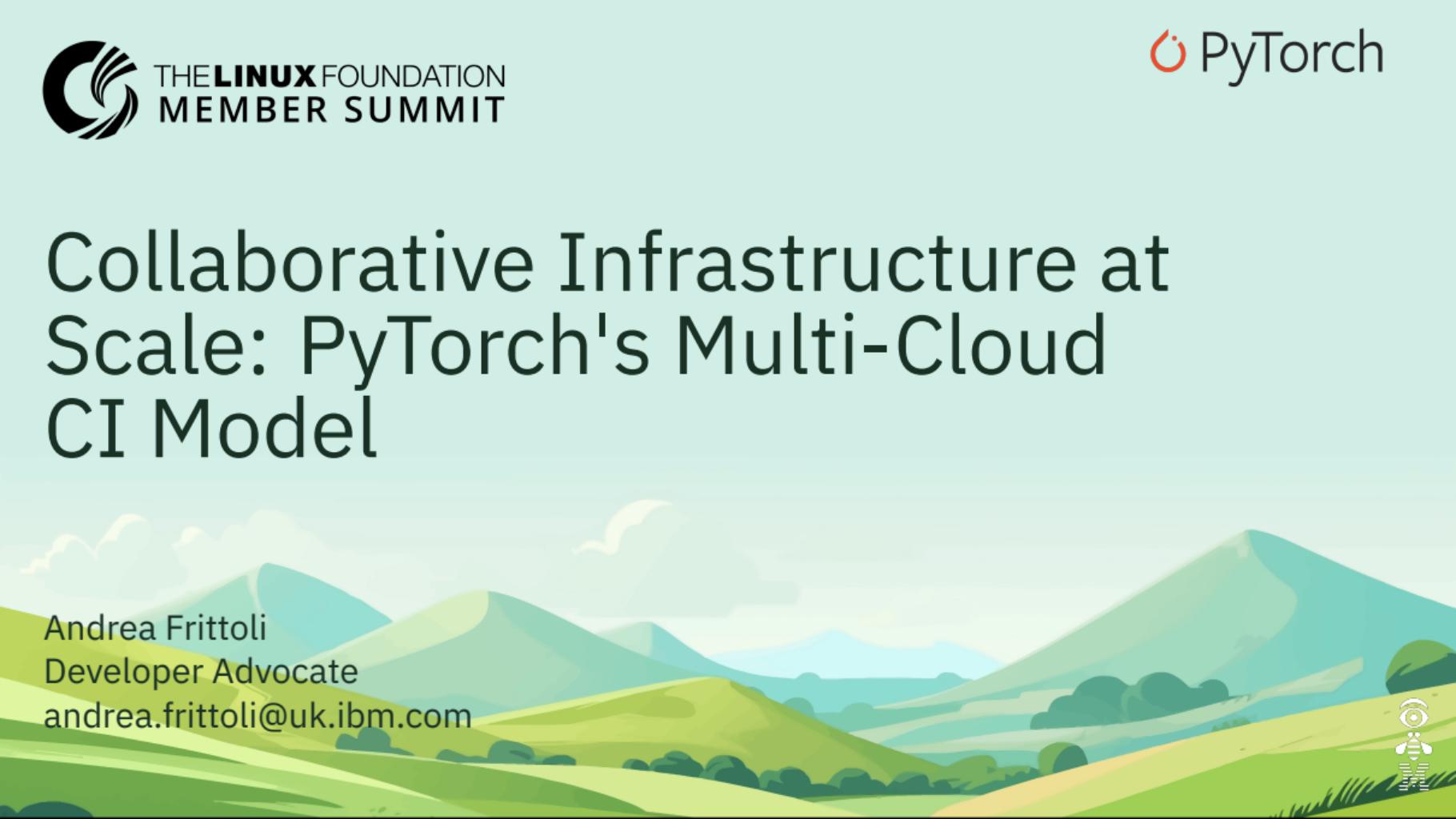


Collaborative Infrastructure at Scale: PyTorch's Multi-Cloud CI Model



Andrea Frittoli
Developer Advocate
andrea.frittoli@uk.ibm.com



Andrea Frittoli

⌚ afrittoli | 💬 andreafrittoli | 🗣 @blackchip76

- › Open Source Advocate @ IBM
- › Lives in Wales, enjoys the wind
- › Multi-Cloud CI Working Group Lead
- › Member of PyTorch Infra Working Group
- › Tekton, CDEvents maintainer





