`pbsacct`: A Workload Analysis System for PBS-Based HPC Systems

Troy Baer
Senior HPC System Administrator
National Institute for Computational Sciences
University of Tennessee

Doug Johnson
Chief Systems Architect
Ohio Supercomputer Center

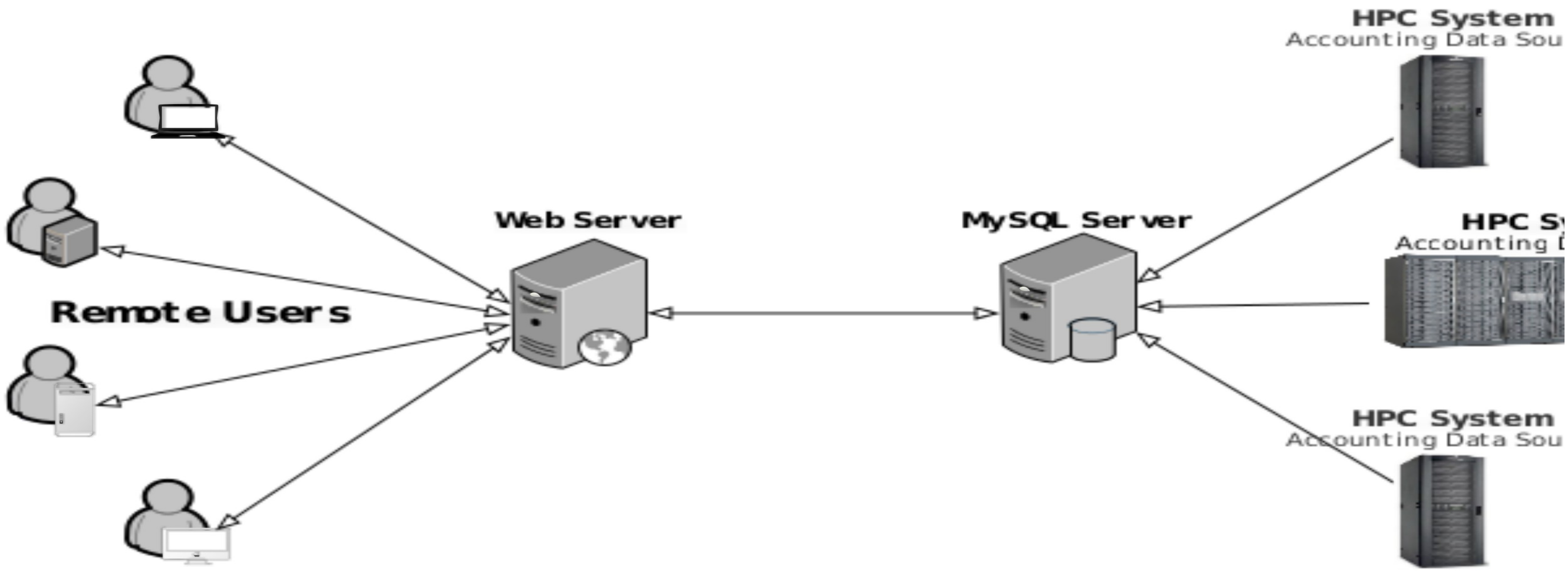
Overview

- **Introduction to `pbsacct`**
- **Technical Overview**
  - **Database Structure**
  - **Data Ingestion**
  - **User Interfaces**
- **Example Deployments**
- **Workload Analysis**
  - **NICS Kraken historical retrospective**
  - **OSC Oakley**
- **Conclusions and Future Work**

Introduction to `pbsacct`

- **`pbsacct` started at Ohio Supercomputer Center in 2005:**
  - **Grew from need to do workload analysis from PBS/TORQUE accounting logs.**
  - **Stores job scripts as well as accounting log data.**
  - **Ability to do on-demand queries on jobs across multiple systems and arbitrary date ranges.**
  - **Despite the name, not an allocation/charging system!**
  - **Open source (GPLv2)**
- **Structure:**
  - **Data sources**
  - **Database (MySQL)**
  - **User interfaces**
- **Development moved to NICS in 2008.**
  - **Available at `http://www.nics.tennessee.edu/~troy/pbstools/`**

pbsacct Architecture



Database Structure

- **Accounting data and scripts are stored in a MySQL database**
- **Two tables:**
  - **`Jobs`**
    - **Job accounting data and scripts**
    - **Used by just about everything**
    - **Indexed by `system`, `username`, `groupname`, `account`, `queue`, `submit_date`, `start_date`, and `end_date` to accelerate queries**
  - **`Config`**
    - **Used to track system changes WRT core count**
    - **Mainly used by web interface to compute utilization**

