

E4S: The Extreme-scale Scientific Software Stack

Release 25.11

Release 25.11 notes
November 14, 2025



High Performance Software Foundation
E4S Team
<https://e4s.io>



U.S. DEPARTMENT OF
ENERGY

Office of
Science

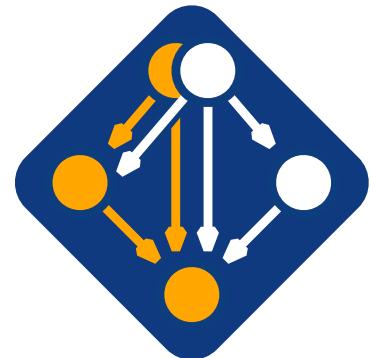
E4S 25.11: What's New?



<https://e4s.io>

- E4S includes 125+ HPC-AI packages on aarch64, x86_64, and ppc64le platforms.
- All new website [<https://e4s.io>] with an OpenAI based chatbot to simplify access to E4S documentation.
- Support for NVIDIA Blackwell on x86_64 and aarch64 (Grace-Blackwell) architectures.
- Support for Rocky Linux 9.6 with Hopper and Blackwell (x86_64 and aarch64), Ubuntu 24.04 LTS.
- Spack 1.0.2 [<https://spack.io>] integration.
- All new E4S Spack build cache [<https://cache.e4s.io/25.11>] with over 7500 optimized binaries.
- AI software stack with Python 3.12.11 including packages like NVIDIA BioNeMo™, NVIDIA NeMo™, Google Agent Development Kit (adk), Vllm, HuggingFace CLI, TensorFlow, PyTorch, Google.genai (Gemini API), OpenAI (API), TorchBraid, Pandas, Scikit-Learn, JAX, OpenCV, LBANN and Codium, Jupyter, and Marimo notebooks.
- HPC Applications include: CP2K, DealII, FFTX, GROMACS, LAMMPS, Nek500, Nekbone, NWChem, OpenFOAM, WarpX, WRF, Quantum Espresso, and Xyce with GPU support where available.
- CUDA upgraded to 12.9 (aarch64, x86_64), ROCm upgraded to 6.4.3, oneAPI upgraded to 2025.2.
- Adaptive Computing's Heidi web-based platform for multi-user, multi-node, ParaTools Pro for E4S™ cloud images on AWS, Microsoft Azure, Google Cloud, IBM Cloud, and OCI with NVIDIA GPUs with SLURM or Torque.

E4S Spack Integration



- E4S is a curated, Spack based distribution of HPC-AI software.
- Major Changes in Spack 1.0.2
 - Compilers as First-Class Dependencies: Compilers are now treated as proper dependencies in the concretization process, leading to clearer and more reproducible environments.
 - Stable Package API: Spack 1.0 introduces a stable API for package development, improving long-term maintainability and easing contributions.
 - Concurrent Builds: Builds can now run concurrently, leveraging parallel jobs and increasing throughput on multi-core machines.
 - Updated Install Tree Layout: The default install tree format is revamped for better organization and reproducibility.
 - Content-Addressed Build Caches: Binary caches now use content-based addressing, improving the reliability and provenance of shared binaries.
 - Improved Git Provenance: Enhanced mirroring and fetching mechanisms for package sources and dependencies.

Visit spack.io

 github.com/spack/spack



E4S: Extreme-scale Scientific Software Stack



<https://e4s.io>

About E4S

- E4S is an **ecosystem for science** and a community effort to provide open-source software packages for developing, deploying and running scientific applications on HPC platforms.
- E4S has built a comprehensive, coherent software stack that enables application developers to productively develop highly parallel applications that effectively target diverse exascale architectures.
- E4S provides a curated, Spack based software distribution of 125+ HPC (TAU, Trilinos, PETSc, OpenFOAM, Gromacs, Nek5000, LAMMPS), EDA (e.g., Xyce), and AI/ML packages (e.g., Google ADK, NVIDIA NeMo™, NVIDIA BioNeMo™, Vllm, HuggingFace CLI, TensorFlow, PyTorch, OpenCV, TorchBraid, Scikit-Learn, Pandas, JAX, LBANN optimized for GPUs where available).
- Base images and full featured containers (with GPU support) and DOE LLVM containers.
- Commercial support for E4S through ParaTools, Inc. for installation, maintaining an issue tracker, and ECP AD engagement.
- E4S for clouds: Adaptive Computing's Heidi with ParaTools Pro for E4S™ image for **AWS, GCP, IBM Cloud, Azure, OCI**.
- With E4S Spack binary build caches, E4S supports both bare-metal and containerized deployment for GPU based platforms.
 - x86_64, ppc64le (IBM Power 10), aarch64 (ARM64) with support for CPUs and GPUs from NVIDIA, AMD, and Intel
 - Container images on DockerHub and E4S website of pre-built binaries of ECP ST products.
- e4s-chain-spack.sh to chain two Spack instances allows us to install new packages in home directory and use other tools.
- e4s-cl container launch tool allows binary distribution of applications by swapping MPI in the containerized app w/ system MPI.
- e4s-alc is an à la carte tool to customize container images by adding system and Spack packages to an existing image.
- E4S 25.11 released on November 14, 2025: https://e4s.io/talks/E4S_25.11.pdf



Updated E4S website <https://e4s.io>

The screenshot shows a web browser window displaying the E4S website at <https://e4s.io>. The page has a light gray header bar with the E4S logo and the text "E4S - An HPC-AI Software Ecosystem for Science" followed by "Advancing Science through Trusted Software". To the right are links for "Use E4S", "Join E4S", "About E4S", and "Contact Us". The main content area has a blue background featuring a stylized white "E4S" logo. The title "E4S: HPC-AI Software Ecosystem for Science" is centered in large white font. Below it is the subtitle "Performance-portable libraries & tools for advanced computing". There are two buttons: "News & Events" and "Get E4S". The main text on the page welcomes visitors to E4S, describing it as an open-source, community-driven collection of high-quality HPC and AI libraries and tools. It highlights its role as an "Extreme-scale Scientific Software Stack" for developing, deploying, and running applications across various architectures. The text also mentions support from the U.S. Department of Energy (DOE) and its partners. Below this, there are four white rectangular boxes with rounded corners, each containing a blue icon and text: "Performance Portability" (square icon), "Curated Ecosystem" (list icon), "Reproducible Environments" (sun icon), and "Continuous Validation" (checkmark icon). Each box also contains a brief description of the feature.

E4S - An HPC-AI Software Ecosystem for Science
Advancing Science through Trusted Software

Use E4S Join E4S About E4S Contact Us

E4S: HPC-AI Software Ecosystem for Science

Performance-portable libraries & tools for advanced computing

[News & Events](#) [Get E4S](#)

Welcome to **E4S**, the *HPC-AI Software Ecosystem for Science* — an open-source, community-driven collection of high-quality HPC and AI libraries and tools. E4S is an **Extreme-scale Scientific Software Stack** that enables scientists, developers, and institutions to develop, deploy, and run performance-portable applications across CPUs and GPUs from NVIDIA, AMD, Intel, and Arm. Ready to adapt to new architectures as they emerge.

Supported by the **U.S. Department of Energy (DOE)** and its partners, E4S accelerates scientific innovation on systems ranging from laptops to exascale supercomputers.

Performance Portability
Run anywhere—from laptops to exascale supercomputers using modern GPU and CPU architectures.

Curated Ecosystem
Over 120 interoperable libraries and tools for HPC and AI workflows, validated for research and production use.

Reproducible Environments
Deploy via Spack, pre-built containers for Docker, Singularity, and commercial cloud environments.

Continuous Validation
E4S packages are tested across DOE leadership and cloud platforms for quality and reproducibility.

Chatbot integration in E4S Website

The screenshot shows the E4S website interface. At the top, there's a navigation bar with icons for search, refresh, and user profile. The URL bar shows <https://e4s.io>. Below the header, there are four main features: "Performance Portability", "Curated Ecosystem", "Reproducible Environments", and "Continuous Validation". Each feature has a brief description and a small icon. Below these, there are two large call-to-action boxes: "Chat with the E4S Bot" and "Visit the E4S Product Catalog". Each box contains a speech bubble icon, a title, a description, and a blue button. Further down, there's a section titled "Trusted by the HPC-AI Community" with a paragraph of text and a "PESO" logo at the bottom left.

Performance Portability
Run anywhere—from laptops to exascale supercomputers using modern GPU and CPU architectures.

Curated Ecosystem
Over 120 interoperable libraries and tools for HPC and AI workflows, validated for research and production use.

Reproducible Environments
Deploy via Spack, pre-built containers for Docker, Singularity, and commercial cloud environments.

Continuous Validation
E4S packages are tested across DOE leadership and cloud platforms for quality and reproducibility.

Chat with the E4S Bot
Ask questions, explore tools, or get recommendations from the E4S Guide Bot.

[Chat with the E4S Guide Bot](#)

Visit the E4S Product Catalog
Browse member products, product families, with extensive links, via the E4S Product Catalog

[Browse the E4S Product Catalog](#)

Trusted by the HPC-AI Community

E4S is developed and supported by contributors from U.S. national laboratories, universities, and industry partners. It is available as both open-source and commercial distributions such as [ParaTools Pro for E4S™](#) for commercial cloud platforms such as AWS, Azure, GCP, OCI, and IBM Cloud.

© 2025 E4S Project, a Series of LF Projects, LLC. For website terms of use, trademark policy and other project policies please see [lfprojects.org](#).
Supported by the U.S. Department of Energy Office of Advanced Scientific Computing Research.

PESO

Chatbot integration in E4S Website

The screenshot shows a web browser window with the URL <https://chatgpt.com/g/g-69010fb294b0819195727cf1a7ccc792-software-ecosystem-for-science-guide>. The page title is "Software Ecosystem for Science Guide 5.1". On the left, there is a sidebar with various options: "New chat", "Search chats", "Library", "Atlas", "Projects", "GPTs", "Explore", and "Software Ecosystem for Scienc...". Below these are sections for "Chats", "Profiling tools in E4S", "E4S libraries for PDE solvers", and "Example chat: Ask anything". The main content area features a circular icon with a cube symbol and the title "Software Ecosystem for Science Guide" by Michael Heroux. It describes the guide as an expert on scientific libraries and tools in the E4S ecosystem. Below the title are four questions: "Which E4S libraries should I use for parallel PDE...", "How do I install and configure PETSc with Spack?", "Can you explain how Kokkos interacts with Trilinos?", and "What profiling tools in E4S help with performance...". At the bottom, there is a "Ask anything" input field with a microphone icon and a "Upgrade" button. The PESO logo is visible in the bottom left corner.

Software Ecosystem for Science Guide 5.1

New chat

Search chats

Library

Atlas

Projects

GPTs

Explore

Software Ecosystem for Scienc...

Chats

Profiling tools in E4S

E4S libraries for PDE solvers

Example chat: Ask anything

By Michael Heroux

Expert on scientific libraries and tools in the E4S (Ecosystem for Science) ecosystem.

Which E4S libraries should I use for parallel PDE...

How do I install and configure PETSc with Spack?

Can you explain how Kokkos interacts with Trilinos?

What profiling tools in E4S help with performance...

Ask anything

Sameer Shende
Free

Upgrade

PESO

ss

7

E4S Bot trained on E4S content on OpenAI's ChatGPT

The screenshot shows a web browser window with the URL <https://chatgpt.com/g/g-69010fb294b0819195727cf1a7ccc792-software-ecosystem-for-science-guide/c/691571>. The page title is "Software Ecosystem for Science Guide 5.1". The main content is a question: "What profiling tools in E4S help with performance optimization?". Below the question, a block of text states: "E4S provides a comprehensive suite of profiling and performance analysis tools that help scientists and developers understand and optimize performance on heterogeneous HPC systems. According to the *E4S Performance Tools Report*, the key profiling tools are:"

1. TAU (Tuning and Analysis Utilities)

- Role:** The most versatile profiler/tracer in E4S; known as the "Swiss Army knife" of HPC performance tools.
- Instrumentation methods:**
 - `tau_exec` runtime interception — no recompilation required.
 - Source instrumentation (PDT)** — automatically inserts timers into C/C++/Fortran code.
 - Compiler-based hooks** — uses compiler flags like `-finstrument-functions`.
- Capabilities:**
 - Profiling (summary of where time is spent).
 - Tracing (timeline of events across processes and GPUs).
 - GPU kernel timing (via CUDA/HIP/oneAPI).
 - Memory, communication, and I/O profiling.
- Integration:** Works with PAPI for hardware counters and Score-P for trace collection.

At the bottom of the page, there is a "Ask anything" button and a note: "New version of GPT available - Continue chatting to use the old version, or start a [new chat](#) for the latest version."

The left sidebar of the browser window shows a navigation menu with the following items:

- New chat
- Search chats
- Library
- Atlas
- Projects
- GPTs
 - Explore
 - Software Ecosystem for Scienc...
- Chats
 - Profiling tools in E4S
 - E4S libraries for PDE solvers
 - Example chat: Ask anything

In the bottom left corner of the browser window, there is a logo for "PESO" and a user profile for "Sameer Shende Free". There is also an "Upgrade" button.

E4S Bot

The screenshot shows a web browser window with the URL `chatgpt.com`. The page title is "Launch GPT-OSS 120B E4S". On the left, there is a sidebar with various options like "New chat", "Search chats", "Library", "Atlas", "Projects", "GPTs", "Explore", and "Software Ecosystem for Scien...". Below this is a "Chats" section with items: "Launch GPT-OSS 120B E4S" (which is highlighted), "Profiling tools in E4S", "E4S libraries for PDE solvers", and "Example chat: Ask anything". At the bottom of the sidebar, it says "Sameer Shende Free" and has an "Upgrade" button.

In the main content area, a message box contains the question: "How can I launch OpenAI's 120B parameter OSS model using E4S?". Below this, a note says "Thought for 16s >".

