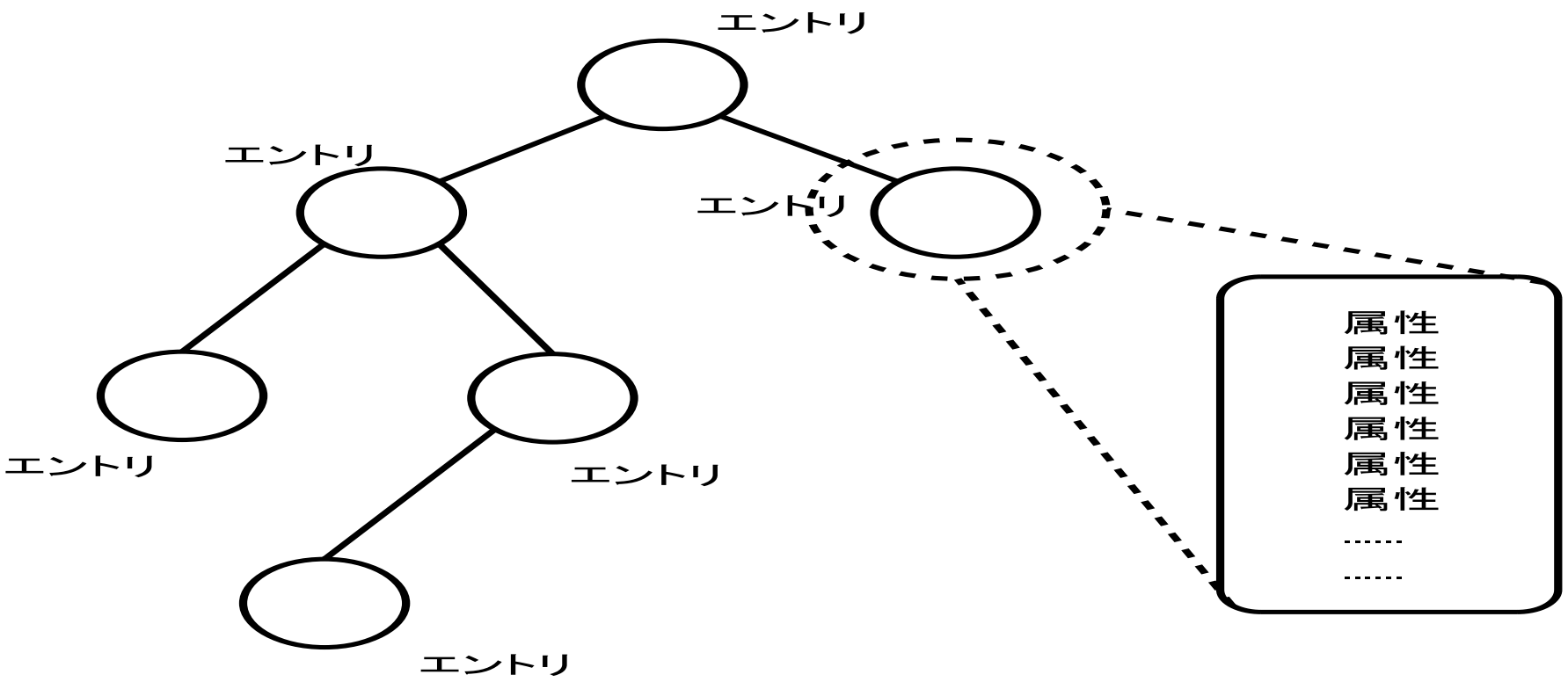
Lightweight Directory Access Protocol

- ディレクトリにアクセスにするためのプロトコル
  - 検索、比較 追加、更新、削除等

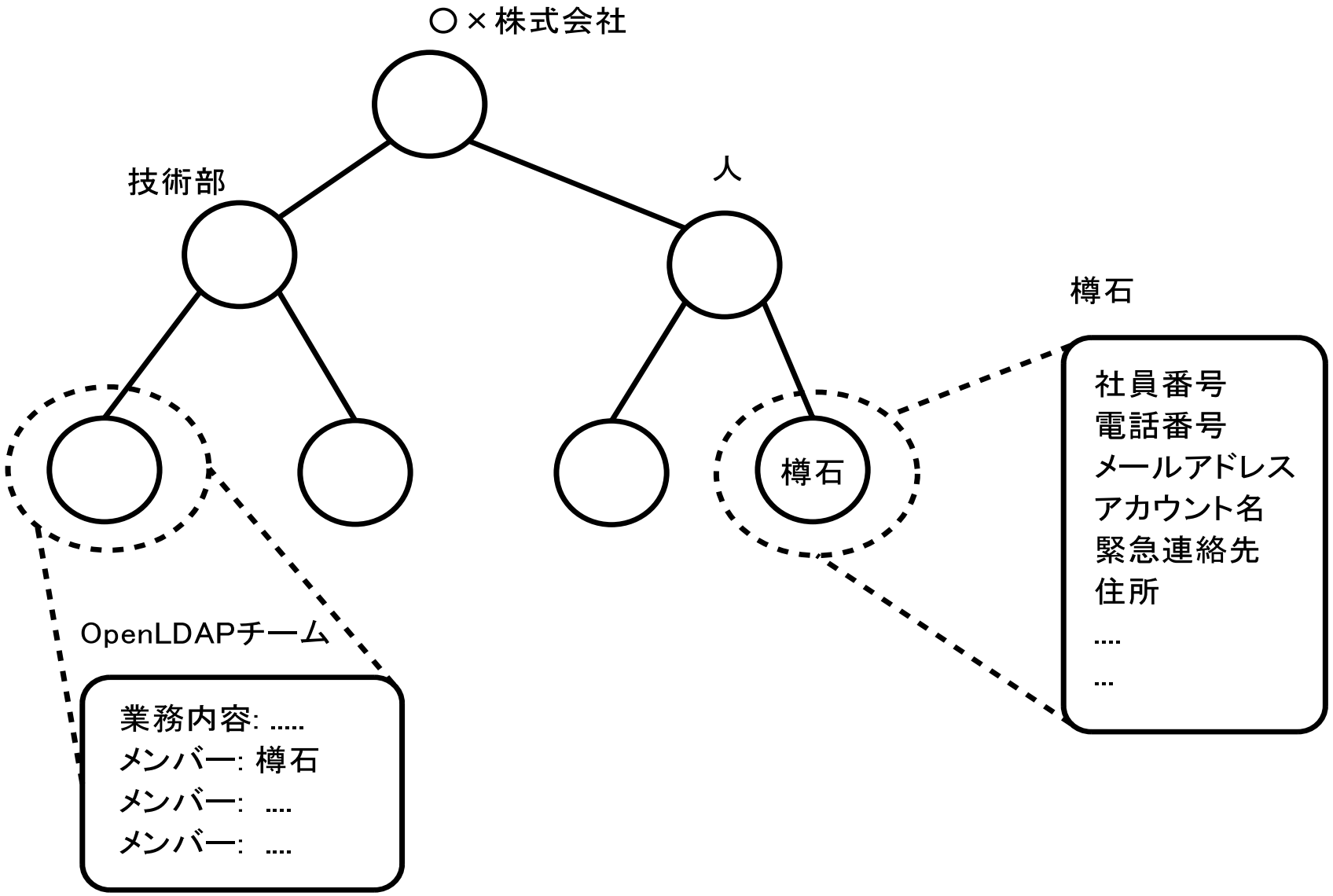


LDAP(X500)のディレクトリモデル

- 木構造で関連付けられたエントリの集合
- エントリは属性の集合



ディレクトリの例


○×株式会社
技術部
人
樽石
社員番号
電話番号
メールアドレス
アカウント名
緊急連絡先
住所
....
...
樽石
OpenLDAPチーム
業務内容: .....
メンバー: 樽石
メンバー: ....
メンバー: ....

名前付モデル - 識別名 (DN)

DIT (*1) 上におけるエントリの呼び方 (パス)

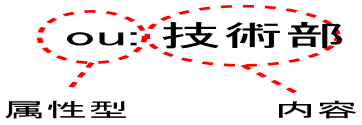
- 識別名 = 相対識別名(RDN),<親の識別名>

(*1)
Directory

属性 (Attribute)

エントリの要素

- 属性 = 属性型: 内容



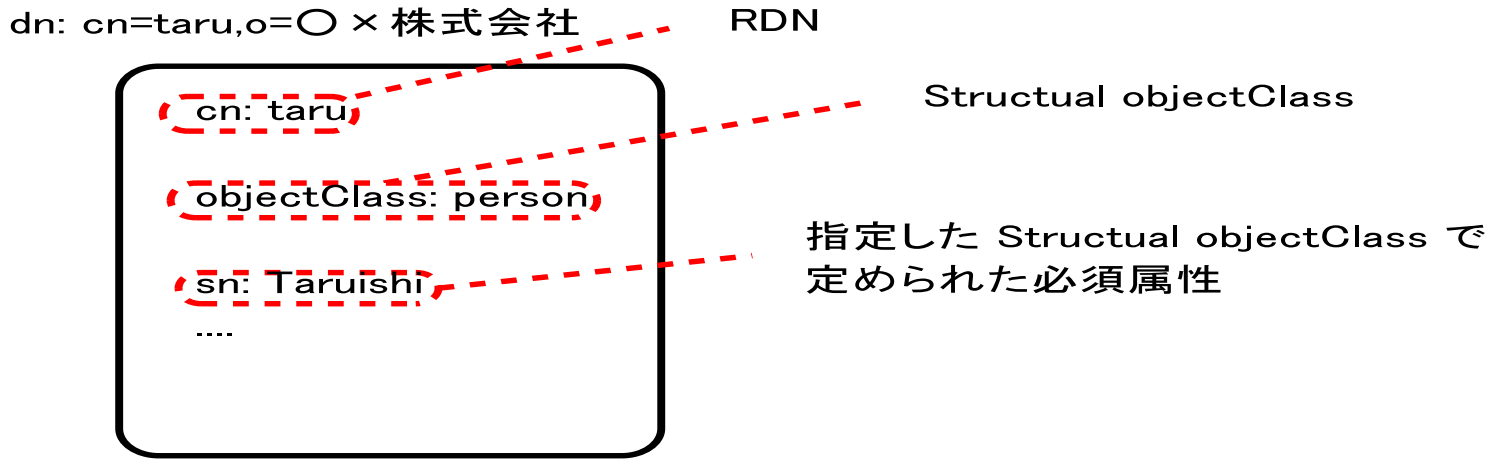
代表的な属性型

ou（OrganizationalUnit）：組織名
cn（Common Name）　：一般名
sn（SurName）　：名字
uid　：POSIXアカウント名
telephoneNumber　：電話番号
dc（DomainContext）　：ドメイン名の構成要素

- ディレクトリ管理者が独自の属性型を定義することが可能
- 一般的なものに関しては(上記等)あらかじめRFCで定義

エントリの必須属性

- (Structual) objectClass
  - エントリが保持する属性の種類
- RDN



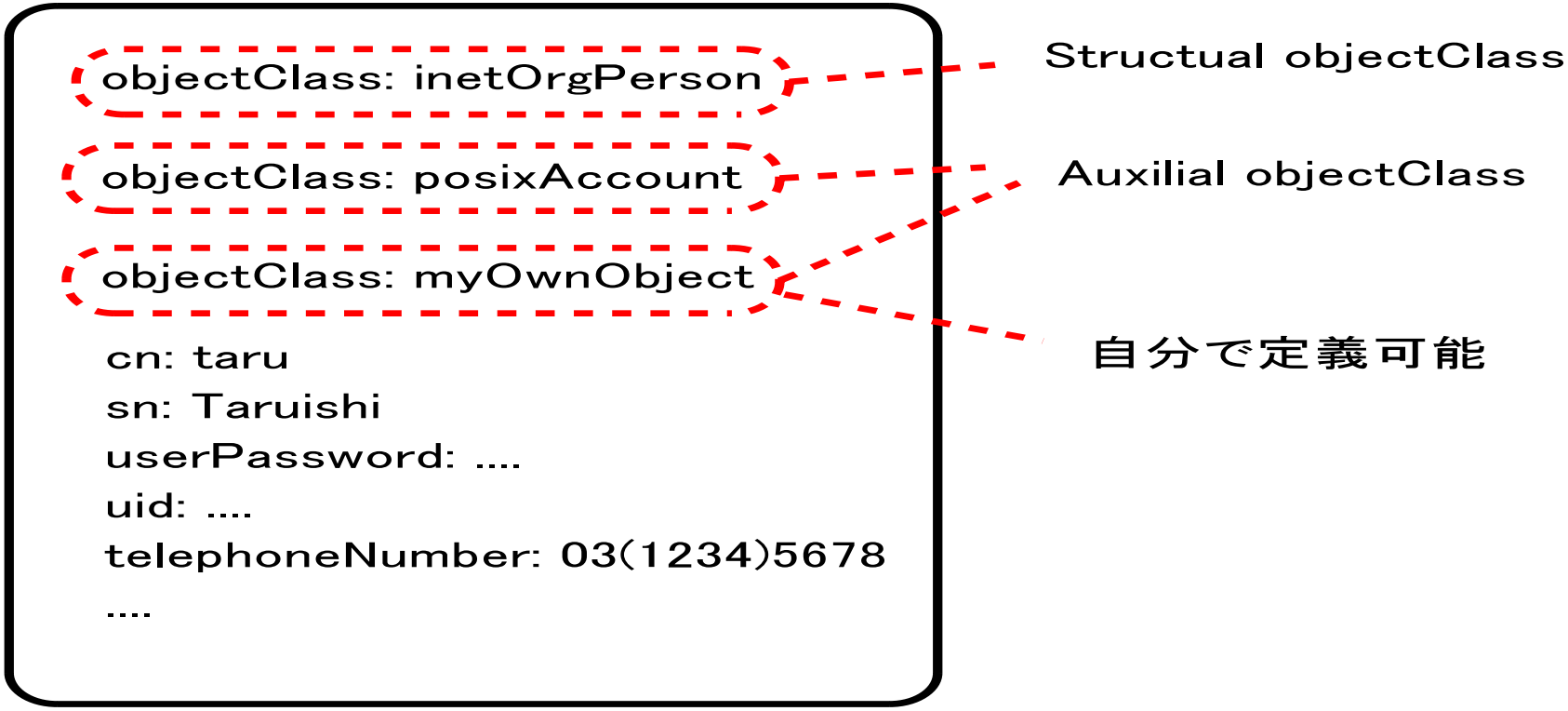
・必須属性はスキーマで記述

```
objectClass ( 2.5.6.6 NAME 'person'
    DESC 'RFC2256: a person'
    SUP top STRUCTUAL
    MUST ( sn $ sn )
    MAY ( userPassword $ telephoneNumber $ seeAlso $ description ) )
```

