OpenStack (essex) かじってみました

ENOG18

2012/12/21

株式会社　創風システム
田畑　紀彦

始まりはいつもの無茶振り？

- 今年の春先に…
  - うちもクラウド環境欲しいよねー
  - OpenStack とかってどうなのよ？
  - ちょっと試してみないー？(強制)

OpenStack のクラウド環境って #1

- 詳細はこちらのサイトをご覧いただくとして

日本OpenStackユーザ会
http://openstack.jp/frontpage.html
本家プロジェクト
http://www.openstack.org/

OpenStack のクラウド環境って #2

- 機能の構成
  - OpenStack Identity (Keystone)
  - OpenStack Object Storage (Swift)
  - OpenStack Compute (Nova)
  - OpenStack Image Service (Glance)
  - OpenStack Dashboard (Horizon)

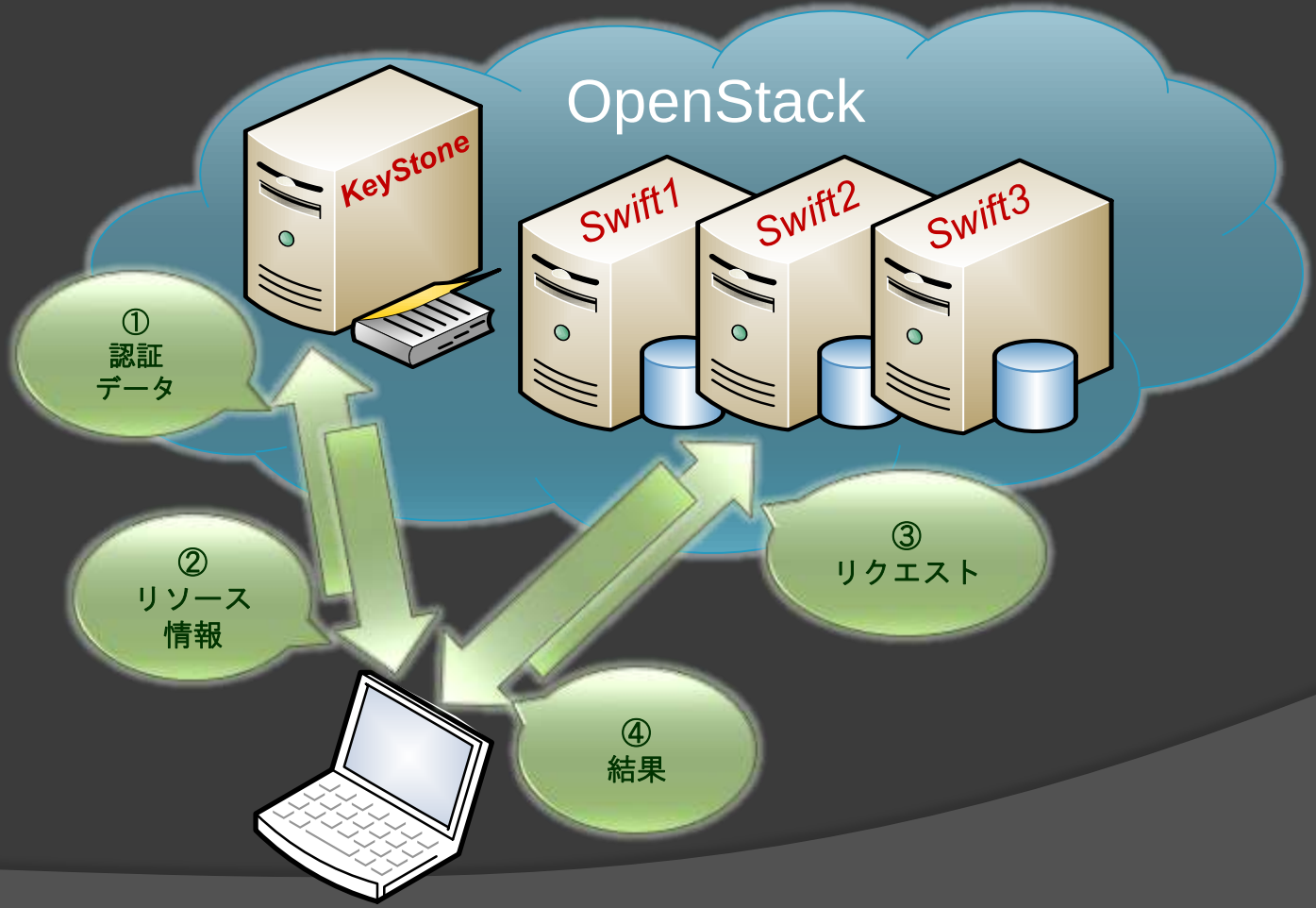
※今回は、諸般の事情により上2つだけ触ってみました

OpenStack のお作法？

- 今回の検証範囲では…
  - RESTを使ってアクセスや制御を行う
  - リソースにアクセスするにはKeystone経由で
  - データ交換には、JSONまたはXMLを使用
  - Amazon S3互換APIも実装されている※

