



PENGUIN COMPUTING
SUPERCOMPUTING SIMPLIFIED

HPC in the Cloud with Penguin Computing on Demand ("POD")

June 2011

Agenda

- (Brief) Introduction to Penguin Computing, Inc.
- Introduction to POD
- Case Studies
- POD Shell

Penguin provides Linux HPC Solutions

Linux Systems



Optimized for Linux

- Intel® & AMD® Rackmount Servers
- Storage
- Networking
- Infrastructure
- GPGPUs
- Professional Workstations

Cluster Management Software



Ease of Management

- Scyld ClusterWare™
- Scyld TaskMaster™
- Enterprise Linux
- Compilers & Tools

HPC as a Service - Penguin on Demand



Elastic Computing

- On-demand environment
- HPC 'optimized'
- Tiered 'Pay-as-you-go' pricing
- Premium set-up and support services

Professional Services and Engineering



Linux and Cluster Expertise

- Factory Integration
- Onsite Installation
- Training
- Product Support
- Software Support
- Customized Service Engagements

Large Cluster Example: Georgia Institute of Technology



- Center for the Study of Systems Biology
- 10,000+ AMD cores
- 53+ TFLOPS
- 300+ TB of storage
- Top 100 ranking



Software Product Line



Remote administrators/users



On-premise systems

Insight

- Complete physical or virtual cluster management GUI
- 1:1 match with Scyld CW functionality
- Hardware and workflow monitoring
- Integrated log viewer for maintenance and diagnostics

Beoweb

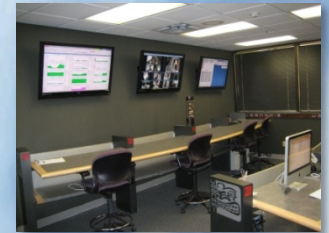
- Exposes Scyld CW functionality through a secure Web interface (<https://>)
- Enables Web sites and applications to dynamically interact with a Scyld cluster
- Dynamically present cluster and job status to users

PODTools

- Convenient and secure method to submit jobs and data to POD
- Automatically returns results when complete
- Customize workflow through a scriptable interface
- Generate ad hoc reports to see core hour and GB usage

Scyld ClusterWare

- Cluster management interface
- Minimizes effort required to set-up, update and maintain a cluster
- Guarantees consistency across all servers
- Provides complete, tested and supported cluster software stack



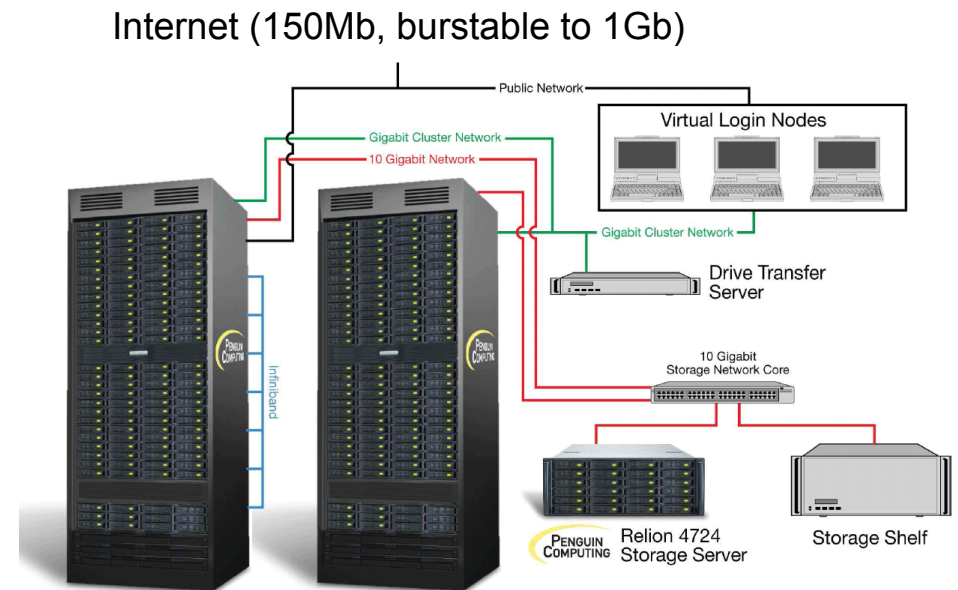
HPC in the Cloud

Introduction to POD

- What is POD
- Is POD Cloud Computing?
- How POD compares/contrasts to Amazon's EC2
- The new EDU POD
- Setting up your account
- Transferring data to/from POD
- Setting up your environment
- Running jobs
- GPU computing on POD
- Running your software
- Checking your usage and storage (PODReport)
- POD's fee structure