objectClass

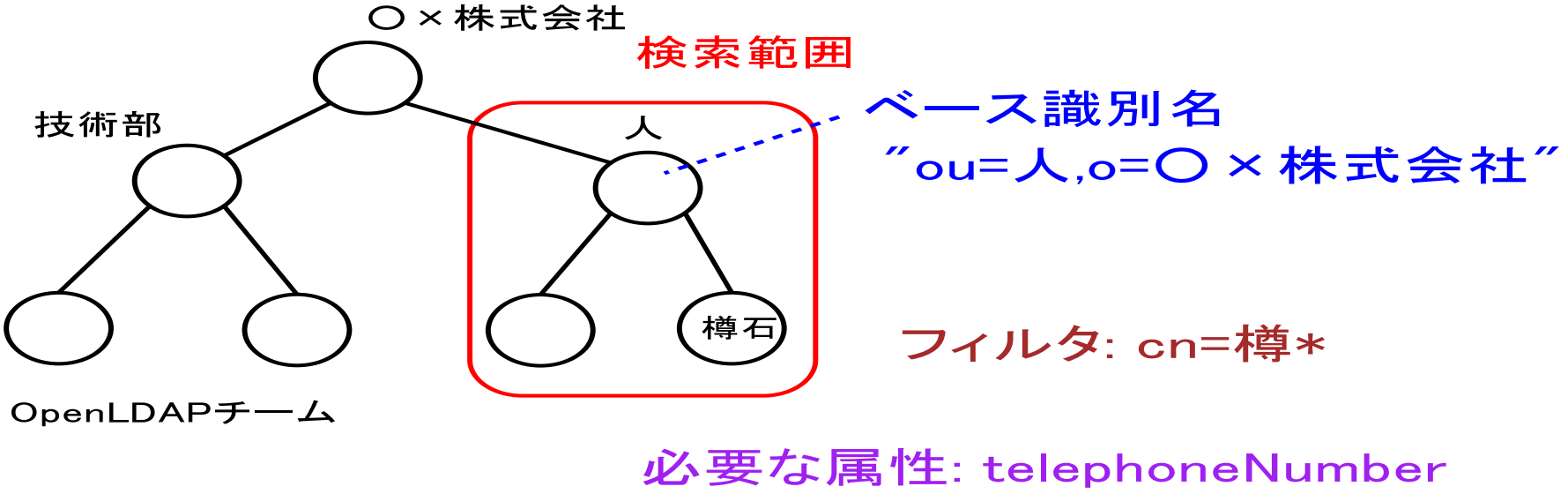
エントリが保持する属性を指定

- 1つの Structual objectClass
- 任意の数の Auxilial (補助) objectClass



検索方法

- ベース識別名の指定
- 検索範囲の指定 (ベース、one level, サブツリー)
- 検索フィルタ式による検索



- 検索フィルタに一致したエントリが返される
- エントリの特定の属性のみを受け取ることが可能（電話番号のみ等）

検索フィルタ式

<属性>=<内容>

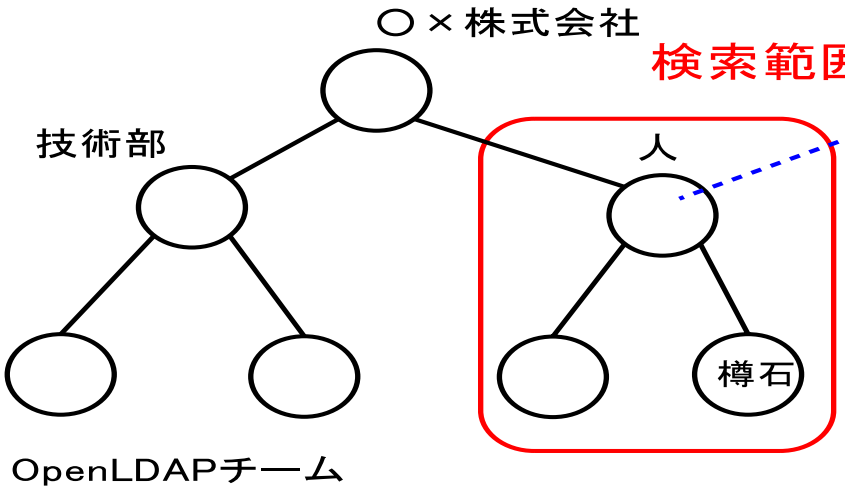
- objectClass=*
- cn=taru*
- age>=18

論理演算

- (&(cn=taru*)(age>=18))

検索の例

ldapsearch -x -h ldap -s sub -b 'ou=人,o=○×株式会社' cn='樽*' telephoneNumber



検索範囲

ベース識別名
"ou=人,o=○×株式会社"

フィルタ: cn=樽*

必要な属性: telephoneNumber

- 検索フィルタに一致したエントリが返される
- エントリの特定の属性のみを受け取ることが可能（電話番号のみ等）

LDAPをサポートした代表的な製品例

Novell eDirectory (TM)
Sun iPlanet(TM)
OpenLDAP
NEC EDS (TM)

OpenLDAPの特徴

非コピーレフトなオープンソース (BSDスタイル)

標準ベース

- 標準化に関わっている人が作成している

LDAPサーバ、クライアント一式

- LDAPサーバ
  - slapd, slurpd
- サーバ管理用コマンドラインツール
  - slapadd, slapcat 等
- クライアント作成に必要なライブラリ
  - libldap 等
- そのライブラリを利用したクライアントツール
  - ldapsearch, ldapadd 等

slapdの特徴

LDAPv3

認証、セキュリティ

- SASL認証
- startTLS (SSL)

柔軟なアクセスコントロール

- 正規表現によるエントリ指定
- ホスト名、バインド(login)

OpenLDAP(2.1)のインストールと起動

コンパイルに必要なライブラリ

- Berkley DB 4.1 以降

コンパイル&インストール

./configure; make; sudo make install

起動

~# /usr/local/libexec/slapd

動作確認

~$ /usr/local/bin/ldapsearch -x -s base -b "" +

- ○rootDSEとよばれる特別なエントリの検索

OpenLDAPの設定

slapd.conf ファイルを編集し、再起動

再起動の方法

- ~# sudo killall slapd
- ~# /usr/local/libexec/slapd

パッケージからインストールしている場合は /etc/init.d/slapd restart で良い

slapd.confの文法

行指向の文法

- 先頭の文字が空白の場合、直前の行と連結される

アクセス制御設定の例

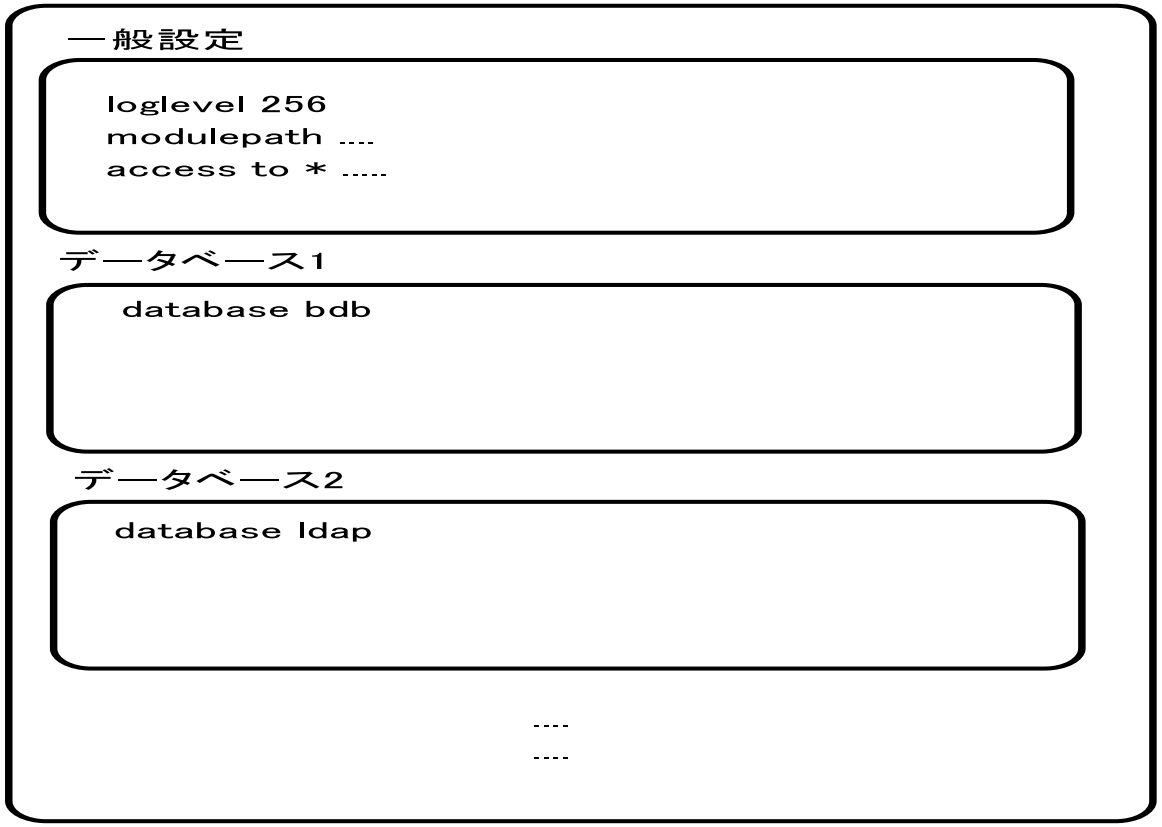
```
access to *
        by dn="cn=admin,dc=vass" write
        by * read
```

slapd.confの文法 (2)

一般設定

バックエンドの設定 (いまのところ設定項目はない)

データベース毎の設定



エントリの追加

LDIF - ディレクトリの操作方法を記述するための書式

```
dn: dc=my-domain,dc=com
objectclass: top
objectclass: domain
dc: my-domain
```

```
~$ ldapadd -x -D "cn=Manager,dc=my-domain,dc=com" < test.ldif
```

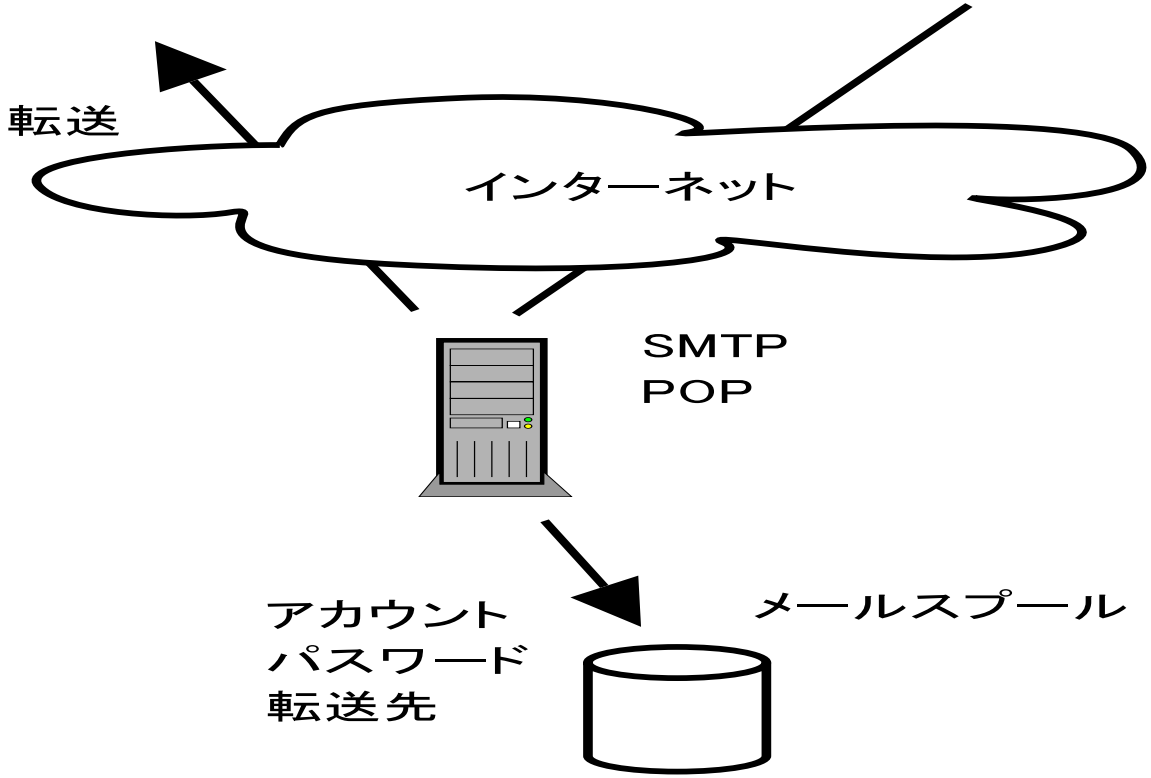
属性の追加

dc=my-domain,dc=comに’objectClass: person’の属性を追加

```
dn: dc=my-domain,dc=com
changetype: modify
add: objectClass
objectClass: person
```