Quick steps — launch OpenAI's gpt-oss-120b on E4S with vLLM

Short answer: since recent E4S releases include vLLM, you can run the gpt-oss 120B model directly with vLLM's `serve` command on a machine in your E4S environment (recommended: single 80 GB GPU like an H100 / A100-80GB or a multi-GPU setup). The cookbook from OpenAI gives a one-line server command and examples for calling it with the OpenAI-compatible API. OACISS +1

1) Preconditions (hardware & E4S)

- **GPU:** gpt-oss-120b is intended for $\geq 60-80$ GB VRAM (fits on a single H100 / A100-80GB or multi-GPU). If you only have smaller GPUs you'll need model parallel / multi-GPU setup or run the 20B model instead. OpenAI Cookbook +1
- **E4S image:** E4S 25.06+ lists vLLM in the stack — meaning vLLM may already be in the image or available as a module/container. If `vllm` is already on `PATH` inside your E4S environment, you can skip pip installs. OACISS

2) Minimal, working flow (commands you can run right now in an E4S compute node)

If `vllm` is in the E4S image and you have an appropriate GPU:

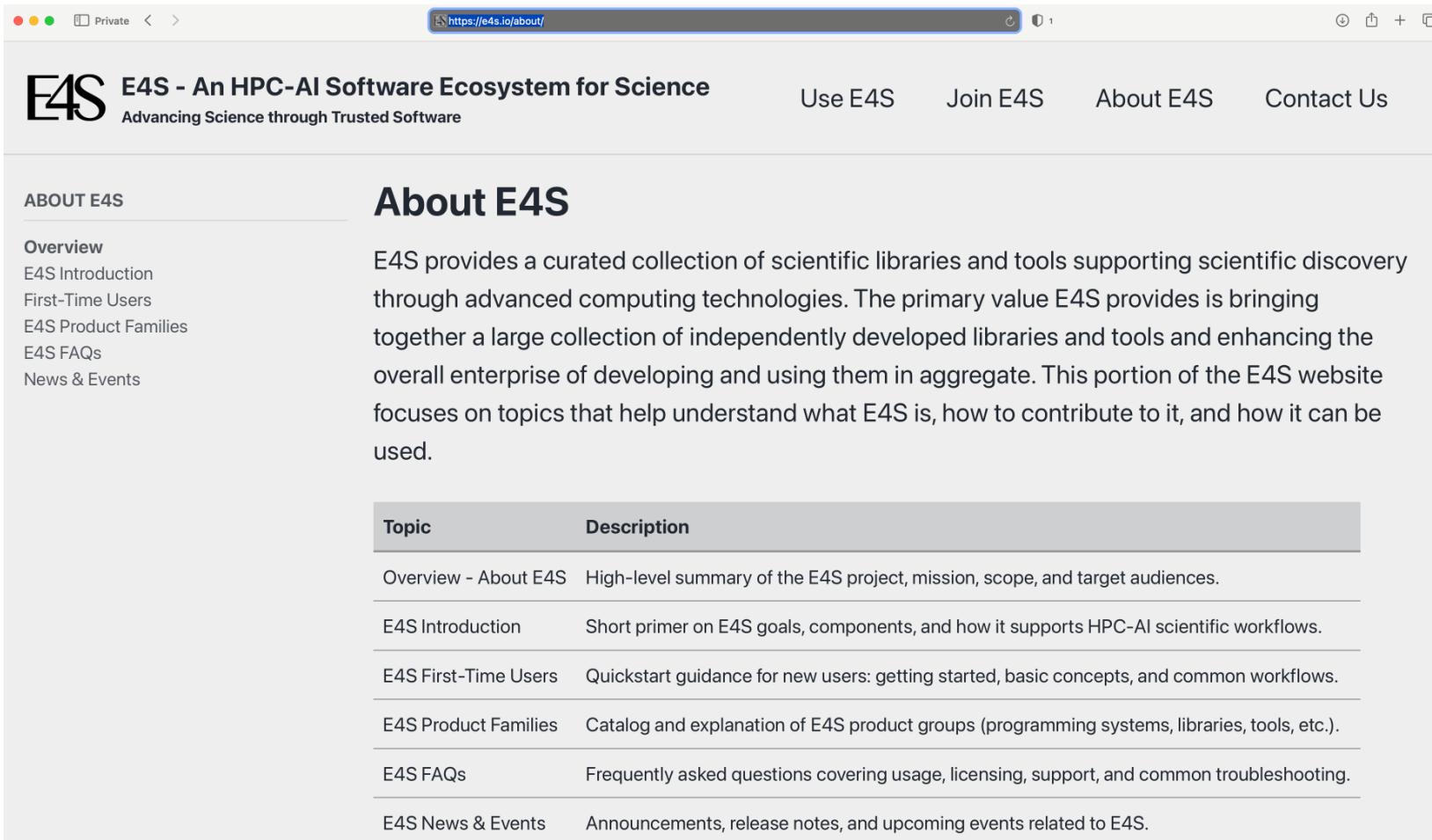
```
bash
# make sure you are on the node with the target GPU (and modules/drivers loaded by you
# then just:
vllm serve openai/gpt-oss-120b
```

That will (by default) download the model from Hugging Face and spin up an OpenAI-compatible server at `http://localhost:8000/v1`. Example client call (Python OpenAI SDK pointed at your local server):

+ Ask anything

ChatGPT can make mistakes. Check important info.

E4S Documentation



The screenshot shows a web browser window displaying the E4S documentation website at <https://e4s.io/about/>. The page has a header with the E4S logo and the text "E4S - An HPC-AI Software Ecosystem for Science" and "Advancing Science through Trusted Software". The header also includes links for "Use E4S", "Join E4S", "About E4S", and "Contact Us". On the left, there's a sidebar titled "ABOUT E4S" with links to "Overview", "E4S Introduction", "First-Time Users", "E4S Product Families", "E4S FAQs", and "News & Events". The main content area is titled "About E4S" and contains a paragraph about the project's mission and value. Below this is a table with rows for each topic listed in the sidebar.

Topic	Description
Overview - About E4S	High-level summary of the E4S project, mission, scope, and target audiences.
E4S Introduction	Short primer on E4S goals, components, and how it supports HPC-AI scientific workflows.
E4S First-Time Users	Quickstart guidance for new users: getting started, basic concepts, and common workflows.
E4S Product Families	Catalog and explanation of E4S product groups (programming systems, libraries, tools, etc.).
E4S FAQs	Frequently asked questions covering usage, licensing, support, and common troubleshooting.
E4S News & Events	Announcements, release notes, and upcoming events related to E4S.



E4S Product Catalog

The screenshot shows a web browser displaying the E4S Product Catalog. The page has a header with the E4S logo and navigation links for "Use E4S", "Join E4S", "About E4S", and "Contact Us". On the left, there's a sidebar titled "USE E4S" with links to "Overview", "E4S Guide Bot", "E4S Documentation", "E4S Product Catalog", "Finding and Installing with Spack", "E4S-Specific Spack Installation", "E4S Container Download", "E4S Container Installation", "E4S Container Launch", "Test with E4S", and "ParaTools Pro for E4S™". The main content area is titled "E4S Product Catalog" and contains a paragraph about the catalog, a search bar, and a table of products.

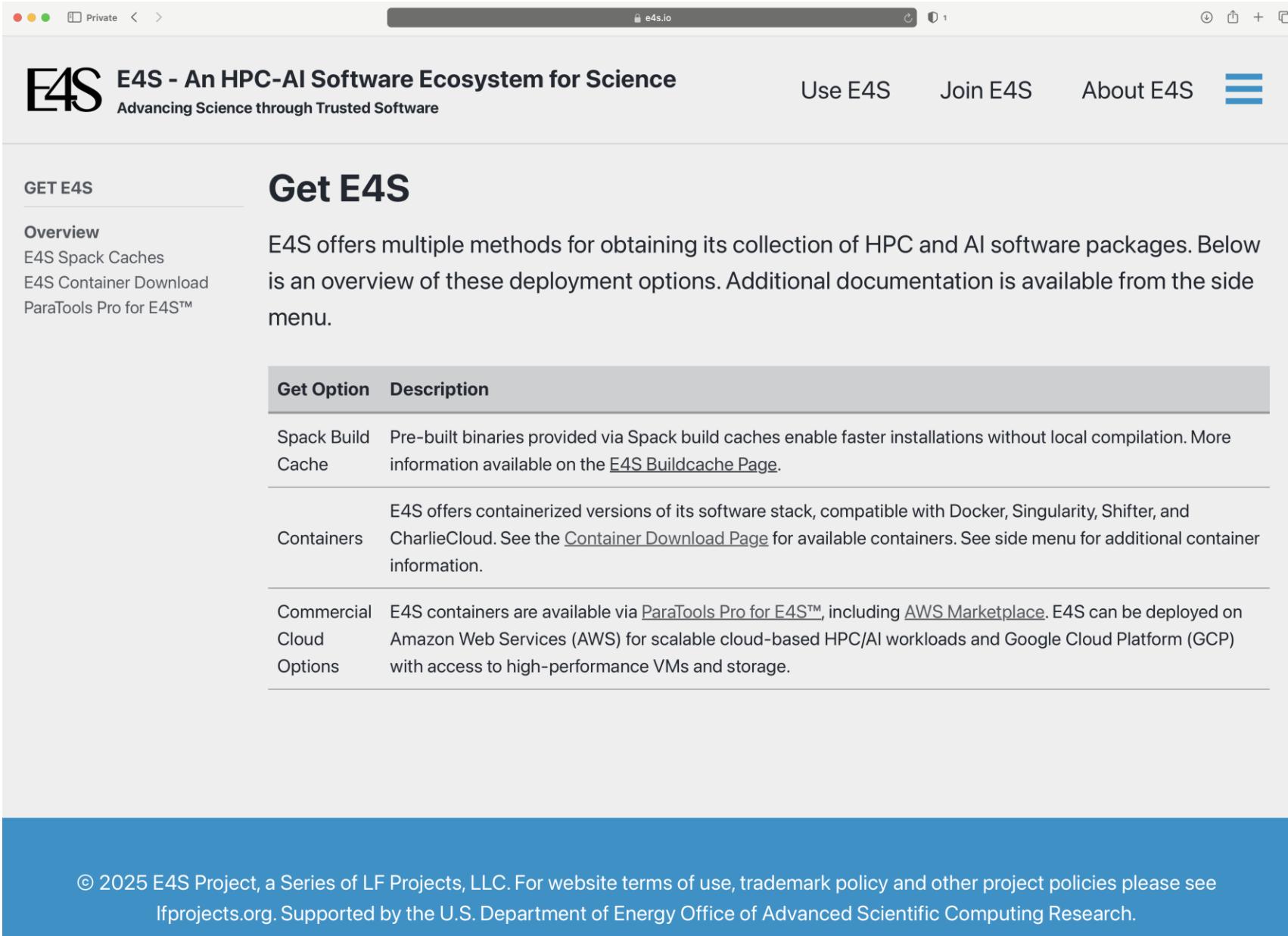
E4S Product Catalog

The E4S Product Catalog provides a convenient searchable and sortable table that provides per-product information for all primary E4S products.

Search Area **AI**

Name	Area	Description	Last Updated	Details
DEEPHYPER	AI	DeepHyper is a powerful Python package for automating machine learning tasks	2025-10-27	Open
HOROVOD	AI	a distributed deep learning training framework	2023-06-12	Open
JAX	AI	Autograd and XLA, brought together for high-performance numerical computing	2025-11-11	Open
KERAS	AI	a multi-backend deep learning framework	2025-10-30	Open
LBANN	AI	an open-source, HPC-centric, deep learning training framework	2025-05-09	Open
NEMO	AI	NVIDIA NeMo Framework is a scalable and cloud-native generative AI framework	2025-11-10	Open
OPENAI-PYTHON	AI	The OpenAI Python library provides convenient access to the OpenAI REST API	2025-11-10	Open
OPENCV	AI	Open Source Computer Vision Library	2025-08-01	Open
PANDAS	AI	powerful Python data analysis toolkit	2025-11-01	Open
PYTORCH	AI	Provides accelerated tensor computation and deep neural networks	2025-11-11	Open
SCIKIT-LEARN	AI	a Python module for machine learning	2025-11-05	Open
TENSORFLOW	AI	An end-to-end open source platform for machine learning	2025-10-28	Open

E4S: Bare-metal installation, Containers, and Cloud images



The screenshot shows the E4S website at e4s.io. The header includes the E4S logo, the text "E4S - An HPC-AI Software Ecosystem for Science" and "Advancing Science through Trusted Software", and navigation links for "Use E4S", "Join E4S", "About E4S", and a menu icon. On the left, a sidebar titled "GET E4S" lists "Overview", "E4S Spack Caches", "E4S Container Download", and "ParaTools Pro for E4S™". The main content area is titled "Get E4S" and describes multiple methods for obtaining software packages. A table provides details on Spack Build Cache, Containers, and Commercial Cloud Options.

Get E4S

E4S offers multiple methods for obtaining its collection of HPC and AI software packages. Below is an overview of these deployment options. Additional documentation is available from the side menu.

Get Option	Description
Spack Build Cache	Pre-built binaries provided via Spack build caches enable faster installations without local compilation. More information available on the E4S Buildcache Page .
Containers	E4S offers containerized versions of its software stack, compatible with Docker, Singularity, Shifter, and CharlieCloud. See the Container Download Page for available containers. See side menu for additional container information.
Commercial Cloud Options	E4S containers are available via ParaTools Pro for E4S™ , including AWS Marketplace . E4S can be deployed on Amazon Web Services (AWS) for scalable cloud-based HPC/AI workloads and Google Cloud Platform (GCP) with access to high-performance VMs and storage.

© 2025 E4S Project, a Series of LF Projects, LLC. For website terms of use, trademark policy and other project policies please see [lfprojects.org](#). Supported by the U.S. Department of Energy Office of Advanced Scientific Computing Research.

Download E4S Containers: Rocky Linux 9.6 and Ubuntu 24.04 LTS

The screenshot shows the E4S website at e4s.io. The page title is "E4S Container Download". On the left, there's a sidebar with "GET E4S" and links to "Overview", "E4S Spack Caches", "E4S Container Download", and "ParaTools Pro for E4S™". The main content area has a heading "Acquiring E4S Containers" and text about the availability of Docker and Singularity images for X86_64, PPC64LE, and AARCH64 architectures, based on Ubuntu 24.04 and Rocky 9.6. It also mentions minimal base images for CI pipelines. Below this is a section titled "Container Releases" with a list of links to download Docker and Singularity images for various configurations.

E4S Container Download

Acquiring E4S Containers

The current E4S container offerings include Docker and Singularity images capable of running on X86_64, PPC64LE, and AARCH64 architectures. Our full E4S Release images are based on Ubuntu 24.04 and Rocky 9.6 (x86_64, aarch64, ppc64le). In addition to offering a full E4S image containing a comprehensive selection of E4S software released on a bi-annual cycle, we also offer a set of minimal base images suitable for use in Continuous Integration (CI) pipelines where Spack is used to build packages.