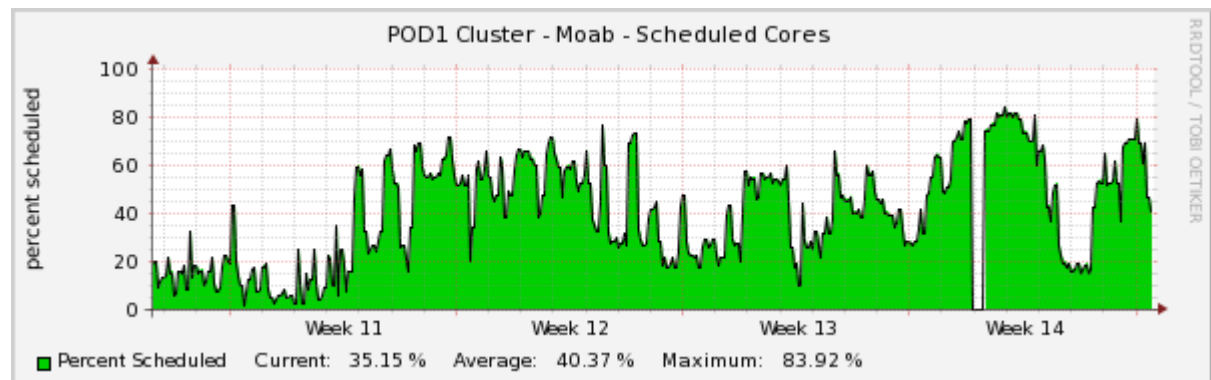
What is POD

- On-demand HPC resource
- Virtualized login node
 - > single core
 - > 2GB RAM
 - > (typical, greater capacities available)
- Physical compute nodes
 - > 2.67GHz Xeon processors
 - > 8 cores per processor
 - > 4GB / core DDR2 RAM
 - > 1TB scratch per compute node
- 10 Gig direct attached NFS storage
- Panasas HPPFS Available
- GigE, 10Gig and IB interconnect
- Deployed in 'scalable units'
- 1 to 480 cores available on-demand



POD usage highlights

- 5 million commercial jobs in 2010
- Users range from grad students to large financial institutions
- Highly variable utilization, but capacity designed to minimize (eliminate) queues
- Utilization characteristic of a variety of job types
- Secure computing
 - > end-to-end data encryption available
 - > users not shared across server boundaries



Is POD Cloud Computing

- Depends on your definition of “cloud computing”
- **Wikipedia:** “**Cloud computing** refers to the logical computational resources (data, software) accessible via a computer network, rather than from a local computer.”
 - > Based on Wikipedia, yes.
- **InfoWorld:** “Some analysts and vendors define **cloud computing** narrowly as an updated version of utility computing: basically **virtual servers** available over the Internet. Others go very broad, arguing anything you consume outside the firewall is ‘in the cloud,’ including conventional outsourcing.”
 - > Based on InfoWorld narrow definition, not really.
- **SearchCloudComputing.com:** “A cloud service has three distinct characteristics that differentiate it from traditional hosting. It is sold on demand, typically by the minute or the hour; it is elastic -- a user can have as much or as little of a service as they want at any given time; and the service is fully managed by the provider (the consumer needs nothing but a personal computer and Internet access).”
 - > Based on SearchCloudComputing.com, yes.

POD – HPC as a Service

	Definition	Examples	HPC Retail Price Point
HPC as a Service	Hybrid (virtual and physical) HPC cloud infrastructure plus HPC services and support	Penguin Computing	\$0.20/physical core hour * \$0.20/GB-month
Platform as a Service	Virtualized infrastructure plus Web services to deliver cloud computing	AWS	\$1.60 per virtual server hour (CCI)
Infrastructure as a Service	<u>Outsourced</u> storage, hardware, servers and networking	Rackspace	\$0.96 per server hour (monthly contract)
Hardware as a Service	Servers and network for rent	Numerous ...	\$0.08 per server hour and up (monthly contract)

** typically we see only 50 to 70% actual core utilization on HPC servers so the actual cost of Amazon, Google or Microsoft is 1.4 to 2 times the core hour rate shown*

Comparing POD and EC2



Description	amazon web services™	POD	Comments
High-speed Intel compute nodes	✓	✓	POD is bare-metal computing
Fast access to local storage	✓	✓	POD does not charge to move data
GPU enabled servers	✓	✓	
Cluster software stack included?	✗	✓ (included)	EC2 requires that you partner with a 3 rd party
HPC expertise?	✗	✓ (included)	Penguin has dedicated HPC customer support
Other compute nodes available	✗	✓	We have 2.6 GHz Harpertown, 2.93GHz Nehalem, 2.93 GHz Westmere, 2.3 GHz AMD MC 12C – more are added every month
Customized Linux compute environment?	Available	✓ (included)	EC2 instances must be created by the user
Bill by the core hour?	✗	✓	EC2 billed by the server hour (rounded up)
Data locality?	✗	✓	No guarantee where Amazon's data is stored
Troubleshooting?	\$\$\$	✓ (included)	HPC applications are not plug 'n play
Dynamically scalable?	Limited	1-480 cores on-demand	EC2 provides up to 8 CCIs, but you can request more...