第二部 メールサーバとの連携

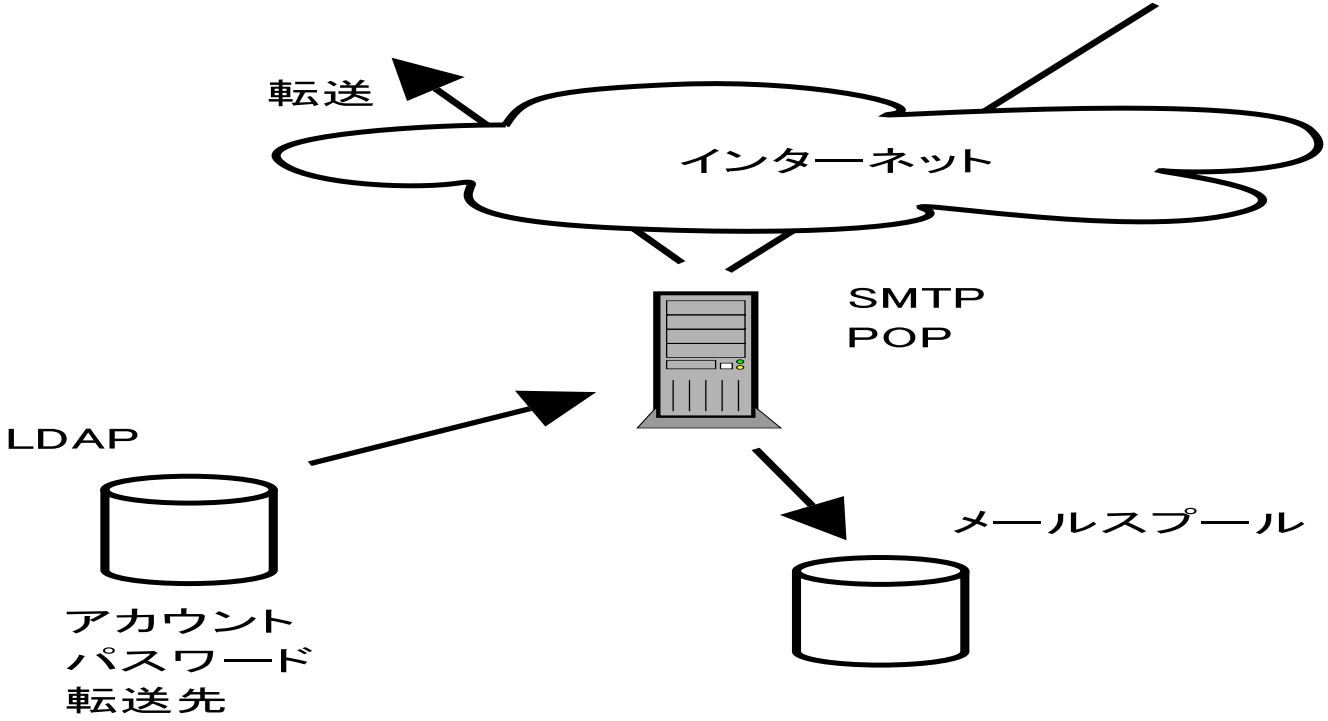
メールサーバ



SMTPサーバ（MDA）
- ローカル記憶装置からアカウント情報を取得
  /etc/passwd, ~/.forward 等

POPサーバ
- SMTPサーバと同様

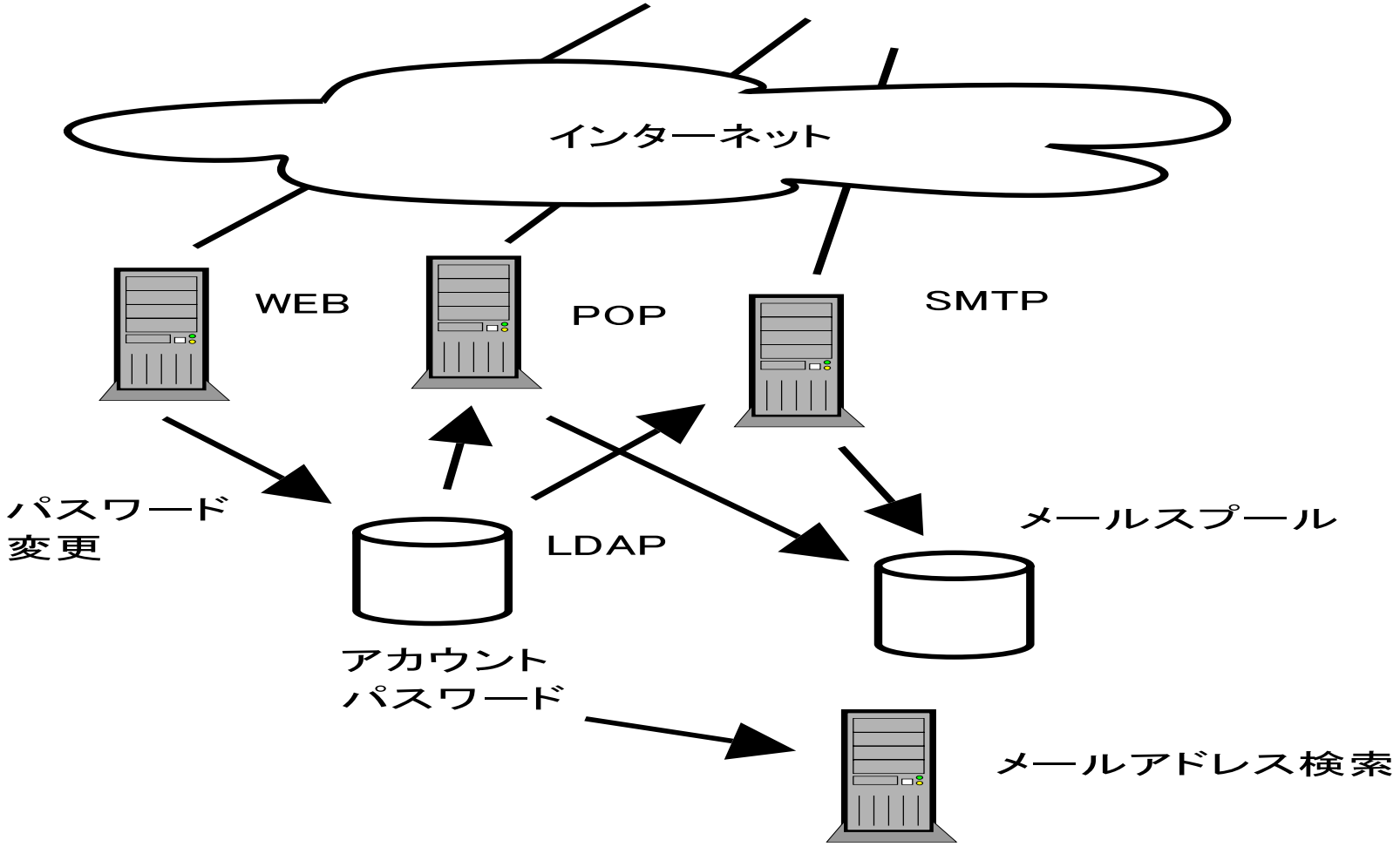
連携イメージ



SMTPサーバ（MDA）
- LDAPからアカウント情報を取得し、存在するユーザであれば、メールスプールに配置
- LDAP上に転送情報があればそちらに転送

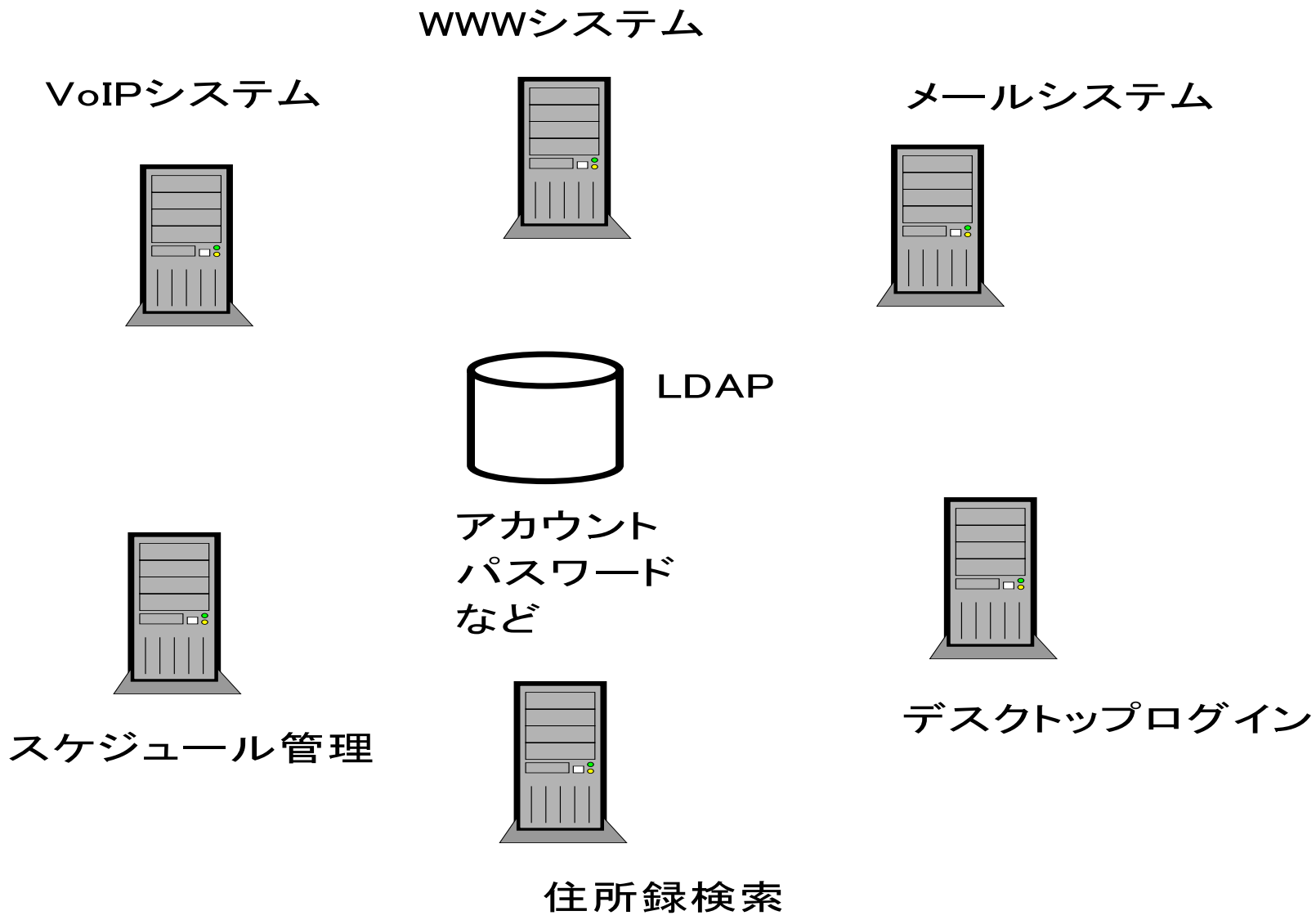
POPサーバ
- LDAPからアカウントとパスワード情報を取得し,認証に成功すればPOPアクセスを許可

LDAPを利用する利点



- アカウント情報の共有
- 標準化されたエントリ変更手続き

LDAPを利用する利点 (2)



LDAPとpostfixの連携

postfix

- 高速なオープンソースメールサーバ
- 標準でLDAPをサポート
- 理解しやすい設定項目

インストール

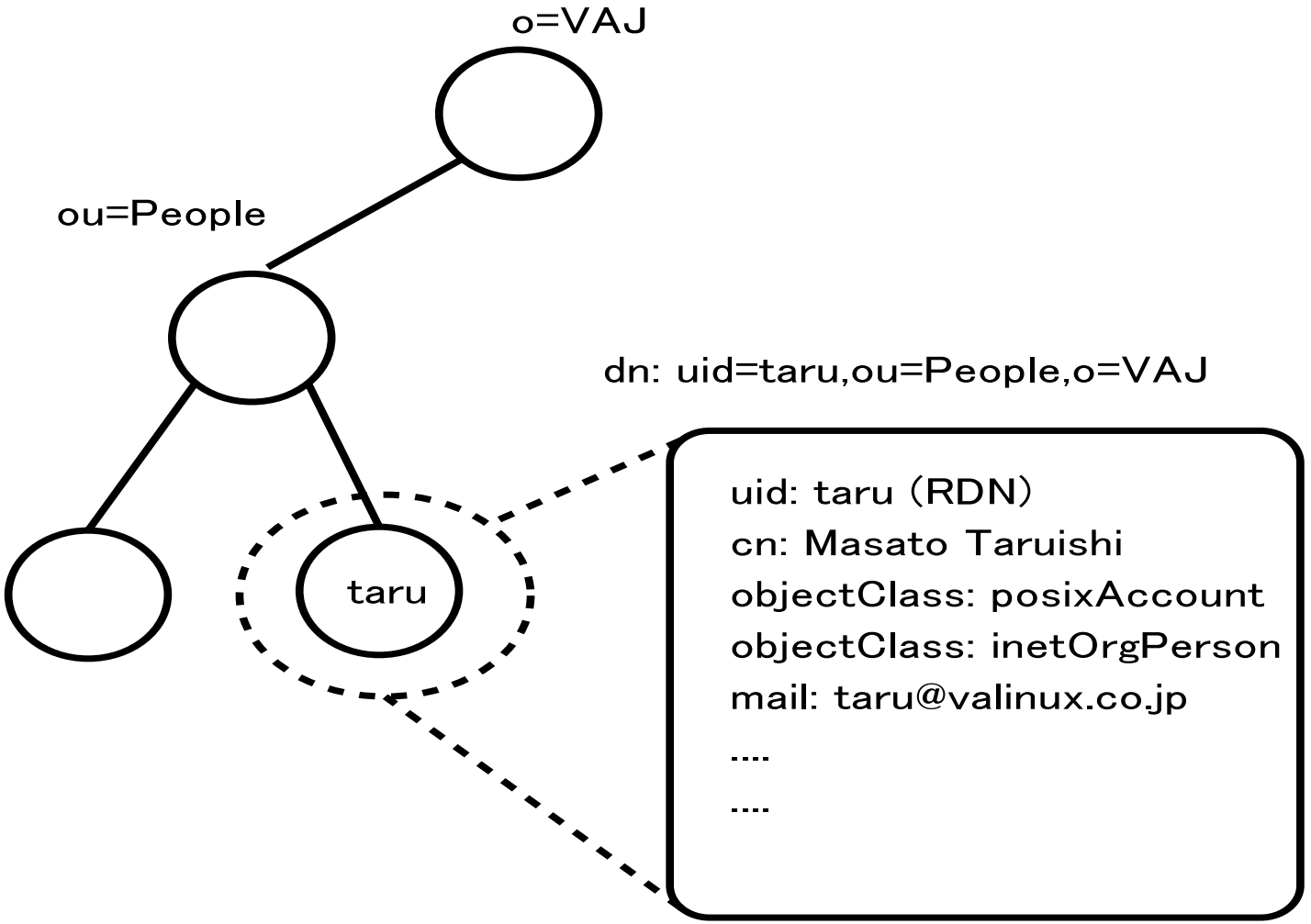
```
$ cd postfix-1.1.2/
$ make tidy
$ make makefiles CCARGS="-I/usr/local/include -DHAS_LDAP"
AUXLIBS="-L/usr/local/lib -lldap -llber"
$ make
$ bin/su -c "make install"
```

postfixのLDAPサポート確認

```
$ /usr/sbin/postconf -m
static
nis
regexp
environ
ldap
btree
hash
```

アカウント情報の登録

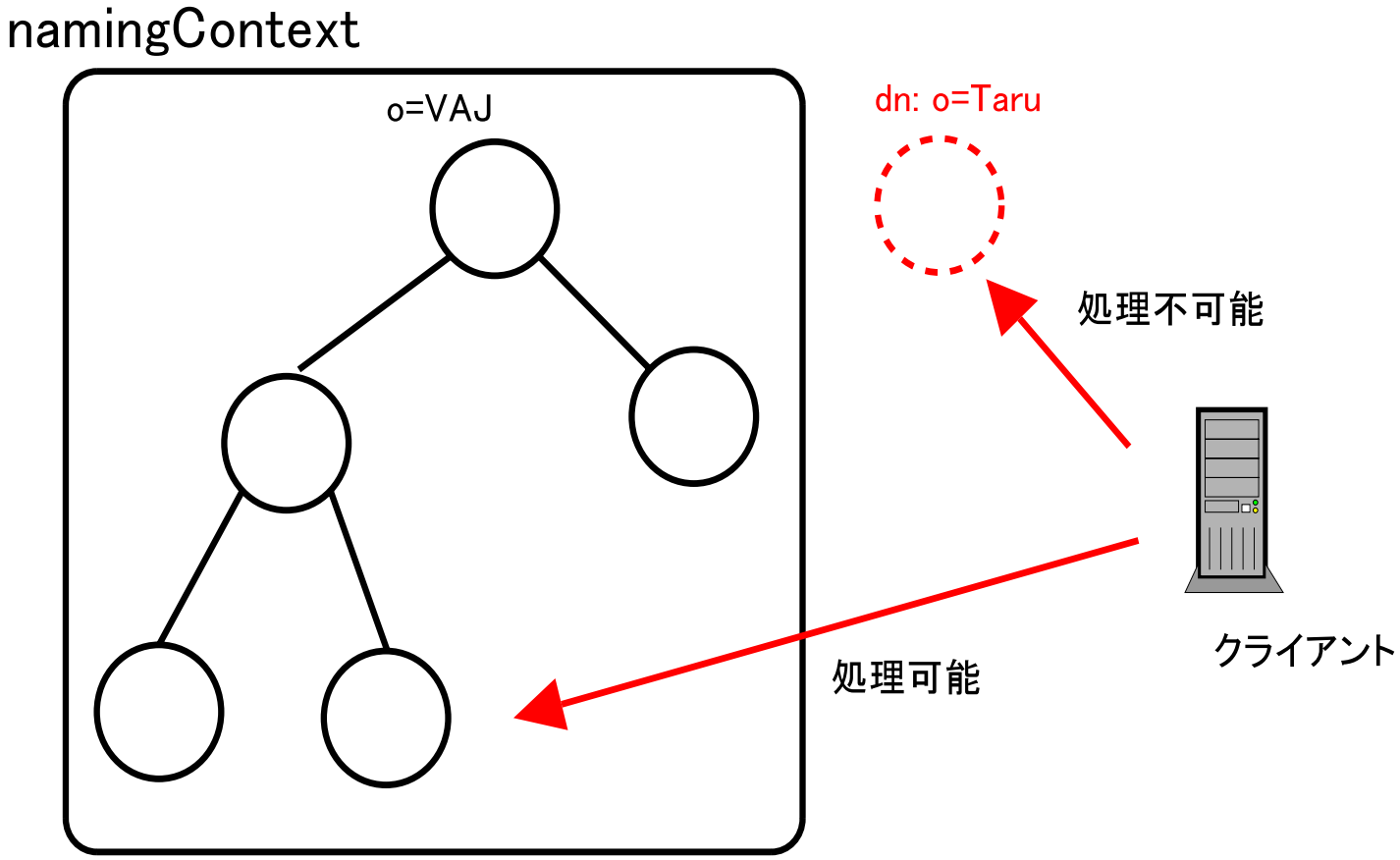
ディレクトリの設計



データベースの作成

namingContextの追加

- サーバが処理可能なディレクトリ



namingContextの追加

slapd.confにデータベース設定を加え、再起動

```
database bdb

suffix "o=VAJ"
rootdn "cn=Manager,o=VAJ"
rootpw secret
directory /usr/local/var/openldap/vaj/
index objectClass eq
```

namingContextの確認

```
$ ldapsearch -s base -b "" namingContexts
```

階層の登録

ou=People,o=VAJを作成するLDIF

```
dn: o=VAJ
objectClass: organization
o: VAJ

dn: ou=People, o=VAJ
objectClass: organizationalUnit
ou: People
```

```
$ ldapadd -x -D "cn=Manager,o=VAJ" -w secret < people.ldif
```