Contents

The Infrastructure Challenge

PyTorch Foundation

Governance Model

Technical Architecture

Results & Next Steps

Q&A



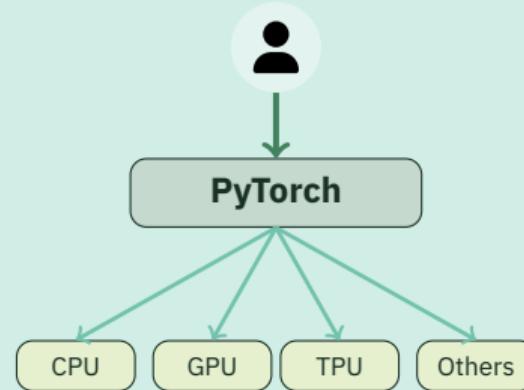
The Infrastructure Challenge



What is PyTorch?

A Leading Open Source ML Framework:

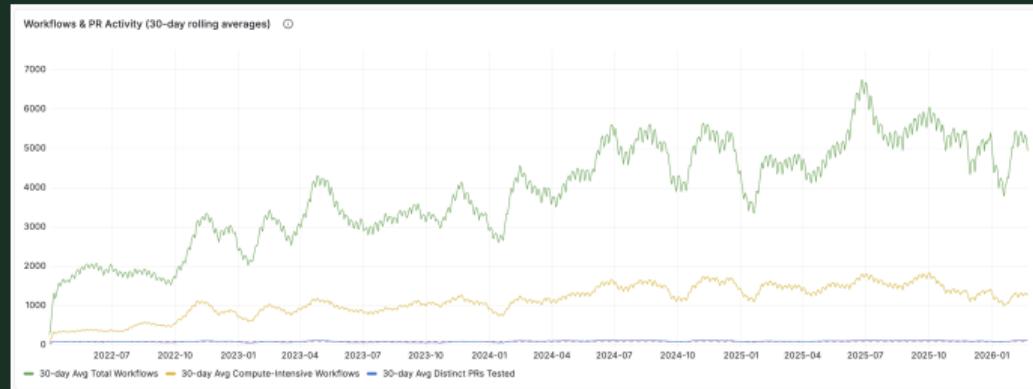
- › Researchers, data scientists, ML engineers
- › Research and production ML workloads
- › (Distributed) Training, LLM, RL, Inference
- › Python API, C++ Core (libtorch)
- › Eager and Graph Mode (Inductor)
- › Rapid growth and adoption across industry



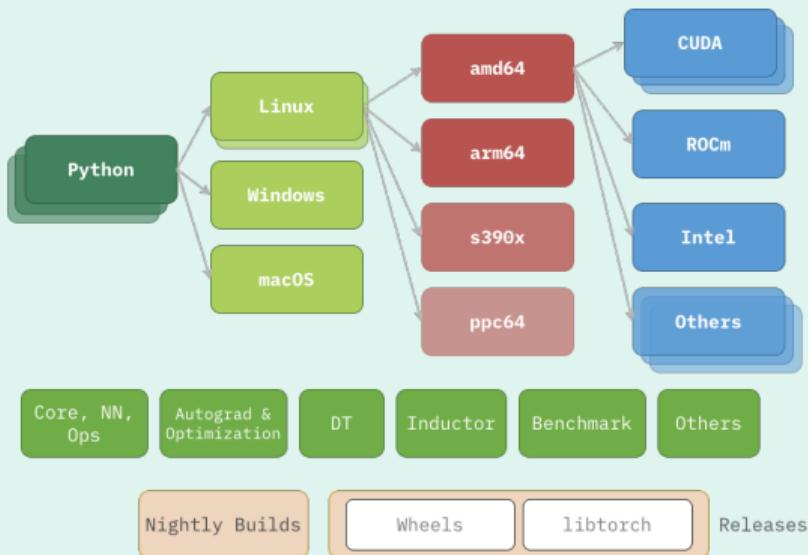
CI/CD Infrastructure Today

- › ~1M\$ monthly infrastructure costs
- › ~1M hours of compute per month
- › ~100 distinct PRs/day
- › ~1.3k Compute-intensive Workflows/day
- › Growing year over year

pytorch/pytorch repo only, numbers are estimated

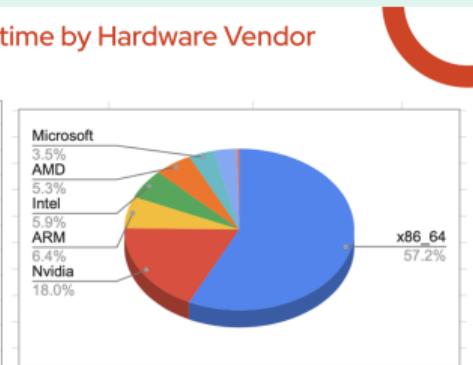


An Expanding Test Matrix



Combined Hardware Runtime by Hardware Vendor
September 2025

Combined Runtime By Hardware Vendor	
Vendor	Total
x86_64	834,085 hours
Nvidia	262,878 hours
ARM	93,000 hours
Intel	86,090 hours
AMD	77,517 hours
Microsoft	51,515 hours
Apple	46,812 hours
IBM	6,245 hours
N/A	0 hours
1,458,142 hours	



