

CONTINUOUS INTEGRATION IN A CRAY MULTIUSER ENVIRONMENT

BEN LENARD
HPC Systems & Database
Administrator



May 22nd, 2018
Stockholm, Sweden

ARGONNE



Director: Paul Kerns

Managed by: UChicago Argonne, LLC

Budget: \$750 million (FY 2017)

Workforce:

3,200 total employees (FTEs)

1,623 scientists and engineers

270 postdoctoral scholars

569 graduate and undergrad students

274 joint faculty

8,300+ facility users

- The first science and engineering research national laboratory in the U.S.
- Argonne integrates world-class science, engineering, and user facilities to deliver innovative research and technologies.
- Argonne creates new knowledge that addresses the scientific and societal needs of our nation.

ACKNOWLEDGEMENT

This research used resources of the Argonne Leadership Computing Facility, which is a DOE Office of Science User Facility supported under Contract DE-AC02-06CH1135



U.S. DEPARTMENT OF
ENERGY

Office of
Science



THE UNIVERSITY OF
CHICAGO

ARGONNE LEADERSHIP COMPUTING FACILITY SUPERCOMPUTERS

Theta Intel/Cray [Production]

- 4,392 nodes
- 16 GB MCDRAM, 192GB RAM per Node
- Peak flop rate: 9.65 PF
- 10 PB Lustre Filesystem

Mira - IBM BG/Q [Production]

- ⊙ 49,152 nodes / 786,432 cores, Peak flop rate: 10 PF
- ⊙ 786 TB RAM

Cooley - Cray/NVIDIA [Production]

- ⊙ 126 nodes / 1512 Intel Haswell CPU cores
- ⊙ 126 NVIDIA Tesla K80 GPUs
- ⊙ 48 TB RAM / 3 TB GPU memory
- ⊙ Peak flop rate: 223 TF

Storage

- ⊙ Home: 1.44 PB raw capacity
- ⊙ Scratch:
 - fs0 - 26.88 PB raw, 19 PB usable; 240 GB/s sustained
 - fs1 - 10 PB raw, 7 PB usable; 90 GB/s sustained
- ⊙ Tape: 21.25 PB of raw archival storage [17 PB in use]



CONTINUOUS INTEGRATION (CI)

What is it?

- The ability to checkout code from a software repository
- The ability to compile the code
 - On-Demand or on a set schedule
- The ability to test the code to verify it still functions as expected.
- Ideally this provides better code for the project since there's consistent testing.

CI IN ALCF

- In 2017, users started inquiring about a CI solution
- Since we are an open science user facility, our users are located globally
- ALCF's Requirements for a CI solution:
 - Security
 - Multiplatform Support
 - Easy of Use
 - Integration with various software repositories
 - Maintainable
 - Cobalt integration
 - Actively maintained

THE SOLUTION

- After considering various options, we deployed a open source Jenkins solution
- ALCF has extensive knowledge of Jenkins since it used for their internal software development
- It is actively developed with a long term support release
- It provides the project level segregation the we require
- Integrates within the ALCF environment
 - X86_64 and PPC hardware, as well as any environment with a JRE
- *ALCF already creates a linux group per project

THE SOLUTION (CONT)

- Easy of use
 - Jenkins has a large following with tutorials on-line
- Security
 - 2FA for user logins
 - Logging of user actions to a central logging service
- Project isolation
 - We isolate our projects based on Linux groups
- Integration with software repositories external to ALCF
 - The ALCF does not host software repositories
 - Git, SVN, Mercurial, etc.