アカウント情報の追加

アカウント taru を作成するLDIF

```
dn: uid=taru,ou=People,o=VAJ
uid: taru
cn: Masato Taruishi
objectClass: posixAccount
objectClass: inetOrgPerson
userPassword: {CRYPT}h9Z4VF89hXpl.
loginShell: /bin/bash
uidNumber: 1000
gidNumber: 1000
homeDirectory: /home/taru
sn: Taruishi
mail: taru@debian.org
```

```
$ ldapadd -x -D "cn=Manager,o=VAJ" < taru.ldif
```

パスワードの管理

平文のまま保存するのは危険

- slappasswd を使ってパスワード情報を暗号化

```
$ /usr/local/sbin/slappasswd -h "{CRYPT}"
New password:
Re-enter new password:
{CRYPT}h9Z4VF89hXpl.
```

メール転送先をLDAPから取得

postfixのmain.cf

```
alias_maps = ldap:ldapalias
```

- 転送先情報をLDAPのldapalias表から取得する指定

ldapalias表の作成

postfixのmain.cf

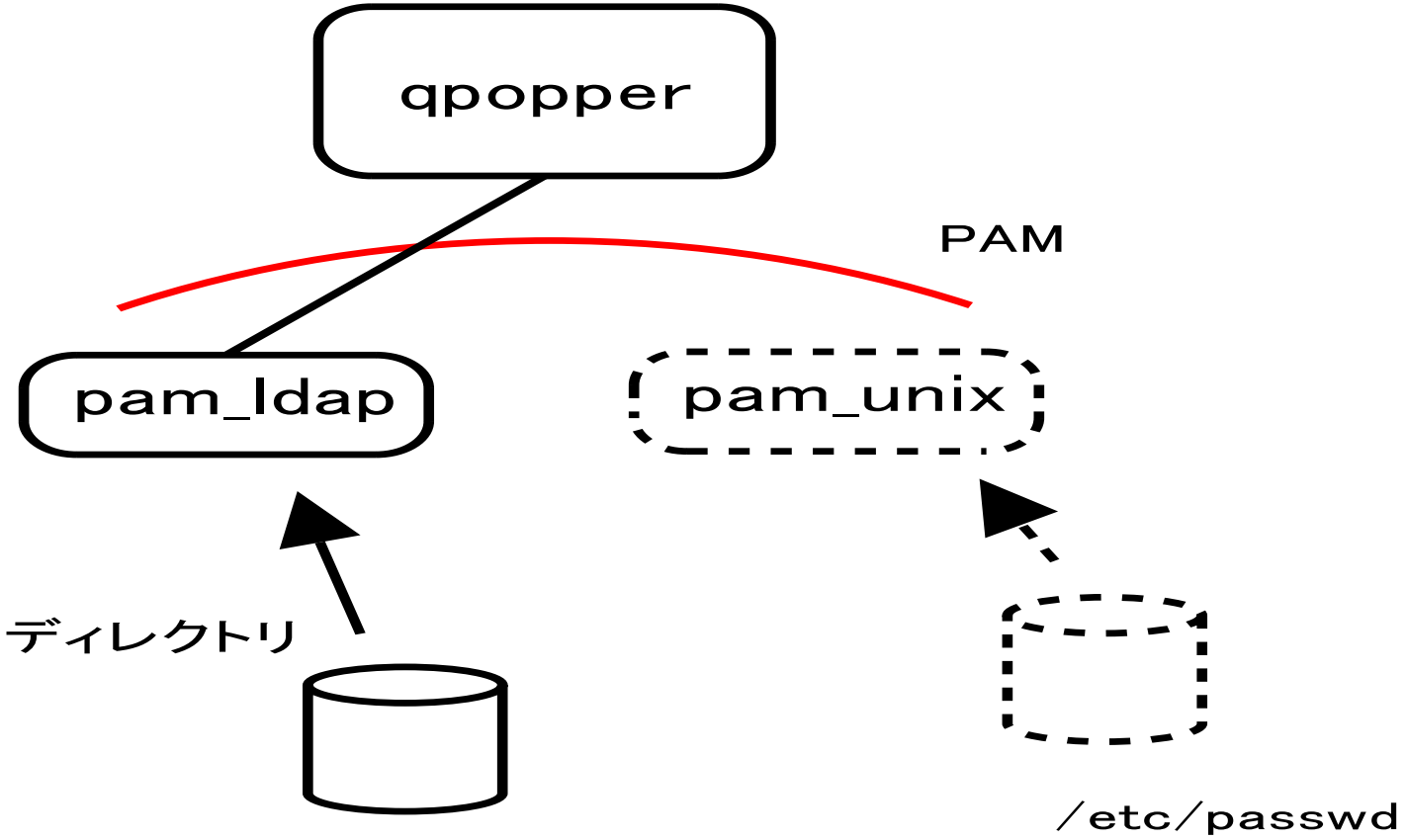
```
ldapalias_server_host = localhost
ldapalias_search_base = ou=People,o=VAJ
ldapalias_scope = sub
ldapalias_query_filter = (uid=%s)
ldapalias_result_attribute = mail
```

```
$ postmap -q taru ldap:ldapalias
taru@debian.org
```

pop サーバからLDAPを使う

qpopperからpam_ldapを利用する

- PAMのLDAPバックエンド



pam_ldapのインストールと設定

インストール

```
$ ./configure
$ make
$ /bin/su -c "make install"
```

設定

/etc/ldap.conf (Debianでは/etc/pam_ldap.conf)

/etc/ldap.conf

pam_ldapとnss_ldapの両方の設定を格納するファイル

pam_ldapに必要な設定

```
host 127.0.0.1
base ou=People,o=VAJ
ldap_version 3
pam_password crypt
```

qpopperの認証にLDAPを使う

/etc/pam.d/qpopper

```
auth       sufficient   pam_ldap.so
auth       required     pam_unix_auth.so shadow

account    sufficient   pam_ldap.so
account    required     pam_unix_acct.so
```

第三部 大規模システムへのLDAPの適用

大規模システムの特徴

考慮すべき問題点

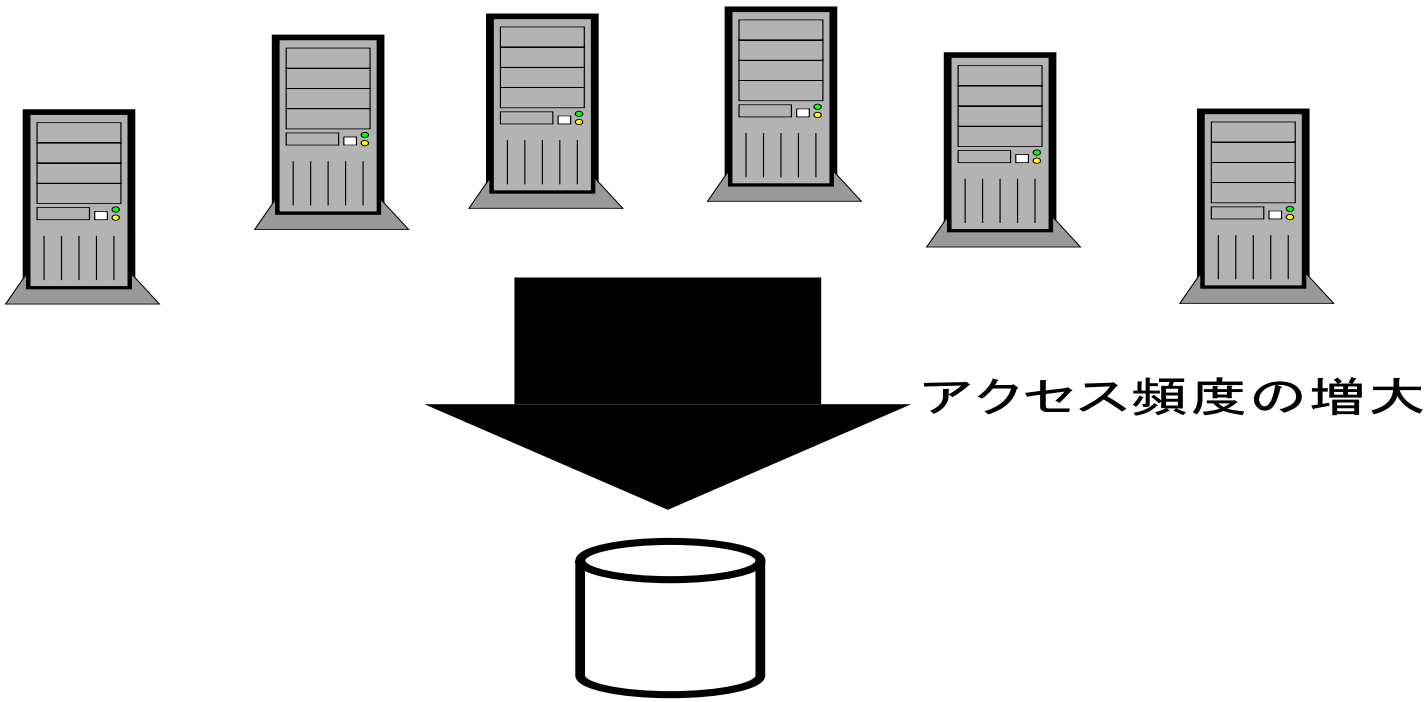
- 高負荷
- リソース枯渇
- システム障害

問題点 - 高負荷

- LDAPサーバを利用するユーザの増大
  - アクセス頻度の増大
- エントリ数増大による影響

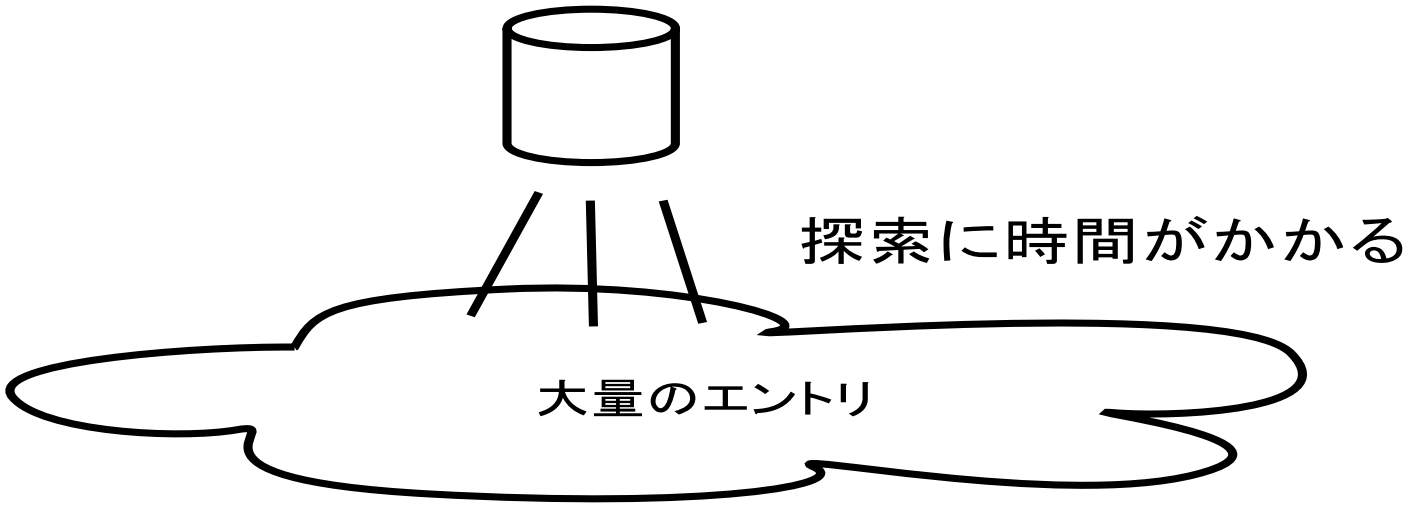
利用ユーザ数の増大

- 検索や更新の頻度が増える
  - サーバマシン性能の限界
  - ディスク入出力のオーバヘッド
  - 帯域の圧迫



エントリ数増大による影響

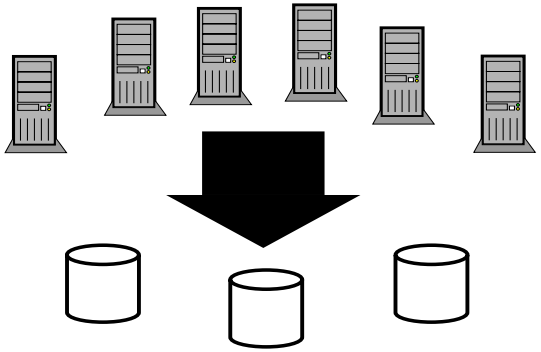
- エントリ数に関係した検索や更新アルゴリズムの計算量
  - 毎回全件探索をすると非常に遅い



高負荷対策

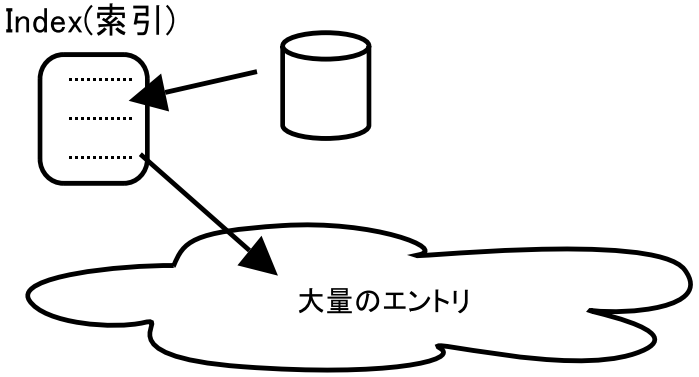
負荷分散

- サーバを複数台用意し、アクセスを分散させる



計算量の軽減

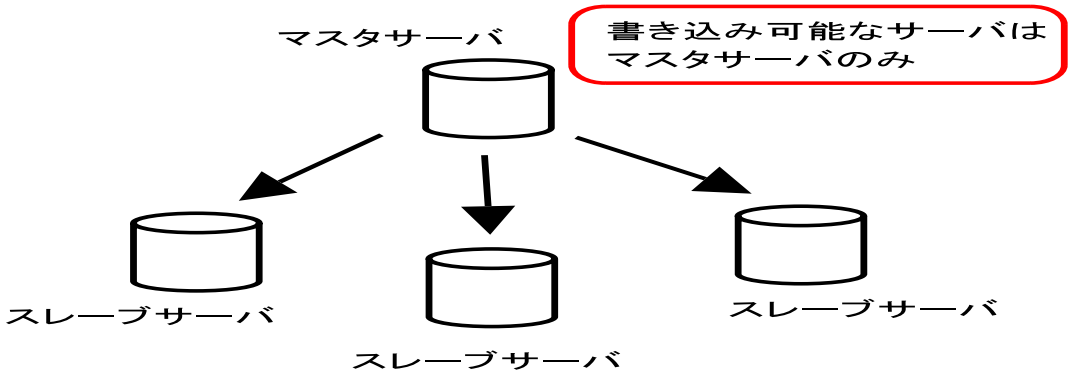
- index を作成する



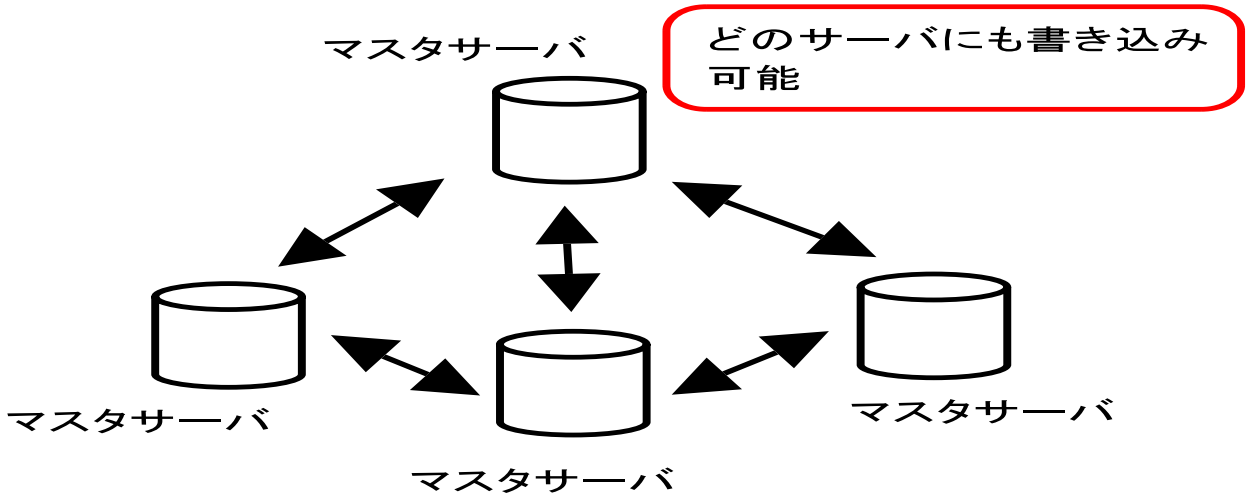
負荷分散

ディレクトリの複製(レプリケーション)