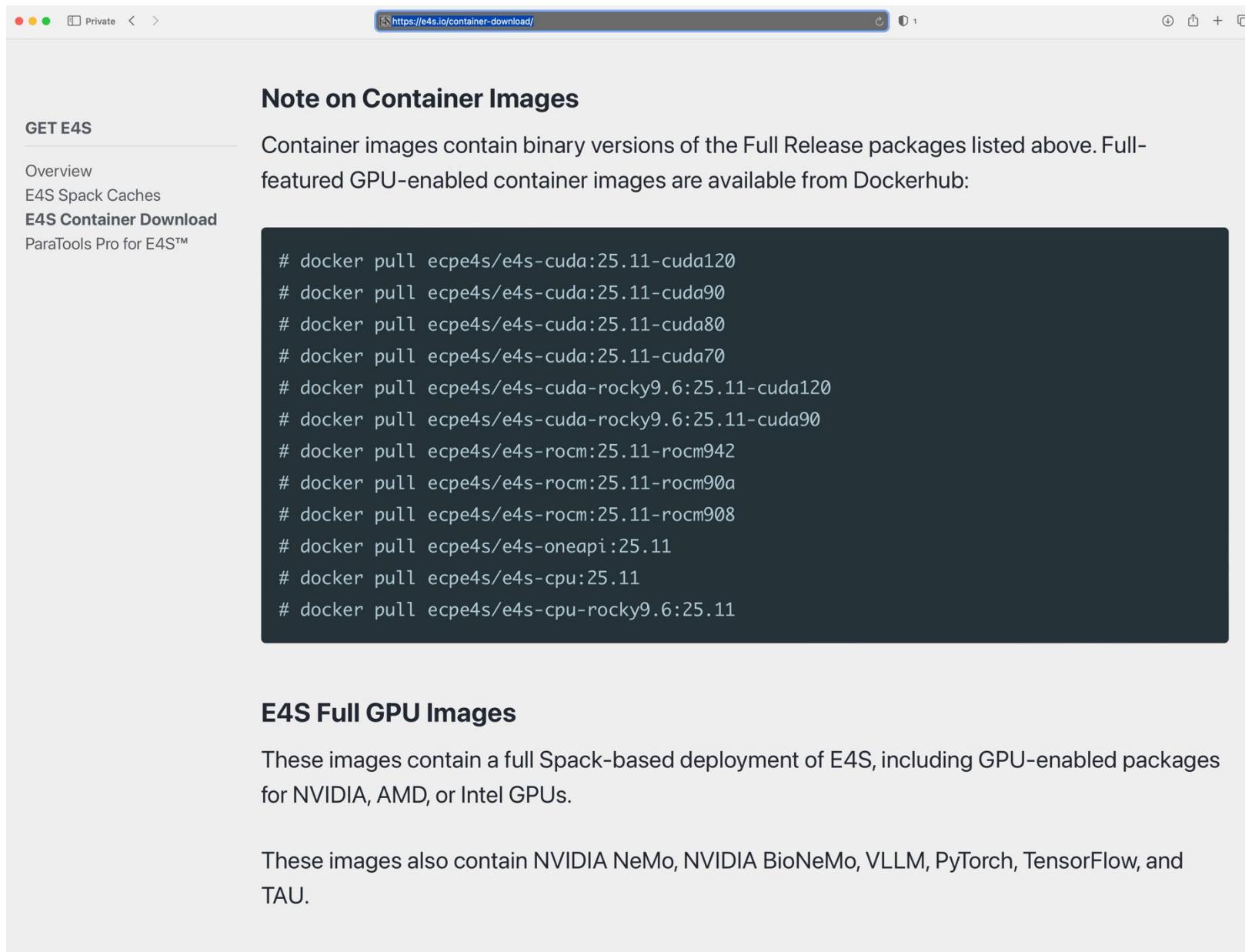
Docker images are available on the [E4S Docker Hub](#). Please see the [E4S 25.11 Release Notes](#).

Container Releases

- [Docker Downloads - CPU only - Ubuntu](#)
- [Docker Downloads - CPU only - Rocky](#)
- [Docker Downloads - CUDA - Ubuntu](#)
- [Docker Downloads - CUDA - Rocky](#)
- [Docker Downloads - ROCm](#)
- [Docker Downloads - OneAPI](#)
- [Singularity x86_64 Download - CPU only - Ubuntu](#)
- [Singularity x86_64 Download - CPU only - Rocky](#)
- [Singularity x86_64 Download - CUDA 80 - Ubuntu](#)
- [Singularity x86_64 Download - CUDA 90 - Ubuntu](#)
- [Singularity x86_64 Download - CUDA 90 - Rocky](#)
- [Singularity x86_64 Download - CUDA 120 - Ubuntu](#)
- [Singularity x86_64 Download - CUDA 120 - Rocky](#)
- [Singularity x86_64 Download - ROCm gfx942 - Ubuntu](#)
- [Singularity x86_64 Download - ROCm gfx90a - Ubuntu](#)
- [Singularity x86_64 Download - ROCm gfx908 - Ubuntu](#)
- [Singularity x86_64 Download - OneAPI - Ubuntu](#)
- [Singularity ppc64le Download - CUDA 70 - Ubuntu](#)
- [Singularity aarch64 Download - CPU only - Ubuntu](#)
- [Singularity aarch64 Download - CUDA 80 - Ubuntu](#)
- [Singularity aarch64 Download - CUDA 90 - Ubuntu](#)
- [Singularity aarch64 Download - CUDA 90 - Rocky](#)

- Docker and Singularity
- ARM64 (aarch64), x86_64, and ppc64le
- Support for GPUs:
 - NVIDIA
 - AMD
 - Intel
- GPU Runtimes:
 - CUDA 12.9
 - ROCm 6.4.3
 - oneAPI 2025.2
- Languages:
 - C/C++/Fortran
 - Python
 - Rust
 - Julia
 - Chapel ...
- OSes:
 - Rocky Linux 9.6
 - Ubuntu 24.04 LTS

E4S container images available on DockerHub



The screenshot shows a web browser window with the URL <https://e4s.io/container-download/> in the address bar. The page content is as follows:

Note on Container Images

Container images contain binary versions of the Full Release packages listed above. Full-featured GPU-enabled container images are available from Dockerhub:

```
# docker pull ecpe4s/e4s-cuda:25.11-cuda120
# docker pull ecpe4s/e4s-cuda:25.11-cuda90
# docker pull ecpe4s/e4s-cuda:25.11-cuda80
# docker pull ecpe4s/e4s-cuda:25.11-cuda70
# docker pull ecpe4s/e4s-cuda-rocky9.6:25.11-cuda120
# docker pull ecpe4s/e4s-cuda-rocky9.6:25.11-cuda90
# docker pull ecpe4s/e4s-rocm:25.11-rocm942
# docker pull ecpe4s/e4s-rocm:25.11-rocm90a
# docker pull ecpe4s/e4s-rocm:25.11-rocm908
# docker pull ecpe4s/e4s-oneapi:25.11
# docker pull ecpe4s/e4s-cpu:25.11
# docker pull ecpe4s/e4s-cpu-rocky9.6:25.11
```

E4S Full GPU Images

These images contain a full Spack-based deployment of E4S, including GPU-enabled packages for NVIDIA, AMD, or Intel GPUs.

These images also contain NVIDIA NeMo, NVIDIA BioNeMo, VLLM, PyTorch, TensorFlow, and TAU.

E4S 25.11 container images available on DockerHub

The screenshot shows a browser window displaying the Docker Hub website at <https://hub.docker.com/r/ecpe4s/e4s-cuda>. The page title is "ecpe4s/e4s-cuda". The repository was created by [ecpe4s](#) and was last updated about 4 hours ago. The description states: "E4S [https://e4s.io] image with CUDA 12.9. Includes NVIDIA NeMo(TM), NVIDIA BioNeMo(TM), VLLM." The repository is categorized under "IMAGE" and includes tags for "MACHINE LEARNING & AI", "DEVELOPER TOOLS", and "OPERATING SYSTEMS". It has 0 stars and 3.2K views. A "Manage Repository" button is visible in the top right corner.

Overview Tags

E4S, an HPSF project, is an HPC-AI software ecosystem for science. It's a curated, Spack based collection of scientific libraries and tools that form the foundation of some of the world's most advanced scientific applications. E4S curates the largest single collection of open-source GPU-enabled libraries and tools for scientific applications, including support for equation-based modeling and simulation, and AI for science. All E4s public-domain software is thoroughly tested for interoperability and portability to multiple computing architectures and will continue to be enhanced and expanded to address architectural changes and emerging new architectures. While E4S supports many products and distributions, users can confidently select any subset of functionality. We build and test the whole so you can select what you need. The key features of E4S 25.06 support a timely expansion of the ecosystem's AI portfolio and include:

- Support for the NVIDIA Blackwell GPU architecture on x86_64 and aarch64 (Grace-Blackwell) with CUDA v12.9.

Tag summary

Recent tags [25.11](#)

Content type
Image

Digest
sha256:8b82152f6...

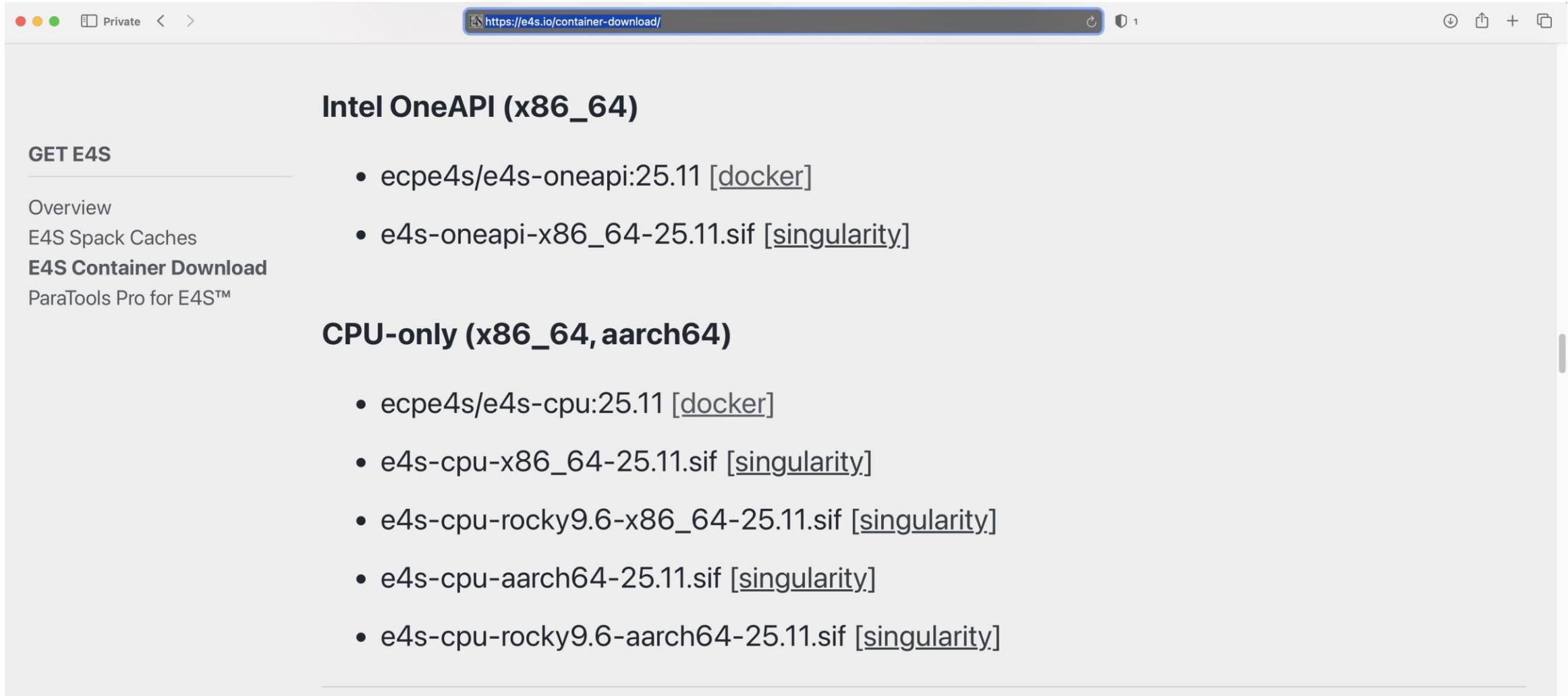
Size
33.2 GB

Download E4S containers: NVIDIA and AMD GPUs

The screenshot shows a web browser window with the URL <https://e4s.io/container-download/>. The page content is organized into two main sections:

- AMD ROCm (x86_64)**
 - [ecpe4s/e4s-rocm:25.11 \[docker\]](#)
 - [e4s-rocm942-x86_64-25.11.sif \[singularity\]](#)
 - [e4s-rocm90a-x86_64-25.11.sif \[singularity\]](#)
 - [e4s-rocm908-x86_64-25.11.sif \[singularity\]](#)
- NVIDIA CUDA (X86_64, PPC64LE, AARCH64)**
 - [ecpe4s/e4s-cuda:25.11 \[docker\]](#)
 - [e4s-cuda80-x86_64-25.11.sif \[singularity\]](#)
 - [e4s-cuda90-x86_64-25.11.sif \[singularity\]](#)
 - [e4s-cuda120-x86_64-25.11.sif \[singularity\]](#)
 - [e4s-cuda90-rocky9.6-x86_64-25.11.sif \[singularity\]](#)
 - [e4s-cuda120-rocky9.6-x86_64-25.11.sif \[singularity\]](#)
 - [e4s-cuda70-ppc64le-25.11.sif \[singularity\]](#)
 - [e4s-cuda80-aarch64-25.11.sif \[singularity\]](#)
 - [e4s-cuda90-aarch64-25.11.sif \[singularity\]](#)
 - [e4s-cuda120-aarch64-25.11.sif \[singularity\]](#)
 - [e4s-cuda90-rocky9.6-aarch64-25.11.sif \[singularity\]](#)
 - [e4s-cuda120-rocky9.6-aarch64-25.11.sif \[singularity\]](#)

E4S Containers: Intel oneAPI (CPU/GPU) and CPU only



The screenshot shows a web browser window with the URL <https://e4s.io/container-download/> in the address bar. The page content is as follows:

Intel OneAPI (x86_64)

GET E4S

- `ecpe4s/e4s-oneapi:25.11` [[docker](#)]
- `e4s-oneapi-x86_64-25.11.sif` [[singularity](#)]

CPU-only (x86_64, aarch64)

- `ecpe4s/e4s-cpu:25.11` [[docker](#)]
- `e4s-cpu-x86_64-25.11.sif` [[singularity](#)]
- `e4s-cpu-rocky9.6-x86_64-25.11.sif` [[singularity](#)]
- `e4s-cpu-aarch64-25.11.sif` [[singularity](#)]
- `e4s-cpu-rocky9.6-aarch64-25.11.sif` [[singularity](#)]

E4S Base Containers with GPU runtimes and MPI

The screenshot shows a web browser window with the URL <https://e4s.io/container-download/>. The page is titled "GPU Base Images". On the left, there's a sidebar with "GET E4S" and links to "Overview", "E4S Spack Caches", "E4S Container Download", and "ParaTools Pro for E4S™". The main content area has three sections: "AMD ROCM (X86_64)", "NVIDIA Multi-Arch (X86_64, PPC64LE, AARCH64)", and "Intel OneAPI (X86_64)". Each section lists Docker and Singularity image links.