The New EDU POD

- Owned and operated by the Penguin Computing on Demand staff
- Hosted at Indiana University data center
- AMD based compute infrastructure (shared environment)
 - > Altus 1804 servers
 - > 48 cores per server (Opteron 6174 12C, 2.2 GHz)
 - > 128GB, DDR3-1333 RAM per server
- Lustre scratch file system (100TB)
- QDR (40Gbps) Infiniband
- Scyld Cloud Management System
- Internet, Internet2 connectivity



Setting up your POD account

- Click the link to the POD Order Form
 - Provide the Services and Infrastructure you think you need
 - A POD Account Representative will email you to review your requirements and establish your POD account
- Log into your POD Portal and review/accept the agreements and fees
- POD will contact you for your IP address and open the firewall
- You can now log in...



- Transferring data to/from POD

- SCP

Upload bandwidth	Time to transfer 1GB	Time to transfer 50GB	Time to transfer 200GB
1 Mbps	2 hours	5 days	Take your 2 week vacation...
10 Mbps	13 minutes	11 hours	2 days
50 Mbps	3 minutes	2 hours	9 hours
150 Mbps	53 seconds	45 minutes	3 hours

- POD Disk Caddy (free, but the user pays the FedEx fees)
- PODShell remote data staging (more on this later)

- Dedicated Storage Node serves as a physical Login Node
- User data transferred to POD on disks can be immediately available
- User can have complete control over data transferred (software RAID, encryption, etc.)
- Discounts apply to quarterly or yearly contracts



Setting up your environment

- *Preparing your Application*
 - > Upload (scp) your custom code or download (wget) OpenSource code
 - > Configure the code with PREFIX=\$HOME/projectname
 - > All \$HOME paths are available on compute nodes
- *Compiling your Application*
 - > Use available env-modules to load libraries that might be necessary to compile (cuda, openmpi, etc)
 - > To see available modules: *module avail*
 - > To load an available module (example: GNU OpenMPI): *module load openmpi/gnu*
 - > Env-modules will automatically set PATH, LD_LIBRARY_PATH and MANPATH to load the proper locations.
 - > For instance, In the case of *module load openmpi/gnu*, you will be provided with OpenMPI's mpicc, mpif90, mpirun, etc.
- *Prepare a Job Script to Run your Application*
 - > Write a PBS or SGE .sub file that will run your application
 - > Use env-module commands inside job scripts to load things like OpenMPI's mpirun
 - > Job script examples are provided upon request

Running jobs

- **Submission: How to request nodes, connectivity, etc.**
 - > To see node availability: pbsnodes
 - > To see queue availability: qstat -q
- **Submission: How to request notifications**
 - > Either with qsub or inside the .sub script:
 - > #PBS -M emailaddress@yourdomain.com
 - > #PBS -m abe
- **Submission: Specifying environment variables**
 - > Not sure what you are looking for here - please explain.
- **Submission: Setting dependencies**
 - > Job dependencies are configured using qsub -W. To make job 2 dependent on job 1:
 - qsub job1.sub
 - 1.scyld
 - qsub -W depend=afterok:1.scyld job2.sub

Running jobs (continued)

- **Submission: Submitting your job**
 - > Use qsub to submit your job script
 - > qsub <jobscript.sub>
- **Runtime: Check job status**
 - > Use qstat to check your job status.
 - qstat
 - qstat -an
 - Using beostatus remote
- **Runtime: Deleting a job**
 - > Use qdel to delete a job that is Running or in Queue
 - > qdel <jobid>

GPU computing on POD

- POD's CUDA-enabled nodes have NVidia C1060s (or later) installed. To request a CUDA-enabled node, request your job with the 'C1060' PBS node attribute. If any type of card is sufficient, you can use the 'cuda' attribute.
- For instance:
 - > #PBS -l nodes=1:ppn=8:C1060

 - Or:

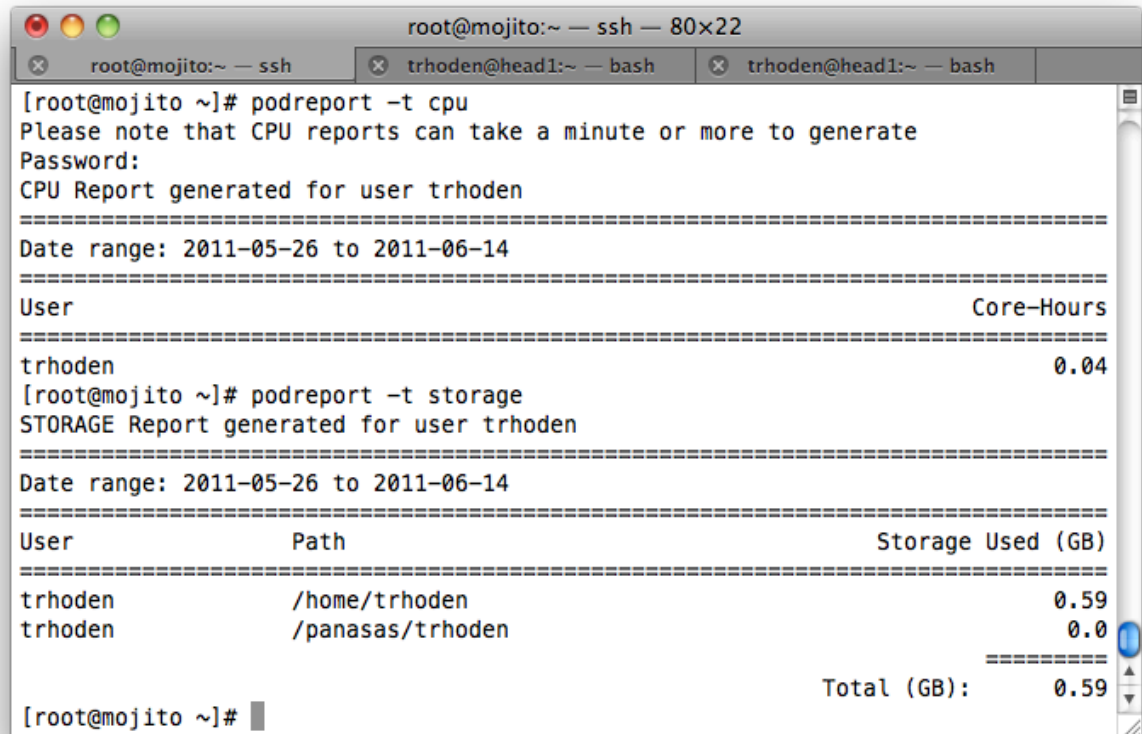
 - > qsub -l nodes=1:ppn=8:cuda

Running your own software

- GCC, CUDA and OpenMPI compilers are provided in the POD environment so that customer can install applications in \$HOME, which is shared out to the compute nodes. For example, to load OpenMPI's mpicc, mpif90 and mpiCC, use the command: `module load openmpi/gnu`
- If applications need to be shared amongst multiple users, POD administrators can install the application in a shared /opt space, which is also mounted by compute nodes
- Applications installed in /opt for groups of users are loaded into your shell environment using the module load command. For example: `module load R`
- Commercial applications like Matlab and Lumerical that require license servers can be configured to request licenses from your license server
- Alternatively, POD may also host your license server if permissible by the vendor's license agreements

Checking your usage and storage (PODReport)

- Query core hour usage of completed jobs
- Query storage utilization
- Custom report dates
- Current billing-cycle
- Previous billing cycle
- Group reports



```
root@mojito:~ — ssh — 80x22
root@mojito:~ — ssh  trhoden@head1:~ — bash  trhoden@head1:~ — bash
[root@mojito ~]# podreport -t cpu
Please note that CPU reports can take a minute or more to generate
Password:
CPU Report generated for user trhoden
=====
Date range: 2011-05-26 to 2011-06-14
=====
User                                                    Core-Hours
=====
trhoden                                                    0.04
[root@mojito ~]# podreport -t storage
STORAGE Report generated for user trhoden
=====
Date range: 2011-05-26 to 2011-06-14
=====
User          Path          Storage Used (GB)
=====
trhoden      /home/trhoden  0.59
trhoden      /panasas/trhoden  0.0
=====
Total (GB):  0.59
[root@mojito ~]#
```


POD Offerings and Prices

Service	Fee
Core hour rates	\$0.10 to \$0.20 pch (plus application uplift)
Dedicated servers	\$800 to \$1,650 per month
On-demand Storage and Fast-access Scratch Storage	\$200 per TB-month
Combination Login/Storage Node (48TB raw)	\$1600/month (customer supplies disks)
Persistent Login Node	<ul style="list-style-type: none">• Virtual Machine - \$79.95/month• Physical Server - \$290/month
Bandwidth	<ul style="list-style-type: none">• Included
Support	<ul style="list-style-type: none">• Included
Disk transfers	<ul style="list-style-type: none">• Included

Case Study: Dolby Laboratories

- Develop encoding IP
 - > HDTV, Blu Ray etc.
- Scalable compute intensive algorithms
- Iterative development process
 - > Faster turnaround of encoding jobs ⇒ Increased development efficiency
- Massive amounts of 'raw' data as encoding input
- Limited 'in-house' resources and admin capabilities
- Uneven workload over time

The POD Solution

- Migration of scalable algorithms to POD
- Scale out to 100 cores per job
- Data management
 - 'Raw' data upload through POD's caddy service
 - Processed encoded data downloaded
- Reduced job runtime from 2 days ⇒ 2 hours



“The use of POD is contributing significantly to making our development process more efficient”

Gopi Lakshminarayanan,
Director of Software
Development

Case Study: Earthmine Inc.