* Data from PT Foundation AWS Account + PyTorch HUD. This does not include self-hosted runner costs by member provided runners.



The Challenges

Community Challenges:

- › Maintain high-quality end-user experience
- › Preserve contributor workflow
- › Provide clear path for vendor engagement

Scale Challenges:

- › **Engineering:** Managing a Diverse Fleet, Access to Platforms
- › **Financial:** Reconcile Community Wishes with Budget Constraints

Security Challenges:

- › Trust and Consistency in a diverse environment
- › Central Administration
- › Trusted Builds

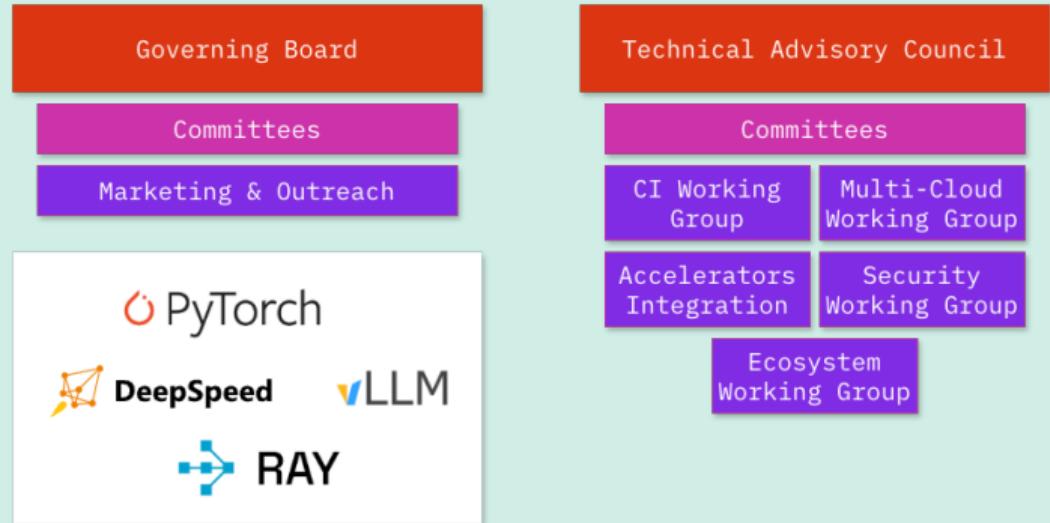


PyTorch Foundation & Working Groups



The PyTorch Foundation

- › Part of the Linux Foundation
- › Neutral home for PyTorch project
- › Hosts multiple projects
- › Thought Leadership (GB)
- › Technical Leadership (TAC)
- › Marketing & Outreach



Working Groups

Multi-Cloud CI

Cloud Agnostic Infrastructure:

- CI/CD jobs and Supporting Infra
- Metrics and Monitoring
- Fleet Management

Vendor-managed Runner Pools

CI Infra

Responsible for the CI/CD infra:

- Develop and maintain tools
- Operate the AWS/GitHub fleet
- Execute Releases

Accelerator Integration

Software Integration:

- Developer Guide
- Framework Improvement

Reference Test Framework

CI Event Relay Infrastructure

Security

- PyTorch Security Triage
- Tools and reports
- CI/CD Security



Governance Model



Photo by Khampha Phimmachak, CC0



Personas: A Balancing Act

End Users

- Stable software
- Available on my dev platform
- Available on my prod platform
- Following the industry at speed

Platform Vendors

- PyTorch for my platform
- Demonstrate platform compatibility
- Low bar to contribution of infrastructure
- Clear guidance and processes

Project Maintainers

- Project Success
- Stay relevant for end-users
- High quality secure builds
- Manageable CI/CD System (scale)
- Smooth CI/CD experience for contributors

PyTorch Foundation

- Ensure Vendor Neutrality
- Vendor participation encouraged
- Fair representation
- Financial sustainability