GPU Base Images

These images come with MPICH, CMake, and the relevant GPU SDK – either AMD ROCm, NVIDIA CUDA Toolkit and NVHPC, or Intel OneAPI.

AMD ROCM (X86_64)

- `ecpe4s/e4s-base-rocm:25.11` [[docker](#)]
- `e4s-base-rocm-25.11.sif` [[singularity](#)]

NVIDIA Multi-Arch (X86_64, PPC64LE, AARCH64)

- `ecpe4s/e4s-base-cuda:25.11` [[docker](#)]
- `e4s-base-cuda-x86_64-25.11.sif` [[singularity](#)]
- `e4s-base-cuda-aarch64-25.11.sif` [[singularity](#)]
- `e4s-base-cuda-ppc64le-25.11.sif` [[singularity](#)]

Intel OneAPI (X86_64)

- `ecpe4s/e4s-base-oneapi:25.11` [[docker](#)]
- `e4s-base-oneapi-25.11.sif` [[singularity](#)]

Minimal E4S Spack containers

The screenshot shows a web browser window with the URL <https://e4s.io/container-download/>. The page is titled "Minimal Spack". On the left, there's a sidebar with "GET E4S" and links to "Overview", "E4S Spack Caches", "E4S Container Download", and "ParaTools Pro for E4STM". The main content area has a section titled "Minimal Spack" with the text: "This image contains a minimal setup for using Spack 0.22.0 w/ GNU compilers". Below this is a section titled "X86_64, PPC64LE, AARCH64" containing a bulleted list of container images:

- ecpe4s/ubuntu20.04 [[docker](#)]
- ecpe4s-ubuntu20.04-x86_64-24.02.sif [[singularity](#)]
- ecpe4s-ubuntu20.04-ppc64le-24.02.sif [[singularity](#)]
- ecpe4s-ubuntu20.04-aarch64-24.02.sif [[singularity](#)]

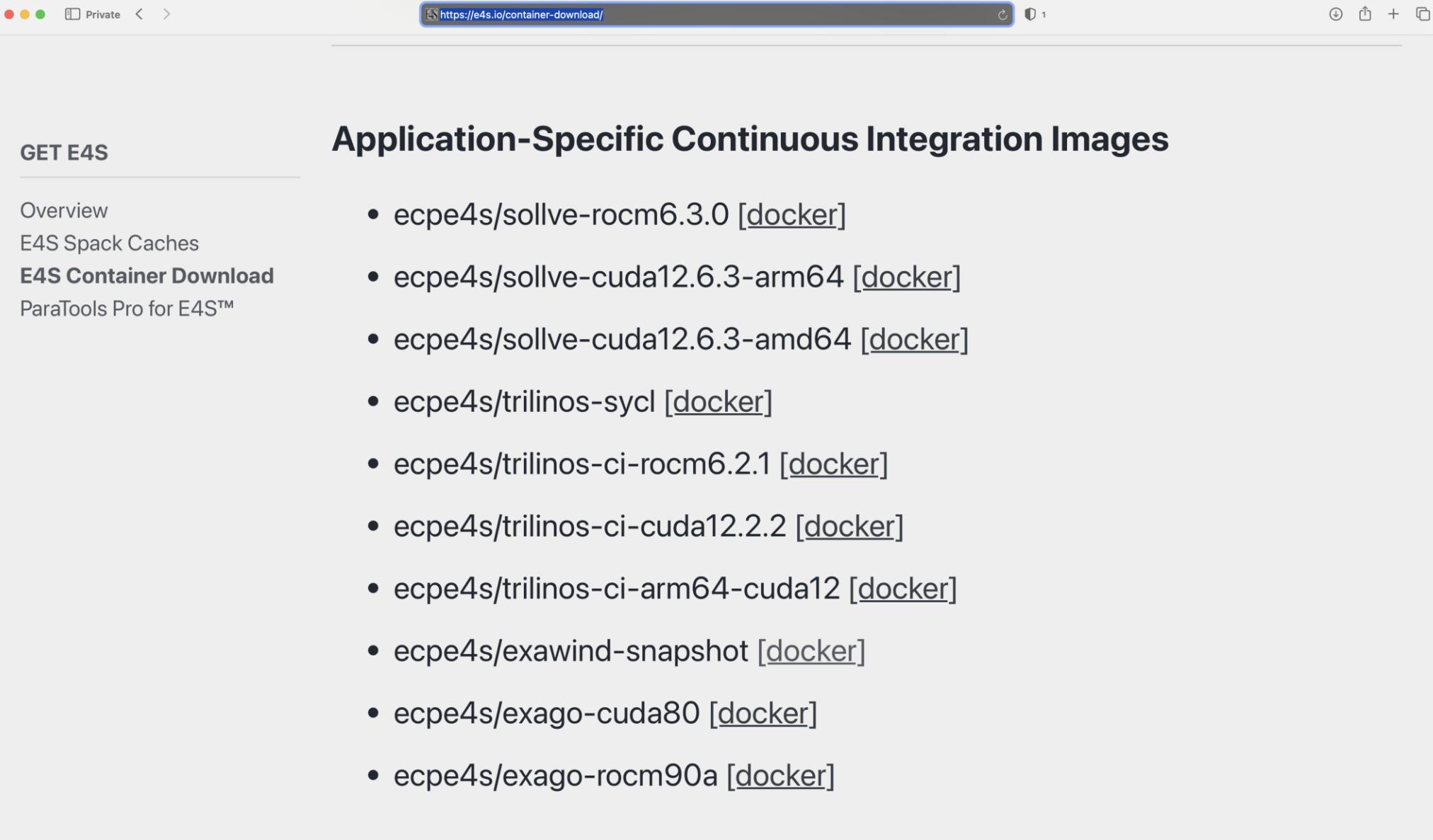
DOE LLVM E4S Image

This multi-architecture image contains E4S products compiled with DOE LLVM 16 and Flang using Spack

Multi-Arch (X86_64, PPC64LE, AARCH64)

- ecpe4s/e4s-doe-llvm:23.05 [[docker](#)]
- e4s-doe-llvm-x86_64-23.05.sif [[singularity](#)]
- e4s-doe-llvm-aarch64-23.05.sif [[singularity](#)]
- e4s-doe-llvm-ppc64le-23.05.sif [[singularity](#)]

E4S Application Specific Container Images for CI: Customization



The screenshot shows a web browser window with the URL <https://e4s.io/container-download/> in the address bar. The page content is as follows:

GET E4S

Overview
E4S Spack Caches
E4S Container Download
ParaTools Pro for E4S™

Application-Specific Continuous Integration Images

- ecpe4s/sollve-rocm6.3.0 [[docker](#)]
- ecpe4s/sollve-cuda12.6.3-arm64 [[docker](#)]
- ecpe4s/sollve-cuda12.6.3-amd64 [[docker](#)]
- ecpe4s/trilinos-sycl [[docker](#)]
- ecpe4s/trilinos-ci-rocm6.2.1 [[docker](#)]
- ecpe4s/trilinos-ci-cuda12.2.2 [[docker](#)]
- ecpe4s/trilinos-ci-arm64-cuda12 [[docker](#)]
- ecpe4s/exawind-snapshot [[docker](#)]
- ecpe4s/exago-cuda80 [[docker](#)]
- ecpe4s/exago-rocm90a [[docker](#)]

E4S Container Images for CI: Gitlab Runners

The screenshot shows a web browser window with the URL <https://e4s.io/container-download/>. The page title is "Minimal Continuous Integration Images". On the left, there's a sidebar with "GET E4S" and links to Overview, E4S Spack Caches, E4S Container Download, and ParaTools Pro for E4S™. The main content area is divided into sections for different architectures:

- X86_64**
 - ecpe4s/ubuntu22.04-runner-x86_64 [[docker](#)] [[github](#)]
 - ecpe4s/ubuntu20.04-runner-x86_64 [[docker](#)] [[github](#)]
 - ecpe4s/ubuntu18.04-runner-x86_64 [[docker](#)] [[github](#)]
 - ecpe4s/rhel8-runner-x86_64 [[docker](#)] [[github](#)]
 - ecpe4s/rhel7-runner-x86_64 [[docker](#)] [[github](#)]
- PPC64LE**
 - ecpe4s/ubuntu22.04-runner-ppc64le [[docker](#)] [[github](#)]
 - ecpe4s/ubuntu20.04-runner-ppc64le [[docker](#)] [[github](#)]
 - ecpe4s/ubuntu18.04-runner-ppc64le [[docker](#)] [[github](#)]
 - ecpe4s/rhel8-runner-ppc64le [[docker](#)] [[github](#)]
 - ecpe4s/rhel7-runner-ppc64le [[docker](#)] [[github](#)]
- AARCH64**
 - ecpe4s/ubuntu22.04-runner-aarch64 [[docker](#)] [[github](#)]
 - ecpe4s/ubuntu20.04-runner-aarch64 [[docker](#)] [[github](#)]
 - ecpe4s/rhel8-runner-aarch64 [[docker](#)] [[github](#)]

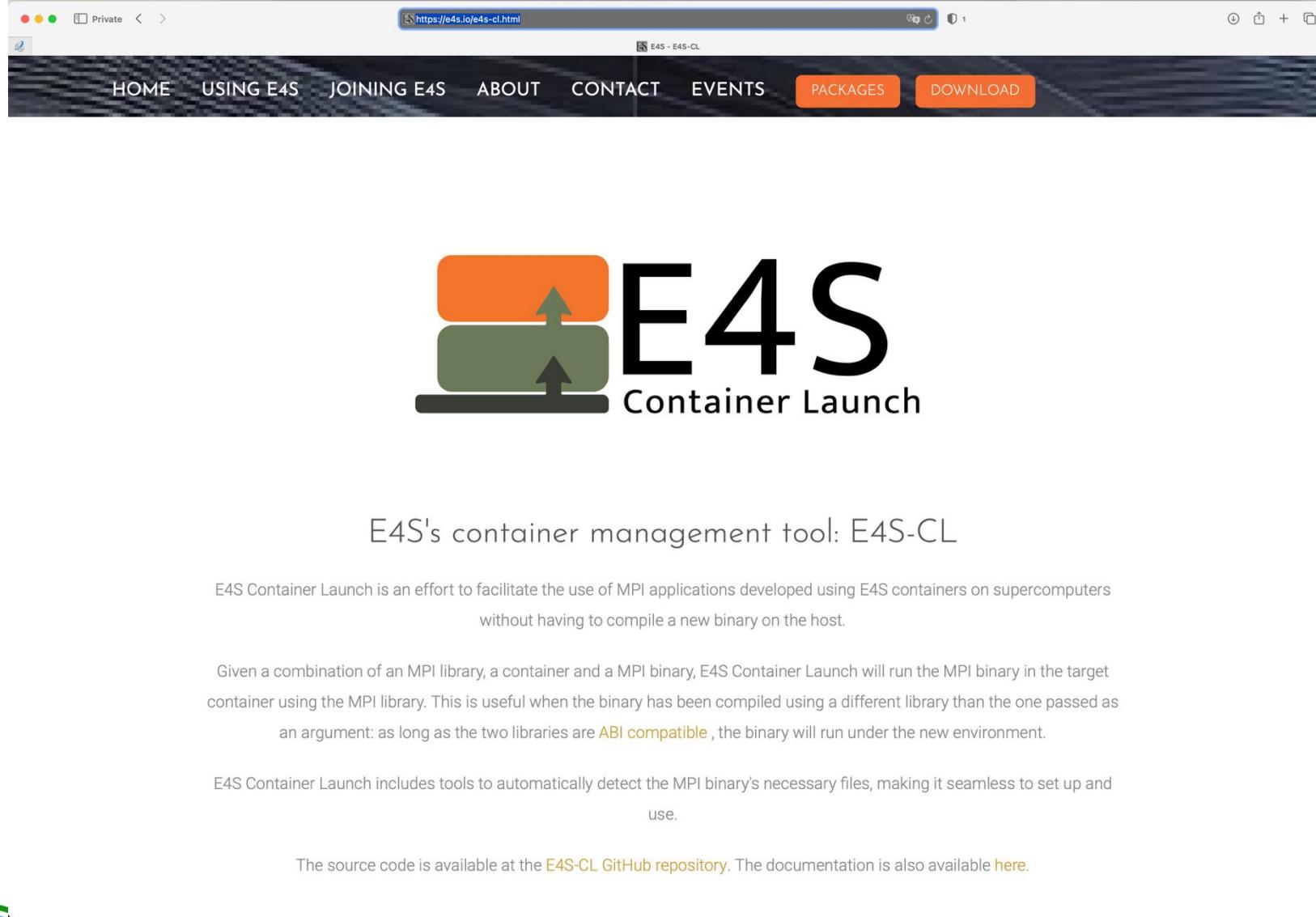
E4S Tools: E4S à la carte or e4s-alc: Customize container images

The screenshot shows the GitHub repository page for E4S-Project/e4s-alc. The repository has 368 commits, 5 issues, and 1 pull request. The README file contains a section titled "E4S à la Carte" which describes the tool's purpose: to facilitate the generation of Dockerfiles and Singularity definition files infused with OS packages, Spack packages, as well as custom commands. The repository has 7 stars, 4 watchers, and 1 fork. It includes sections for Releases (with one latest release at v1.0.2), Packages (no packages published), and Contributors (listing FrederickDeny, PlatinumCD, spoutn1k, and sameershende).