○シングルマスタレプリケーション

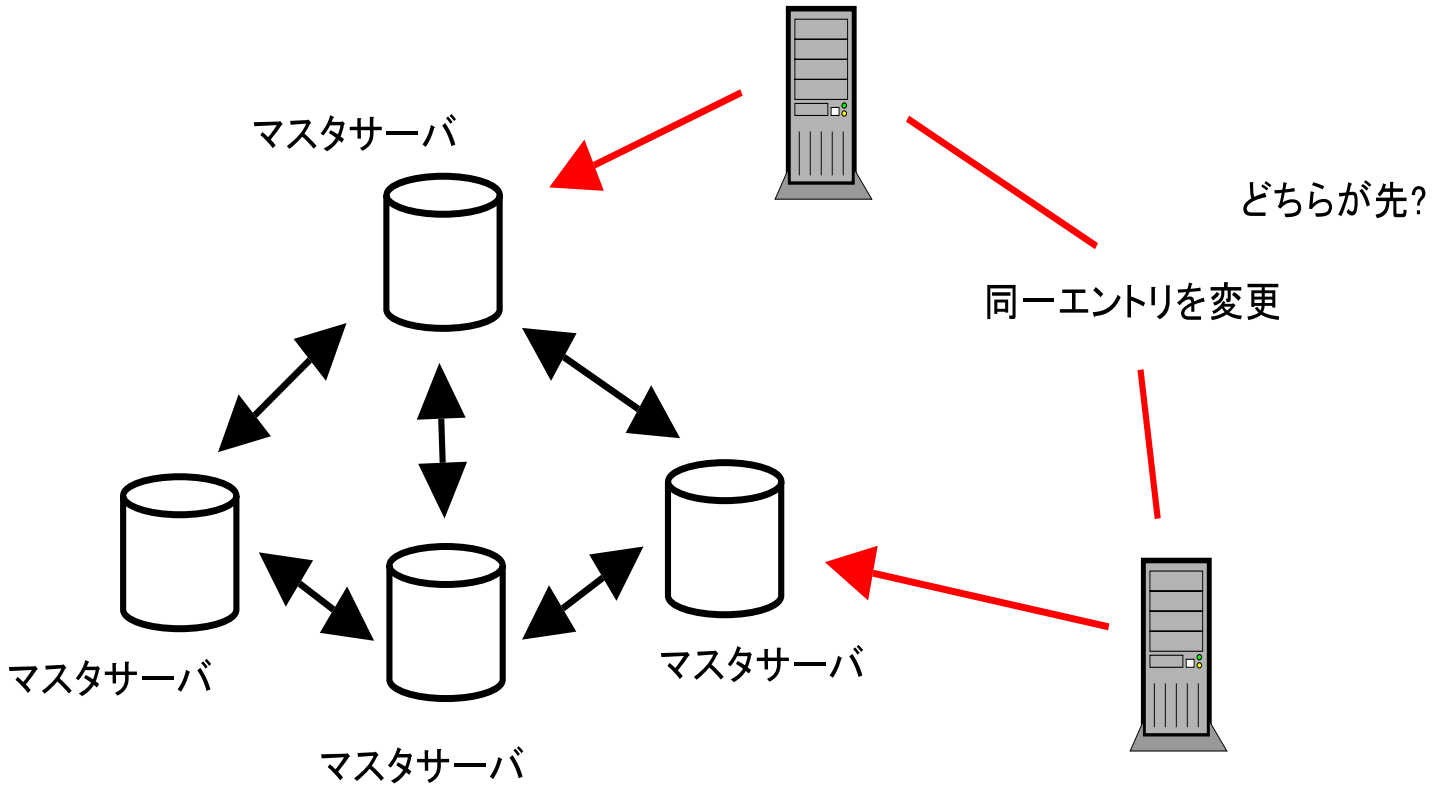


○マルチマスタレプリケーション



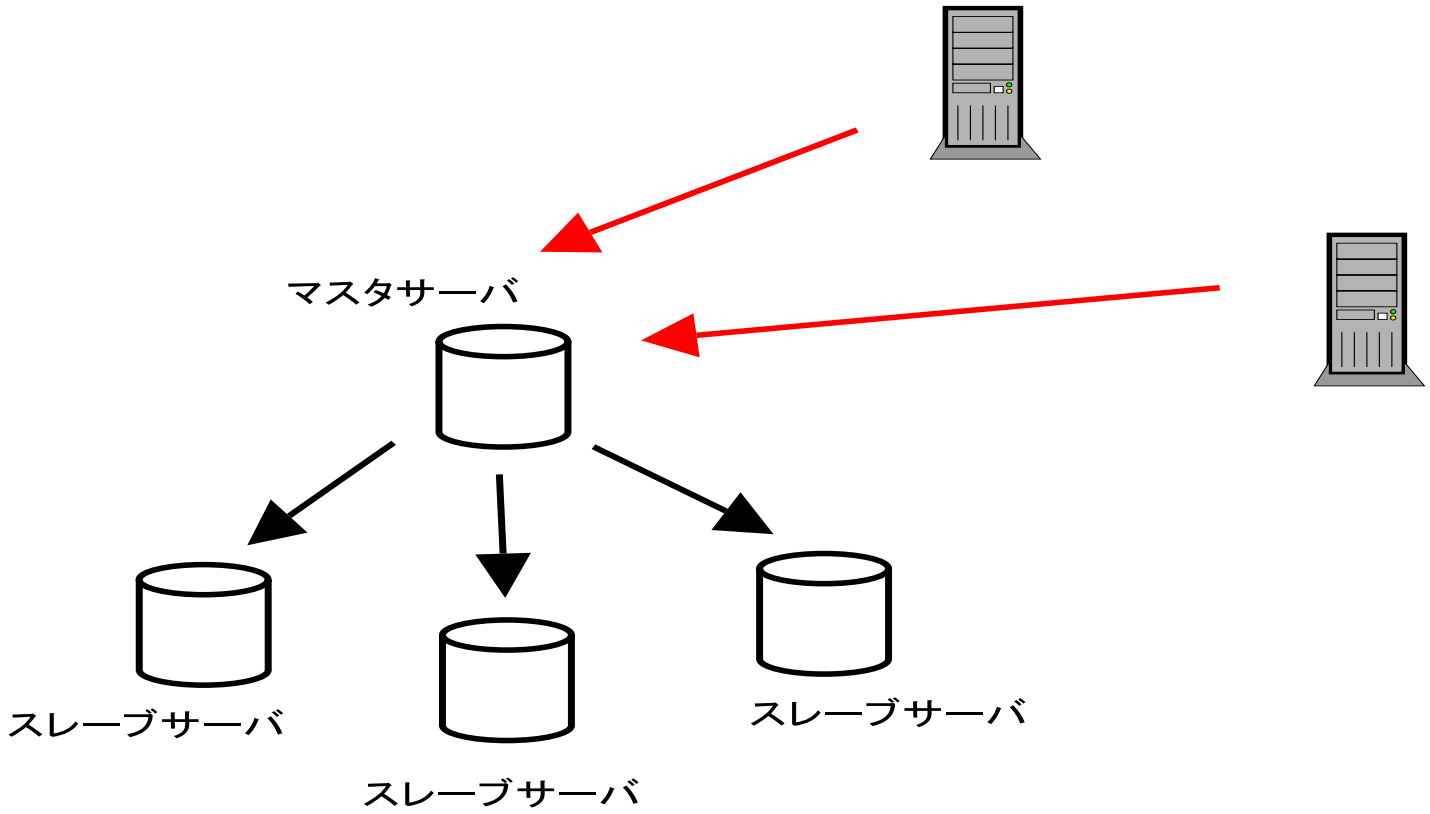
マルチマスタレプリケーション

- どの複製サーバにも書き込みが可能
- 一台のサーバが停止しても別のサーバに書き込みが可能
- 書き込みの整合性をとることが難しくなる



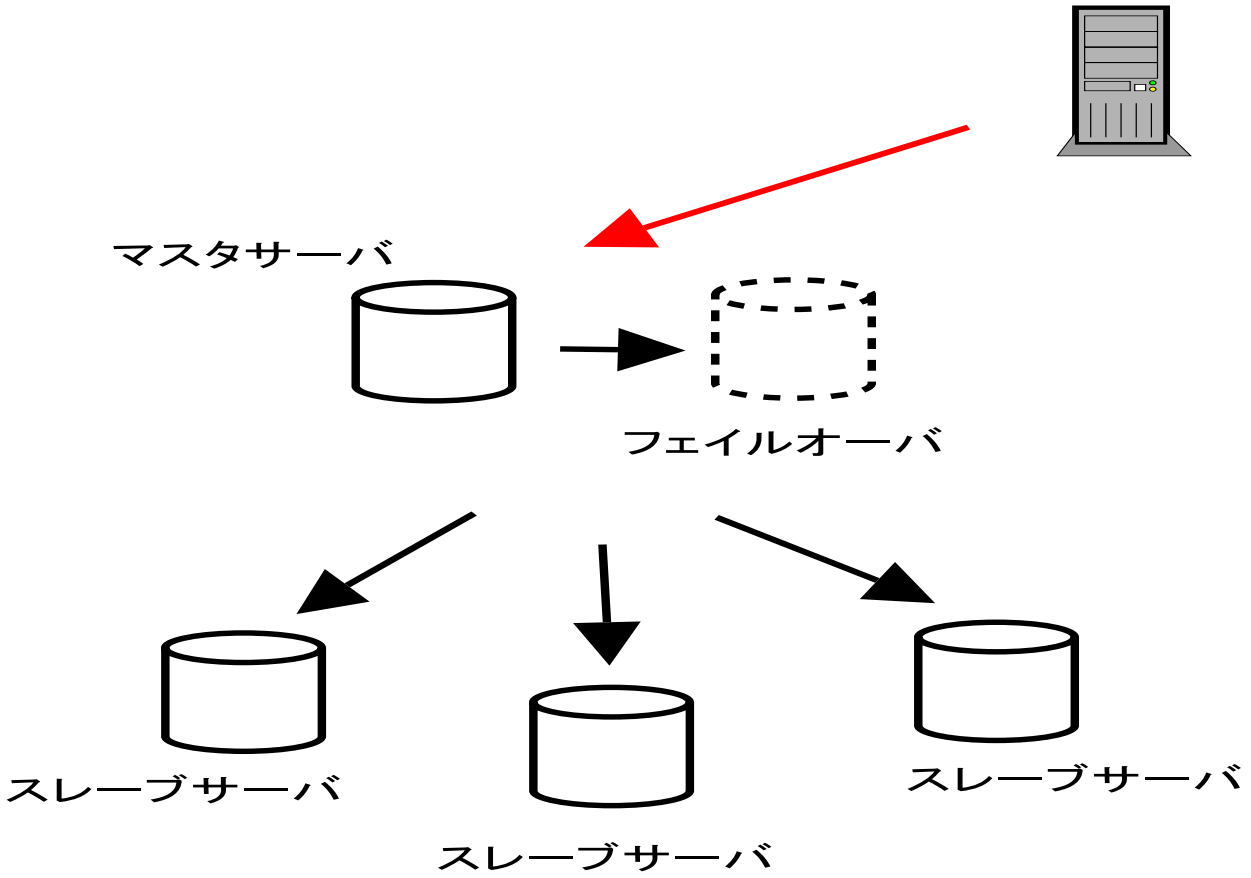
シングルマスタレプリケーション

- 書き込みの整合性確保が単純
- クライアント側で書き込みとそれ以外のサーバリストを管理
- マスタサーバが停止すると書き込みが停止



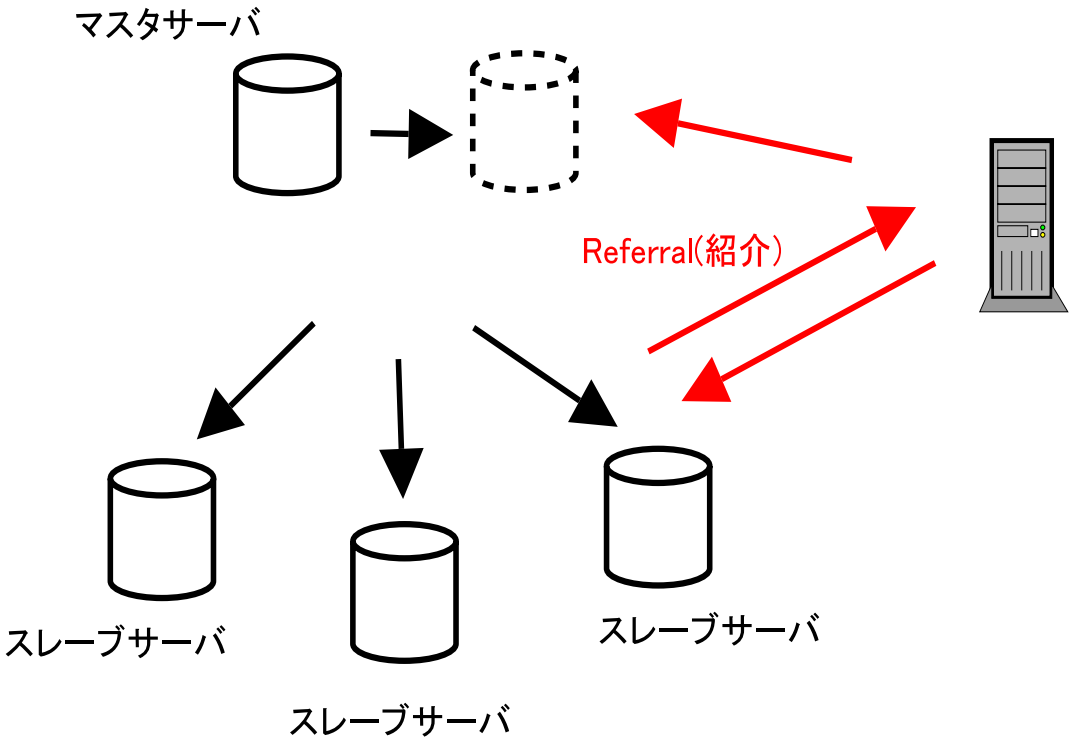
フェイルオーバ

サービス停止時に別のサーバがサービスを代理



Referral (紹介)