Integration Models

Three Ways to Contribute Infrastructure:

- › **Public Cloud Credits:** Financial contribution for shared infrastructure
- › **Vendor-Managed Runner Pools:** Tightly integrated with GitHub Actions
- › **Vendor-Managed Test Infrastructure:** Loosely integrated with GitHub

	Community	Scale	Security
Advantages	Flexible models accommodate different vendor capabilities	Outsourcing engineering with vendor-managed setups	More loosely coupled systems cannot compromise core CI system
Challenges	Hide the underlying infra complexity to contributors	Consistent tech stack required for public cloud credits. Consistent metrics and monitoring across the board.	Ensure security best practices on vendor managed infrastructure



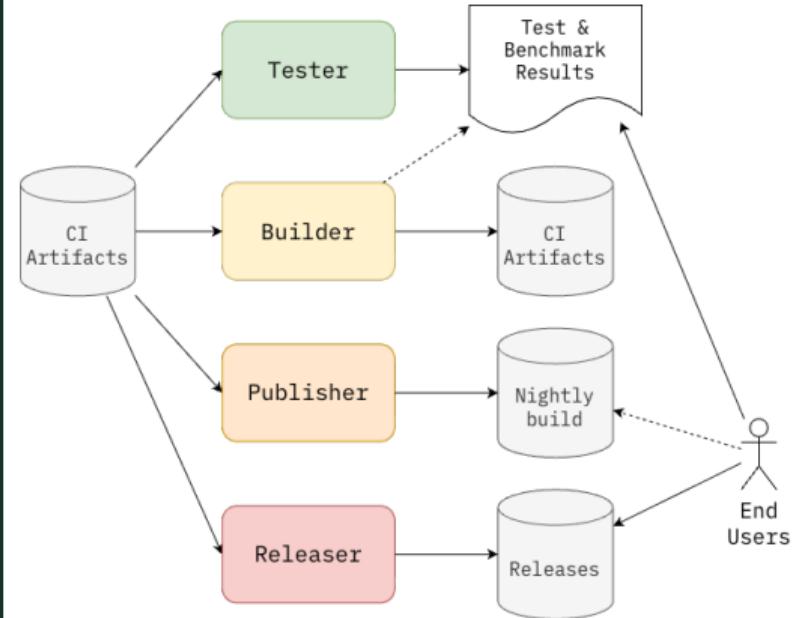
Roles & Requirements

Least privilege access to cloud resources:

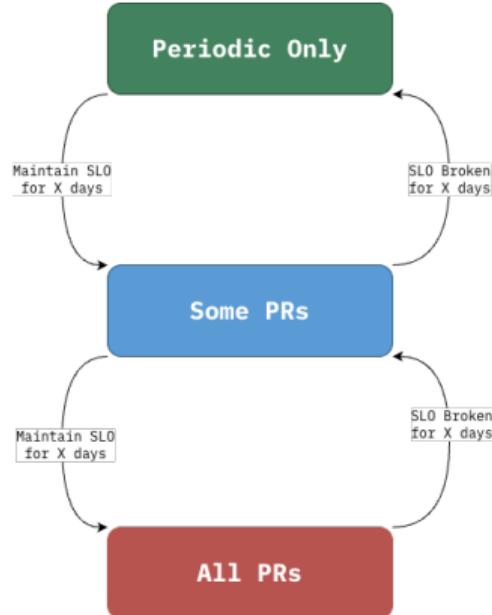
- › Jobs specify nodes access level required
- › Runners are assigned a role
- › Roles define the level of access to resources

Requirements for Runners:

- › Configuration
- › Provisioning
- › Monitoring
- › Security



Triggers & SLOs



Trigger levels for CI/CD jobs:

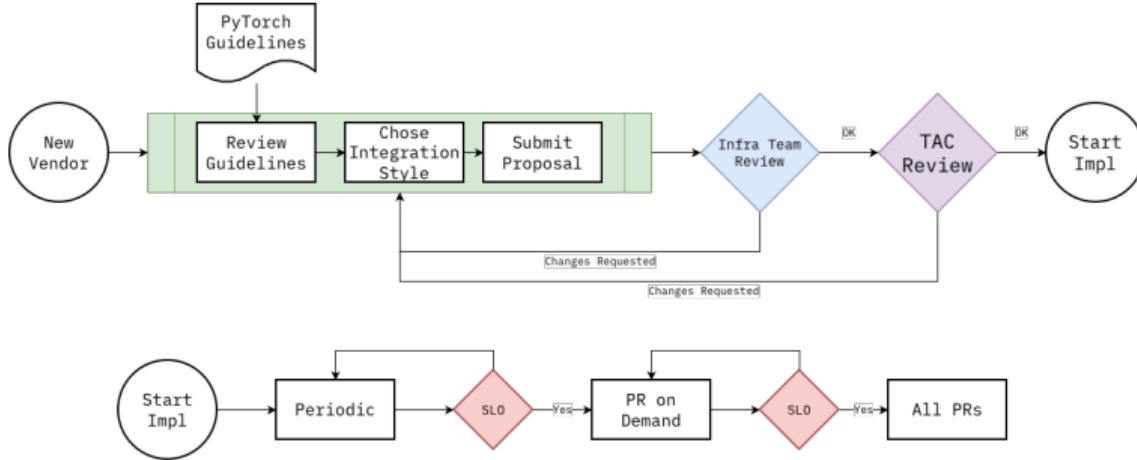
- On-demand only
- Periodic nightly
- Periodic multi-times per day
- Specific PRs only
- All PRs

Service Level Objectives:

- Availability %
- Reliability (Num of infra failures)
- Average Job Queue Time (p95/3 months)
- Engineering Commitment (on-call, slack, meetings)



Contribution Process



- › Working Group provides guidance and recommendations
- › Infra team verify checks for compliance
- › TAC Community for final approval
- › Transparent process with clear requirements
- › Monitoring for SLOs



Central Administration & Visibility

Maintaining Manageability at Scale:

- › Common software stack across clouds
Reusable by vendors too
- › Self-service Tools with Central Admin Override
- › Vendor playbooks, reusable IaaC

Visibility & Control:

- › Public Dashboards: Transparency for community and vendors
- › Cost Tracking: Attribution per vendor and workload type
- › Performance Metrics: Build times, success rates, SLA compliance

A screenshot of a GitHub Actions dashboard for the pytorch/pytorch repository. The interface shows a timeline of recent pull requests (PRs) with columns for Time, SHA, Commit, PR, Author, and a detailed view pane. The timeline spans from 12:39 pm on Jan 26 to 8:46 am on Jan 27, 2024. The commits are color-coded by author.

Time	SHA	Commit	PR	Author
12:39 pm	F77D8E0	[reverted] Implement RA reader in clang...	#172801	cd8-mira
0:54 am	4E7B0C0	add flitlock: Fix build error due to missing...	#172802	cd8-mira
0:55 am	2D9E8F0	[reverted] Add a request function to XML to...	#172803	pranjaygupta
9:13 am	3951D90	[reverted] Update both npy-arrays comment for #1...	#172804	pranjaygupta
9:13 am	5B85A00	[reverted] Implement RA reader in clang...	#172805	cd8-mira
1:17 pm	4E7B0C0	[reverted] Add a request function to XML to...	#172806	cd8-mira
5:40 am	2D9E8F0	[reverted] Implement RA reader in clang...	#172807	cd8-mira
5:20 am	4E7B0C0	[reverted] Add a request function to XML to...	#172808	cd8-mira
5:20 am	2D9E8F0	[reverted] Implement RA reader in clang...	#172809	cd8-mira
8:28 am	32128E0	[reverted] [WIP] Implement RA reader in clang...	#172810	cd8-mira
8:46 am	8e4a190	[X] Change kxx.c test with long time exp...	#172809	cd8-mira

A screenshot of the GitHub Actions Runner Service dashboard. It displays a summary card with the following counts: Total Runners (4), Active (0), Offline (4), and Pending (0). Below this, there's a section for Recent Activity showing three log entries related to runner creation and configuration.

Total Runners	Active	Offline	Pending
4	0	4	0

Recent Activity
batch_create_runner by cd8-mira@github.org created 2 users Jan 26, 2024 at 9:39 PM
batch_update_runner by cd8-mira@github.org deleted 2 users Jan 26, 2024 at 9:40 PM
label_policy_update by cd8-mira@github.org updated Jan 26, 2024 at 9:40 PM

A screenshot of the GitHub Actions Runner Service security events dashboard. It lists recent events categorized by Event Type (All Types), Severity (All Severities), and User Identity. The events shown are all of high severity and involve actions like batch_create_runner and batch_delete_runner.

