iRODS
Advanced Features

Michael Wan

mwan@diceresarch.org

http://irods.org/

iRods advanced features

- Data Transfer modes
- Structured file implementation
- iRods FUSE implementation

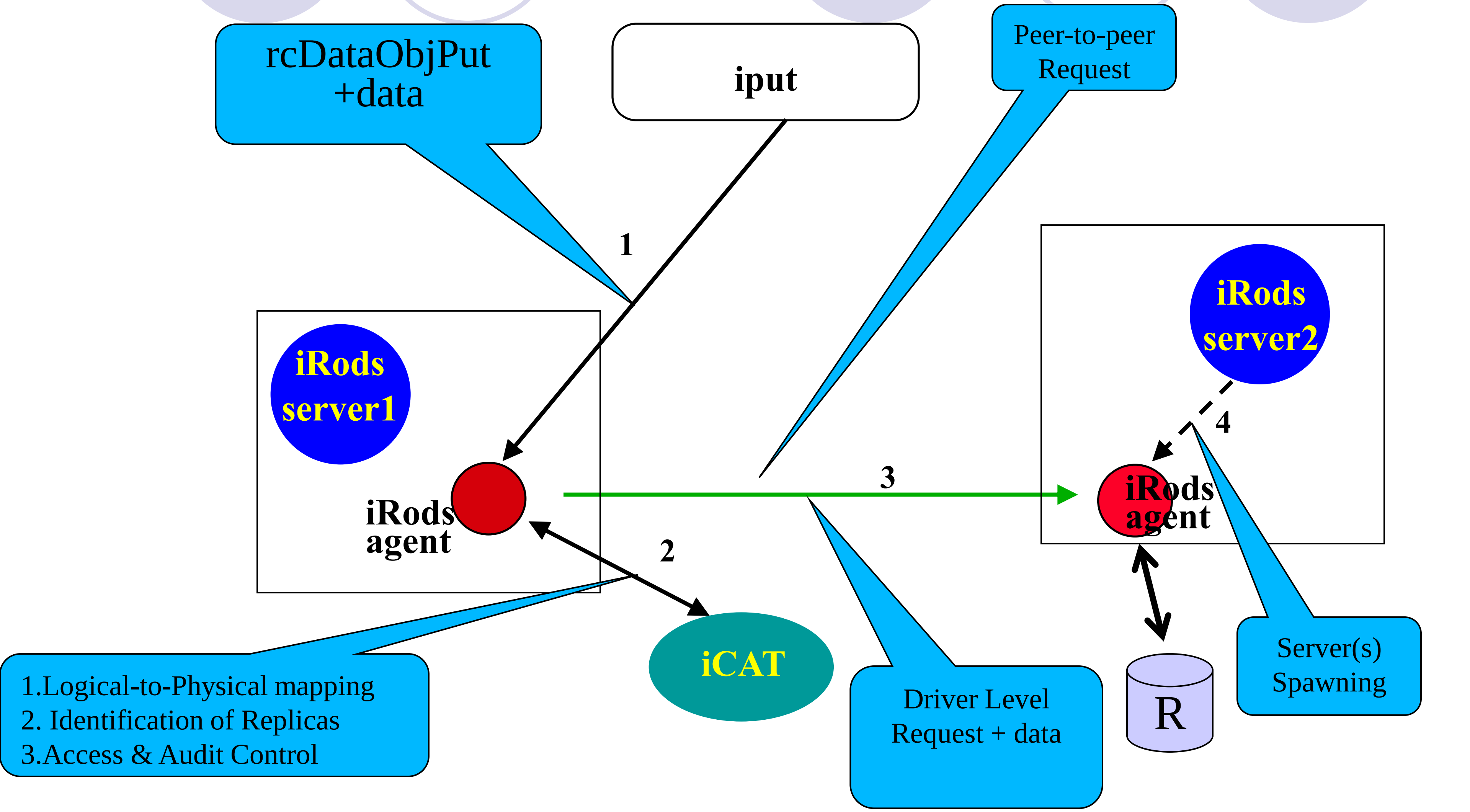
Data Transfer

- Three modes
  - Sequential
    - file size <= 32 MB (MAX_SZ_FOR_SINGLE_BUF in rodsdef.h)
    - Single request packet – request + data
    - Data transfer could require 2 hops
  - Parallel
    - Use multi-threads for data transfer
    - Client initiates multiple connections to server
    - Single hop for data transfer
    - Supported by all types of data transfer
      - Client/server – put, get
      - Server/server – copy, replicate, phymove, etc
    - Sequential or parallel is automatic
    - Tuning - msiSetNumThreads(sizePerThrInMb, maxNumThr, windowSize)
      - numThr = fileSize/sizePerThrInMb + 1
    - Iput –N numThr