- Specialized in 3D street level imagery
- Library of 3D street level data/API's for web applications
 - > Allows for application integration of interactive street level maps
 - > Images spatially accurate
- Eight 2D images per location need to be processed to create accurate 3D 'image'
 - > Image processing computationally expensive
 - > Uneven workload dependent on # of data collection vehicles
 - > Limited in-house resources for data processing

The POD Solution

- POD provides 'overflow' capacity
- Submission of up to 40,000 jobs per day, 300,000 per month
- Integration with Amazon EC2 for Earthmine cloud services

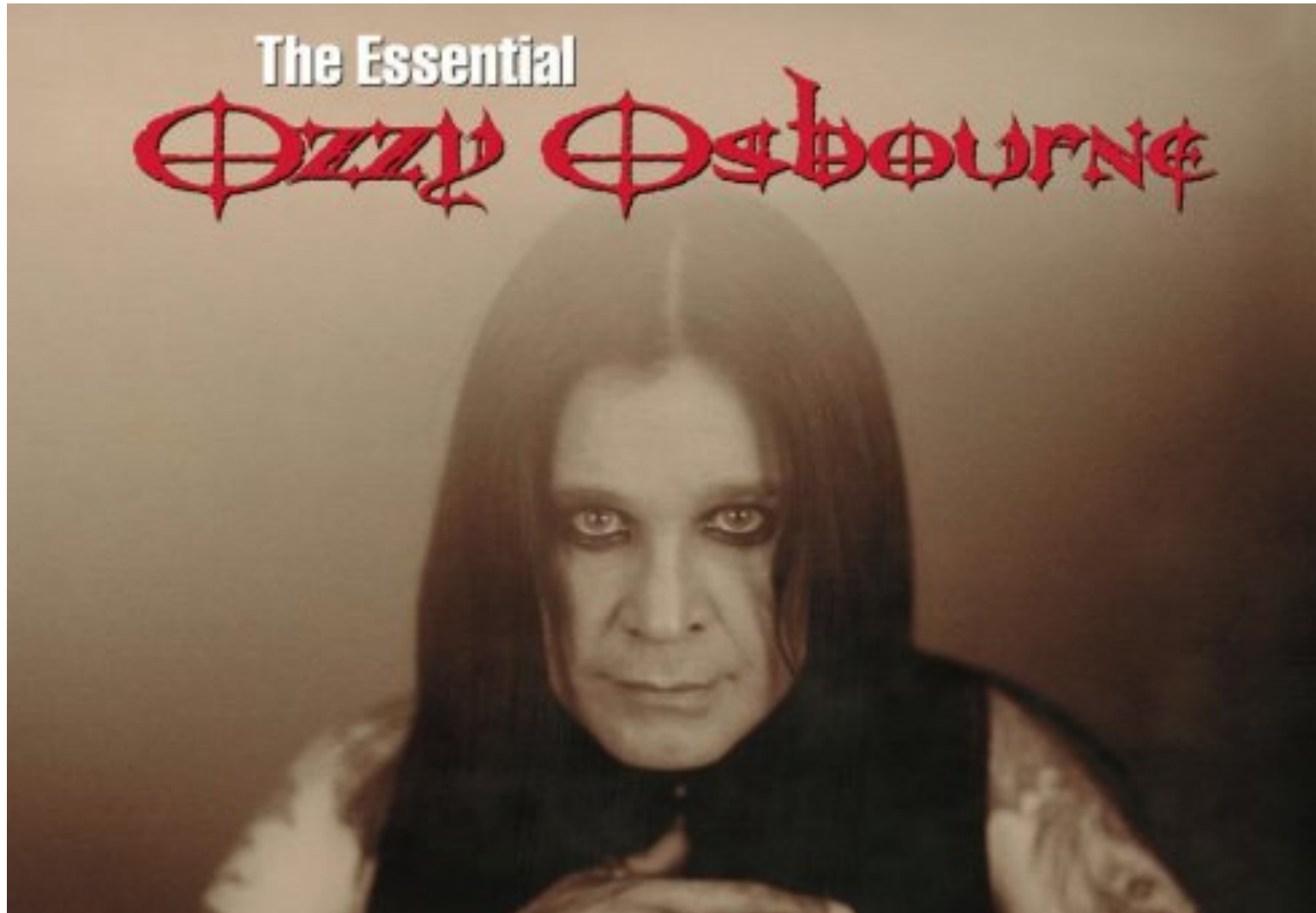


“Overall, the experience has been great. POD is fast, reliable, and works as described. In particular, POD has provided excellent support and Penguin has gone out of its way many times to accommodate our requests for technical help and to meet our fast-changing needs.”