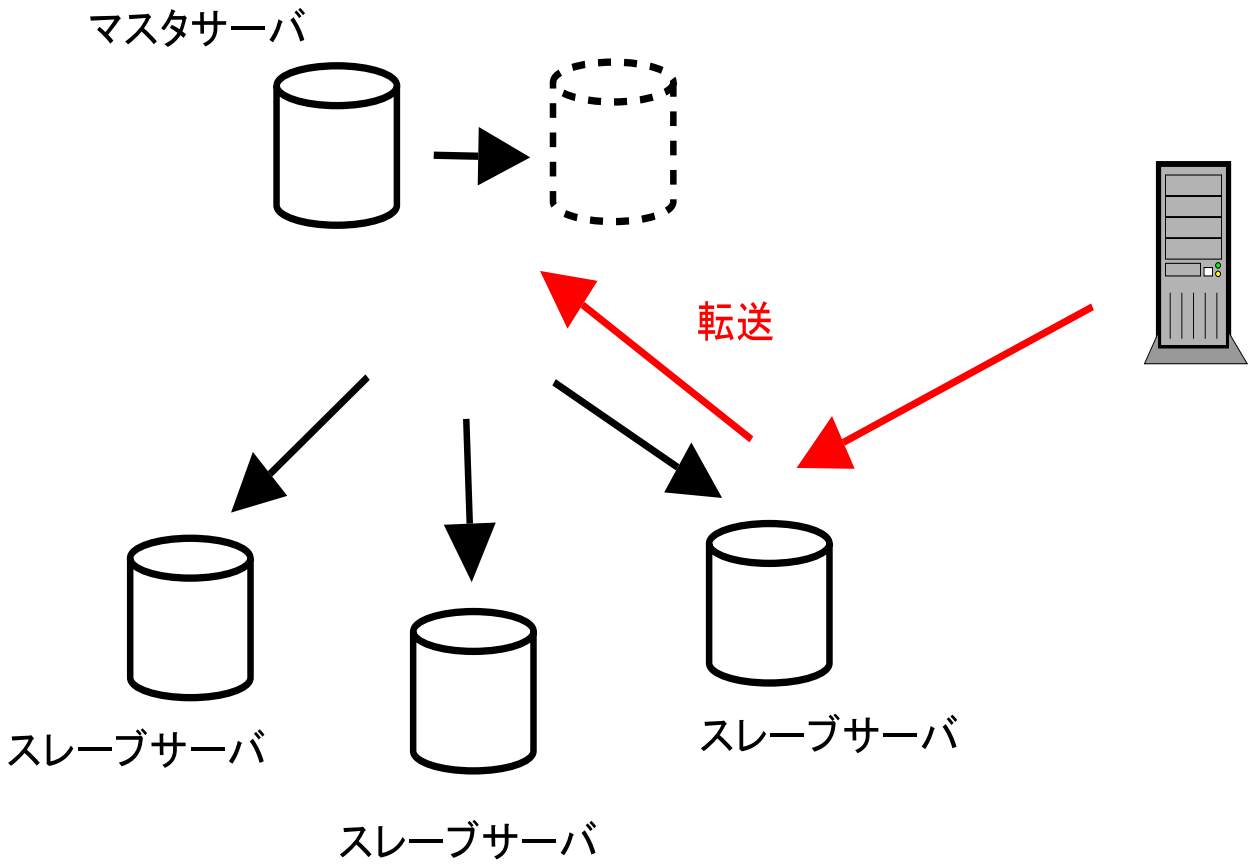
読込専用サーバに書込要求が来た場合に書込サーバを教える(書込用、読込|用|のサーバリストの管理が不要)



シングルマスタ+Referral+フェイルオーバを用いる事で更新環境の高可用性を実現可能

Chain

書込要求をマスタサーバに転送する
- OpenLDAP では実装されていない



OpenLDAPによるレプリケーション

シングルマスタレプリケーション+Referral

マスタサーバの設定

```
replogfile /usr/local/var/openldap-data/replogfile

database bdb
...
replica host=slave:389
        bindmethod=simple
        binddn="cn=Manager,o=VAJ"
        credentials=secret
...
```

OpenLDAPによるレプリケーション (2)

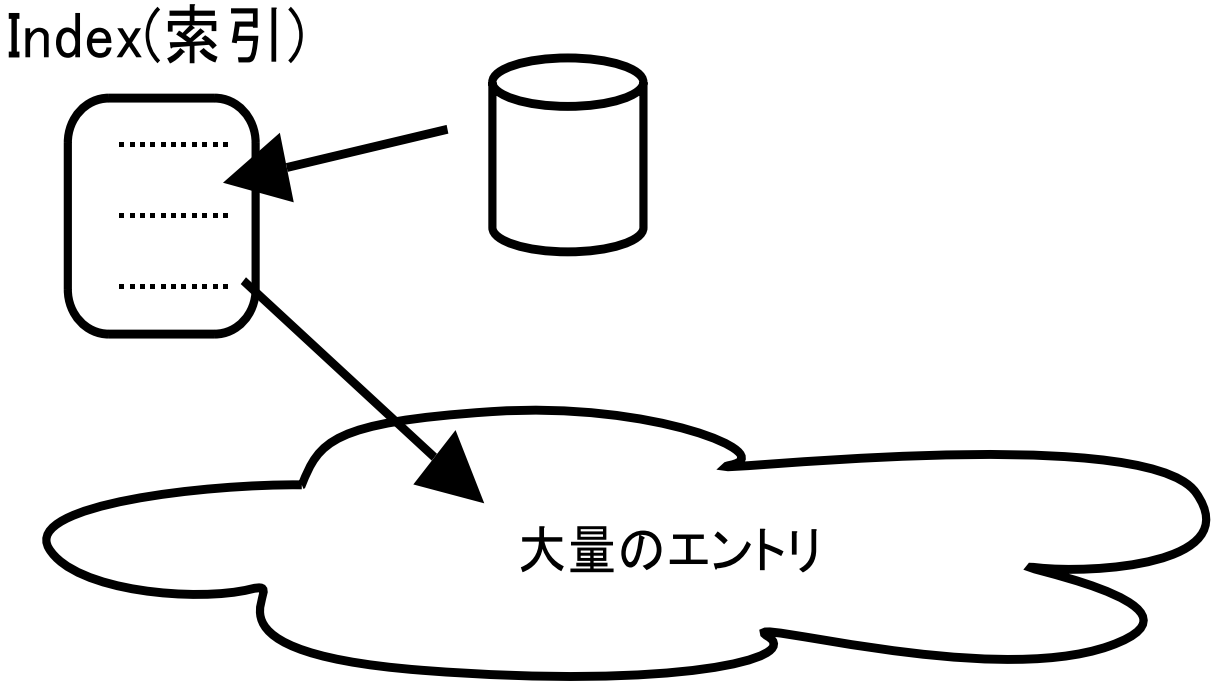
スレーブサーバの設定

```
...
database bdb
...
updatedn "cn=Manager,o=VAJ"
updateref "ldap://master:389/"
...
```

indexの活用

索引を引く事により計算量を大幅に削減

- 書き込み時に索引作成分のオーバヘッドがかかる
- 適切な索引の指定



indexの設定

slapd.conf

index {<attrlist>|default} [pres,eq,approx,sub,<special>]

```
...
database bdb
...
index objectClass,uid eq
index cn,sn eq,sub
...
```

index設定を変更した場合はslapindex(8)を呼ぶ

高負荷対策のまとめ

負荷分散

- レプリケーションによる検索の分散
- 更新の分散は(検索機能と比べると)メリットが少ない
  - トランザクションマネージャに負荷が集中

LDAPサーバの改良、高性能ディスクの使用

index の活用

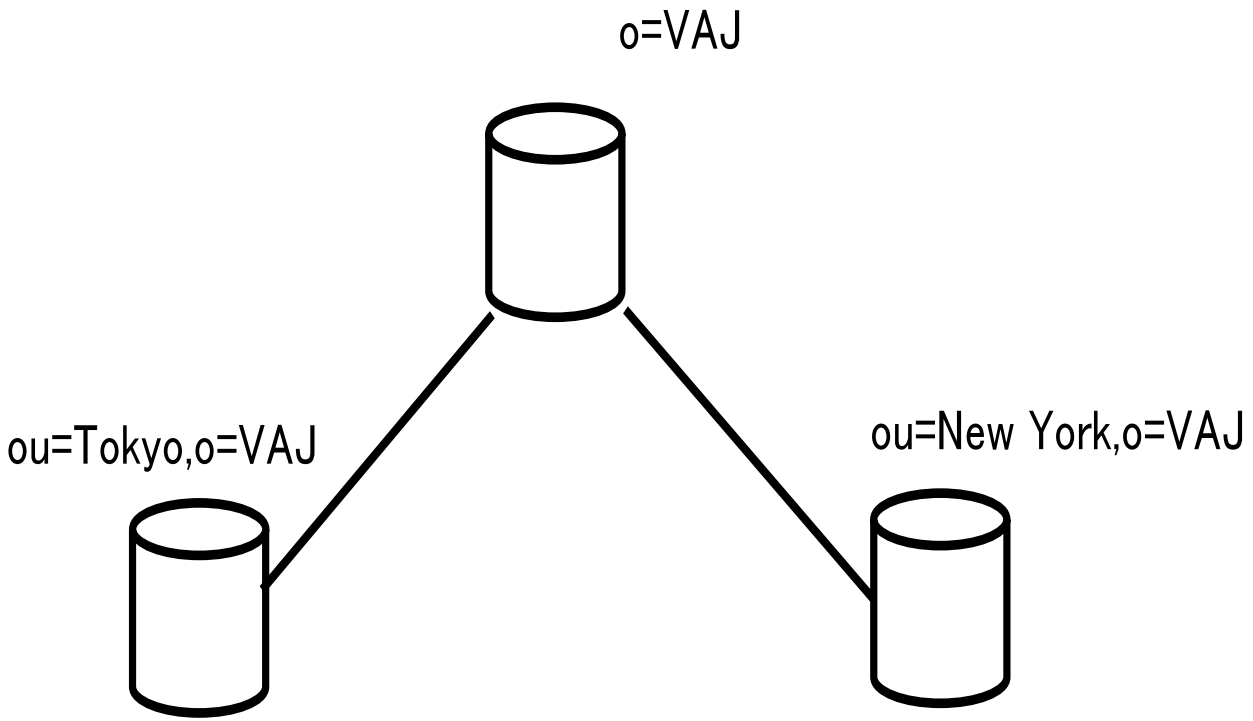
- 更新負荷とのトレードオフ

問題点 - リソース枯渇

- ディスク容量
- ファイルシステムによるファイルサイズの制限
- 固定長整数を利用するアプリケーション
  - UNIX のユーザID

リソース枯渇対策 - ディスク容量

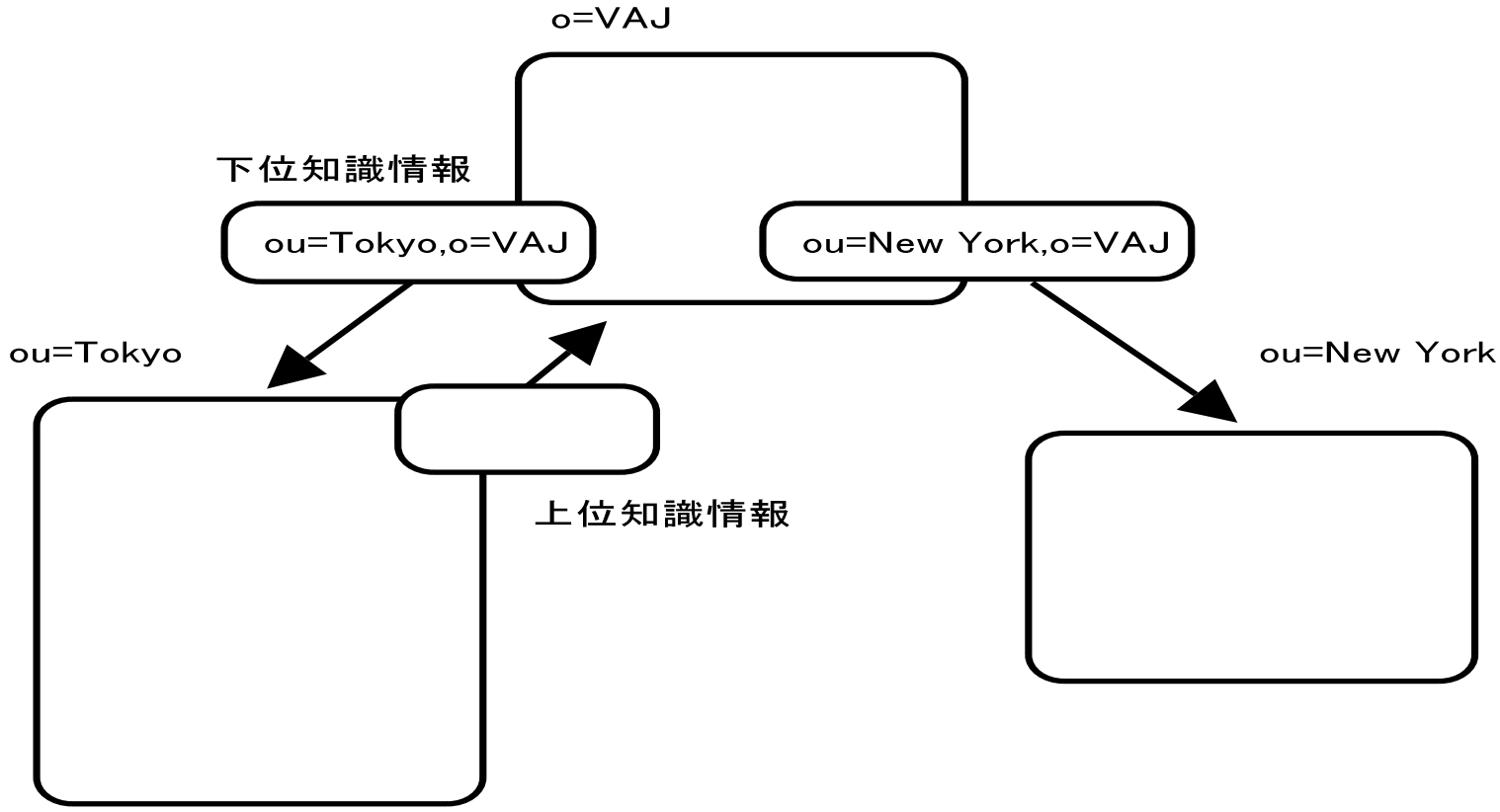
- ディスクの増量
- ディレクトリ階層毎に分散
  - (組織的な分散管理にもなる)



下位/上位知識情報

そのサーバが管理しないディレクトリに対する情報

- 上位知識情報 (Superior Knowledge Information)
- 下位知識情報 (Subordinate Knowledge Information)