OpenStack のお作法？

- 目的のリソースにアクセスするまでの流れ

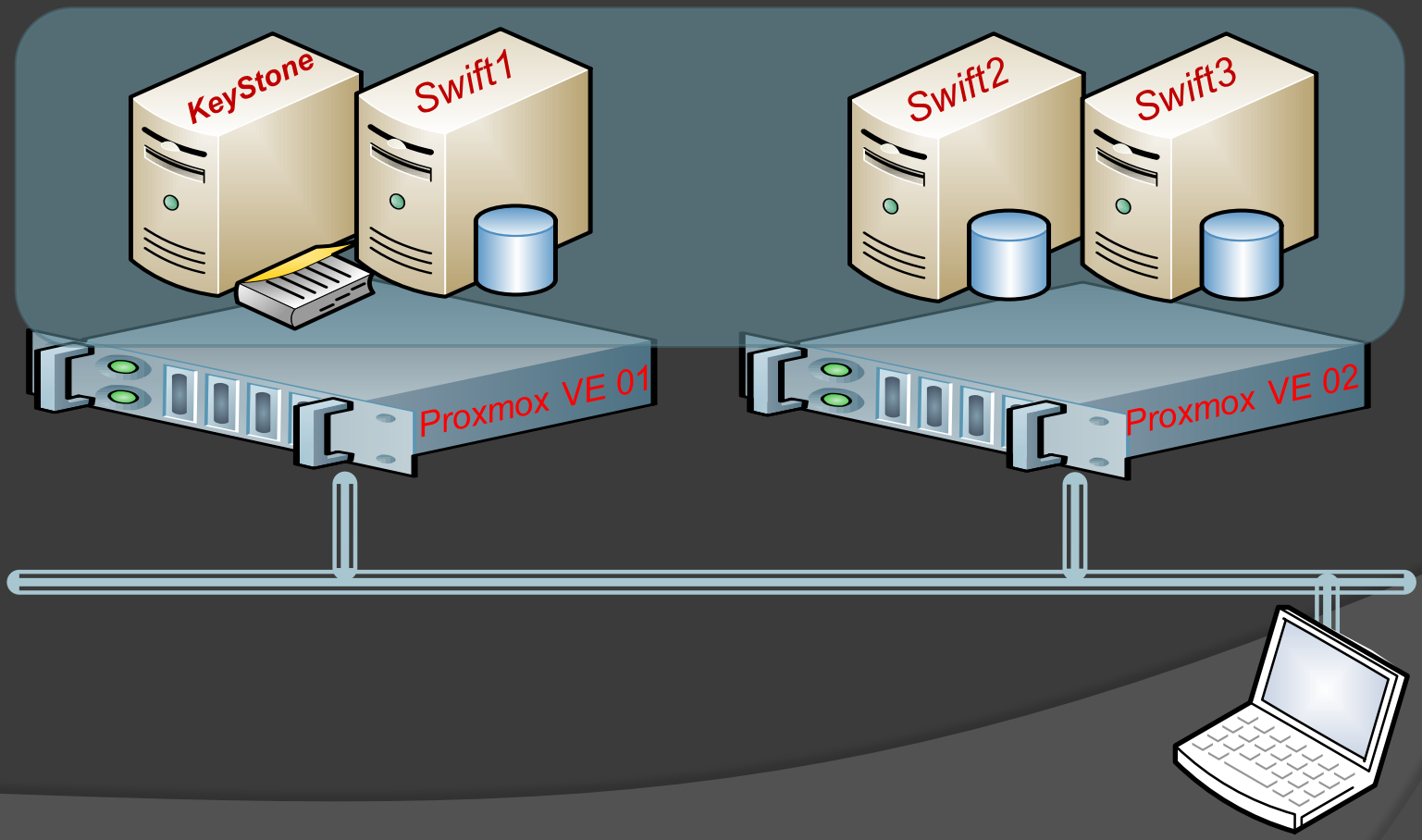


早速検証環境の構築(を強要w)

- 実験機(~~余ってるがらくた~~)を用意
- インストールと調整を某氏に丸投げ
- 数日待つと…

検証環境ができました！

- 大体こんな感じだそうです



さっそく触ってみたものの…

- CLIツールからのアクセスは普通にOK！
- …あれ？クライアントからつながらない？？
- 対応するクライアントが見つからない！

検証時点の結論

- 既存のアプリケーションではほぼ未対応
- OpenStack のCUIツールしか検証とれてない
- CUIツールと連携取るのが面倒
- 今回はSWIFTにアクセスできればOKだけど…

…それっぽいのを作るしかないか orz

SWIFT の操作コマンド

- 概ね以下の機能に集約される？
  - バケットの作成
  - オブジェクトのアップロード／ダウンロード
  - バケット／ファイルの一覧
  - オブジェクトの削除(バケットも)
  - メタデータの登録／削除

検証コード作成の基本ポリシー

- Webアプリに取り込みやすい事
  - 実装言語はPHPを採用
  - 使いやすい様にクラスライブラリ化する

検証コード作成の基本ポリシー

- 実装する機能は極力シンプルに
  - リクエスト機能はGET/PUT/DELETEのみ
  - https使用時になんちゃって証明書を許容
  - もちろんエラートラップも最小限

検証コード作成の基本ポリシー

- ある程度大きなファイルも扱いたい
  - ファイルを指定サイズに分割転送
  - 分割されたファイルを結合して取得

こんなのできました！

- クラスライブラリの公開関数
  - constructor(URI, TenantName, UserName, Password)
  - getTokenInfo()
  - getBucketList()
  - getObjectList(bucketName = "")
  - putObject(sPath = "", objectName = "", dSize = -1)