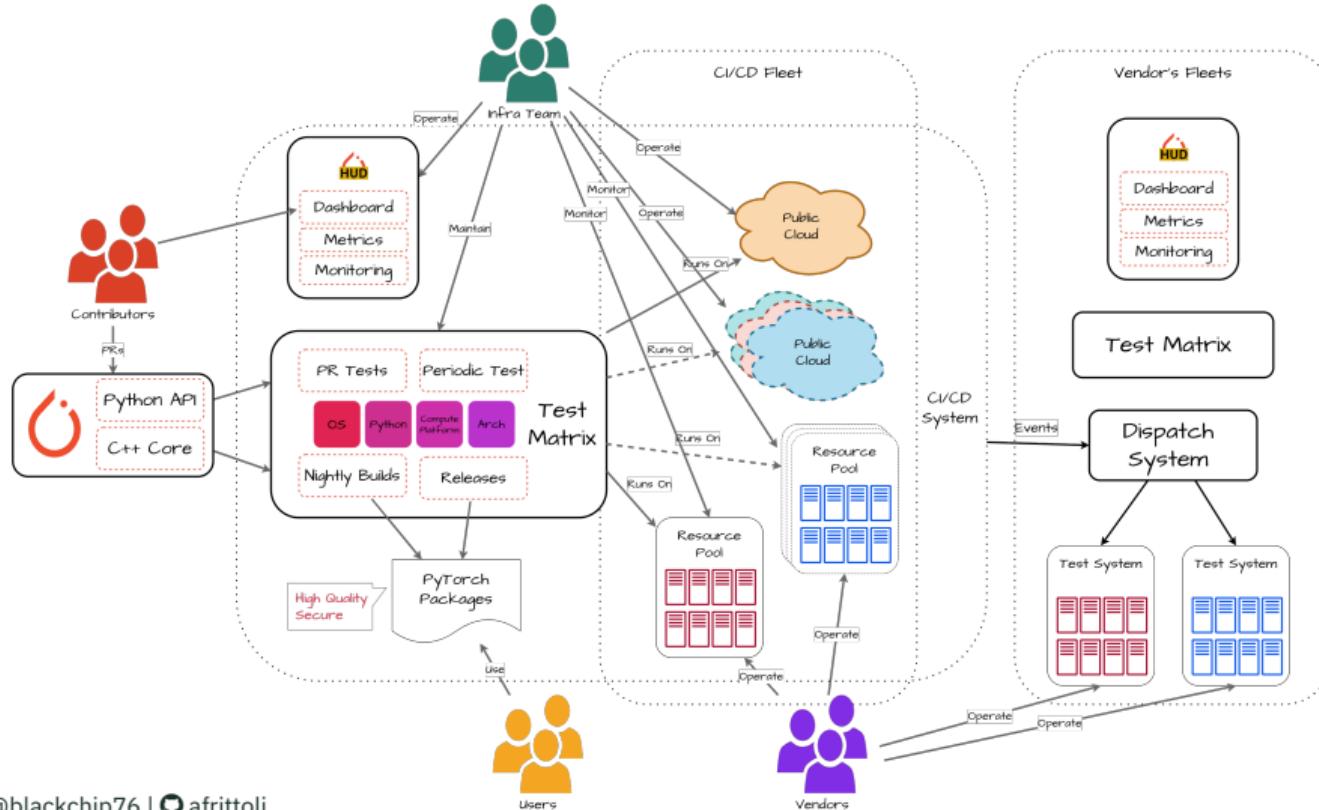
Event Type	Severity	User Identity	Timestamp	Action	Target	Details
batch_create_runner	high	cd8-mira@github.org	Jan 26, 2024 at 9:39 PM	-	-	Details
batch_delete_runner	high	cd8-mira@github.org	Jan 26, 2024 at 9:40 PM	-	-	Details
label_policy_update	medium	alice@github.org	Jan 26, 2024 at 9:40 PM	alice-number-dashc12	-	Details
batch_delete_runner	high	cd8-mira@github.org	Jan 26, 2024 at 9:40 PM	-	-	Details



Technical Architecture



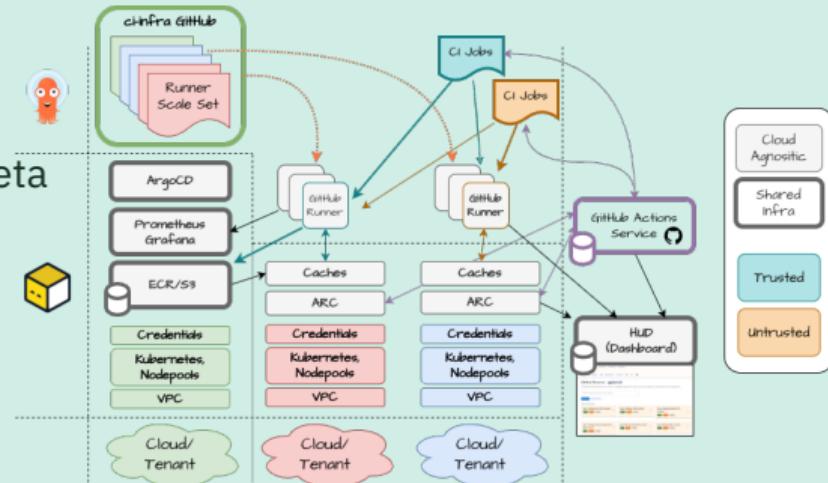
High Level Architecture



Public Cloud Infrastructure

Current AWS Setup:

- > Primary infrastructure hosted on AWS
- > Two accounts, funded by AWS Credits and Meta
- > Self-hosted GitHub Actions runners
- > Managed by PyTorch Infra team
- > Autoscaling based on workload demand



Multi-Cloud & Infra Working Group Initiatives:

- > **Portable CI Jobs:** Run in a container, remove cloud-specific assumptions
- > **Reusable Autoscaler:** Developing portable autoscale based on ARC
- > **Infrastructure as Code:** Reusable OpenTofu modules



Vendor-Managed Runner Pools

Challenges:

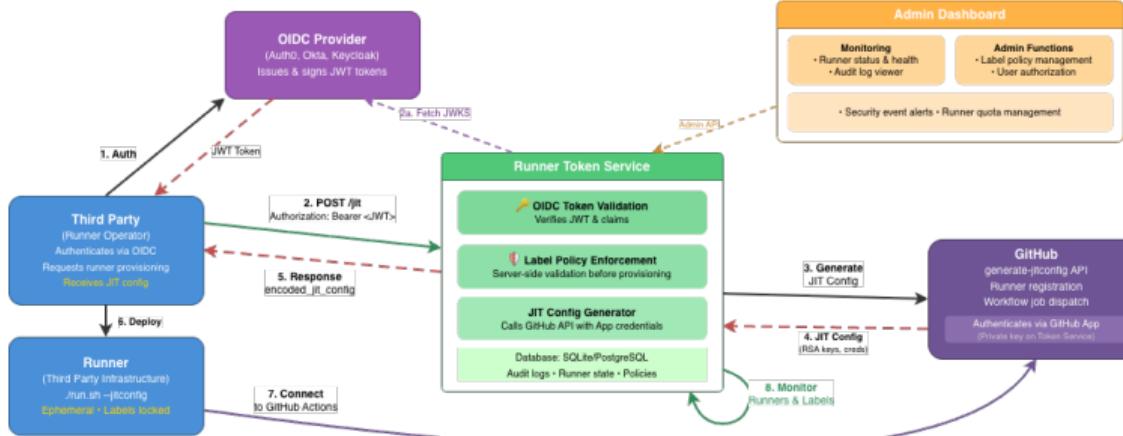
- Long lived admin credentials
- Runner metadata cannot be enforced
- Lack of central administration
- Lack of auditability

Proposed Solution:

- GHA Runner Token Service (GHARTS)
- Use LFID for authentication
- No Admin Credentials to vendors
- Ephemeral, non reusable credentials
- Enforcement of metadata and quota
- Centralized administration
- Auditability



GHARTS Provisioning Flow



- Vendor authenticates using existing credentials (LFID)
- Vendor requests a JIT config using the JWT
- GHARTS verifies AuthN and AuthZ for vendor
- GHARTS requests JIT config from GitHub
- Vendor uses JIT config to provision a runner
- JIT lasts 1h, and can only be used Once



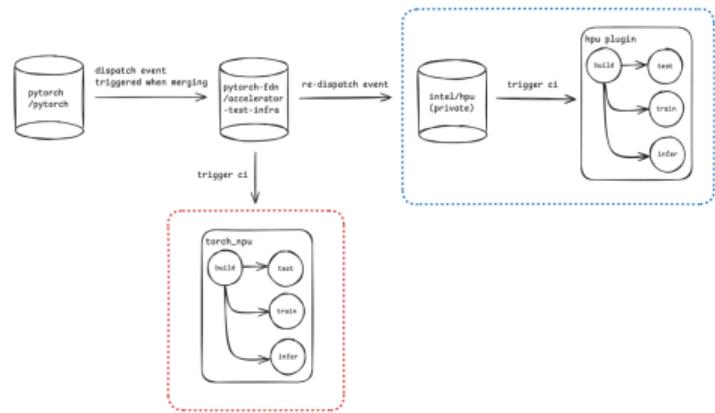
Vendor-Managed Test Systems

Accelerator Integration Working Group Initiative:

- Loosely coupled with PyTorch CI
- Events relayed to external repository
- Vendors provide test infrastructure and environments
- Out-of-tree test workflows

Use Cases:

- Specialized hardware testing (TPUs, custom accelerators)



Old Issues, New Issues

PyTorch Queue Time Analysis

* All datetime values are in UTC. Local: 2026-02-24 21:36:29, UTC Time: 2026-02-25 05:36:29

* Data is collected every 30 minutes, including all jobs in queue at that time. ⓘ

Chart Type



New Challenges:

- > Multiple routes to Integration
- > Address several challenges, but
- > Creates extra complexity

Ongoing Work to Solve:

- > Collaboration across working groups
- > Same visibility of test results for all routes
- > Consistent metrics
- > Single administrative interface

max queue time line

avg queue time line

1.2d

13.9h

Time Range Last 3 Days Granularity Half Hour

Search Category

workflow name

By default, shows data for all queued jobs. Using filter and checkbox below for specific items.

Select items:

Filter

Select All Unselect All

- pull
- inductor
- operator_microbenchmark
- inductor-A100-perf-nightly
- inductor-perf-nightly-h100
- windows-binary-libtorch-debug
- trunk
- linux-aarch64-binary-manywheel
- periodic
- inductor-unitest
- windows-arm64-binary-wheel
- trunk-rocm-sandbox
- windows-binary-wheel

Search

Click to apply filter changes



Results & Lessons Learned



Platform Coverage Expansion

Achievements:

- › **Public Cloud Credits:** AWS, Meta funding
- › **Runner Pools:** IBM (via GHARTS), additional vendors onboarding
- › **Vendor Systems:** AMD, Intel, NVIDIA, ARM for specialized accelerator testing
- › Significant increase in platform coverage across CPU architectures and accelerators
- › Reduced time-to-market for new platform support
- › Improved test reliability through dedicated infrastructure



Vendor Onboarding Experience

Positive Feedback:

- › Clear expectations and requirements documentation
- › Reasonable onboarding timeline with good support
- › Responsive technical support from maintainers
- › Fair and transparent governance model

Ongoing Initiatives:

- › IBM trialing GHARTS for runner pool integration
- › Red Hat collaborating on vendor system visibility improvements
- › Vendors actively sharing experiences and best practices



Key Lessons Learned

Principles for Collaborative Infrastructure:

- › **Fair Governance:** Protect everyone's interests, vendors can't veto but have voice
- › **Engage Vendors Early:** Let them help build solutions, not just consume them
- › **Empower Maintainers:** Give infra team authority and tools to manage at scale
- › **Security First:** Build security in from day one, not as afterthought
- › **Flexible Models:** Multiple contribution paths reduce friction and attrition

Challenges:

- › Balancing vendor needs with project sustainability
- › Standardization across diverse platforms and requirements



Applicability to Other LF Projects

Shared Challenges:

- › Accepting vendor infrastructure
- › Maintaining project neutrality
- › Ensuring transparency
- › Achieving financial sustainability

This Model Can Help:

- › Proven governance framework
- › Technical architecture patterns
- › Onboarding playbook



Future Directions

Ongoing Activities:

- › Complete GHARTS rollout with IBM and additional vendors
- › Enhance monitoring and observability across all integration models
- › Expand portable CI/CD tooling for multi-cloud deployments
- › Standardize event streaming with OpenTelemetry/CDEvents

Long-term Goals:

- › Share governance and technical patterns with other LF projects
- › Build community of practice around collaborative infrastructure
- › Continuously refine based on vendor and maintainer feedback



Key Takeaways

- › **Sustainable Scaling:** Distribute infrastructure costs while maintaining project control
- › **Governance Matters:** Fair rules enable vendor participation without compromising autonomy
- › **Security & Transparency:** Non-negotiable foundations that build trust
- › **Flexible Models:** Multiple contribution paths reduce friction and increase adoption
- › **Reusable Pattern:** This model can help other Linux Foundation projects





Questions?



References & Contact

Resources:

- › PyTorch Foundation: <https://pytorch.org/foundation>
- › PyTorch CI/CD: <https://github.com/pytorch/pytorch/wiki/CI>
- › GitHub: <https://github.com/pytorch/pytorch>
- › Multi-Cloud Infrastructure WG: PyTorch Slack #infra-wg

Contact:

- › Andrea Frittoli
- › andrea.frittoli@uk.ibm.com
- ›  @blackchip76 |  afrittoli