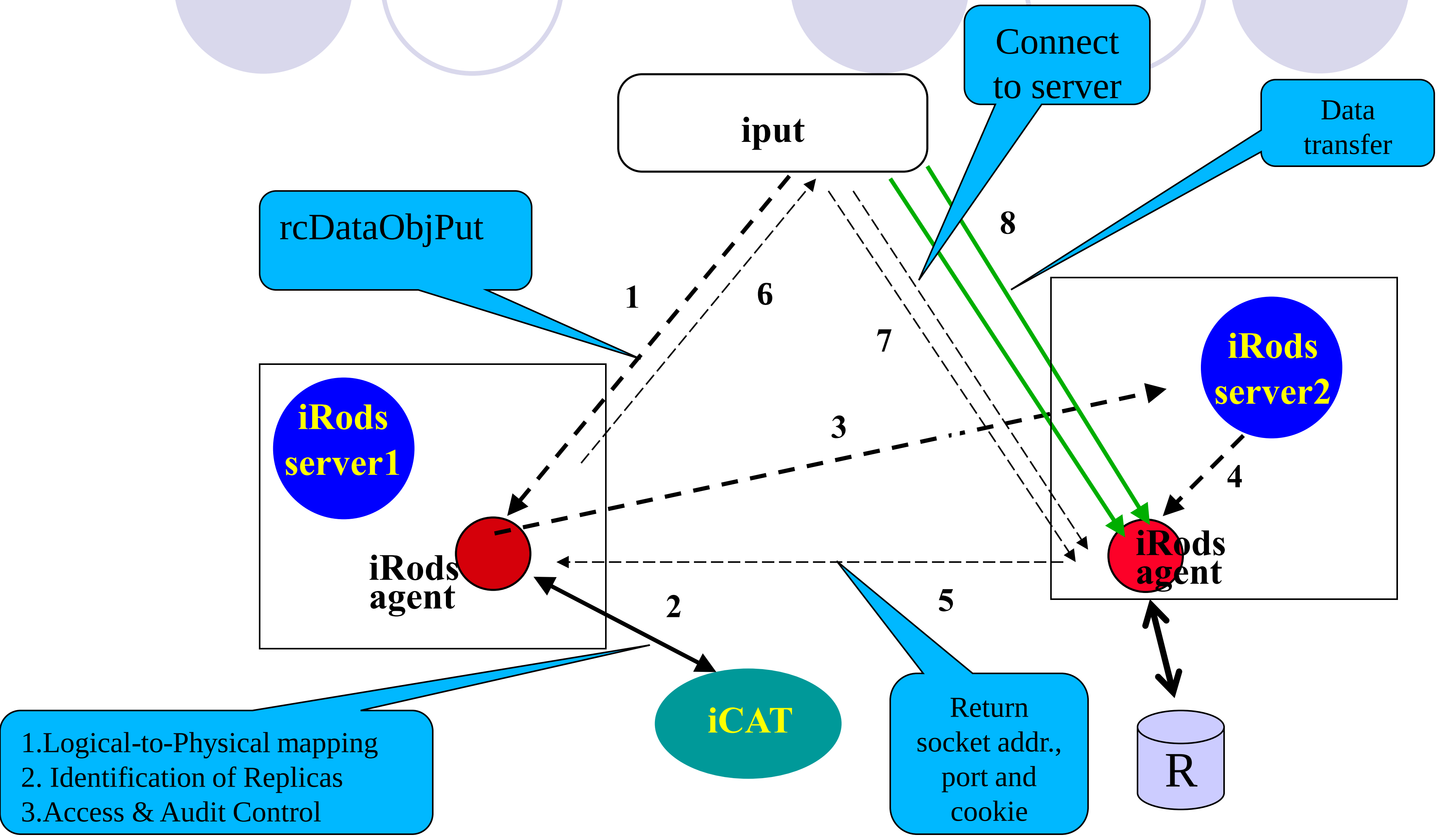
RBUDP Data Transfer

- RBUDP - Reliable Blast UDP
  - Developed by Eric He, Jason Leigh, Oliver Yu and Thomas Defanti of U of Ill at Chicago
  - Use UDP protocol
  - iput –Q
  - Sender sends (blasts) out data at a predetermined rate (600,000 kbits/s).
  - Env variable rbudpSendRate – change default rate
  - Each packet has a sequence number
  - At end of each transfer, receiver sends a bit map of packets it has not receivied
  - Sender sends the missing packets.
  - Env variable budpPackSize – change default packet size (8192 bytes)
  - Use memory mapped file for I/O
  - For robust network, 10-20% improvement