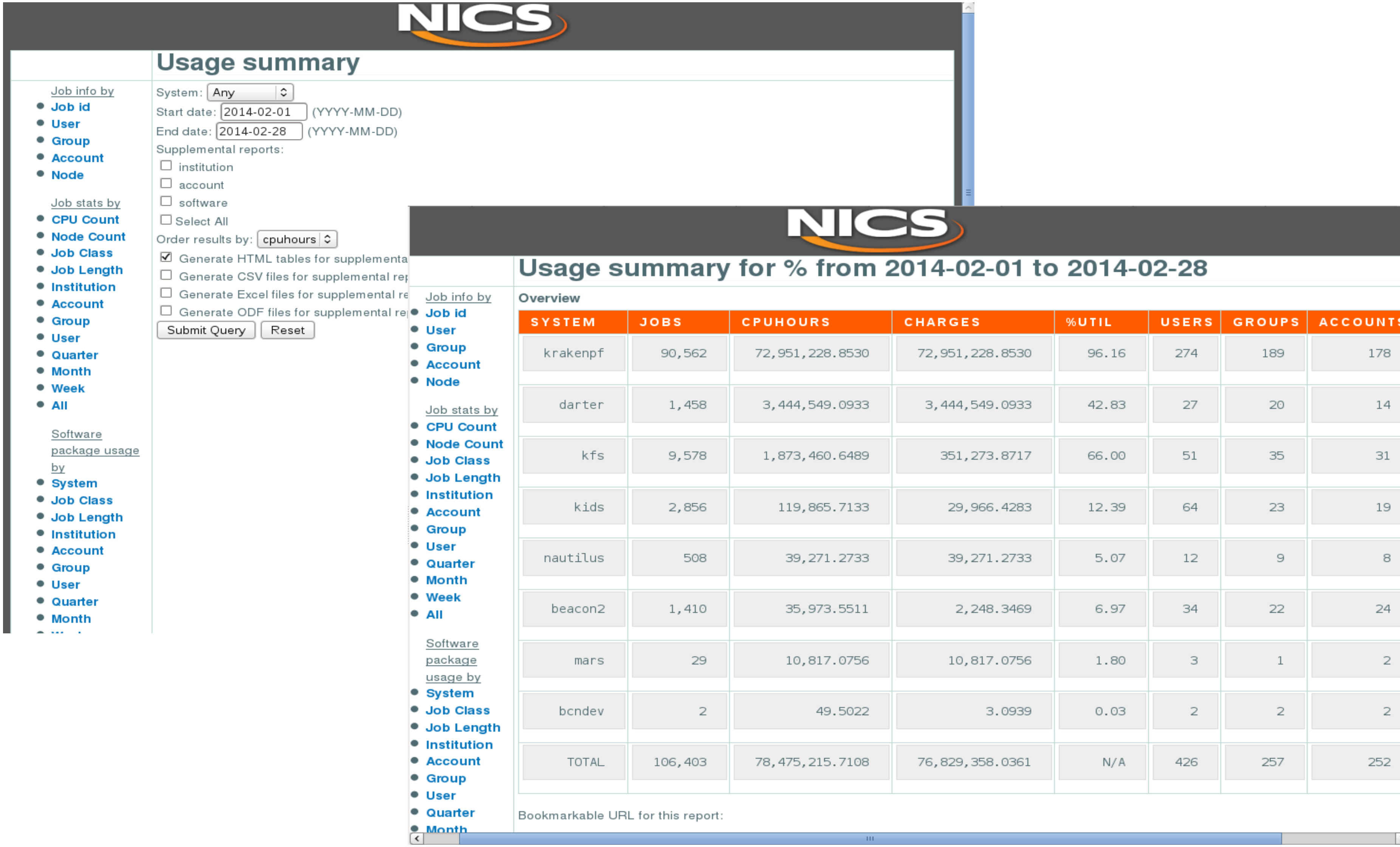
Data Ingestion

- **Accounting data comes in from hosts that run `pbs_server`:**
  - **A Perl script called `job-db-update` parses the accounting logs in `$PBS_HOME/server_priv/accounting` and inserts the results into the database.**
  - **Typically run out of a `cron` job (hourly, daily, etc.).**
- **Job scripts can also be captured on hosts that run `pbs_server`:**
  - **`dnotify`- or `inotify`-based daemon watches for new files created in `$PBS_HOME/server_priv/jobs`.**
  - **When new `.SC` files are created in the `jobs` directory, daemon launches a Perl script called `spool-jobscripts`.**
  - **`spool-jobscripts` copies the `.SC` files to a temp directory and launches another Perl script called `jobscript-to-db`, which inserts the scripts into the database.**
  - **This is done to be able to keep up with high throughput situations where there may be thousands of short-running jobs in flight and the database might not be able to keep up.**