John Ristevski,
CTO Earthmine



Case Study: Cofactor Genomics



Hard Rock, Life Science and POD



- Ozzy Osbourne's genome analyzed on SOLiD 4
- Cofactor generated 39 Gigabases (13 x coverage)
- Full sequence data shipped to POD on hard drive
- Secondary Analysis: Bioscope mapping-pairing engine on POD used for mapping sequence to reference (HG18)
- Tertiary Analysis: SNPs and small indels identified and analyzed by Bioscope pipeline and analyzed by Knome
- Results
 - > Novel variant in the *ADH4* alcohol metabolizing gene
 - > Variant for caffeine sensitivity
 - > variants in *TTN* (deafness and Parkinsonism) and *CLTCL1* (brain chemistry)
- Detailed coverage at <http://www.bio-itworld.com>

Case Study



“Penguin provides the cluster management technology and expertise to support our next generation gene sequencer”

- Tim Burcham, Vice President



Solution Provided

- Life Technologies is a world leader in gene sequencing systems with 250 customers world-wide, growing at 100% per year
- Life Tech’s on-demand SaaS solution LifeScopeCloud.com is hosted on POD
- Penguin provides the first CE Mark certified BioScope cluster appliance world-wide
- Each Life Technologies customer can generate 1TB of gene sequencing data per week per machine.

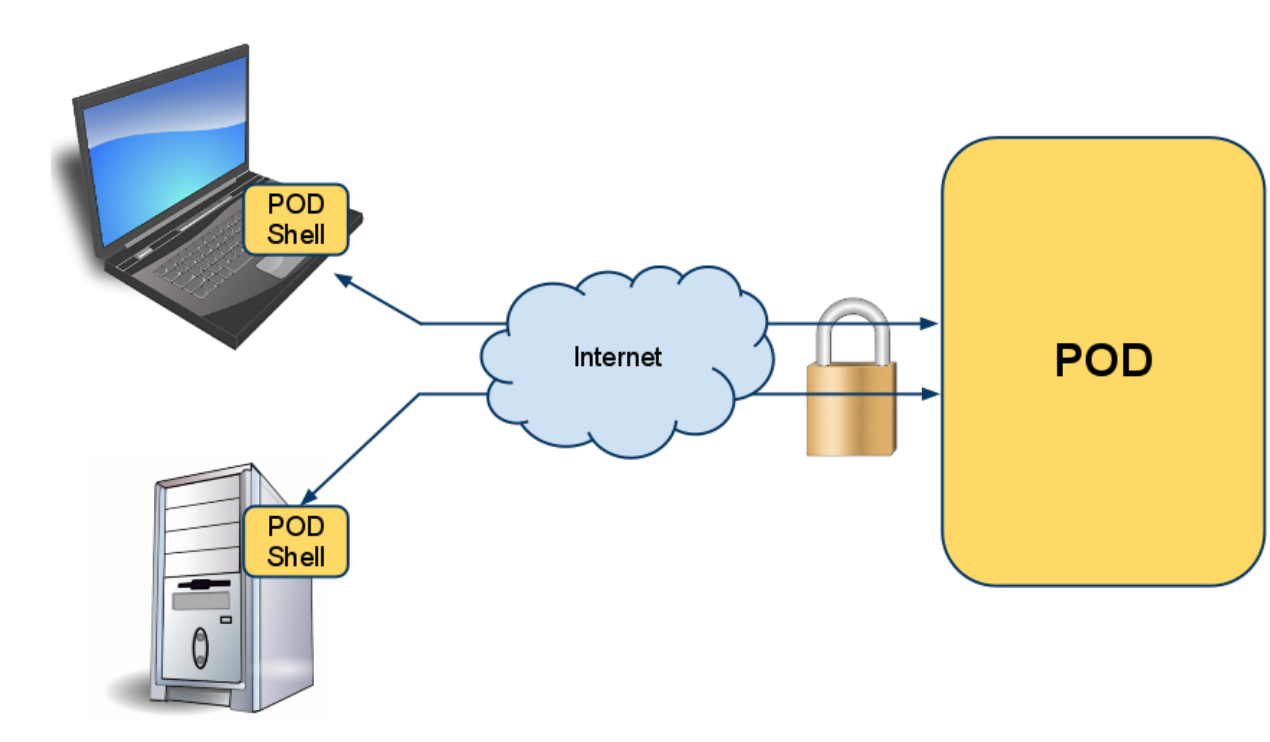


PODShell (live demonstrations)