THE SOLUTION (CONT)

- Manageable
 - We have deployed this solution with all open source plugins
 - Limited customization needed
- Centralization
 - Jenkins provides a central location for managing various jobs for the project
- The compiling of code or execution is occurring on the login nodes and generic x86_64; therefore, the Jenkins VM is lightweight

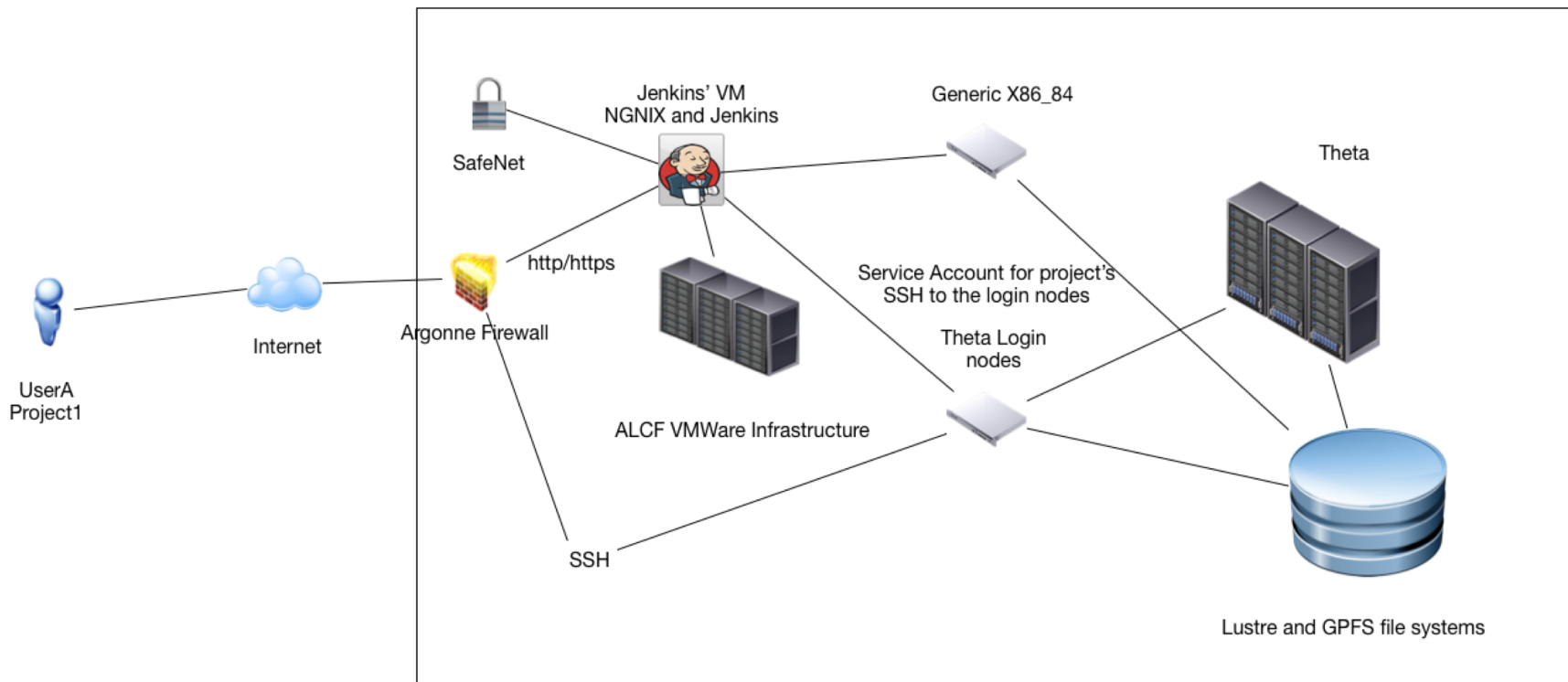
HOW JENKINS IS DEPLOYED AT THE ALCF

- How Jenkins is deployed
 - The Jenkins service is deployed in a VM since VM Ware provides the ability to grow the server on-demand
 - VM Ware also provides High Availability
 - Slaves are deployed on bare metal servers
 - The Jenkins data directory is hosted on a NFS appliance which also provides for snapshotting
- Nginx is deployed as the webserver front-end
 - We deployed NGINX as the webserver so we can decouple the webserver if needed

WHY NOT JUST USE CRON?