<https://github.com/E4s-Project/e4s-alc>

- Add new system packages
- Add new Spack packages
- Add new tarballs
- Customize the container image
- Start with a base image
- Add packages
- Create a new container image!

E4S Tools: e4s-cl: Container Launch tool for MPI applications

A screenshot of a web browser displaying the E4S Container Launch website at https://e4s.io/e4s-cl.html. The page features a dark header with navigation links: HOME, USING E4S, JOINING E4S, ABOUT, CONTACT, EVENTS, PACKAGES (highlighted in orange), and DOWNLOAD. Below the header is a large E4S logo consisting of a stylized orange and green block icon followed by the text "E4S Container Launch". The main content area contains the following text:

E4S's container management tool: E4S-CL

E4S Container Launch is an effort to facilitate the use of MPI applications developed using E4S containers on supercomputers without having to compile a new binary on the host.

Given a combination of an MPI library, a container and a MPI binary, E4S Container Launch will run the MPI binary in the target container using the MPI library. This is useful when the binary has been compiled using a different library than the one passed as an argument: as long as the two libraries are **ABI compatible**, the binary will run under the new environment.

E4S Container Launch includes tools to automatically detect the MPI binary's necessary files, making it seamless to set up and use.

The source code is available at the [E4S-CL GitHub repository](#). The documentation is also available [here](#).

- Distribute your MPI application as a binary with an E4S image
- While deploying on a system substitute the embedded containerized MPI in application with the system/vendor MPI
- Use inter-node network interfaces efficiently for near native performance!

e4s-cl: A tool to simplify the launch of MPI jobs in E4S containers

- E4S containers support replacement of MPI libraries using MPICH ABI compatibility layer and Wi4MPI [CEA] for OpenMPI replacement.
- Applications binaries built using E4S can be launched with Singularity using MPI library substitution for efficient inter-node communications.
- e4s-cl is a new tool that simplifies the launch and MPI replacement.
 - e4s-cl init --backend [singularity|shifter|docker] --image <file> --source <startup_cmds.sh>
 - e4s-cl srun -n <N> <command>

- Usage:

```
% e4s-cl init --backend singularity --image ~/images/e4s-gpu-x86.sif --source ~/source.sh  
% cat ~/source.sh  
  . /spack/share/spack/setup-env.sh  
  spack load trilinos+cuda  
  
% e4s-cl srun -n 4 ./a.out
```



<https://github.com/E4S-Project/e4s-cl>

E4S Tools: e4s-chain-spack.sh to customize software stack

```
sameer@mothra:~$ ls ~/images
e4s-cuda80-x86_64-25.06.sif
sameer@mothra:~$ singularity run --nv ~/images/e4s-cuda80-x86_64-25.06.sif
Singularity> /etc/e4s/e4s-chain-spack.sh ~/spack
Cloning into '/home/sameer/spack'...
remote: Enumerating objects: 686113, done.
remote: Counting objects: 100% (976/976), done.
remote: Compressing objects: 100% (463/463), done.
remote: Total 686113 (delta 772), reused 518 (delta 510), pack-reused 685137 (from 3)
Receiving objects: 100% (686113/686113), 230.82 MiB | 37.06 MiB/s, done.
Resolving deltas: 100% (326280/326280), done.

-----
Configuration SUCCESS!
```

```
Downstream: /home/sameer/spack
Upstream: /spack
```

To use the downstream Spack instance, run the following command in your shell:

```
. /home/sameer/spack/share/spack/setup-env.sh
-----
```

```
Singularity> . /home/sameer/spack/share/spack/setup-env.sh
```

```
Singularity> spack find valgrind
==> Error: No package matches the query: valgrind
Singularity> spack install valgrind
[+] /usr/local/mpich/install/mpich (external mpich-4.2.3-47excoypwhfmhx57rfs6reouvninugcf)
[+] /usr (external glibc-2.35-a7drdl4t1x4bu3mzhor75pskvd3pdot6)
[+] /spack/opt/spack/linux-ubuntu22.04-x86_64_v3/gcc-11.4.0/gcc-runtime-11.4.0-f63c77kavzjtpmnhud2oyfaxagwjzla
[+] /spack/opt/spack/linux-ubuntu22.04-x86_64_v3/gcc-11.4.0/boost-1.86.0-6qkv24gbidwxhllgah6jrkym5ev2cng5
[+] /spack/opt/spack/linux-ubuntu22.04-x86_64_v3/gcc-11.4.0/gmake-4.4.1-qp5blvcyuzghqsrp2ew6gq2nlos34b2
==> Installing valgrind-3.23.0-feuxx36lsqp7quzmhmo4opbiadwpsars [6/6]
==> No binary for valgrind-3.23.0-feuxx36lsqp7quzmhmo4opbiadwpsars found: installing from source
==> Fetching https://mirror.spack.io/_source-cache/archive/c5/c5c34a3380457b9b75606df890102e7df2c702b9420c2ebef9540f8b5d56264d.tar.bz2
==> Ran patch() for valgrind
==> valgrind: Executing phase: 'autoreconf'
==> valgrind: Executing phase: 'configure'
==> valgrind: Executing phase: 'build'
==> valgrind: Executing phase: 'install'
==> valgrind: Successfully installed valgrind-3.23.0-feuxx36lsqp7quzmhmo4opbiadwpsars
  Stage: 3.78s. Autoreconf: 0.01s. Configure: 48.56s. Build: 37.71s. Install: 2.97s. Post-install: 0.60s. Total: 1m 33.97s
[+] /home/sameer/spack/opt/spack/linux-ubuntu22.04-x86_64_v3/gcc-11.4.0/valgrind-3.23.0-feuxx36lsqp7quzmhmo4opbiadwpsars
Singularity> spack load valgrind
Singularity> which valgrind
/home/sameer/spack/opt/spack/linux-ubuntu22.04-x86_64_v3/gcc-11.4.0/valgrind-3.23.0-feuxx36lsqp7quzmhmo4opbiadwpsars/bin/valgrind
```

Specify location of downstream
Spack installation directory

Source downstream Spack's
setup-env.sh

Install a new Spack package
in downstream Spack directory

Load new package (valgrind)
using spack load



E4S Tools: e4s-chain-spack.sh to customize software stack

```
Singularity> which valgrind
/home/sameer/spack/opt/spack/linux-ubuntu22.04-x86_64_v3/gcc-11.4.0/valgrind-3.23.0-feuxx36lsqp7quzmhmo4opbiadwpsars/bin/valgrind
Singularity> valgrind --help
usage: valgrind [options] prog-and-args

tool-selection option, with default in []:
--tool=<name>          use the Valgrind tool named <name> [memcheck]
available tools are:
memcheck cachegrind callgrind helgrind drd
massif dhat lackey none exp-bbv

basic user options for all Valgrind tools, with defaults in []:
-h --help                show this message
--help-debug              show this message, plus debugging options
--help-dyn-options        show the dynamically changeable options
--version                 show version
-q --quiet                run silently; only print error msgs
-v --verbose               be more verbose -- show misc extra info
--trace-children=no|yes   Valgrind-ise child processes (follow execve)? [no]
--trace-children-skip=patt1,patt2,...  specifies a list of executables
                                         that --trace-children=yes should not trace into
--trace-children-skip-by-arg=patt1,patt2,... same as --trace-children-skip=
                                         but check the argv[] entries for children, rather
                                         than the exe name, to make a follow/no-follow decision
--child-silent-after-fork=no|yes omit child output between fork & exec? [no]
--vgdb=no|yes|full        activate gdbserver? [yes]
                                         full is slower but provides precise watchpoint/step
--vgdb-error=<number>     invoke gdbserver after <number> errors [99999999]
                                         to get started quickly, use --vgdb-error=0
                                         and follow the on-screen directions
--vgdb-stop-at=event1,event2,... invoke gdbserver for given events [none]
                                         where event is one of:
                                         startup exit abexit valgrindabexit all none
--track-fds=no|yes|all    track open file descriptors? [no]
                                         all includes reporting stdin, stdout and stderr
--time-stamp=no|yes       add timestamps to log messages? [no]
--log-fd=<number>         log messages to file descriptor [2=stderr]
--log-file=<file>         log messages to <file>
--log-socket=ipaddr:port log messages to socket ipaddr:port
--enable-debuginfod=no|yes query debuginfod servers for missing
                                         debuginfo [yes]

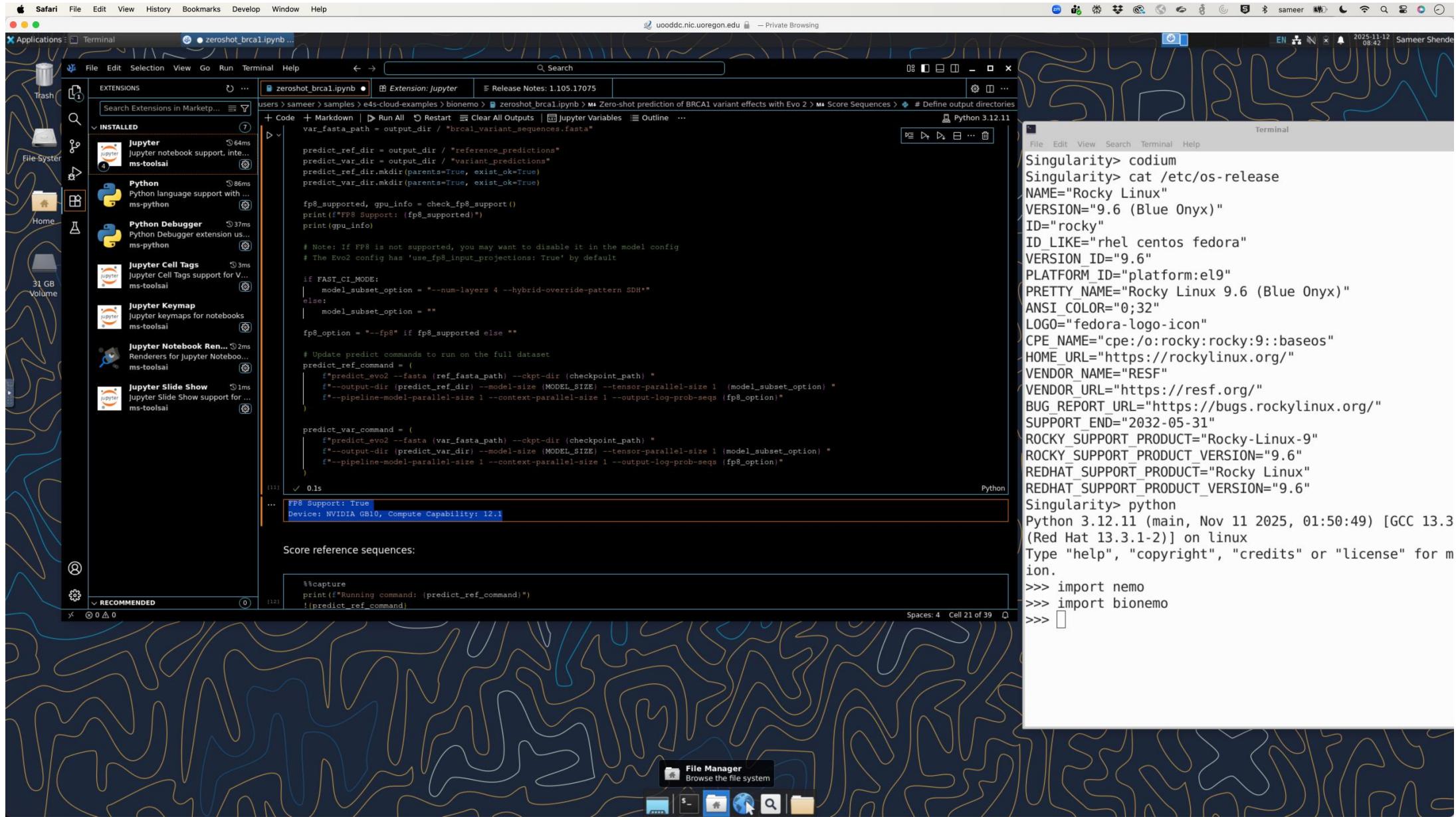
user options for Valgrind tools that report errors:
--xml=yes                emit error output in XML (some tools only)
--xml-fd=<number>          XML output to file descriptor
--xml-file=<file>          XML output to <file>
--xml-socket=ipaddr:port  XML output to socket ipaddr:port
--xml-user-comment=STR    copy STR verbatim into XML output
--demangle=no|yes          automatically demangle C++ names? [yes]
--num-callers=<number>     show <number> callers in stack traces [12]
--error-limit=no|yes      stop showing new errors if too many? [yes]
--exit-on-first-error=no|yes exit code on the first error found? [no]
--error-exitcode=<number>  exit code to return if errors found [0=disable]
--error-markers=<begin>,<end> add lines with begin/end markers before/after
                                         each error output in plain text mode [none]
--show-error-list=no|yes|all show detected errors list and
                                         suppression counts at exit [no].
                                         all means to also print suppressed errors.
                                         same as --show-error-list=yes
-s
```

Downstream Spack's package
is loaded in your environment

e4s-chain-spack.sh helps
customize the software stack
using upstream /spack
(read-only in the container) for
package dependencies while
installing a new package in the
downstream Spack in your
writable home directory.