- What is POD Shell
- How it is set up at Caltech
- Submitting Jobs
- Switching between TORQUE and SGE
- Data Staging
- Scripting / workflow
- Security
- Report examples

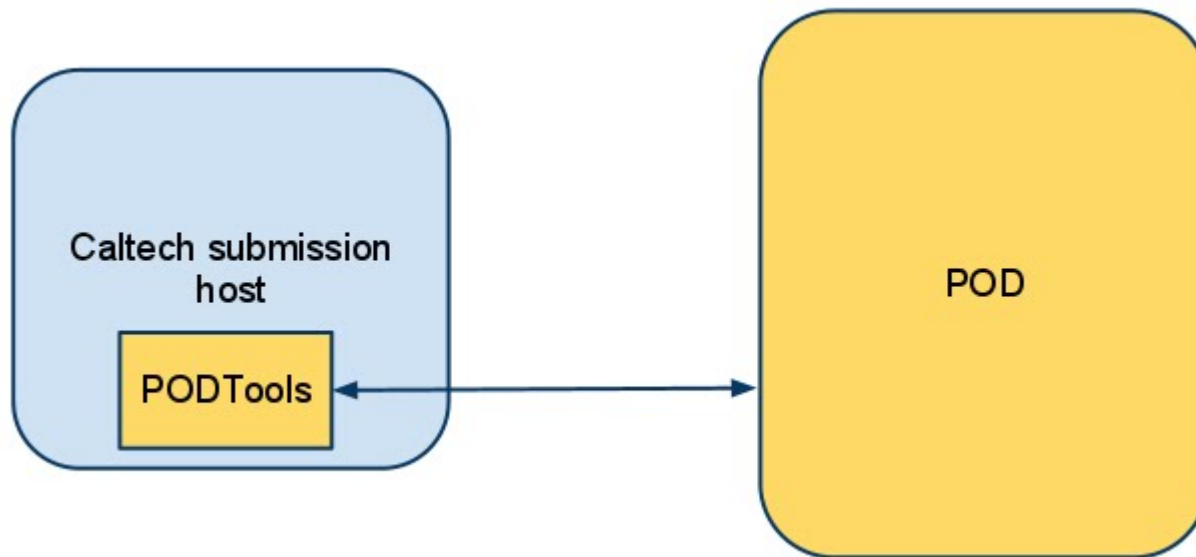
PODShell – overview

- Secure, remote access to POD
 - > Submit jobs directly to POD without SSH/SCP
 - > Copy data to/from POD, either as part of a compute job or independent (data staging)



How PODShell is set up at Caltech

- Dedicated submission host with same libraries and compilation environment as POD
 - > Enables upload of binary compatible executables
- Automated user creation based on LDAP attributes-
- Caltech/Penguin jointly managed



PODShell system requirements/compatibility

- PODShell is installable on RHEL4 or RHEL5 systems (or CentOS)
- We distribute RPMs for:
 - > PODTools (includes PODShell and PODReport)
 - > Python 2.6.5
 - > Python libraries/utilities
- RHEL6 support in Q3 2011
- Internet access
- Firewall open to ports 5000, 10000-10100

PODShell – submitting jobs

- POD supports two schedulers, TORQUE and SGE
- PODShell acts as a “remote qsub”
- Submit your job script with `podsh submit`

```
[root@mojito ~]# podsh submit test.sub  
Job Submitted successfully. Job ID: 1586
```

- PODShell can submit to different schedulers from the command like with `--sched=` option:

```
[root@mojito ~]# podsh submit --sched=SGE test.sub  
Job Submitted successfully. Job ID: 1587
```

PODShell – data staging

- PODShell can copy your data for you
 - > As part of a job, or independently
- Use `--stagein` and/or `--stageout` options

```
[root@mojito ~]# podsh submit --stagein=mydata.bin  
test.sub  
Job Submitted successfully. Job ID: 1588
```

- Your job will automatically be held until stagein completes!

```
[root@mojito ~]# podsh submit --stageout=\~/  
myresults.log test.sub  
Job Submitted successfully. Job ID: 1589
```

- Stageout will not occur until after the job finishes

PODShell – data staging (continued)

- Combine the power of stagein and stageout to create a single command that executes your entire workflow.
- Without POD Shell
 - > SCP Data from local machine to POD
 - > SSH to host and submit job
 - > Status job from SSH session
 - > Or wait for job email
- Once job is completed, SCP data from POD to your local machine
- With POD Shell:
 - > All steps accomplished by one command

```
[root@mojito ~]# podsh submit
--stagein=mydata:~/data --
stageout=~\~/myresults.log
myjob.sub
Job Submitted successfully.
Job ID: 1590
```


