High Availability iRODS System (HAIRS)

Yutaka Kawai, KEK

Adil Hasan, ULiv
Outline

- Introduction
- iRODS HA system with Director
- Large File Transfer
- Speed Performance
- Summary
Introduction

- Replication enables high availability (HA) system for catalog service
  - Replicate by back-end, i.e. iRODS
  - Replicate by front-end;
    - i.e. AMGA (ARDA[1] Metadata Grid Application)
      - Metadata Catalogue of EGEE’s gLite 3.1 Middleware
      - Back-end: Oracle, PostgreSQL, MySQL, SQLite
      - http://amga.web.cern.ch/amga/

- The current iRODS HA is implemented by replicating ICAT DB with PgPool tool [2]
  - A problem when iRODS server fails
  - Solve the problem by using Director
The Current iRODS HA

- ICAT DB replication by Pgpool

Change the server info in .irodEnv
Problem of the current HA

- Even if the iRODS server fails, clients still continue to access the same server without noticing the failure.

```
Need to change server info in .irodEnv
```

March 25th, 2010  iRODS User Meeting @ UNC in Chapel Hill, USA  -- Yutaka (KEK), Adil (ULiv)
Solution by using Director

- Place a Director between Client and Server
  - Monitor the iRODS server statuses
  - Load balance to the iRODS servers

March 25th, 2010
iRODS User Meeting @ UNC in Chapel Hill, USA  -- Yutaka (KEK), Adil (ULiv)
How to Implement Director?

- UltraMonkey[^3]
  - Linux based director
  - Low cost but not so high speed
  - Need some steps to setup

- Hardware Director
  - High cost and high speed
  - Easy to setup (?)
  - Cisco, HP, etc.
UltraMonkey

- UltraMonkey consists of 3 components
  - Linux Virtual Server (LVS) : Load balancing
  - ldirectord : Monitoring real servers
  - Linux–HA (LHA) : Monitoring directors

- LVS and ldirectord are used here
  - LVS : Provide Virtual IP for load balance
  - ldirectord : Monitoring iRODS service
  - LHA : Future use for director redundancy
Virtual IP for load balance

iRODS Client

.100
192.168.1.0/24

VIP 192.168.1.200

Linux Director

.240
192.168.2.0/24

iRODS Real Servers

.101
.102

iRODS Client can specify only this VIP in .irodsEnv

Gateway of Real Servers is Director
Monitoring iRODS service

- Idirector monitors iRODS real servers
  - Polling server status via iRODS control port

```
<MsgHeader_PI>
  <type>RODS_VERSION</type>
  <msgLen>182</msgLen>
  <errorLen>0</errorLen>
  <bsLen>0</bsLen>
  <intInfo>0</intInfo>
</MsgHeader_PI>

<Version_PI>
  <status>-4000</status>
  <relVersion>rods2.1</relVersion>
  <apiVersion>d</apiVersion>
  <reconnPort>0</reconnPort>
  <reconnAddr></reconnAddr>
  <cookie>0</cookie>
</Version_PI>
```
Outline

- Introduction
- iRODS HA system with Director
- Large File Transfer
- Speed Performance
- Summary
Large File Transfer

- iRODS uses parallel ports to transfer a large file.
  - Smaller than 32MB file is transferred through iRODS control port #1247.

- iRODS catalog server directs a server to open parallel ports to transfer a large file
  - iRODS clients can directly connect with the server through the parallel ports.
Process of Large File Transfer

▸ Steps to transfer a large file in iRODS

1. iRODS Client
   - iput a large file

2. iRODS Server
   - Find physical location to store

3. iRODS Server w/o ICAT
   - Start service for Parallel I/O

4. iRODS Server w/o ICAT
   - Physical Data
   - PostgreSQL
   - File Transfer via Parallel I/O
Large File Transfer w/ Director

- Need to confirm whether Director interferes in transferring a large file or not

- The physical storage should be located out of the local network of iRODS real servers
  - Director handles only iRODS catalog server IP
  - Director cannot manage all of the parallel ports
Process using Director

- Works as same as normal case
  - Only one additional step between (1) and (2)
Outline

- Introduction
- iRODS HA system with Director
- Large File Transfer
- Speed Performance
- Summary
Speed Performance

▸ Test Program
  ▹ concurrent-test in iRODS package
  ▹ iput, imeta, iget, imv
  ▹ 1000 entries
  ▹ Servers are VMs (Xen) on same physical machine
    ▪ Client is located on the different machine

▸ Test Case
  ▹ Case1: Normal case.
    ▪ Client directly accesses one iRODS server.
  ▹ Case2: Using a director.
    ▪ Client accesses one iRODS server via Director.
  ▹ Case3: Load sharing case.
    ▪ Client accesses two iRODS servers via Director.
Speed Performance (cont’d)

- Using a Director (Case2)
  - About 10% slower than no Director (Case1)
  - Reasonable to consider tradeoff between speed and availability
Speed Performance (cont’d)

- Load sharing case (Case3)
  - About 5% slower than Case2
  - The concurrent-test is not suitable under such a Load balanced system.
  - Need a program using multi-clients or multi-threading methods.
Opinions in this study

▸ Network limitation
  ▹ Director works as NAT. Difficult to place iRODS catalog servers in different subnets.
  ▹ But the problem depends on NAT technology. We hope some NAT vender can implement extensions.

▸ Speed Performance
  ▹ The “concurrent-test” consumes overhead. The result 10% slow is in one of the worst cases. We may see less than 10% in actual uses.

▸ PostgreSQL only?
  ▹ How about other DB services? They have the same tools as PgPool?
  ▹ Back–end replication is enough? Front–end replication should be considered for iRODS?
Summary

▸ iRODS HA system
  ▸ The current approach using only PgPool
  ▸ The new approach using Director
  ▸ The new one can solve the current problem

▸ Large File Transfer
  ▸ iRODS large file transfer works well when using Director

▸ Speed Performance
  ▸ Director results in the speed performance of concurrent-test getting slower 10%

▸ Future works
  ▸ Apply this solution to other catalog services
References


Back up
Outline

- Introduction
- iRODS HA system with Director
- Large File Transfer
- Speed Performance
- Future works (apply to RNS application)
- Summary
Outline

- Introduction
- iRODS HA system with Director
- Large File Transfer
- Speed Performance
- Future works (apply to RNS application)
- Summary
Outline

- Introduction
- iRODS HA system with Director
- Large File Transfer
- Speed Performance
- Future works (apply to RNS application)
- Summary
Outline

» Introduction
» iRODS HA system with Director
» Large File Transfer
» Speed Performance
» Future works (apply to RNS application)
» Summary
What is RNS?

- RNS: Resource Namespace Service
  - RNS offers a simple standard way of mapping names to endpoints within a grid or distributed network [4]
  - The latest version is available here; https://forge.gridforum.org/sf/go/doc8272

- Java based RNS application is being developed by Osaka University and Tsukuba University
  - This application is similar to iRODS
  - The other kind of RNS application is Grid Shell of Genesis II by The Virginia Center for Grid Research (VCGR) [5].
Apply to RNS application??

- Derby can do replication?
  - No load-sharing in the above example
Issues in RNS application

- Several issues to be solved
  - Derby is not enough to work replication as same as using PostgreSQL w/Pgpool
  - Need some developments to replace Derby by PostgreSQL
  - The catalog implementation in the current RNS application has specific IP addresses