Research Computing Web Portal
Overview, Motivation, and History

Dan Dietz
ddietz@purdue.edu

March 3, 2017
Components

There are a number of different management components to the web portal:

- User accounts
- Queues
- Unix groups
- Storage
- News
- Contact reports
- Orders

I’ll provide a quick overview and the history and motivation behind some of these.
User Accounts
Prior to 2011, user account management was a manual and tedious process:

- Faculty sends rcac-help a ticket to add new student
- Support staff picks up the ticket some time later
- Sometimes student would request, necessitating another loop to approve with faculty
- Goes to web tool to add ACMAINT role to create account
- Logs into cluster, types PBS commands to let student submit to the faculty’s queue
- Send new user the welcome packet
This process had some obvious problems

- **Slow:**
  - Turnaround time on a request often took hours, days, or even weeks
  - Average completion time was found to be 6.2 days (Cumberland; 2011)
  - Requests not made directly by faculty often added days of delay waiting for approval

- **Unreliable:**
  - Prone to human errors
  - Original request would sometimes be vague and would require several back-and-forths to gather required information
  - No accountability - only way to know who and when was through compiling huge stack of tickets
All and all, this slow process added significant costs to time-to-science for new users, and looked bad on us.

- In 2010, support handled 500+ tickets regarding user management.
- Assuming each request takes 15 minutes to handle, this is over 125 staff hours spent on mundane tasks in 2010 (Colby, Dietz, Smith, & Cumberland; 2014).
- Rapid growth of Community Cluster and Data Depot programs would have made this a nightmare in the years to come.
In 2011, we set out to build a self-service web-based tool:

- Faculty directly manage access
- User initiated requests
- Email confirmations to all parties at all steps
- Account creation totally automated
- Zero support staff involvement!

New system was initially piloted by a few friendly users with the deployment of Hansen. System was quickly rolled out to rest of Hansen users, then back ported to Steele and Coates a few months later.
User Accounts

User management interface allows:

- Faculty to manage membership in queues and Unix groups
- Faculty to delegate assistant or top student to manage on their behalf
- Students to initiate a request to the faculty for access
- Staff to see and manage access
### User Accounts

#### Managers

<table>
<thead>
<tr>
<th>Name</th>
<th>Halstead</th>
<th>Radon</th>
<th>Hathl</th>
</tr>
</thead>
<tbody>
<tr>
<td>ddietz</td>
<td>✓</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- **Daniel Dietz**
  - ddietz ✓
  - workq ✓
  - Hadoop

#### Members

<table>
<thead>
<tr>
<th>Name</th>
<th>Halstead</th>
<th>Radon</th>
<th>Hathl</th>
</tr>
</thead>
<tbody>
<tr>
<td>Kevin D Colby</td>
<td>✓</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Lev Gorenstein</td>
<td>✓</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Preston M Smith</td>
<td>✓</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Tanner C Strom</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- **Kevin D Colby**: colbykd ✓
- **Lev Gorenstein**: lev ✓
- **Preston M Smith**: psmith ✓
- **Tanner C Strom**: tstrom
User Accounts

Since the deployment of the user account management tool in Oct 2011:

- Over 12,000 user-queue additions
- Over 5,000 user-queue removals
- Over 1,200 user initiated requests

If we assume each of these was a ticket, 15 minutes each, that’s over 4,250 staff hours. In the last year alone, there were over 5,000 additions/removals. That works out to 2/3 of an FTE dedicated just to user management alone.
As part of the new user account management tool, we also desired a tool for managing the queues themselves. Previous process included a wall of sticky notes and a pile of emails to keep track of who owned what.

- Full accounting and tracking of sales
- Automate queue deployment, PBS command knowledge not required
- Controls what options faculty have in their user management interface
- Tweak queue parameters, PBS knowledge not required
## Queues

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Priority</td>
<td>1000</td>
</tr>
<tr>
<td>Dedicated Reservation</td>
<td>No</td>
</tr>
<tr>
<td>Scheduling Policy</td>
<td>Whole Node Scheduling</td>
</tr>
<tr>
<td>Default Walltime (hrs)</td>
<td>0.5</td>
</tr>
<tr>
<td>Max Walltime (hrs)</td>
<td>720</td>
</tr>
<tr>
<td>Max Jobs Queued</td>
<td>12000</td>
</tr>
<tr>
<td>Max Jobs Queued per User</td>
<td>5000</td>
</tr>
<tr>
<td>Max Jobs Running</td>
<td>none</td>
</tr>
<tr>
<td>Max Jobs Running per User</td>
<td>none</td>
</tr>
<tr>
<td>Max Cores per Job</td>
<td>1040</td>
</tr>
<tr>
<td>Max Jobs per Iteration Factor</td>
<td>2</td>
</tr>
<tr>
<td>Max Jobs per User Iteration Factor</td>
<td>1</td>
</tr>
<tr>
<td>User ACL Enabled</td>
<td>Yes</td>
</tr>
<tr>
<td>Group ACL</td>
<td></td>
</tr>
<tr>
<td>Subcluster(s)</td>
<td>a</td>
</tr>
</tbody>
</table>
# Queues

<table>
<thead>
<tr>
<th>Date</th>
<th>Nodes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dec 13, 2016</td>
<td>15</td>
</tr>
<tr>
<td>Jan 11, 2017</td>
<td>52</td>
</tr>
<tr>
<td>Jan 17, 2017</td>
<td>52</td>
</tr>
<tr>
<td>Today</td>
<td>52</td>
</tr>
<tr>
<td>End of Life</td>
<td>0</td>
</tr>
</tbody>
</table>

Legend: Purchases, Loans

- Sell Nodes
- Loan Nodes

Size: nodes / cores
Date Range: to end of cluster life
Sell to: (Select Group) (Select Queue)

Make Sale, Cancel
Queues

- Managed 700+ queues
- Managed 750+ node sales
- Managed 350+ node loans
Unix groups
Unix groups are technically limited to 8 characters in ACMAINT, despite modern Linux systems supporting more than 8 characters.