  - getObjectHeader(objectName = "")
  - getObject(objectName = "", dPath = "")
  - deleteObject(objectName = "")

こんなのできました！

- constructor(URI, TenantName, UserName, Password)

  クラスの実体化を行う際に認証用の情報をセットします。

  - URI　　：Keystoneのアドレスを記入します。
  - TenantName　：テナント名を指定します。
  - UserName　　：ユーザ名を指定します。
  - Password　　：パスワードを指定します。

こんなのできました！

- getTokenInfo()

  認証情報を元に、リソースへのアクセスに使う情報を取得します。

こんなのできました！

- getBucketList()

  登録済みのバケットの一覧をテキスト形式で返却します。
  ※後述のgetObjectList()でも代替可能

こんなのできました！

- getObjectList(bucketName)

  バケット内に格納されているオブジェクトの一覧をテキスト形式で返却します。

こんなのできました！

- putObject(sPath, objectName, dSize)

  ソースパスに指定したファイルをバケット名を含むフルパスで指定したオブジェクトとしてアップロードします。

  - sPath　　：ソースパスを指定します。
  - objectName　：オブジェクト名を指定します。
  - dSize　　：ファイルの分割サイズを指定します。

こんなのできました！

- getObject(objectName, dPath)

  バケット名を含むフルパスで指定したオブジェクトの内容を保存先パスに出力します。

  - objectName　：オブジェクト名を指定します。
  - dPath　　：保存先パスを指定します。

こんなのできました！

- deleteObject(objectName)