PODShell – checking job status

- `podsh status` gives you output similar to `qstat`
- Summary or detailed view, plus data staging information

```
[root@mojito ~]# podsh status
```

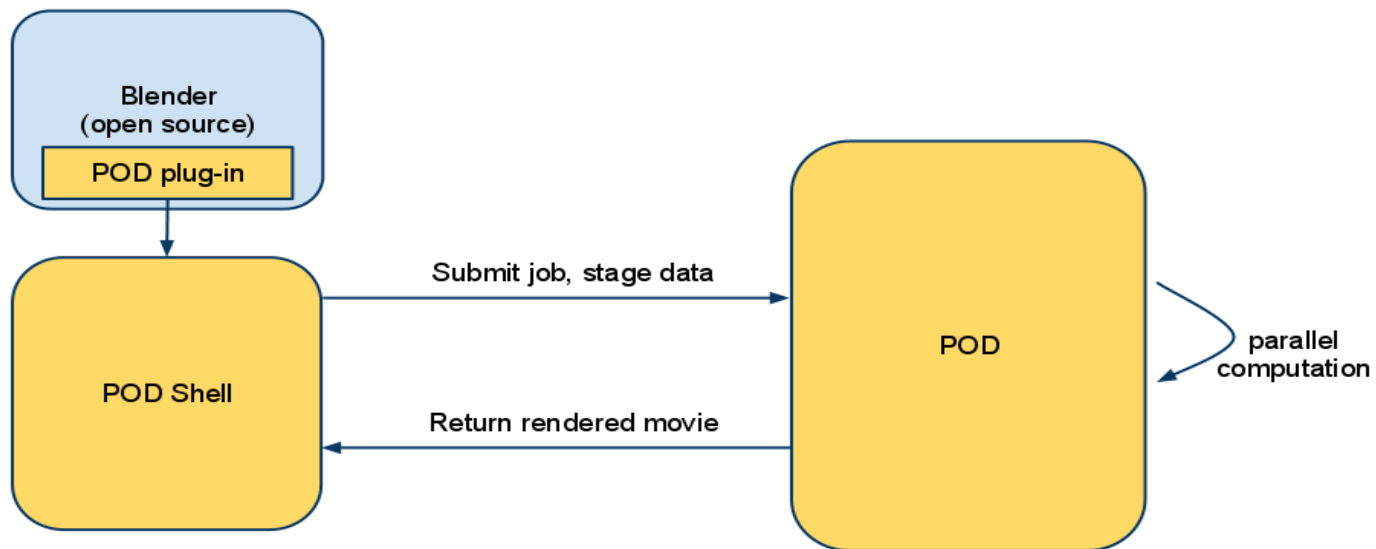
```
=====
Id           Type      State      Job Name      S Stage-in      Stage-out
-----
361          COMPUTE  COMPLETE  N/A           FINISHED      NONE
1214         COMPUTE  COMPLETE  N/A           FINISHED      NONE
1373         COMPUTE  COMPLETE  N/A           FINISHED      FINISHED
1590         COMPUTE  RUNNING   test.sub      R NONE        NONE
```

```
[root@mojito ~]# podsh status 1590
```

```
ID: 1590
  Type: COMPUTE
  State: RUNNING
  Join_Path = oe
  fault_tolerant = False
  exec_host = n86/0
  Resource_List.needsnodes = 1:ppn=1
  Resource_List.walltime = 00:05:00
```

PODShell – it's scriptable...

- Since PODShell is a command line interface, it's scriptable
- Allows automated workflows to POD without having to script SSH access and data transfers
- Can be called from other programs with a “plug-in” capability
- At SC10, we demoed a Blender plug-in built on PODShell that exported the scene to POD, rendered the frames, created the movie/animation, then copied result back to client machine



PODShell – security

- PODShell communicates with POD using HTTPS and TCP
- HTTPS used for everything except data staging
- TCP sockets used for data staging
- BOTH encrypted with TLSv1 (successor to SSLv3)
- POD does not accept unencrypted connections with PODShell

PODReport

- PODReport is part of our PODTools “suite” of tools – which also include PODShell
- Allows for custom usage reports for core-hours and storage

```
[root@mojito ~]# podreport -t cpu
Please note that CPU reports can take a minute or more to generate
Password:
CPU Report generated for user trhoden
=====
Date range: 2011-05-26 to 2011-06-14
=====
User                                                                                               Core-Hours
=====
trhoden                                                                                               0.04
[root@mojito ~]# podreport -t storage
STORAGE Report generated for user trhoden
=====
Date range: 2011-05-26 to 2011-06-14
=====
User          Path                                                                                               Storage Used (GB)
=====
trhoden       /home/trhoden                                                                                           0.59
trhoden       /panasas/trhoden                                                                                       0.0
=====
Total (GB):    0.59
```