E4S: NVIDIA BioNeMo™ on NVIDIA Grace-Blackwell architecture



E4S: NVIDIA Grace-Blackwell CUDA 120 Rocky Linux 9.6 image

```
$ singularity run --nv e4s-cuda120-rocky9.6-aarch64-25.11.sif
Singularity>
Singularity> nvidia-smi -L
GPU 0: NVIDIA GB10 (UUID: GPU-233b4e95-c45e-17bd-40d5-daa39dbbd475)
Singularity> spack find -x
-- linux-rocky9-aarch64 / %c,cxx,fortran=gcc@13.3.1 -----
adios2@2.10.2 caliper@2.12.1 hpre@2.33.0 magma@2.9.0 petsc@3.24.0 slepc@3.24.0 sundials@7.5.0 tau@2.35 umpire@2025.03.0
amrex@25.10 heffte@2.4.1 libceed@0.12.0 paraview@5.13.3 slate@2025.05.28 strumpack@8.0.0 superlu-dist@9.1.0 trilinos@16.1.0 zfp@1.0.1

-- linux-rocky9-aarch64 / %c,cxx=gcc@13.3.1 -----
cabana@0.7.0 chapel@2.6.0 flecsi@2.4.1 gromacs@2025.3 lammps@20250722 libpressio@0.99.4 omega-h@10.8.6-scorec vtk-m@2.3.0
chai@2025.03.0 fftx@1.2.0 ginkgo@1.10.0 hpctoolkit@2025.0.1 legion@25.03.0 mgard@compat-2023-12-09 upcxx@2023.9.0

-- linux-rocky9-aarch64 / %c=gcc@13.3.1 -----
parsec@4.0.2411

-- linux-rocky9-aarch64 / %cxx,fortran=gcc@13.3.1 -----
tasmanian@8.1

-- linux-rocky9-aarch64 / %cxx=gcc@13.3.1 -----
arborx@1.5 hpx@1.11.0 kokkos@4.7.01 kokkos-kernels@4.7.01

-- linux-rocky9-aarch64 / no compilers -----
e4s-alc@1.0.3 e4s-cl@1.0.5 mpich@4.3.1
==> 42 installed packages
Singularity> module avail
----- /spack/share/spack/modules/linux-rocky9-aarch64 -----
adios2/2.10.2-cuda120 e4s-alc/1.0.3 hpctoolkit/2025.0.1-cuda libceed/0.12.0-cuda120 parsec/4.0.2411-cuda120 tasmanian/8.1-cuda120
amrex/25.10-cuda120 e4s-cl/1.0.5 hpx/1.11.0-cuda120 libpressio/0.99.4-cuda120-openmp petsc/3.24.0-cuda120 tau/2.35-cuda
arborx/1.5-cuda120 fftx/1.2.0-cuda120 hpre/2.33.0-cuda120 magma/2.9.0-cuda120 slate/2025.05.28-cuda120-openmp trilinos/16.1.0-cuda120
cabana/0.7.0-cuda120 flecsi/2.4.1-cuda120 kokkos-kernels/4.7.01-cuda120 mgard@compat-2023-12-09-cuda120-openmp umpire/2025.03.0-cuda120
caliper/2.12.1-cuda120 ginkgo/1.10.0-cuda120-openmp kokkos/4.7.01-cuda120 mpich/4.3.1 strumpack@8.0.0-cuda120-openmp upcxx@2023.9.0-cuda120
chai/2025.03.0-cuda120 gromacs/2025.3-cuda120-openmp lammps@20250722-cuda120-openmp omega-h@10.8.6-scorec-cuda120 sundials/7.5.0-cuda120
chapel/2.6.0-cuda120 heffte@2.4.1-cuda120 legion@25.03.0-cuda70 paraview@5.13.3 superlu-dist@9.1.0-cuda120
zfp@1.0.1-cuda120

Key:
loaded modulepath
Singularity> cat /etc/os-release | grep PRETTY_NAME
PRETTY_NAME="Rocky Linux 9.6 (Blue Onyx)"
Singularity> uname -m
aarch64
Singularity>
```

E4S: NVIDIA x86_64-Blackwell (CUDA 120) Rocky Linux 9.6 image

```
$ singularity run --nv e4s-cuda120-rocky9.6-x86_64-25.11.sif
Singularity>
Singularity> nvidia-smi -L
GPU 0: NVIDIA RTX PRO 6000 Blackwell Server Edition (UUID: GPU-0a3e5316-2d97-bf04-1ed7-011560029cf3)
Singularity> spack find -x
-- linux-rocky9-x86_64_v3 / %c,cxx,fortran=gcc@13.3.1 -----
adios2@2.10.2 heffte@2.4.1 libceed@0.12.0 papi@7.2.0 strumpack@8.0.0 superlu-dist@9.1.0 umpire@2025.03.0
caliper@2.12.1 hypre@2.33.0 magma@2.9.0 slate@2025.05.28 sundials@7.5.0 tau@2.35 zfp@1.0.1

-- linux-rocky9-x86_64_v3 / %c,cxx=gcc@13.3.1 -----
bricks@2023.08.25 cabana@0.7.0 chai@2025.03.0 chapel@2.6.0 cusz@0.14.0 fftx@1.2.0 flecsi@2.4.1 ginkgo@1.10.0 hpctoolkit@2025.0.1 legion@25.03.0 mgard@compat-2023-12-09 vtk-m@2.3.0

-- linux-rocky9-x86_64_v3 / %c=gcc@13.3.1 -----
parsec@4.0.2411

-- linux-rocky9-x86_64_v3 / %cxx,fortran=gcc@13.3.1 -----
tasmanian@8.1

-- linux-rocky9-x86_64_v3 / %cxx=gcc@13.3.1 -----
arborx@1.5 hpx@1.11.0 kokkos@4.7.01 kokkos-kernels@4.7.01

-- linux-rocky9-x86_64_v3 / no compilers -----
e4s-alc@1.0.3 e4s-cl@1.0.5 mpich@4.3.1
=> 35 installed packages
Singularity> module avail
----- /spack/share/spack/modules/linux-rocky9-x86_64_v3 -----
adios2/2.10.2-cuda120 chapel/2.6.0-cuda120 ginkgo/1.10.0-cuda120-openmp kokkos/4.7.01-cuda120 papi/7.2.0-cuda
arborx/1.5-cuda120 cusz/0.14.0-cuda120 heffte/2.4.1-cuda120 legion/25.03.0-cuda70 parsec/4.0.2411-cuda120
bricks/2023.08.25-cuda e4s-alc/1.0.3 hpctoolkit/2025.0.1-cuda libceed/0.12.0-cuda120 slate/2025.05.28-cuda120-openmp
cabana/0.7.0-cuda120 e4s-cl/1.0.5 hpx/1.11.0-cuda120 magma/2.9.0-cuda120 umpire/2025.03.0-cuda120
caliper/2.12.1-cuda120 fftx/1.2.0-cuda120 hypre/2.33.0-cuda120 mgard/compat-2023-12-09-cuda120-openmp
chai/2025.03.0-cuda120 flecsi/2.4.1-cuda120 kokkos-kernels/4.7.01-cuda120 sundials/7.5.0-cuda120
mpich/4.3.1 zfp/1.0.1-cuda120
superlu-dist/9.1.0-cuda120

----- /usr/share/Modules/modulefiles -----
dot module-git module-info modules null use.own

Key:
loaded modulepath
Singularity> cat /etc/os-release | grep PRETTY_NAME
PRETTY_NAME="Rocky Linux 9.6 (Blue Onyx)"
Singularity> uname -m
x86_64
Singularity>
```



E4S: NVIDIA x86_64-Blackwell (CUDA 120) Rocky Linux 9.6 image

```
Singularity> module avail
----- /spack/share/spack/modules/linux-rocky9-x86_64_v3 -----
adios2/2.10.2-cuda120  chapel/2.6.0-cuda120  ginkgo/1.10.0-cuda120-openmp  kokkos/4.7.01-cuda120
arborx/1.5-cuda120    cusz/0.14.0-cuda120  heffte/2.4.1-cuda120          legion/25.03.0-cuda70
bricks/2023.08.25-cuda e4s-alc/1.0.3      hpctoolkit/2025.0.1-cuda   libceed/0.12.0-cuda120
cabana/0.7.0-cuda120   e4s-cl/1.0.5       hpx/1.11.0-cuda120          magma/2.9.0-cuda120
caliper/2.12.1-cuda120 fftx/1.2.0-cuda120  hypre/2.33.0-cuda120        mgard/compat-2023-12-09-cuda120-openmp
chai/2025.03.0-cuda120 flecsi/2.4.1-cuda120  kokkos-kernels/4.7.01-cuda120  mpich/4.3.1
                                                               papi/7.2.0-cuda          tasmanian/8.1-cuda120
                                                               parsec/4.0.2411-cuda120  tau/2.35-cuda
                                                               slate/2025.05.28-cuda120-openmp  umpire/2025.03.0-cuda120
                                                               strumpack/8.0.0-cuda120-openmp  vtk-m/2.3.0-cuda120-openmp
                                                               sundials/7.5.0-cuda120        zfp/1.0.1-cuda120
                                                               superlu-dist/9.1.0-cuda120

----- /usr/share/Modules/modulefiles -----
dot module-git module-info modules null use.own

Key:
loaded modulepath
Singularity> spack find -x
-- linux-rocky9-x86_64_v3 / %c,cxx,fortran=gcc@13.3.1 -----
adios2@2.10.2  heffte@2.4.1  libceed@0.12.0  papi@7.2.0  strumpack@8.0.0  superlu-dist@9.1.0  umpire@2025.03.0
caliper@2.12.1  hypre@2.33.0  magma@2.9.0   slate@2025.05.28  sundials@7.5.0   tau@2.35   zfp@1.0.1

-- linux-rocky9-x86_64_v3 / %c,cxx=gcc@13.3.1 -----
bricks@2023.08.25  cabana@0.7.0  chai@2025.03.0  chapel@2.6.0  cusz@0.14.0  fftx@1.2.0  flecsi@2.4.1  ginkgo@1.10.0  hpctoolkit@2025.0.1  legion@25.03.0  mgard@compat-2023-12-09  vtk-m@2.3.0

-- linux-rocky9-x86_64_v3 / %c=gcc@13.3.1 -----
parsec@4.0.2411

-- linux-rocky9-x86_64_v3 / %cxx,fortran=gcc@13.3.1 -----
tasmanian@8.1

-- linux-rocky9-x86_64_v3 / %cxx=gcc@13.3.1 -----
arborx@1.5  hpx@1.11.0  kokkos@4.7.01  kokkos-kernels@4.7.01

-- linux-rocky9-x86_64_v3 / no compilers -----
e4s-alc@1.0.3  e4s-cl@1.0.5  mpich@4.3.1
=> 35 installed packages
Singularity>
```

E4S NVIDIA Blackwell x86_64 Rocky Linux 9.6 Python Packages

```
$ singularity run --nv e4s-cuda120-rocky9.6-x86_64-25.11.sif
Singularity> nvidia-smi -L
GPU 0: NVIDIA RTX PRO 6000 Blackwell Server Edition (UUID: GPU-0a3e5316-2d97-bf04-1ed7-011560029cf3)
Singularity> uname -m
x86_64
Singularity> which python
/opt/python/pkgs/python-3.12.11/bin/python
Singularity> ls /opt/python/pkgs/python-3.12.11/lib/python3.12/
abc.py          compileall.py      ensurepip      imaplib.py      ntpath.py      _py_abc.py      signal.py      sysconfig.py      unittest
aifc.py         _compression.py   enum.py       imglib.py      nturl2path.py  __pycache__    _sitebuiltins.py tabnanny.py      urllib
_aix_support.py concurrent      filecmp.py     importlib     numbers.py     pyclbr.py      site-packages  tarfile.py      uuid.py
antigravity.py config-3.12-x86_64-linux-gnu fileinput.py inspect.py     opcode.py      py_compile.py  site.py        telnetlib.py    uu.py
argparse.py     configparser.py   fnmatch.py    fractions.py ipaddress.py optparse.py    _pydatetime.py smtplib.py      tempfile.py    venv
ast.py          contextlib.py    fnmatch.py    fractions.py ipaddress.py optparse.py    _pydecimal.py sndhdr.py      test
asyncio         contextvars.py   ftplib.py     json         os.py         pydoc_data    socket.py      socketserver.py test
base64.py       copy.py        functools.py keyword.py    _osx_support.py pydoc.py       socketserver.py textwrap.py    warnings.py
bdb.py          copyreg.py     __future__.py getopt.py     lib2to3       pathlib.py    _pyio.py       sqlite3        this.py       weakref.py
bisect.py       cProfile.py    genericpath.py getopt.py     lib_dynload   pdb.py       _pylong.py    sre_compile.py _threading_local.py threading.py  _weakrefset.py
bz2.py          crypt.py      genericpath.py getopt.py     lib_dynload   pdb.py       _phello__     sre_constants.py timeit.py      webdriver.py
calendar.py    csv.py        getpass.py    gettext.py    locale.py     pickle.py     quopri.py      sre_parse.py   tkinter       xdrlib.py
cgi.py          ctypes.py     gettext.py    glob.py      logging     pipes.py     random.py     ssl.py        tokenize.py   xml
cgib.py         curses.py    glob.py      graphlib.py lzma.py      pkgutil.py    reprlib.py    stat.py       token.py      xmlrpc
chunk.py        dataclasses.py graphlib.py gzip.py      mailbox.py platform.py rlcompleter.py stringprep.py tomllib      zipfile
cmd.py          datetime.py   gzip.py      hashlib.py mailcap.py  plistlib.py  runpy.py      string.py    traceback.py tracemalloc.py zipimport.py
codecs.py       dbm.py        hashlib.py heapq.py     _markupbase.py poplib.py    sched.py      _strptime.py trace.py      zoneinfo
codeop.py       decimal.py   heapq.py     __hello__.py mimetypes.py posixpath.py secrets.py   struct.py      tty.py       tutledemo
code.py         difflib.py   __hello__.py hmac.py      modulefinder.py pprint.py    selectors.py subprocess.py turtle.py      types.py
collections    dis.py       hmac.py      html        multiprocessing profile.py shelf.py     sunau.py      symtable.py  typing.py
_collections_abc.py doctest.py  html        http       netrc.py      pstats.py    shlex.py     shutil.py    _sysconfigdata_linux_x86_64-linux-gnu.py
colorsys.py    email.py     http        idlelib    nntplib.py    pty.py      sunau.py      symtable.py  types.py
_compat_pickle.py encodings  idlelib    nntplib.py nntplib.py    pty.py      shlex.py     shutil.py    _sysconfigdata_linux_x86_64-linux-gnu.py
Singularity> ls -l /opt/python/pkgs/python-3.12.11/lib/python3.12/site-packages/ | wc -l
1337
```

E4S NVIDIA Blackwell x86_64 Rocky Linux 9.6 Python Packages

```
GPU 0: NVIDIA RTX PRO 6000 Blackwell Server Edition (UUID: GPU-0a3e5316-2d97-bf04-1ed7-011560029cf3)
Singularity> which adk
/opt/python/pkg/python-3.12.11/bin/adk
Singularity> which jupyter
/opt/python/pkg/python-3.12.11/bin/jupyter
Singularity> which vlm
/opt/python/pkg/python-3.12.11/bin/vlm
Singularity> which marimo
/opt/python/pkg/python-3.12.11/bin/marimo
Singularity>
Singularity> python
Python 3.12.11 (main, Nov  4 2025, 19:11:59) [GCC 13.3.1 20240611 (Red Hat 13.3.1-2)] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> import torch
>>> import nemo
>>> import bionemo
>>> import jax
>>> import openai
>>> import google.genai
>>> import huggingface_hub
>>> import polars
>>> import tensorflow
>>> import pandas
>>> import sklearn
>>> import cv2
>>> import dsi
>>> import zarr
>>>
>>> import matplotlib
>>> import plotly
>>> import seaborn
>>> import mpi4py
>>> import numpy
>>> import scipy
>>> import polars
>>>
>>> torch.cuda.get_arch_list()
['sm_70', 'sm_75', 'sm_80', 'sm_86', 'sm_90', 'sm_100', 'sm_120', 'compute_120']
>>> torch.cuda.is_available()
True
>>>
Singularity> which nvcc
/usr/local/cuda/bin/nvcc
Singularity> nvcc --version
nvcc: NVIDIA (R) Cuda compiler driver
Copyright (c) 2005-2025 NVIDIA Corporation
Built on Tue_May_27_02:21:03_PDT_2025
Cuda compilation tools, release 12.9, V12.9.86
Build cuda_12.9.r12.9/compiler.36037853_0
Singularity> which codium
/usr/bin/codium
```