  バケット名を含むフルパスで指定したオブジェクトを削除します。バケット自身の削除にも使用します。

  - objectName　：オブジェクト名を指定します。

動かしてみました！　#1/2

- Token情報を取得してみます

```
<?php
$_Debug = true;
// 設定ファイルの取り込み
require_once( './config/config.php');
// ライブラリの取り込み
require_once( './lib/ks20lib.php');
require_once( './lib/fs.php');
```

クラスを実体化しています
KeystoneのURL
テナント名
ユーザIDとパスワード

```
$kstn20 = new keystone20( $_Config[“KeystoneURL"],
                          $_Config["TenantName"],
                          $_Config["UID"], $_Config["PWD"]);

print_r( $kstn20->getTokenInfo());
```

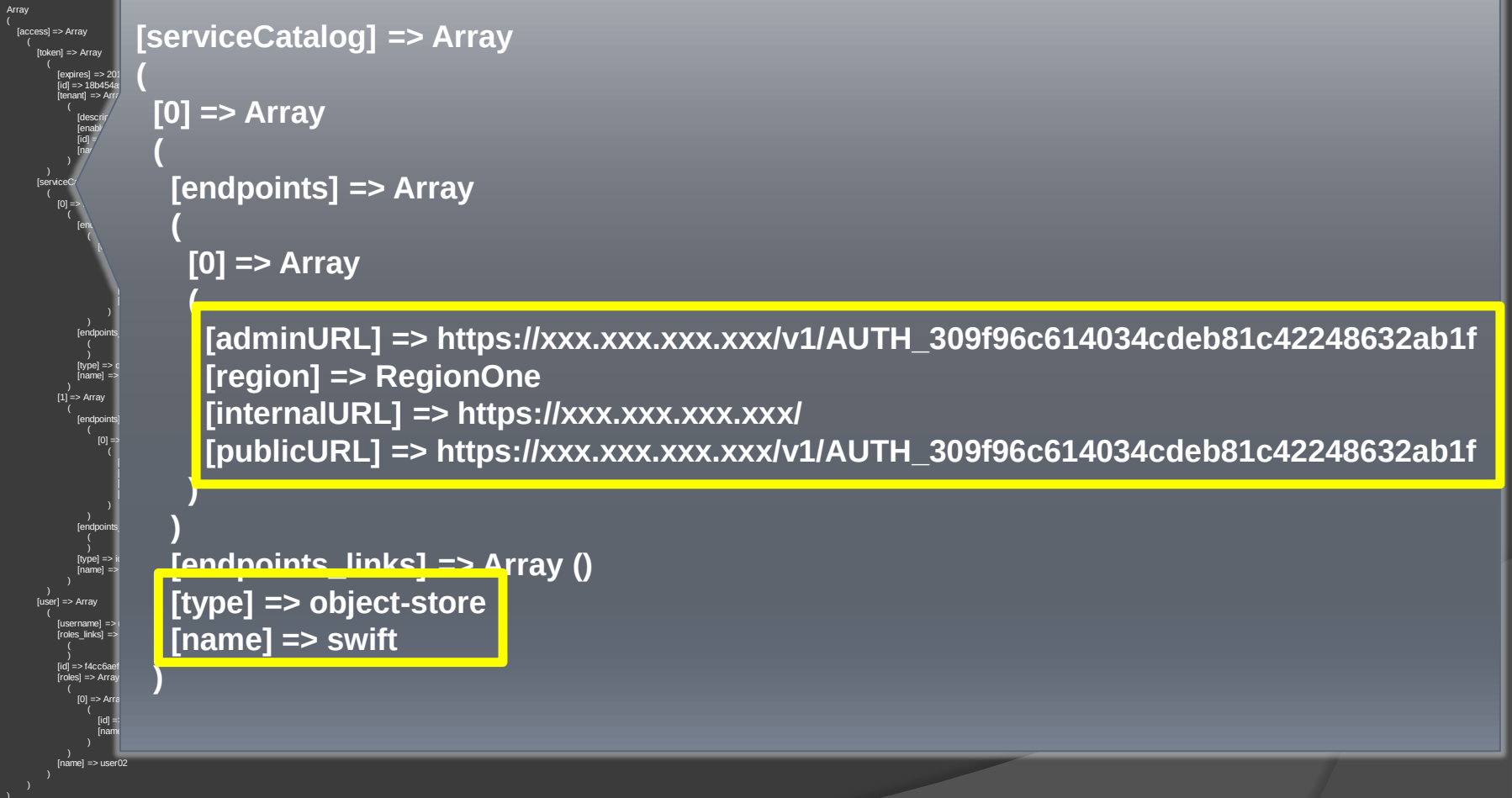
動かしてみました！　#1/2

- Token情報を取得してみます

```
<?php
$_Debug = true;
// 設定ファイルの取り込み
require_once( './config/config.php');
// ライブラリの取り込み
require_once( './lib/ks20lib.php');
require_once( './lib/fs.php');

$kstn20 = new keystone20( $_Config[“KeystoneURL"],
                          $_Config["TenantName"],
                          $_Config["UID"], $_Config["PWD"]);

print_r( $kstn20->getTokenInfo());
```

リソース情報を取得して
表示します。

動かしてみました！　#1/2



```
[serviceCatalog] => Array
(
  [0] => Array
  (
    [endpoints] => Array
    (
      [0] => Array
      (
        [adminURL] => https://xxx.xxx.xxx.xxx/v1/AUTH_309f96c614034cdeb81c42248632ab1f
        [region] => RegionOne
        [internalURL] => https://xxx.xxx.xxx.xxx/