Data transfer – sequential mode



Data Transfer – Parallel or RBUDP modes



Structured Files

- Structured files
  - Files that have their own internal structures
    - Tar, winZip, other archival packages
    - iRods uses these structured files to package and archive data
    - Supports tar files only. More may be coming
      - HAAW files – UK's Hasan and Weiss
- Two usages
  - Data Bundle –ibun command
  - Mounted collections – imcoll command

Data Bundle

- Aggregate a large number of small files into a single self contained structured file
- More efficient to transfer
- More efficient to archive – tape
- ibun command

Data Bundle

- Upload and unbundle a tar file
  - tar -chf testdir.tar -C testdir .
  - iput -vDtar testdir.tar tardir
    - Put the tar in the tardir collection
    - Forget to use –Dtar, isysmeta to change dataType
  - ibun -x tardir/testdir.tar testdir
  - ils -lr testdir
- Bundle an iRods collection into a tar file
  - ibun -cDtar tardir/testdir1.tar testdir
  - iget –v tardir/testdir1.tar
- The tar file and the sub-files resources must be on the same host.

Mounted Collection

- A framework for associating a structured dataset on the server to a collection
- The entire dataset can then be access through this collection using iRods APIs and iCommands
- Individual files and sub-collections are not registered
  - Low overhead
  - No user defined metadata
  - No support for replication
- Current implementation
  - UNIX directory
    - Mount a UNIX directory on a server to a collection
    - All files and subdirectories in this UNIX directory now appears as if they are iRods files and sub-collections
  - Tar structured files
    - Mount a tar file to a collection
    - All files and subdirectories in this tar file now appears as if they are iRods files and sub-collections
  - Easy to add other types of structured files by adding ~20 functions to the structured file driver

Mounted Collection

- Mount a UNIX file directory:
  - imkdir mymount
  - imcoll -m f –R disk1 /tmp/myDir /workshop/home/mwan/mymount
  - ils –Lr mymount
  - icd mymount
  - iput/iget
  - imcoll –U /workshop/home/mwan/mymount
- Mount a tar file
  - imkdir mymount1
  - imcoll –m tar /workshop/home/mwan/tardir/testdir.tar /workshop/home/mwan/mymount1
  - ils –lr mymount1
  - imcoll –U /workshop/home/mwan/mymount1

iRods FUSE

- FUSE
  - Free UNIX kernel implementation
  - Allows users to implement their own file system in User Space
- iRods FUSE
  - Allow normal users to mount their iRods collection to a location directory
  - Access iRods data using normal UNIX commands and system calls
    - Unix command - cp, cat, vi, etc
    - Unix system calls – creat, open, read, write, etc
    - Other I/O library calls should also work.
  - Access control determined by the permission of the Unix mount point

iRods FUSE

- Performance issues
  - UNIX commands and applications make many “stat” calls, same files many times
  - Small read/write calls, less that 10 KB
  - A simple command such as ls, cp can make 30-60 irods calls.
  - iRods 2.0
    - File “stat” cached in memory hash queue. Stale after 10 min
    - Small files (< 1 MB) cached in /tmp/fuseCache
    - env variable "FuseCacheDir“ - change the default cache directory.
  - Much improved, usable

iRods Fuse Example

- Build iRods with Fuse
  - See configure instruction in README in clients/fuse
  - build
    - cd clients/fuse
    - make
- To mount a iRods collection
    - cd clients/fuse/bin
    - iinit
    - icd /tempZone/home/myUser/myCollection
    - mkdir ~/fuseMnt
    - ./irodsFs ~/fuseMnt
- To access iRods files
    - cd ~/fuseMnt
    - ls   should see all files in the /tempZone/home/myUser/myCollection
    - cat, vi of any files should work.

More Information

Michael Wan

mwan@dicerearch.org

http://irods.sdsc.edu