- Jenkins provides the following:
 - Build steps within a job – you can have dependent steps within a job
 - Build timeouts – you can set a duration for a job to run
 - It captures stdout and stderr and keeps this centrally. Jenkins will also prune the logs as defined by the users
 - It also does not start a new job until the currently executing one is completed
 - It provides a secure location, not on the shared filesystem, to store the project's credentials to the software repositories.
 - Lastly, Jenkins provides centralization.

JENKINS IN ALCF



KEY JENKINS PLUGINS

- Folders
 - Allows for project separation
 - Credentials are also stored at the folder level
- Job and Node ownership
 - This also aides in the isolation
 - This plugin ties a project's linux group to a job
- Job restrictions
 - This is how we prevent jobs from running on another project's slave
- SCM Sync Configuration
 - This enables automatic backups of the Jenkins configurations to a Git repo

KEY JENKINS PLUGINS

- Matrix authorization
 - With the use of this and folders, we are able to keep projects isolated
 - This is used in conjunction with the LDAP plugin which feeds it group information
- SSH Slaves
 - The ssh slaves ssh from the VM over a private network to the login nodes or generic x86_64 servers when the demand arises for a build. Slaves are only activated when the demand is in the queue
- LDAP authentication
 - This plugin allows Jenkins to communicate with our LDAP systems
 - LDAP also provides Jenkins with group membership information

JENKINS FROM AN ADMIN STAND POINT

Dashboard [Jenkins]

Jenkins

Ben Lenard | log out

ENABLE AUTO REFRESH

New Item

People

Build History

Project Relationship

Check File Fingerprint

Manage Jenkins

My Views

Credentials

New View

ALCF CI System

add description

S	W	Name	Last Success	Last Failure	Last Duration
		[REDACTED]	N/A	N/A	N/A
		[REDACTED]	N/A	N/A	N/A
		[REDACTED]	N/A	N/A	N/A
		[REDACTED]	N/A	N/A	N/A

Icon: S M L

Legend RSS for all RSS for failures RSS for just latest builds

Build Queue

No builds in the queue.

Build Executor Status

[REDACTED] (offline)

[REDACTED] (offline)

[REDACTED] (offline)

[REDACTED] (offline)

[REDACTED] (offline)

[REDACTED] (offline)

JENKINS FROM THE PROJECT STANDPOINT

The screenshot shows the Jenkins dashboard for the 'ALCF CI System'. The main content area displays a table of build history with columns for Status (S), Warnings (W), Name, Last Success, Last Failure, and Last Duration. The 'Name' column contains a redacted entry. Below the table, there are links for 'Icon: S M L' and several RSS feeds: 'Legend', 'RSS for all', 'RSS for failures', and 'RSS for just latest builds'. On the left sidebar, the 'Build Queue' section shows 'No builds in the queue.' and the 'Build Executor Status' section shows a list of executors, with the first one being redacted.

S	W	Name ↓	Last Success	Last Failure	Last Duration
		[REDACTED]	N/A	N/A	N/A

JENKINS FROM THE PROJECT STANDPOINT (CONT)

Build Queue: No builds in the queue.

Build Executor Status: (offline)

S	W	Name ↓	Last Success	Last Failure	Last Duration
🟢	☀️	[REDACTED]	1 day 6 hr - #17	5 days 9 hr - #15	3 hr 24 min
🔴	☁️	[REDACTED]	24 days - #17	1 day 15 hr - #30	7 sec
🟢	☀️	[REDACTED]	24 days - #4	26 days - #2	0.44 sec

FUTURE JENKINS WORK

- We are currently seeking more ‘friendly’ projects
- We would like to explore workflows with a project in Jenkins. Jenkins provides the framework for workflows natively
- We would like to automate the project on-boarding process
- Direct integration into the Cobalt scheduler using it’s API
- Documentation for our users and catalysts

QUESTIONS?

**UCHICAGO
ARGONNE,LLC**



U.S. DEPARTMENT OF
ENERGY

Argonne National Laboratory is a
U.S. Department of Energy laboratory
managed by UChicago Argonne, LLC.

Argonne 
NATIONAL LABORATORY