We able to workaround this limitation by commandeering another longer-length field, and some trickery on our systems to map each group with the longer name on our side.

The true name of the Unix group is not human readable so we needed a web tool to manage it.
## Unix groups

<table>
<thead>
<tr>
<th>Group Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dan's new tests</td>
<td></td>
</tr>
<tr>
<td>ddtest</td>
<td></td>
</tr>
<tr>
<td>(Add New Custom Unix Group)</td>
<td></td>
</tr>
</tbody>
</table>

### Unix Groups

- ddtest
- ddtest-data
- ddtest-apps
- ddtest-mgr
- ddtest-testers11
- ddtest-testers12
- ddtest-testers13
- ddtest-rewfds
- ddtest-testers14
- ddtest-done
- ddtest-testdir
- ddtest-foobar
- ddtest-footbar
- ddtest-dan
- ddtest-test2
- ddtest-testme
- ddtest-testers
Unix groups

- Faculty can provision their own Unix groups
- Managing 1600+ Unix groups
- Managing 9600+ Unix group memberships
Previous incarnations of group storage systems had a couple problems:

- Required tickets to storage admins to configure new group spaces
- Files became a jumbled mess of permissions
With Data Depot, we resolved these issues with:

- Interfacing with scripts on storage side to configure spaces automatically
- Using extended ACLs

```
setfacl --set="user::rwx,group::rwx,other::r-x,default:user::rwx,
     default:group::rwx,default:other::r-x"
chmod g+s
```

Who wants to remember all that or even understands it? Let’s make it automated.
Storage

Components
User Accounts
Queues
Unix groups

Storage
News
Contact
Reports
Orders

Directory

/depot/ddietz
apps

data
etc

(Add New Directory)
(Add New Top Level Directory)

Current Quota
Future Quota

1 TB
-
-
-
-

Quota
Access Unix Group
ddtest-apps
Read access for ddtest
Yes to directories:
/depot/ddietz/apps

Permissions
Group
ddtest-apps
ddtest
Public

Read
Write

✔
✔
✔
✘

Unallocated space: 0 B / 1 TB

[ Delete Directory ]
Storage

- New spaces and directories deployable in a minute
- Don’t need to know setfacl and GPFS juju to make changes
- 350+ group storage spaces
- 1700+ directories managed
We had a news system provided for us by another group with in ITaP:

- Clunky WYSIWYG editor, formatting was a pain
- No date fields: had to manually push/pull things from current postings on main page
- No date fields: every editor uses different date styles, some confusing for people of different backgrounds, would get a mish-mash of different styles across articles
- Not tied into the rest of the web portal
New news system is better:

- Markdown-like formatting syntax, formatting easily reproducible
- Events posted for the future automatically appear on the main page as they become relevant
- Can use variables inside text, such as date variables for consistency and readability
- Tied into the rest of the web portal. Tag outages to resources and the posting appears on that resource’s page
- Mailing list tied to the user management page, can mail user list with a click
- Nearly 300 news articles posted
Contact Reports

Contact Reports
Previously, we had no way of tracking contacts with users. And so the CRM system was created:

- Tag users and groups
- Can see other reports tagged with the same people
- Others can add comments onto your report
- Users of the tool can follow each other (I set this manually when new people join, need to automate) and receive emails when new posts are made
- Used by several groups, over 2300 contacts filed
As mentioned earlier, node and storage orders were managed by a wall of sticky notes and emails between faculty and business offices. We made that better:

- Let faculty log in, see prices, submit orders, and fill in payment information themselves
- We assign to faculty’s business office
- Faculty’s business office approves
- We fulfill the order
- We, the business office, and the faculty receive emails every step of the way, and can log in at any time to see the order status
## Order #1759

**Set Group**  
Daniel Dietz (ddietz)  
Scientific Applications Analyst  
Research Computing  
Graduate School  
ddietz@purdue.edu

### Awaiting Payment Information

February 7, 2017  8:57am  
Submitted by Daniel Dietz

<table>
<thead>
<tr>
<th>Status</th>
<th>Item</th>
<th>Quantity</th>
<th>Price</th>
<th>Subtotal</th>
</tr>
</thead>
<tbody>
<tr>
<td>Awaiting Approval</td>
<td>Halstead High-Performance Cluster (through Dec 2021)</td>
<td>1</td>
<td>$3,600.00 per node</td>
<td>$3,600.00</td>
</tr>
</tbody>
</table>

**Order Total**  
$3,600.00

### Payment Information

**Example Accounts**

<table>
<thead>
<tr>
<th>Fund</th>
<th>Cost Center</th>
<th>Order</th>
<th>Amount</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Add new payment account**

**Balance Remaining**  
$3,600.00
Orders

- Automatically generates renewal orders for monthly/yearly billed services
- System used by a few groups
- Over 1200 orders placed and fulfilled
- Over $11,000,000 worth of services transacted
Can't stress the most important feature of all this enough: **everything** is tied together and cross-referenced

- Search a user or group and get everything we know about them
- Queues they own, storage they own, all orders they’ve placed, contact information, all contacts we’ve had, etc
- One-stop shop for everything we might know
- Look up the user before going into a meeting with them to be informed