        [publicURL] => https://xxx.xxx.xxx.xxx/v1/AUTH_309f96c614034cdeb81c42248632ab1f
      )
    )
    [endpoints_links] => Array ()
    [type] => object-store
    [name] => swift
  )
```

動かしてみました！　#2/2

- オブジェクトをUPLOADして、リスト表示します

```
<?php
                :
                :
// Bucketを作成する
$kstn20->putObject( "", "/bucket1");

// Bucketの中身を確認する
echo $kstn20->getObjectList( "/bucket1");

// 作成したBucketにファイルをPUTする
$kstn20->putObject( "./testFile.dat", "/bucket1/testFile.dat");

// Bucketの中身を確認する
echo $kstn20->getObjectList( "/bucket1");
```

バケットを作ります

動かしてみました！　#2/2

- オブジェクトをUPLOADして、リスト表示します

```
<?php
                :
                :
// Bucketを作成する
$kstn20->putObje

// Bucketの中身を確認する
echo $kstn20->getObjectList( "/bucket1");

// 作成したBucketにファイルをPUTする
$kstn20->putObject( "./testFile.dat", "/bucket1/testFile.dat");

// Bucketの中身を確認する
echo $kstn20->getObjectList( "/bucket1");
```

バケットは空っぽなので
何も表示されません

動かしてみました！　#2/2

- オブジェクトをUPLOADして、リスト表示します

```
<?php
                :
                :
// Bucketを作成する
$kstn20->putObject( "", "

// Bucketの中身を確認する
echo $kstn20->getObject

// 作成したBucketにファイルをPUTする
$kstn20->putObject( "./testFile.dat", "/bucket1/testFile.dat");

// Bucketの中身を確認する
echo $kstn20->getObjectList( "/bucket1");
```