E4S: NVIDIA Grace-Hopper (CUDA 90) Ubuntu 24.04 LTS image

```
$ singularity run --nv e4s-cuda90-aarch64-25.11.sif
Singularity> spack find -x
-- linux-ubuntu24.04-aarch64 / %c,cxx,fortran=gcc@13.3.0 -----
adios2@2.10.2 axom@0.10.1 heffte@2.4.1 libceed@0.12.0 paraview@5.13.3 slate@2025.05.28 strumpack@8.0.0 superlu-dist@9.1.0 trilinos@16.1.0 zfp@1.0.1
amrex@25.10 caliper@2.12.1 hypre@2.33.0 magma@2.9.0 petsc@3.24.0 slepc@3.24.0 sundials@7.5.0 tau@2.35 umpire@2025.03.0

-- linux-ubuntu24.04-aarch64 / %c,cxx=gcc@13.3.0 -----
cabana@0.7.0 chapel@2.6.0 fftx@1.2.0 ginkgo@1.10.0 hpctoolkit@2025.0.1 legion@25.03.0 mgard@compat-2023-12-09 raja@2025.03.0 vtk-m@2.3.0
chai@2025.03.0 cusz@0.14.0 flecsi@2.4.1 gromacs@2025.3 lammps@20250722 libpressio@0.99.4 omega-h@10.8.6-scorec upcxx@2023.9.0

-- linux-ubuntu24.04-aarch64 / %c=gcc@13.3.0 -----
parsec@4.0.2411

-- linux-ubuntu24.04-aarch64 / %cxx,fortran=gcc@13.3.0 -----
tasmanian@8.1

-- linux-ubuntu24.04-aarch64 / %cxx=gcc@13.3.0 -----
arbortex@1.5 hpx@1.11.0 kokkos@4.7.01 kokkos-kernels@4.7.01 mfem@4.8.0

-- linux-ubuntu24.04-aarch64 / no compilers -----
e4s-alc@1.0.3 e4s-cl@1.0.5 mpich@4.3.1
=> 46 installed packages
Singularity>
exit
$ singularity run --nv e4s-cpu-aarch64-25.11.sif
Singularity> spack find -x
-- linux-ubuntu24.04-aarch64 / %c,cxx,fortran=gcc@13.3.0 -----
adios@1.13.1 axom@0.10.1 darshan-util@3.4.7 h5bench@1.4 papi@7.2.0 plumed@2.9.2 slate@2025.05.28 tau@2.35 zfp@1.0.1
adios2@2.10.2 butterlypack@3.2.0 datatransferkit@3.1.1 heffte@2.4.1 parallel-netcdf@1.14.1 precice@3.3.0 slepc@3.24.0 trilinos@16.1.0
alquimia@1.1.0 caliper@2.12.1 exago@1.6.0 libcatayst@2.0.0 paraview@5.13.3 quantum-espresso@7.5 strumpack@8.0.0 umpire@2025.03.0
amrex@25.10 conduit@0.9.5 globalarray@5.8.2 libceed@0.12.0 petsc@3.24.0 rempi@1.1.0 sundials@7.5.0 wps@4.5
ascent@0.9.5 darshan-runtime@3.4.7 gptune@4.0.0 openmpi@5.0.8 phist@1.12.1 scr@3.1.0 superlu-dist@9.1.0 xyce@7.10.0

-- linux-ubuntu24.04-aarch64 / %c,cxx=gcc@13.3.0 -----
boost@1.88.0 faodel@1.2108.1 glvis@4.4 hdf5-vol-log@1.4.0 libunwind@1.7.2 nco@5.3.4 pdt@3.25.2 swig@4.0.2-fortran upcxx@2023.9.0
cabana@0.7.0 fftx@1.2.0 gmp@6.3.0 hpctoolkit@2025.0.1 metall@0.30 omega-h@10.8.6-scorec pruners-ninja@1.0.1 sz@2.1.12.5 veloc@1.7
chai@2025.03.0 flecsi@2.4.1 gotcha@1.0.8 lammps@20250722 mgard@compat-2023-12-09 openfoam@2412 pumi@2.2.9 sz3@3.2.0 vtk-m@2.3.0
chapel@2.6.0 gasnet@2025.8.0 gromacs@2025.3 lbann@0.104 mpfileutils@0.12 openpmf-api@0.16.1 qthreads@1.18 turbine@1.3.0 warpx@25.04
dyninst@13.0.0 ginkgo@1.10.0 hdf5-vol-cache@v1.1 legion@25.03.0 nccmp@1.9.1.0 papyrus@1.0.2 raja@2025.03.0 umap@2.1.1

-- linux-ubuntu24.04-aarch64 / %c,fortran=gcc@13.3.0 -----
fpm@0.10.0 hypre@2.33.0 nek5000@19.0 netcdf-fortran@4.6.2 plasma@24.8.7 py-petsc4py@3.24.0 wannier90@3.1.0
hdf5@1.14.6 libquo@1.4 nekbone@17.0 netlib-scalapack@2.2.2 py-libensembl@1.5.0 superlu@7.0.0 wrf@4.6.1

-- linux-ubuntu24.04-aarch64 / %c=gcc@13.3.0 -----
aml@0.2.1 argobots@1.2 charliecloud@0.40 hdf5-vol-async@1.7 libnrm@0.1.0 parsec@4.0.2411 py-h5py@3.14.0

-- linux-ubuntu24.04-aarch64 / %cxx,fortran=gcc@13.3.0 -----
tasmanian@8.1

-- linux-ubuntu24.04-aarch64 / %cxx=gcc@13.3.0 -----
arbortex@2.0.1 flit@2.1.0 hpx@1.11.0 kokkos@4.7.01 kokkos-kernels@4.7.01 laghos@3.1 loki@0.1.7 mfem@4.8.0 mpack-variant@1.4.0

-- linux-ubuntu24.04-aarch64 / no compilers -----
e4s-alc@1.0.3 e4s-cl@1.0.5 mpich@4.3.1 nrm@0.1.0 py-cinemasci@1.7.0 py-jupyterhub@1.4.1 stc@0.9.0
=> 123 installed packages
Singularity>
```

E4S: Rocky Linux 9.6 x86_64 CPU image

```
$ singularity run e4s-cpu-rocky9.6-x86_64-25.11.sif
Singularity>
Singularity> cat /etc/os-release | grep PRETTY_NAME
PRETTY_NAME="Rocky Linux 9.6 (Blue Onyx)"
Singularity>
Singularity> spack find -x
-- linux-rocky9-x86_64_v3 / %c,cxx,fortran=gcc@13.3.1 -----
adios@1.13.1    axom@0.10.1      darshan-runtime@3.4.7  gptune@4.0.0      openmpi@5.0.8      plumed@2.9.2      slate@2025.05.28   tau@2.35       xyce@7.10.0
adios2@2.10.2   butterflypack@3.2.0  darshan-util@3.4.7  h5bench@1.4      papi@7.2.0       precice@3.3.0     slepc@3.24.0      trilinos@16.1.0  zfp@1.0.1
alquimia@1.1.0  caliper@2.12.1    datatransferkit@3.1.1 heffte@2.4.1      parallel-netcdf@1.14.1 quantum-espresso@7.5  strumpack@8.0.0    umpire@2025.03.0
amrex@25.10    conduit@0.9.5      exago@1.6.0       libcatalyst@2.0.0  petsc@3.24.0     rempi@1.1.0      sundials@7.5.0     variorum@0.8.0
ascent@0.9.5   cp2k@2025.2      globalarrays@5.8.2  libceed@0.12.0    phist@1.12.1     scr@3.1.0       superlu-dist@9.1.0 wps@4.5
-- linux-rocky9-x86_64_v3 / %c,cxx=gcc@13.3.1 -----
boost@1.88.0    dyninst@13.0.0    ginkgo@1.10.0    hdf5-vol-cache@v1.1  legion@25.03.0    nccmp@1.9.1.0    papyrus@1.0.2     raja@2025.03.0  umap@2.1.1
bricks@2023.08.25 faodel@1.2108.1  glvis@4.4      hdf5-vol-log@1.4.0  libunwind@1.8.3    nco@5.3.4       pdt@3.25.2      swig@4.0.2-fortran upcxx@2023.9.0
cabana@0.7.0    fftx@1.2.0       gmp@6.3.0       hpctoolkit@2025.0.1 metall@0.30      omega-h@10.8.6-scorec pruners-ninja@1.0.1 sz@2.1.12.5
chai@2025.03.0  flecsi@2.4.1    gotcha@1.0.8    lammps@20250722  mgard@compat-2023-12-09 openfoam@2412      pumi@2.2.9      sz3@3.2.0
chapel@2.6.0   gasnet@2025.8.0  gromacs@2025.3  lbann@0.104     mpfileutils@0.12  openpmd-api@0.16.1 qthreads@1.18     turbine@1.3.0
warpx@25.04
-- linux-rocky9-x86_64_v3 / %c,fortran=gcc@13.3.1 -----
fpm@0.10.0     hypre@2.33.0    nek5000@19.0    netcdf-fortran@4.6.2  plasma@24.8.7      py-petsc4py@3.24.0  wannier90@3.1.0
hdf5@1.14.6    libquo@1.4      nekbone@17.0    netlib-scalapack@2.2.2 py-libensemble@1.5.0  superlu@7.0.0     wrf@4.6.1
-- linux-rocky9-x86_64_v3 / %c=gcc@13.3.1 -----
aml@0.2.1      argobots@1.2   charliecloud@0.40  hdf5-vol-async@1.7  libnrm@0.1.0     parsec@4.0.2411   py-h5py@3.14.0
tasmanian@8.1
-- linux-rocky9-x86_64_v3 / %cxx,fortran=gcc@13.3.1 -----
arborx@2.0.1    flit@2.1.0     hpx@1.11.0     kokkos@4.7.01    kokkos-kernels@4.7.01 laghos@3.1    loki@0.1.7    mfem@4.8.0    mpark-variant@1.4.0
-- linux-rocky9-x86_64_v3 / no compilers -----
e4s-alc@1.0.3   e4s-cl@1.0.5   mpich@4.3.1    nrm@0.1.0      py-cinemasci@1.7.0  py-jupyterhub@1.4.1  stc@0.9.0
=> 125 installed packages
Singularity> █
```

E4S: Rocky Linux 9.6 x86_64 CPU image

```
Singularity> module avail
```

/spack/share/spack/modules/linux-rocky9-x86_64_v3							
adios/1.13.1	cp2k/2025.2-openmp	gotcha/1.0.8	libcatalyst/2.0.0	nrm/0.1.0	py-h5py/3.14.0	sz3/3.2.0	
adios2/2.10.2	darshan-runtime/3.4.7	gptune/4.0.0	libceed/0.12.0	omega-h/10.8.6-scorec	py-jupyterhub/1.4.1	tasmanian/8.1	
alquimia/1.1.0	darshan-util/3.4.7	gromacs/2025.3-openmp	libnrm/0.1.0	openfoam/2412	py-libensemble/1.5.0	tau/2.35	
aml/0.2.1	datatransferkit/3.1.1	h5bench/1.4	libquo/1.4	openmpi/5.0.8	py-petsc4py/3.24.0	trilinos/16.1.0	
amrex/25.10	dyninst/13.0.0-openmp	hdf5-vol-async/1.7	libunwind/1.8.3	openpmd-api/0.16.1	qthreads/1.18	turbine/1.3.0	
arbortex/2.0.1	e4s-alc/1.0.3	hdf5-vol-cache/v1.1	loki/0.1.7	papi/7.2.0	quantum-espresso/7.5-openmp	umap/2.1.1	
argobots/1.2	e4s-cl/1.0.5	hdf5-vol-log/1.4.0	metall/0.30	papyrus/1.0.2	raja/2025.03.0	umpire/2025.03.0	
ascent/0.9.5-openmp	exago/1.6.0	hdf5/1.14.6	mfem/4.8.0	parallel-netcdf/1.14.1	rempi/1.1.0	upcxx/2023.9.0	
axom/0.10.1-openmp	faodel/1.2108.1	heffte/2.4.1	mgard/compat-2023-12-09-openmp	parsec/4.0.2411	scr/3.1.0	variorum/0.8.0	
boost/1.88.0	fftx/1.2.0	hpctoolkit/2025.0.1	mpark-variant/1.4.0	pdt/3.25.2	slate/2025.05.28-openmp	veloc/1.7	
bricks/2023.08.25	flecsi/2.4.1	hpx/1.11.0	mpich/4.3.1	petsc/3.24.0	slepc/3.24.0	vtk-m/2.3.0-openmp	
butterflypack/3.2.0-openmp	flit/2.1.0	hypre/2.33.0	mpifileutils/0.12	phist/1.12.1-openmp	stc/0.9.0	wannier90/3.1.0	
cabana/0.7.0	fpm/0.10.0-openmp	kokkos-kernels/4.7.01-openmp	nccmp/1.9.1.0	plasma/24.8.7	strumpack/8.0.0-openmp	warpx/25.04	
caliper/2.12.1	gasnet/2025.8.0	kokkos/4.7.01-openmp	nco/5.3.4-openmp	plumed/2.9.2	sundials/7.5.0	wps/4.5	
chai/2025.03.0	ginkgo/1.10.0-openmp	laghos/3.1	nek5000/19.0	precice/3.3.0	superlu-dist/9.1.0	wrf/4.6.1	
chapel/2.6.0	globalarrays/5.8.2	lammps/20250722-openmp	nekbone/17.0	pruners-ninja/1.0.1	superlu/7.0.0	xyce/7.10.0	
charliecloud/0.40	glvis/4.4	lbann/0.104	netcdf-fortran/4.6.2	pumi/2.2.9	swig/4.0.2-fortran	zfp/1.0.1	
conduit/0.9.5	gmp/6.3.0	legion/25.03.0	netlib-scalapack/2.2.2	py-cinemasci/1.7.0	sz/2.1.12.5		