User Interfaces

- **Command line**
  - **`js` – Look up job script by jobid.**
  - **Want to develop more, but need to figure out a workable security model.**
- **Web**
  - **PHP based, using several add-ons**
    - **PEAR DB**
    - **PEAR Excel**
    - **OpenOffice spreadsheet writer**
    - **jQuery**
  - **Lots of premade reports**
    - **Individual jobs, software usage, utilization summaries...**
    - **Site-specific rules to map job script patterns to applications**
  - **Meant to be put behind HTTPS**

Web Interface Example



SYSTEM	JOBS	CPUHOURS	CHARGES	%UTIL	USERS	GROUPS	ACCOUNTS
krakenpf	90,562	72,951,228.8530	72,951,228.8530	96.16	274	189	178
darter	1,458	3,444,549.0933	3,444,549.0933	42.83	27	20	14
kfs	9,578	1,873,460.6489	351,273.8717	66.00	51	35	31
kids	2,856	119,865.7133	29,966.4283	12.39	64	23	19
nautilus	508	39,271.2733	39,271.2733	5.07	12	9	8
beacon2	1,410	35,973.5511	2,248.3469	6.97	34	22	24
mars	29	10,817.0756	10,817.0756	1.80	3	1	2
bcndev	2	49.5022	3.0939	0.03	2	2	2
TOTAL	106,403	78,475,215.7108	76,829,358.0361	N/A	426	257	252

Example Deployments

- **OSC**
  - **~14.9M job records (~13.4M with job scripts)**
  - **~30GB database size**
  - **Web interface accessed over HTTPS with HTTP Basic authentication against LDAP**
- **NICS**
  - **~5.4M job records (~5.0M with job scripts)**
  - **~13.1GB database size, growth rate of ~600MB/month**
  - **Web interface accessed over HTTPS with RSA Securid one-time password authentication**

Workload Analysis: NICS Kraken Historical Retrospective

- **NICS Kraken**
  - Cray XT5 system with 9,408 dual-Opteron compute nodes
  - Operated in production for NSF from February 4, 2008, to April 30, 2014
  - Batch environment is TORQUE, Cray ALPS, and Moab
  - Queue structure:
    - `batch` (routing queue)
      - `small` (0-512 cores, up to 24 hours)
      - `longsmall` (0-256 cores, up to 60 hours)
      - `medium` (513-8192 cores, up to 24 hours)
      - `large` (8193-49536 cores, up to 24 hours)
      - `capability` (49537-98352 cores, up to 48 hours)
      - `dedicated` (98353-112896 cores, up to 48 hours)
    - `hpss` (0 cores, up to 24 hours)

Kraken Workload Analysis
2009-02-04 to 2014-04-30

Overall

- **4.14M jobs**
- **4.08B core-hours**
- **2,657 users**
- **1,119 projects**

NSF Teragrid/XSEDE

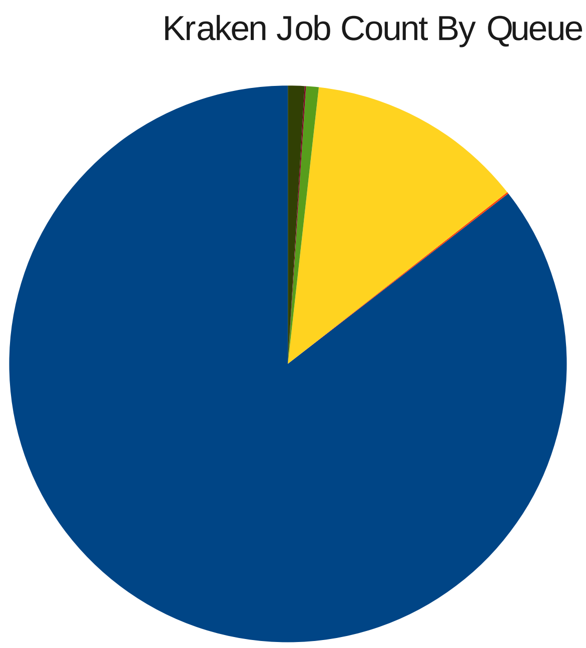
- **3.84M jobs**
- **3.85B core-hours**
- **2,252 users**
- **793 projects**

