iRODS in RADII: Resource Aware Data-centric Collaboration Infrastructure

Tahsin Kabir (tahsin@cs.unc.edu)
Research Assistant at RENCI
iRODS in RADII

• Brief about RADII project
• Why are we using iRODS in RADII?
• Demo of RADII data management policy to iRODS rule and their execution
• Resource aware experiments on iRODS
RADII

• RADII: Resource Aware Data-centric Collaboration Infrastructure
  • Middleware to facilitate data-driven collaborations for domain researchers and a commodity to the science community
  • PI: I. Baldin, Co-PIs: C. Castillo, C. Schmitt, A. Rajasekar
  • Students Involved: Tahsin Kabir, Fan Jiang
  • Reducing the large gap between procuring the required infrastructure and manage data transfers efficiently
    • Integration of data-grid (iRODS) and NIaaS (ORCA) technologies on ExoGENI infrastructure
    • Novel tools to map data processes, computations, storage and organization entities onto infrastructure with intuitive GUI based application
    • Novel data-centric resource management mechanisms for provisioning and de-provisioning resources dynamically throughout the lifecycle of collaborations
Figure: High level diagram of RADII’s capabilities
Why iRODS in RADII?

- RADII Policies to iRODS Rule Language
  - Easy to map policies to iRODS Dynamic PEP
  - Reduced complexity for RADII
- Distributed and Elastic Data Grid
- Resource Monitoring Framework
- Geo-aware Resource hierarchy creation via composable iRODS
- Metadata tagging
RADII Data Management Policy

{ "tag": "red",  
  "create": [ 
    { "user": "rods", "site": "SL" }, 
    { "user": "bob", "site": "UNC"}  
  ], 
  "retrieve": [ 
    { "user": "adam"}  
  ], 
  "update": [ 
    { "user": "bob" }, 
    { "user": "rods" }  
  ], 
  "delete": [ 
    { "user": "rods" }  
  ] 
}

- A file created with the tag **red**
  - if the user is *rods*, store it in site SL
  - if the user is *bob*, store it in site UNC

- Files tagged **red** can only be
  - read by user *adam*
  - updated by user *bob* and *rods*
  - deleted by user *rods*
RADII Data Management
Policy to iRODS Rule

```json
{  "tag": "red",
  "create": [
    {  "user": "rods",
        "site": "SL" },
    {  "user": "bob",
        "site": "UNC"}
  ],
  "retrieve": [
    {  "user": "adam"
  }
  ],
  "update": [
    {  "user": "bob" },
    {  "user": "rods" }
  ],
  "delete": [
    {  "user": "rods" }
  ]
}
```

```c
acRADIIGetOpType( *op_type ){
  *op_type = "null";
  msiGetValByKey($KVPairs,"mode_kw",*mode_kw);
  msiGetValByKey($KVPairs,"flags_kw",*flags_kw);
  if( "*mode_kw" like "384" && "*flags_kw" like "578" ){
    *op_type = "update";
  } else if ( "*mode_kw" like "384" && "*flags_kw" like "0") {
    *op_type = "retrieve";
  }
}

acRADIIRetrieveUpdate{
  *meta = "null";
  acRADIIGetMeta(*meta);
  acRADIIGetOpType(*op_type);
  if( *op_type like "retrieve" ) {
    acRADIIRetrieve(*meta,*userNameClient);
  } else if ( *op_type like "update" ){
    acRADIUpdate(*meta,*userNameClient);
  } else {
    ... 
  }
}
```
Demo of rule injection and enforcement
Resource Awareness

- iRODS RMS provides node specific resource utilization
- End-to-End parameters such as throughput, current network flow is important for judicious placement, replication and retrieval decision
- Created end-to-end Throughput, Latency and instantaneous transfer RX/TX per second monitoring.
- The best server selection based on end-to-end utility value:

\[
Util_{AB} = Throughput_{AB} + CPU_B + Sto_B + NIn_B \\
+ NOut_B + TX_{AB} + RX_{AB} + Latency_{AB}
\]
Experiment Topology

Figure: Experimental Setup Topology
Experimental Setup

- The sites were: UCD, SL, UH, FIU
- Parallel and multithreaded file ingestion from each of the clients
- Total 400GB file ingestion from each client
- One copy at the edge node and another replication based on utile value.

<table>
<thead>
<tr>
<th>Title</th>
<th>File Size Range</th>
<th>Weight on Number of Files</th>
</tr>
</thead>
<tbody>
<tr>
<td>Very Small</td>
<td>1KB - 5MB</td>
<td>40</td>
</tr>
<tr>
<td>Small</td>
<td>5MB - 50MB</td>
<td>40</td>
</tr>
<tr>
<td>Medium</td>
<td>50MB - 500MB</td>
<td>20</td>
</tr>
<tr>
<td>Large</td>
<td>500MB - 5GB</td>
<td>20</td>
</tr>
</tbody>
</table>

Table 1: File Size Distribution Table
Edge Put and Remote Replication Time

Figure: Edge Node Put Time

Figure: Remote Replication Time
Combined Store Response Time

The chart shows the average time taken in seconds for different file size ranges. The x-axis represents file size ranges in KBytes, while the y-axis shows the average time taken in seconds. The chart compares two types of file size distributions: Util and Random.
Read Response Time

The diagram shows the average time taken in seconds for different file size ranges. The x-axis represents the file size range in KBytes, and the y-axis represents the average time taken. The data is grouped by two categories: Util. and Random. The bars for each category and size range provide a visual representation of the response time.
Questions
Figure: Average CPU usage on each of the sites for store experiment
Other Graphs: CPU for Read Exp.

Figure: Average CPU usage on each of the sites for read experiment
Other Graphs: Mem and Runq

Figure: Memory util. over time for each site

Figure: Runq util over time for each site
Other Graphs: Storage and CPU

Figure: Memory util. over time for each site

Figure: CPU util. over time for each site