ファイルをPUTします
十秒程度で約120MB≒80Mbps位？

※内部で100MB単位に分割します

動かしてみました！　#2/2

- オブジェクトをUPLOADして、リスト表示します

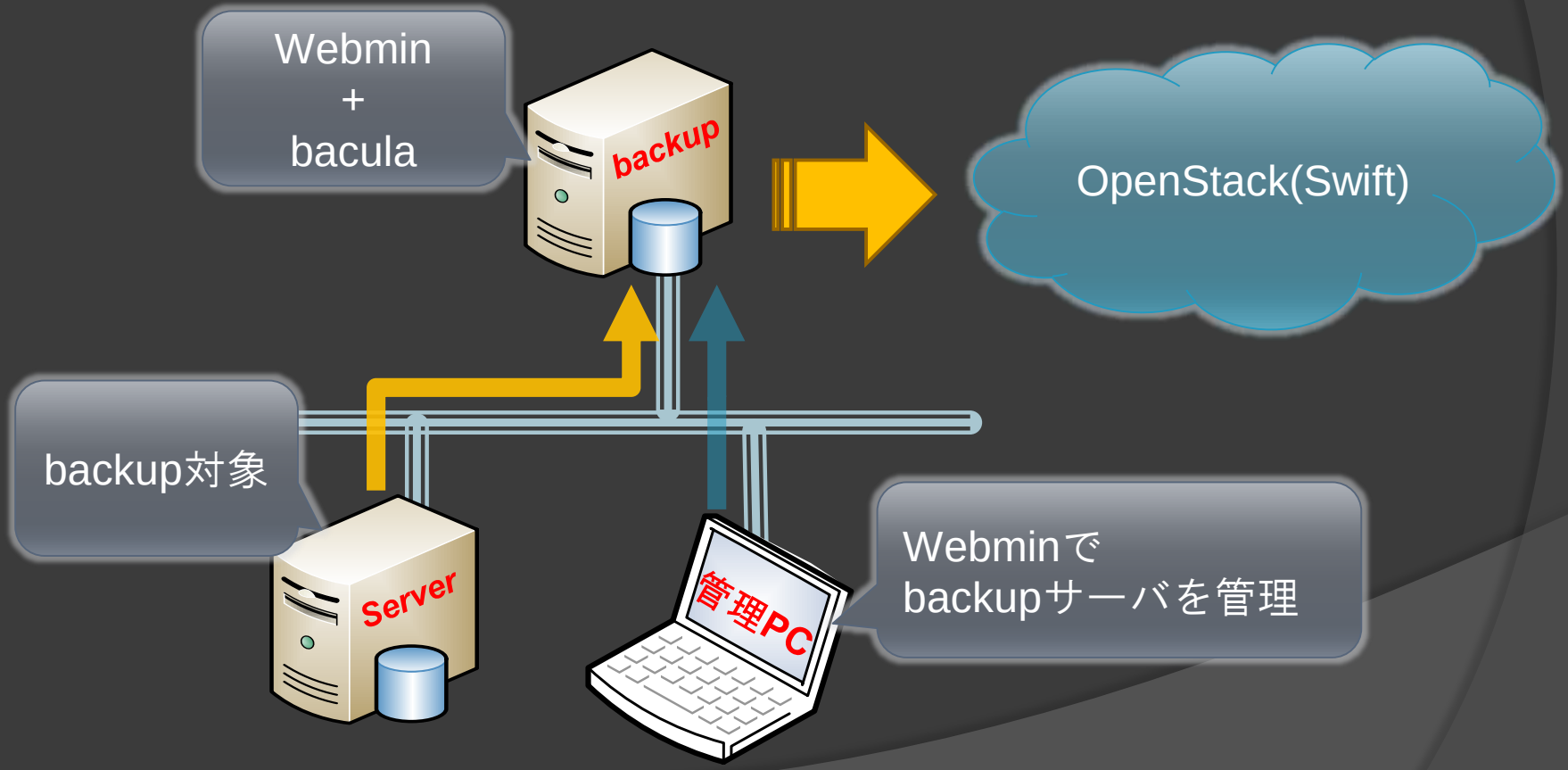
```
<?php
                :
                :
// Bucketを作成する
$kstn20->putObject( "/bucket1");

// Bucketの中身を確認する
echo $kstn20->ge

// 作成したBucketにファイル
$kstn20->putObj                                    at");

// Bucketの中身を確認する
echo $kstn20->getObjectList( "/bucket1");
```