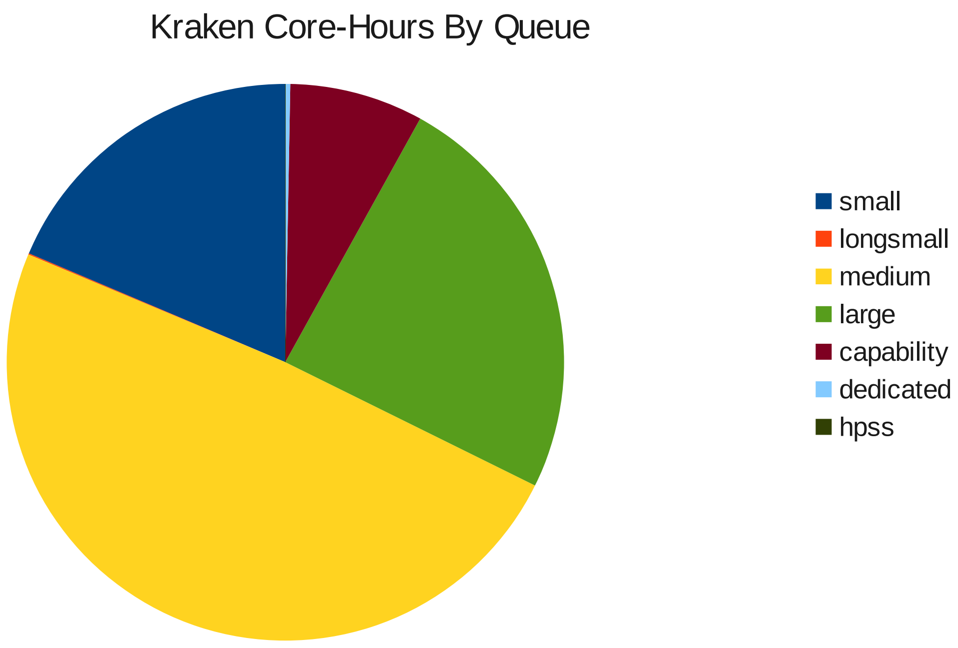
85.6% average utilization (not compensated for downtime)

Kraken Workload Analysis by Queue 2009-02-04 to 2014-04-30

QUEUE	JOBS	CORE HOURS	USERS	PROJECTS
small	3,576,368	768,687,441	2,602	1,090
longsmall	3,570	2,782,681	169	122
medium	488,006	2,003,837,680	1,447	718
large	27,908	983,795,230	521	301
capability	2,807	306,724,698	117	73
dedicated	338	11,765,421	17	7
hpss	36,462	53,285	184	123
TOTAL	4,136,759	4,077,647,799	2,657	1,119

Kraken Workload Analysis by Queue 2009-02-04 to 2014-04-30





Kraken Top 10 Applications by Core Hours 2009-02-04 to 2014-04-30

APP	JOBS	CORE HOURS	USERS	PROJECTS
namd	347,535	421,255,609	358	164
chroma	38,872	178,790,933	17	10
res	58,630	161,570,056	268	190
milc	22,079	146,442,361	37	21
gadget	6,572	131,818,157	29	21
cam	66,267	124,427,700	88	68
enzo	15,077	112,704,917	54	37
amber	103,710	110,938,365	208	120
vasp	148,686	94,872,455	147	85
lammps	137,048	94,398,544	187	127

Workload Analysis: OSC Oakley

- **OSC Oakley**
  - **HP Xeon cluster with 693 compute nodes**
    - **Most nodes are dual-Xeon with 12 cores**
    - **One node is quad-Xeon with 32 cores and 1TB RAM**
    - **64 nodes have 2 Nvidia M2070 GPUs each**
  - **Operated in production since March 19, 2012**
  - **Batch environment is TORQUE and Moab**
  - **Queue structure:**
    - **`batch` (routing queue)**
      - **`serial` (1-12 cores, up to 168 hours)**
      - **`parallel` (13-2040 cores, up to 96 hours)**
      - **`longserial` (1-12 cores, up to 336 hours)**
      - **`longparallel` (13-2040 cores, up to 250 hours)**
      - **`dedicated` (2041-8336 cores, up to 48 hours)**
      - **`hugemem` (32 cores, up to 1 TB mem, up to 48 hours)**

Oakley Workload Analysis
2012-03-19 to 2014-03-14

Overall

- **2.12M jobs**
- **112M core-hours**
- **1,147 users**
- **403 projects**

77.6% average utilization (not compensated for downtime)

Oakley Workload Analysis by Queue 2012-03-19 to 2014-03-14

QUEUE	JOBS	CORE HOURS	USERS	PROJECTS
serial	1,799,890	32,938,880	1,088	387
parallel	324,848	77,614,464	595	256
longserial	36	58,456	5	5
longparallel	158	1,574,567	5	3
hugemem	299	54,466	28	23
TOTAL	2,125,231	112,240,833	1,147	403

Conclusions and Future Work

- **`pbsacct` is feature rich and extensible**
  - **Written in Perl and PHP**
  - **Support for site-specific code**
  - **Scales to millions of jobs across tens of machines**
- **Future work**
  - **Better packaging to ease installation – RPMs?**
  - **Port to another DBMS (e.g. PostGreSQL)?**
  - **Speed up full text job script searches with external indices (e.g. Apache Lucene Solr)?**
  - **Interface with other RMs (Grid Engine, SLURM)?**