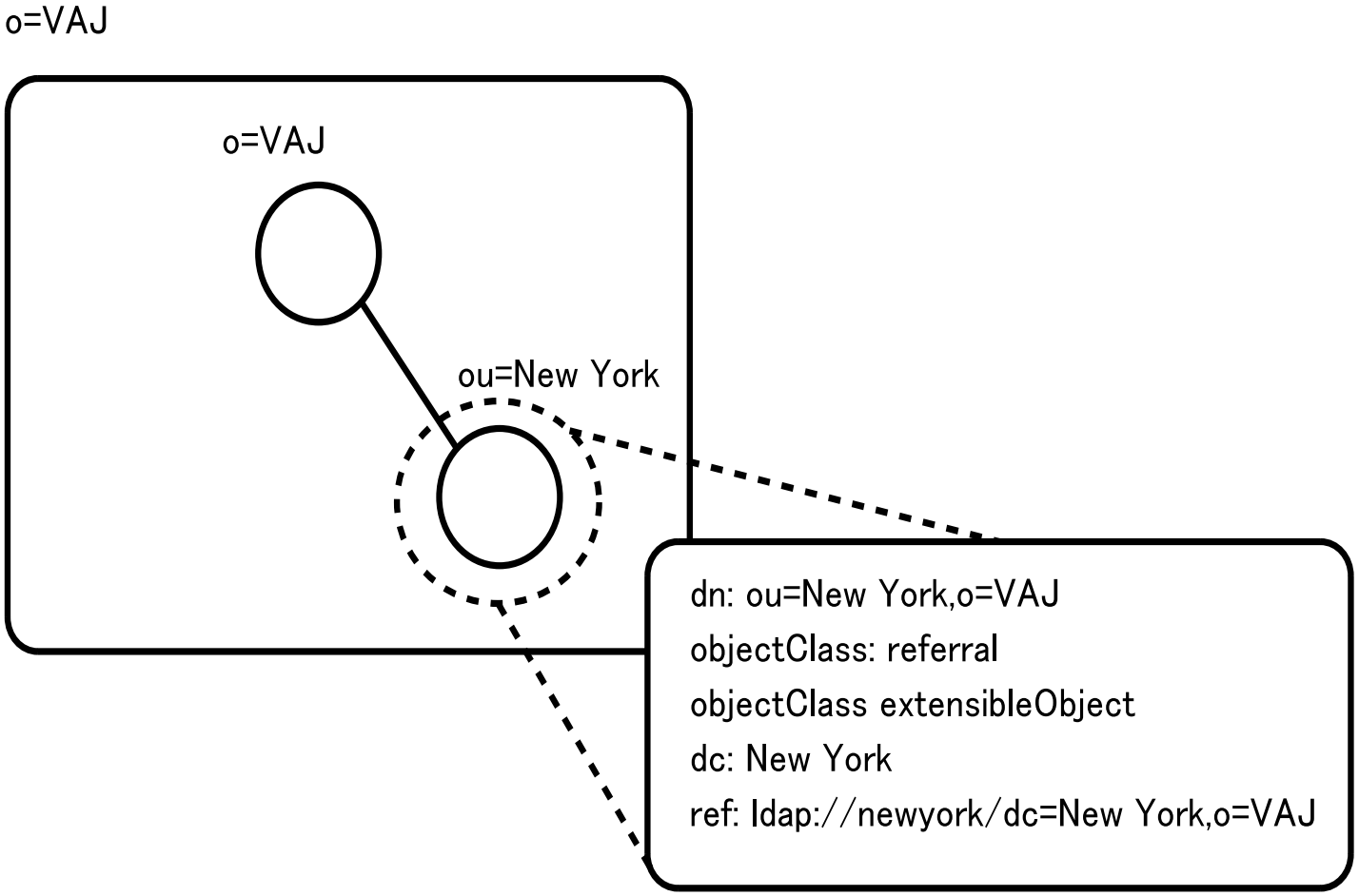
上位知識情報の設定

slapd.conf

```
referral ldap://superior/
```

下位知識情報の設定

ディレクトリに下位知識情報であることを示すエントリを書き込む



固定長数値を扱うアプリケーション

UNIX uidは16bit

- 数十万、数百万のアカウントを管理できない
- qpopperはuidに依存する
- uidに依存しない対策
  - 1アカウントですべてのメールを管理する

リソース枯渇対策のまとめ

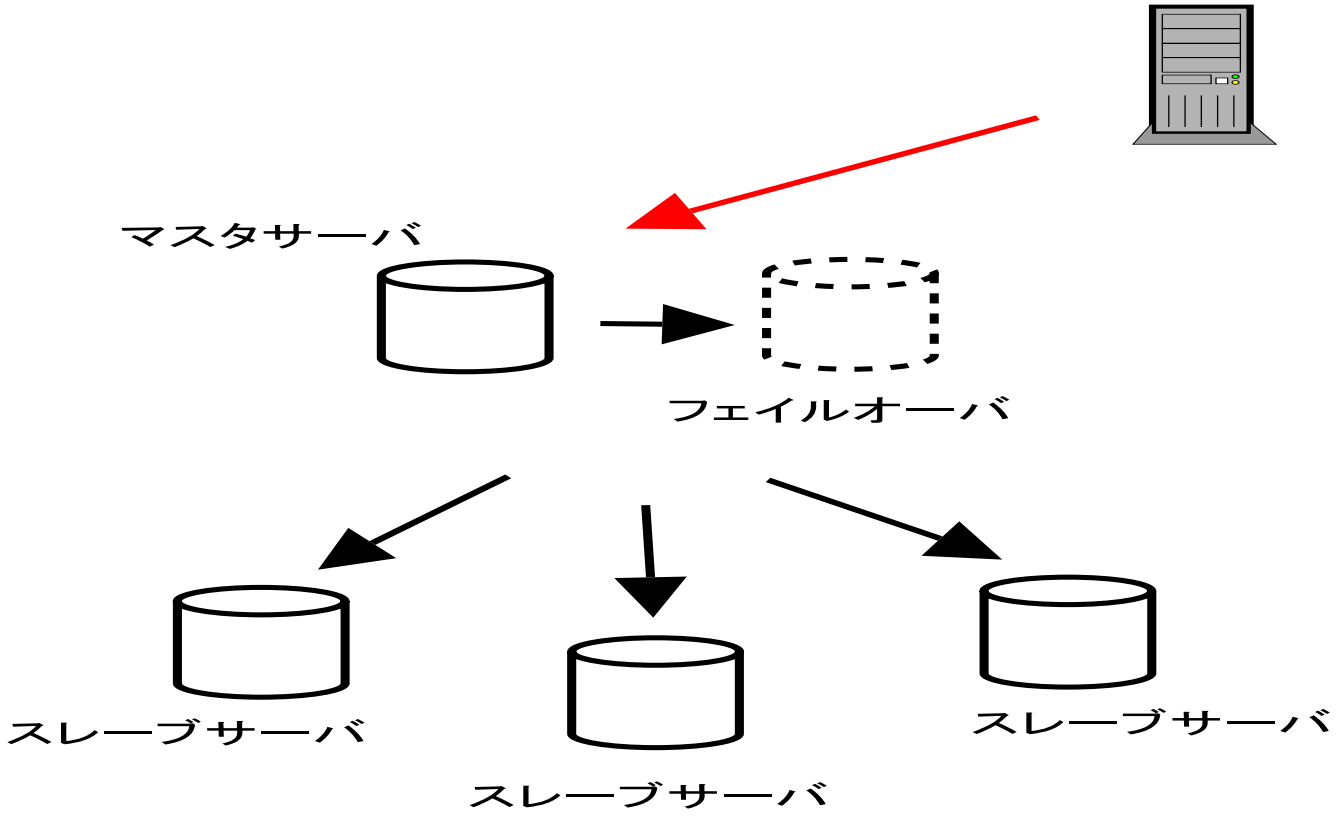
- 分散ディレクトリ
  - 下位/上位知識情報
- 固定長のデータに依存しないか?

問題点 - システム障害

- 高負荷によるトラブル発生の可能性が増大
- データファイル破損の可能性が増大
  - 破損による損失規模が大きい
- システム停止中のアクセス機会損失
  - ユーザへの対応コストの増大

システム障害対策 - トラブル発生

フェイルオーバによるサービスの継続
- 提供サーバが複数台あるサービスでは不要
  - 検索機能が負荷分散されている場合
  - マルチマスタ環境の場合



マスタサーバのフェイルオーバ

必要なコンポーネント

- OpenLDAP - LDAPサーバ
- heartbeat - システムが動作しているか調べる
- mon - サービスが提供されているか調べる

フェイルオーバ環境の構成

heartbeatの役割 (相手の監視)

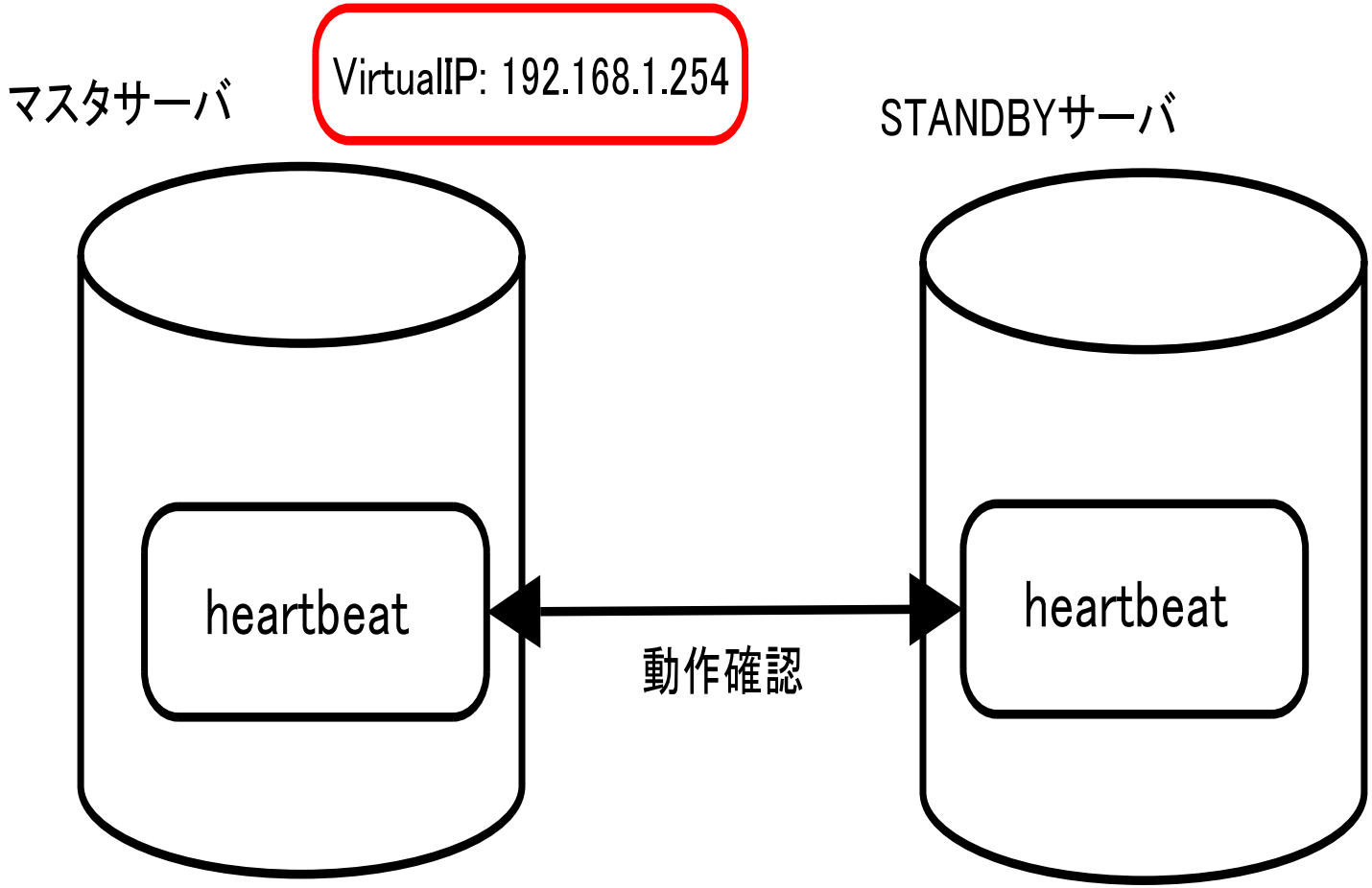
- ACTIVEサーバの動作を監視する(hearbeatプロトコル)
- ACTIVEサーバ全体が停止した時、STANDBYサーバが昇格する

monの役割 (自分の監視)

- ACTIVEサーバ上のLDAPサービスの動作を確認する (LDAPプロトコル)
- LDAPサービスが止まっている場合、heartbeatデーモンを停止する

構成

ACTIVEサーバにVirtual IPを付与
STANDBY時はスレーブモードで動作し、ACTIVEサーバからディレクトリをレプリケーション
フェイルオーバ時にVirtual IPを受け継ぎ、マスタモードで動作するようにLDAPサーバを再起動する



heartbeatのインストール

```
$ ./configure; make; /bin/su -c "make install"
```

monのインストール

必要な perl モジュール

- Peiod-1.20
- Time-HiRes-1.42
- Convert-BER-1.3101
- Mon-0.11
- Convert-ASN-0.16
- perl-ldap-0.26

量が多いため、パッケージを利用するほうが楽

```
$ cd mon-0.99.2/
# install mon /usr/local/sbin/
# install alert.d/* alert/template /usr/local/lib/mon/alert.d/
# install mon.d/*.monitor /usr/local/lib/mon/mon.d/
```

heartbeatの設定

/usr/local/etc/ha.d/hacf
/usr/local/etc/ha.d/authkeys
/usr/local/etc/ha.d/haresources
/usr/local/etc/ha.d/resource.d/ldap

hacfの設定

heartbeat全体(ノードの設定を含む)の設定

```
...
node master       # 初期 ACTIVE サーバのホスト名
node backup       # 初期 STANDBY サーバのホスト名
...
```

authkeysの設定

認証情報の設定

```
auth 2
2 sha1 HI!
```

haresourcesの設定

リソースの設定

```
...
master    172.17.91.254 ldap
...
```

resource.d/ldap

フェイルオーバ時にLDAPサーバを起動するスクリプト

- 標準配布ではない
- インターネットから取得する

heartbeat の動作確認 - 起動

ACTIVE, STANDBY の両方で heartbeat を起動
- /etc/init.d/heartbeat start

ACTIVE側にVirtual IP が付与されていることを確認する
- /sbin/ifconfig

ACTIVE側はマスタモードでLDAPサーバが起動していることを確認する
- ps ax | grep slapd
  - -f slapd.master.conf がある

STANDBY側はスレーブモードでLDAPサーバが起動していることを確認する
- ps ax | grep slapd

heartbeat の動作確認 - フェイルオーバ

ACTIVE側のheartbeatを停止する

- /etc/init.d/heartbeat stop
- ACTIVE側のLDAPサーバが停止していることを確認する
- STANDBY側のLDAPサーバがマスタモードで起動していることを確認する

mon の設定

/usr/local/etc/mon/mon.cf
/usr/local/etc/ha.d/haresources
/usr/local/lib/mon/alert.d/failover.alert

mon.cf の設定

```
alertdir = /usr/local/lib/mon/alert.d
mondir = /usr/local/lib/mon/mon.d
histlength = 100

hostgroup ldapmaster master       # VirutlaIPを持つホスト名

watch ldapmaster
  service ldap
     interval 10s            # 正常時動作チェックの間隔
     failure_interval 2s     # 異常時の回復チェック間隔
     # ldapsearch を行ってサーバの動作をチェックする
     monitor ldap.monitor --basedn='cn=Manager,o=VAJ' --filter='cn=Manager'
--value='organizationalRole'
     period wd {Sun-Sat}
        alert failover.alert
        alertafter 10s    # 異常と判断してからアラートとするまでの時間
        numalerts 1
...
```