以下が表示されます

testFile.dat
testFile.dat/1
testFile.dat/2

組み込んでみました！

- 手持ちの実験機でバックアップデータを同期



組み込んでみました！

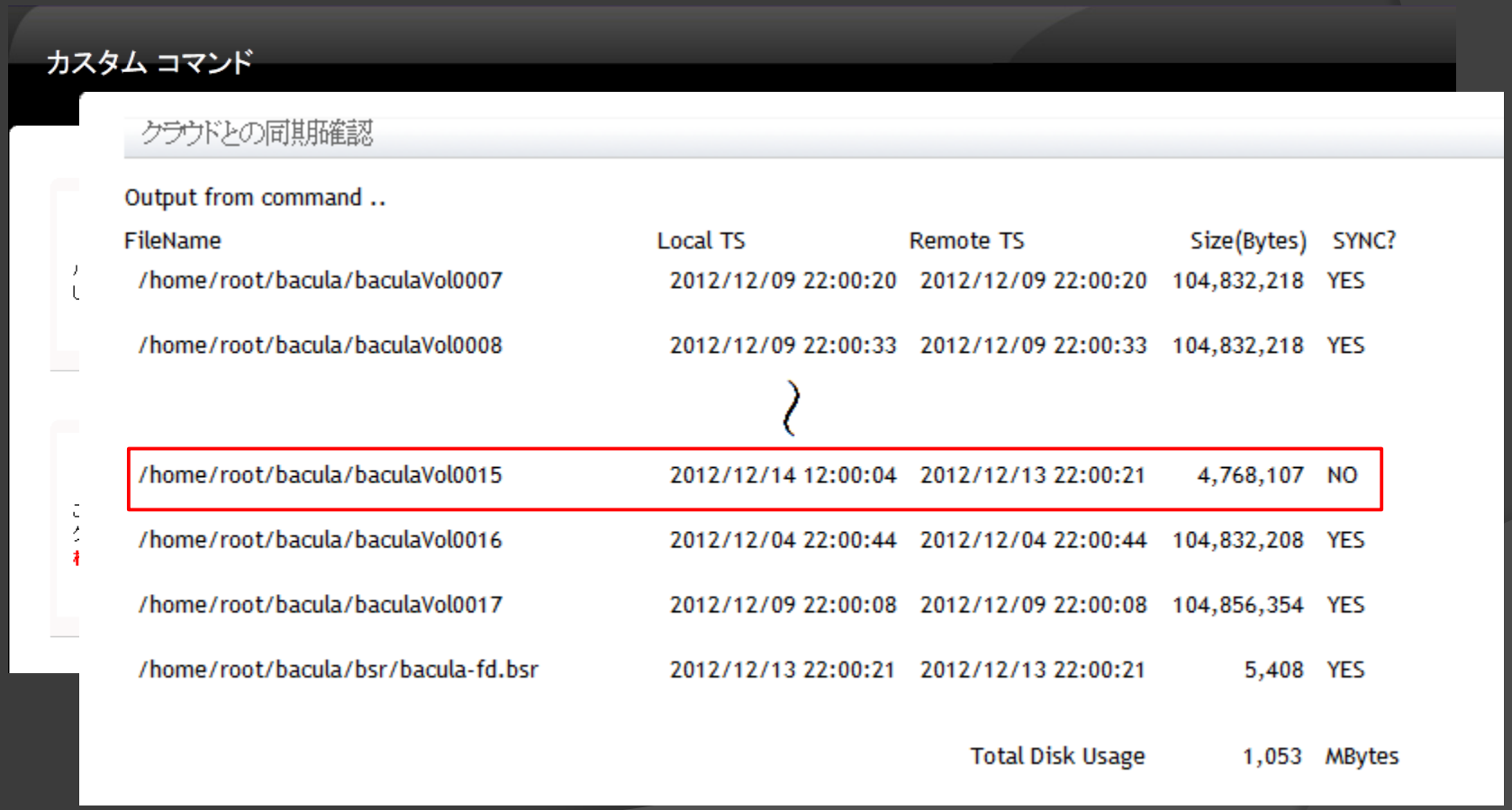
- Bacula はこんな感じの画面です

組み込んでみました！

- 手持ちの実験機でバックアップデータを同期

組み込んでみました！

- 手持ちの実験機でバックアップデータを同期



まとめ

- 評価時点は2012年5～7月頃
- Keystone&Swiftはきちんと動いた
- クライアントアプリの対応状況に難あり？
- もう次のバージョン(folsom)がリリース済み

ご清聴ありがとうございました。