haresourcesの設定

リソースの設定

```
...
master    172.17.91.254 ldap mon
...
```

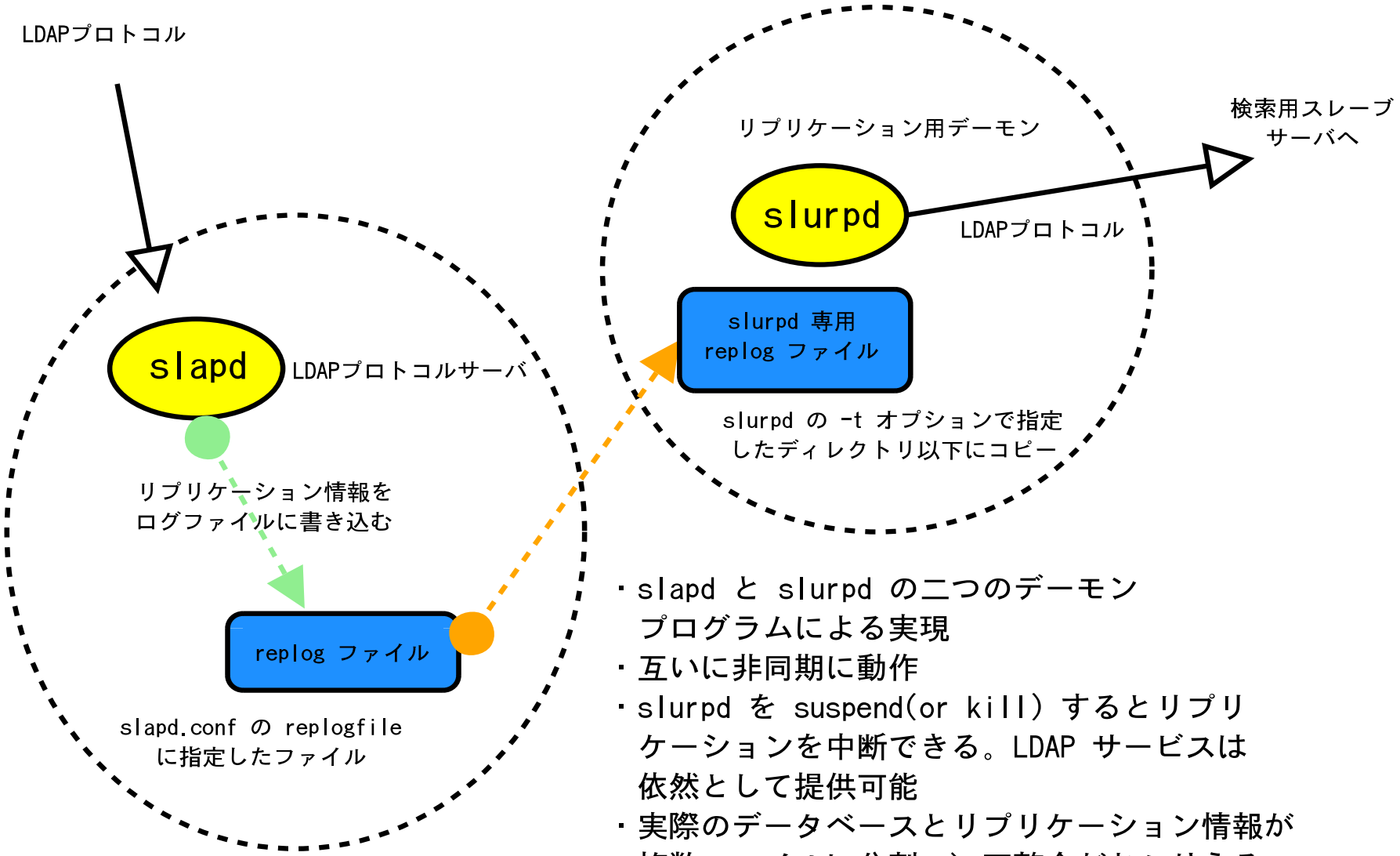
mon の動作確認 - 起動

- heartbeatを起動する
- ACTIVE側のLDAPサーバを停止する
- フェイルオーバが発生したことを確認する

フェイルオーバの問題

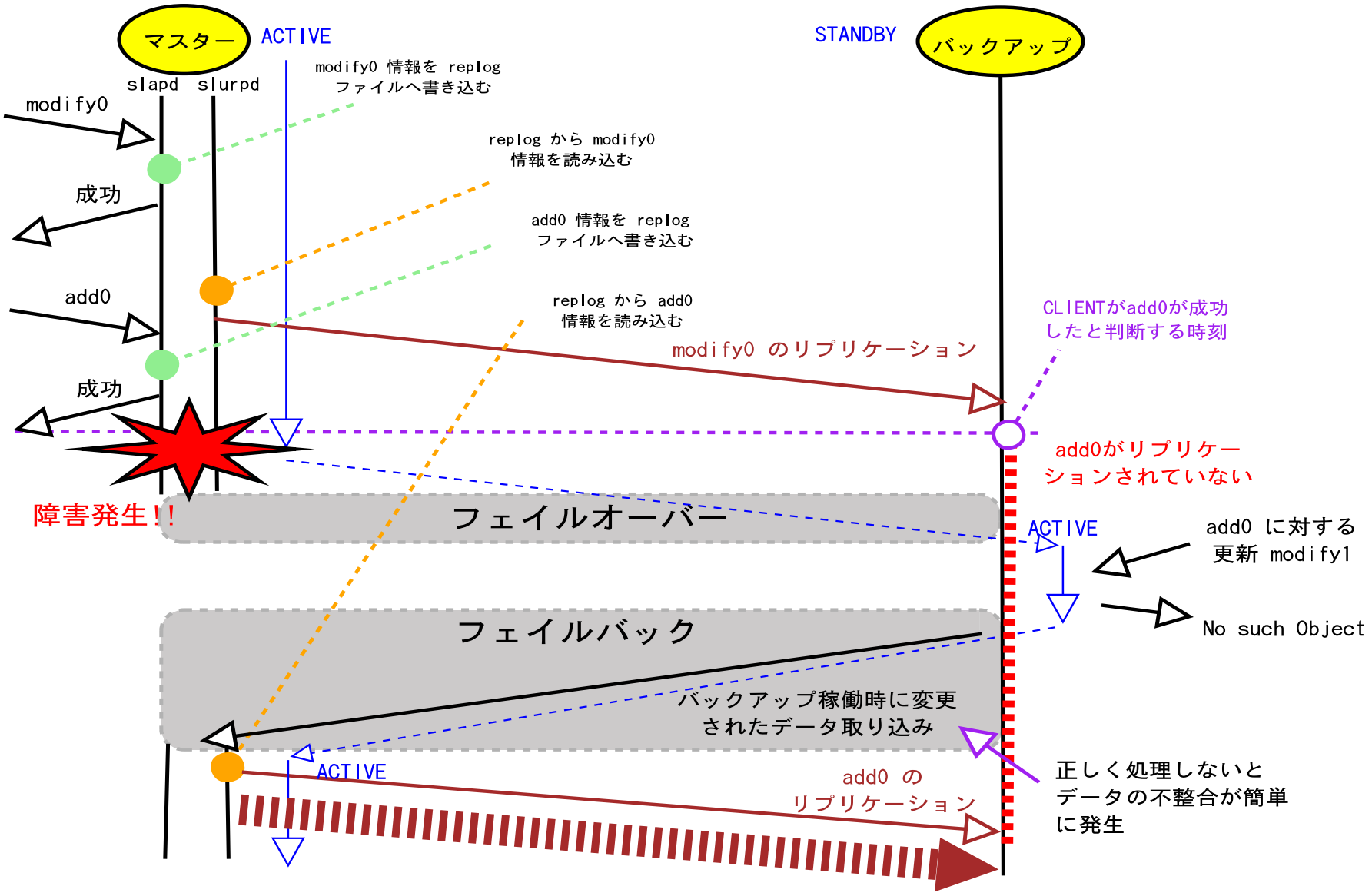
ディレクトリの内容がローカルに保存される
レプリケーションが完了する前にACTIVEサーバが停止すると
フェイルオーバ後に更新内容が反映されていない状態が発生

OpenLDAPでのレプリケーションの仕組み

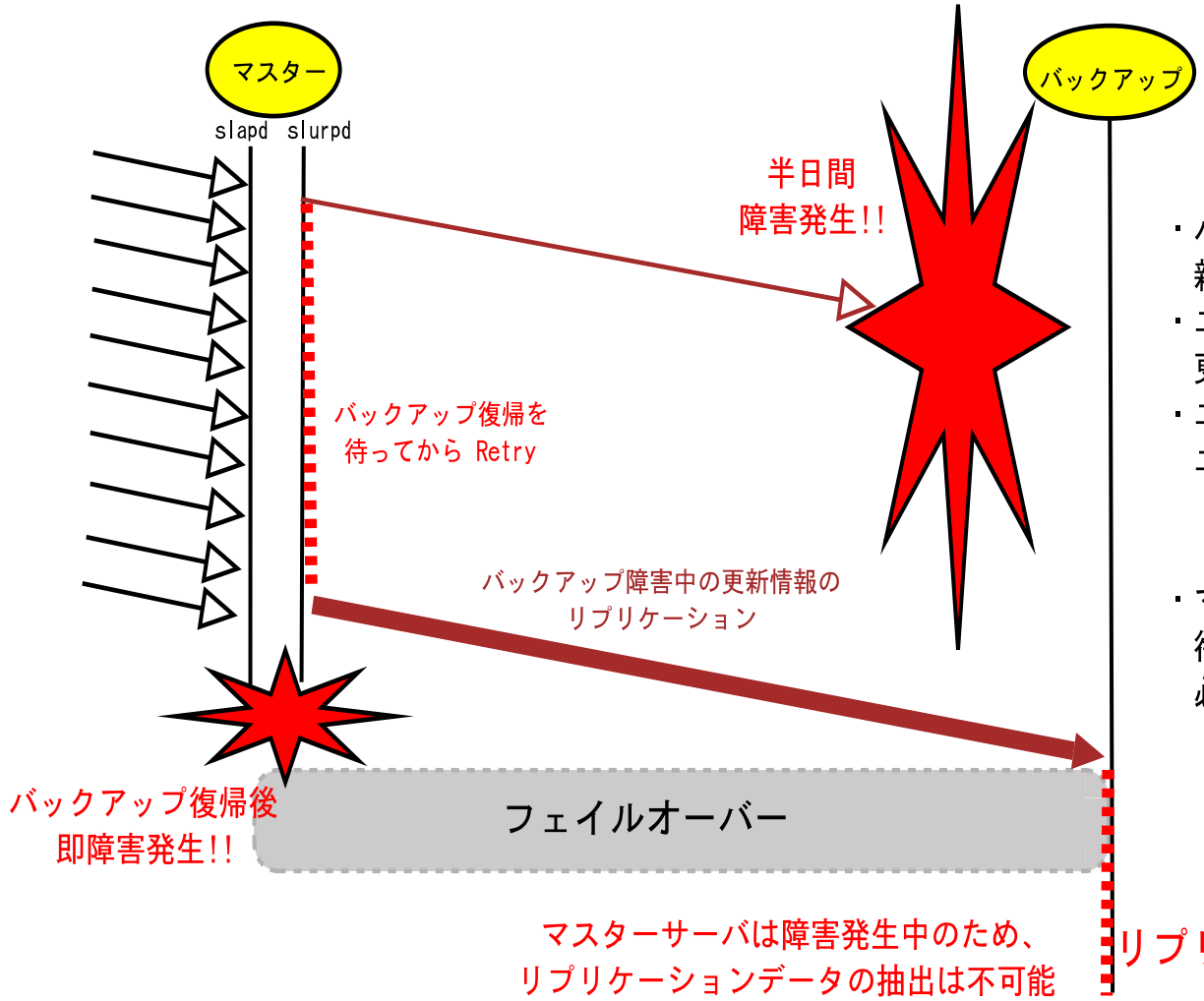


- slapd と slurpd の二つのデーモン プログラムによる実現
- 互いに非同期に動作
- slurpd を suspend(or kill) するとリプリケーションを中断できる。LDAP サービスは依然として提供可能
- 実際のデータベースとリプリケーション情報が複数ファイルに分割 -> 不整合がおこりえる

単純にフェイルオーバした場合



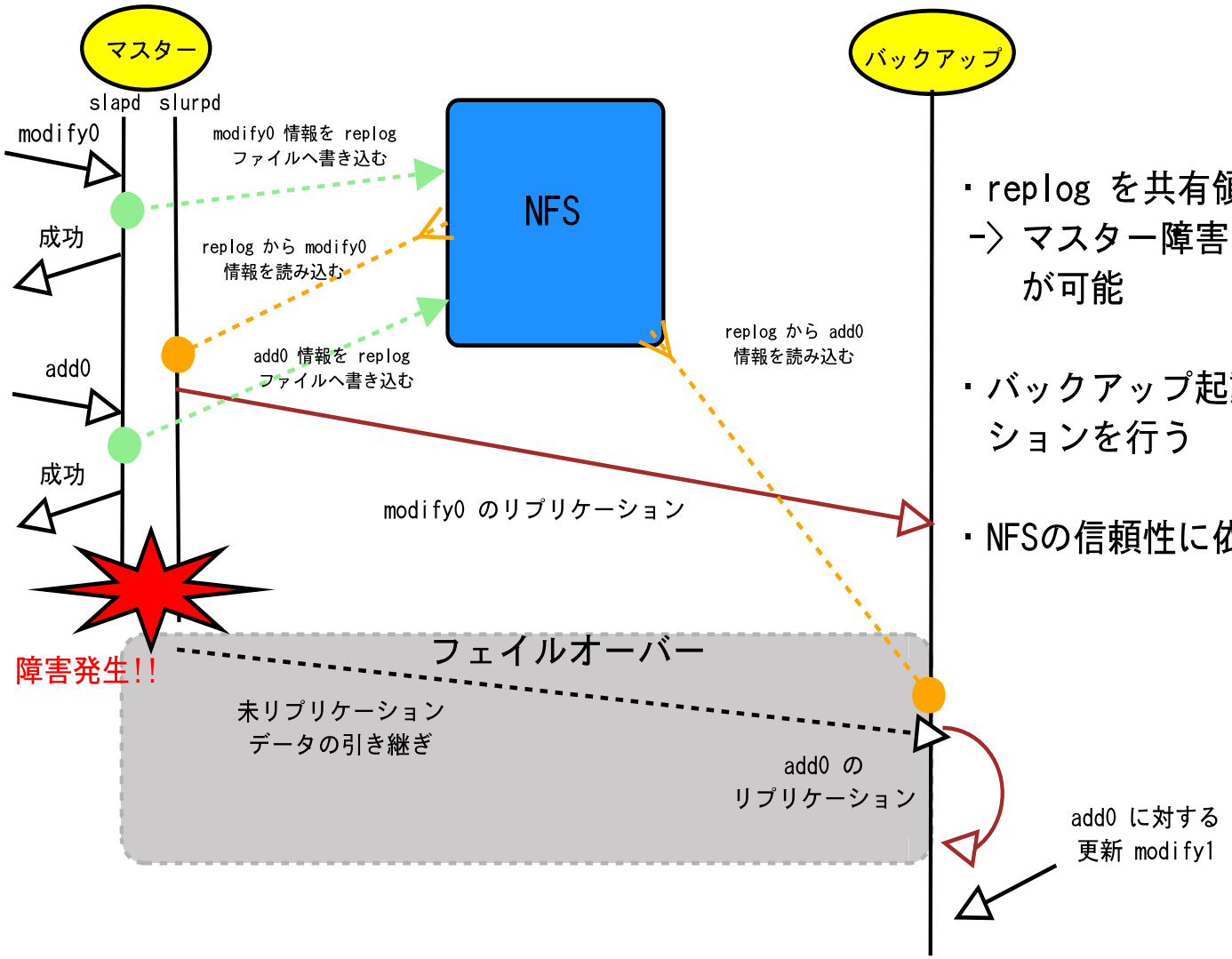
最悪のケース



- パスワードが変更されていた場合、
  新しいパスワードで認証できなくなる
- エントリが追加されていた場合、
  更新を行うと No such object になる
- エントリが削除されていた場合、
  エントリが突然復活する

- マスター復帰後の自然リプリケーションを
  待つとすると、マスターの早急の復旧が
  必要となる。

レプリケーションタイムラグの短縮



- replog を共有領域(NFS)に保存
  -> マスター障害時にファイル参照が可能
- バックアップ起動時にリプリケーションを行う
- NFSの信頼性に依存

システム障害対策 - その他

- バックアップ
- 障害解析用のログ
- 適切なシステム管理者

システム障害対策のまとめ

- 信頼性の高いハードウェア
- 高可用性環境の実現
- 障害発生時の冷静な対応

まとめ

- LDAPの概要、利点
- OpenLDAPによる環境構築
- メールサーバ環境の構築
- フェイルオーバシステムの構築

付録

関連リンク

- OpenLDAP - http://www.openldap.org/
- Berkeley DB - http://www.sleepycat.com/
- heartbeat - http://www.linux-ha.org/
- mon -
  - http://www.us.kernel.org/pub/software/admin/mon/
- postfix - http://www.postfix.org/
- qpopper - http://www.eudora.com/qpopper/
- pam-ldap - http://www.padl.com/OSS/pam_ldap.html
- 大規模メールシステム -
  - http://www.valinux.co.jp/technical/ldap/