Key:

loaded modulepath

Singularity> █



E4S Intel oneAPI Ubuntu 24.04 LTS CPU/GPU image

```
$ singularity run e4s-oneapi-x86_64-25.11.sif
Singularity> spack find -x
-- linux-ubuntu24.04-x86_64_v3 / %c,cxx,fortran=gcc@13.3.0 -----
adios@1.13.1    axom@0.10.1      darshan-runtime@3.4.7  gptune@4.0.0      openmpi@5.0.8      plumed@2.9.2      slate@2025.05.28  tau@2.35       xyce@7.10.0
adios2@2.10.2   butterflypack@3.2.0  darshan-util@3.4.7  h5bench@1.4      papi@7.2.0       precice@3.3.0     slepc@3.24.0     trilinos@16.1.0  zfp@1.0.1
alquimia@1.1.0  caliper@2.12.1    datatransferkit@3.1.1 heffte@2.4.1      parallel-netcdf@1.14.1 quantum-espresso@7.5 strumpack@8.0.0     umpire@2025.03.0
amrex@25.10    conduit@0.9.5     exago@1.6.0       libcatalyst@2.0.0  petsc@3.24.0     rempi@1.1.0      sundials@7.5.0     variorum@0.8.0
ascent@0.9.5   cp2k@2025.2     globalarrays@5.8.2  libceed@0.12.0    phist@1.12.1    scr@3.1.0       superlu-dist@9.1.0 wps@4.5

-- linux-ubuntu24.04-x86_64_v3 / %c,cxx,fortran=oneapi@2025.2.1 -
amrex@25.10    heffte@2.4.1     petsc@3.24.0     sundials@7.5.0     tau@2.35       tau@2.35

-- linux-ubuntu24.04-x86_64_v3 / %c,cxx=gcc@13.3.0 -----
boost@1.88.0    dyninst@13.0.0   ginkgo@1.10.0    hdf5-vol-cache@v1.1  legion@25.03.0    nccmp@1.9.1.0    papyrus@1.0.2    raja@2025.03.0  umap@2.1.1
bricks@2023.08.25 faodel@1.2108.1  glvis@4.4      hdf5-vol-log@1.4.0   libunwind@1.8.3   nco@5.3.4       pdt@3.25.2      swig@4.0.2-fortran upcxx@2023.9.0
cabana@0.7.0    fftx@1.2.0     gmp@6.3.0      hpctoolkit@2025.0.1 metall@0.30    omega-h@10.8.6-scorec pruners-ninja@1.0.1 sz@2.1.12.5
chai@2025.03.0  flecsi@2.4.1    gotcha@1.0.8    lammps@20250722   mgard@compat-2023-12-09 openfoam@2412    pumi@2.2.9      sz3@3.2.0      vtk-m@2.3.0
chapel@2.6.0    gasnet@2025.8.0  gromacs@2025.3  lbann@0.104     mpfileutils@0.12  openpmd-api@0.16.1 qthreads@1.18    turbine@1.3.0  warpx@25.04

-- linux-ubuntu24.04-x86_64_v3 / %c,cxx=oneapi@2025.2.1 -----
cabana@0.7.0    ginkgo@1.10.0   hpctoolkit@2025.0.1 upcxx@2023.9.0

-- linux-ubuntu24.04-x86_64_v3 / %c,fortran=gcc@13.3.0 -----
fpm@0.10.0     hypre@2.33.0    nek5000@19.0    netcdf-fortran@4.6.2 plasma@24.8.7    py-petsc4py@3.24.0  wannier90@3.1.0
hdf5@1.14.6    libquo@1.4     nekbone@17.0    netlib-scalapack@2.2.2 py-libensembl@1.5.0 superlu@7.0.0     wrf@4.6.1

-- linux-ubuntu24.04-x86_64_v3 / %c=gcc@13.3.0 -----
aml@0.2.1      argobots@1.2   charliecloud@0.40  hdf5-vol-async@1.7  libnrm@0.1.0     parsec@4.0.2411   py-h5py@3.14.0

-- linux-ubuntu24.04-x86_64_v3 / %c=oneapi@2025.2.1 -----
aml@0.2.1

-- linux-ubuntu24.04-x86_64_v3 / %cxx,fortran=gcc@13.3.0 -----
tasmanian@8.1

-- linux-ubuntu24.04-x86_64_v3 / %cxx=gcc@13.3.0 -----
arborx@2.0.1    flit@2.1.0     hpx@1.11.0     kokkos@4.7.01   kokkos-kernels@4.7.01 laghos@3.1    loki@0.1.7    mfem@4.8.0    mpark-variant@1.4.0

-- linux-ubuntu24.04-x86_64_v3 / %cxx=oneapi@2025.2.1 -----
arborx@2.0.1    kokkos@4.7.01   kokkos-kernels@4.7.01

-- linux-ubuntu24.04-x86_64_v3 / no compilers -----
e4s-alc@1.0.3   e4s-cl@1.0.5   intel-oneapi-mpi@2021.16.0 nrm@0.1.0     py-cinemasci@1.7.0  py-jupyterhub@1.4.1  stc@0.9.0
=> 139 installed packages
Singularity>
```

E4S IBM ppc64le Power image with GPU

```
$ singularity run e4s-cuda70-ppc64le-25.11.sif
Singularity> lscpu | grep POWER
Model name:          POWER10 (architected), altivec supported
Singularity> spack find +cuda
-- linux-ubuntu20.04-ppc64le / %c,cxx,fortran=gcc@9.4.0 -----
amrex@25.10  caliper@2.12.1  heffte@2.4.1  hypre@2.33.0  paraview@5.13.3  petsc@3.24.0  slepc@3.24.0  sundials@7.5.0  tau@2.35  umpire@2025.03.0  umpire@2025.03.0
axom@0.10.1  exago@1.6.0  hiop@1.0.0  magma@2.9.0  petsc@3.24.0  slate@2025.05.28  strumpack@8.0.0  superlu-dist@9.1.0  umpire@6.0.0  umpire@2025.03.0  zfp@1.0.1

-- linux-ubuntu20.04-ppc64le / %c,cxx=gcc@9.4.0 -----
bricks@2023.08.25  camp@0.2.3  camp@2025.03.0  fftx@1.2.0  ginkgo@1.10.0  hwloc@2.12.2  mgard@compat-2023-12-09  omega-h@10.8.6-scorec  raja@2025.03.0  upcxx@2023.9.0
cabana@0.7.0  camp@2025.03.0  chai@2025.03.0  flecsi@2.4.1  hpctoolkit@2024.01.1  lammps@20250722  nvcomp@2.2.0  raja@0.14.0  raja@2025.03.0  vtk-m@2.3.0

-- linux-ubuntu20.04-ppc64le / %c=gcc@9.4.0 -----
flux-core@0.73.0  parsec@4.0.2411

-- linux-ubuntu20.04-ppc64le / %cxx,fortran=gcc@9.4.0 -----
tasmanian@8.1

-- linux-ubuntu20.04-ppc64le / %cxx=gcc@9.4.0 -----
blaspp@2025.05.28  hpx@1.11.0  kokkos@4.6.02  kokkos@4.7.01  kokkos-kernels@4.7.01  lapackpp@2025.05.28  mfem@4.8.0
=> 53 installed packages
Singularity> spack find -x
-- linux-ubuntu20.04-ppc64le / %c,cxx,fortran=gcc@9.4.0 -----
adios@1.13.1  axom@0.10.1  darshan-util@3.4.7  h5bench@1.4  nwchem@7.2.3  petsc@3.24.0  slate@2025.05.28  sundials@7.5.0  trilinos@16.1.0  zfp@1.0.1
alquimia@1.1.0  butterflypack@3.2.0  datatransferkit@3.1.1  heffte@2.4.1  openmpi@5.0.8  petsc@3.24.0  slate@2025.05.28  sundials@7.5.0  umpire@2025.03.0
amrex@25.10  caliper@2.12.1  exago@1.6.0  heffte@2.4.1  papi@7.2.0  plumed@2.9.2  slepc@3.24.0  superlu-dist@9.1.0  umpire@2025.03.0
amrex@25.10  caliper@2.12.1  exago@1.6.0  hypre@2.33.0  parallel-netcdf@1.14.1  quantum-espresso@7.5  slepc@3.24.0  superlu-dist@9.1.0  wps@4.5
ascent@0.9.5  conduit@0.9.5  globalarrays@5.8.2  libcatalyst@2.0.0  paraview@5.13.3  rempi@1.1.0  strumpack@8.0.0  tau@2.35  xce@7.10.0
axom@0.10.1  darshan-runtime@3.4.7  gptune@4.0.0  magma@2.9.0  paraview@5.13.3  scr@3.1.0  strumpack@8.0.0  tau@2.35  zfp@1.0.1

-- linux-ubuntu20.04-ppc64le / %c,cxx=gcc@9.4.0 -----
bolt@2.0  chai@2025.03.0  fftx@1.2.0  gmp@6.3.0  lammps@20250722  metall@0.30  omega-h@10.8.6-scorec  qthreads@1.18  turbine@1.3.0  vtk-m@2.3.0
boost@1.88.0  chai@2025.03.0  flecsi@2.4.1  gotcha@1.0.8  lammps@20250722  mgard@compat-2023-12-09  omega-h@10.8.6-scorec  raja@2025.03.0  umap@2.1.1
bricks@2023.08.25  chapel@2.6.0  gasnet@2025.8.0  hdf5-vol-cache@v1.1  lbann@0.104  mgard@compat-2023-12-09  papyrus@1.0.2  raja@2025.03.0  upcxx@2023.9.0
bricks@2023.08.25  dyninst@13.0.0  ginkgo@1.10.0  hdf5-vol-log@1.4.0  legion@25.03.0  mpfileutils@0.12  pdt@3.25.2  swig@4.0.2-fortran  upcxx@2023.9.0
cabana@0.7.0  faodel@1.2108.1  ginkgo@1.10.0  hpctoolkit@2024.01.1  libpressio@0.99.4  nccmp@1.9.1.0  pruners-ninja@1.0.1  sz@2.1.12.5  veloc@1.7
cabana@0.7.0  fftx@1.2.0  glvis@4.4  hpctoolkit@2024.01.1  libunwind@1.7.2  nco@5.3.4  pumi@2.2.9  sz3@3.2.0  vtk-m@2.3.0

-- linux-ubuntu20.04-ppc64le / %c,fortran=gcc@9.4.0 -----
fpm@0.10.0  hypre@2.33.0  nek5000@19.0  netcdf-fortran@4.6.2  plasma@24.8.7  py-petsc4py@3.24.0  wannier90@3.1.0
hdf5@1.14.6  libquo@1.4  nekbone@17.0  netlib-scalapack@2.2.2  py-libensemble@1.5.0  superlu@7.0.0  wrf@4.6.1

-- linux-ubuntu20.04-ppc64le / %cxx,gcc@9.4.0 -----
aml@0.2.1  argobots@1.2  charliecloud@0.40  flux-core@0.73.0  flux-core@0.73.0  hdf5-vol-async@1.7  libnrm@0.1.0  parsec@4.0.2411  parsec@4.0.2411  py-h5py@3.14.0

-- linux-ubuntu20.04-ppc64le / %cxx,fortran=gcc@9.4.0 -----
fortrilinos@2.3.0  tasmanian@8.1  tasmanian@8.1

-- linux-ubuntu20.04-ppc64le / %cxx=gcc@9.4.0 -----
arbortex@2.0.1  flit@2.1.0  hpx@1.11.0  hpx@1.11.0  kokkos@4.7.01  kokkos-kernels@4.7.01  kokkos-kernels@4.7.01  laghos@3.1  loki@0.1.7  mfem@4.8.0  mfem@4.8.0  mpark-variant@1.4.0

-- linux-ubuntu20.04-ppc64le / no compilers -----
e4s-alc@1.0.3  e4s-cl@1.0.5  exaworks@0.1.0  mpich@4.3.1  nrm@0.1.0  py-cinemasci@1.7.0  py-jupyterhub@1.4.1  stc@0.9.0
=> 158 installed packages
Singularity> █
```

E4S: AMD MI300A GPU support with ROCm 6.4.3

```
$ singularity run e4s-rocm942-x86_64-25.11.sif
Singularity> which hipcc
/opt/rocm/bin/hipcc
Singularity> hipcc --version
HIP version: 6.4.43484-123eb5128
AMD clang version 19.0.0git (https://github.com/RadeonOpenCompute/llvm-project roc-6.4.3 25224 d366fa84f3fdcb4b10847ebd5db572ae12a34fb)
Target: x86_64-unknown-linux-gnu
Thread model: posix
InstalledDir: /opt/rocm-6.4.3/lib/llvm/bin
Configuration file: /opt/rocm-6.4.3/lib/llvm/bin/clang++.cfg
Singularity> which adk
/opt/python/pkgs/python-3.12.11/bin/adk
Singularity> which codium
/usr/bin/codium
Singularity> which python
/opt/python/pkgs/python-3.12.11/bin/python
Singularity>
Singularity> module avail
----- /spack/share/spack/modules/linux-ubuntu24.04-x86_64_v3 -----
amrex@25.10-gfx942      e4s-alc@1.0.3          heffte@2.4.1-gfx942      libceed@0.12.0-gfx942    petsc@3.24.0-gfx942      superlu-dist@9.1.0-gfx942  upcxx@2023.9.0-gfx942
arborx@2.0.1-gfx942     e4s-cl@1.0.5          hpctoolkit@2025.0.1-rocm  magma@2.9.0-gfx942      raja@2025.03.0-gfx942      tasmanian@8.1-gfx942    vtk-m@2.3.0-gfx942
cabana@0.7.0-gfx942-roc cabana@0.7.0-gfx942   hypre@2.33.0-gfx942     mfem@4.8.0-gfx942      slepc@3.24.0-gfx942      tau@2.35-rocm
caliper@2.12.1-gfx942   gasnet@2025.8.0-gfx942 kokkos@4.7.01-gfx942    mpich@4.3.1           strumpack@8.0.0-gfx942-openmp  trilinos@16.1.0-gfx942
chai@2025.03.0-gfx942   ginkgo@1.10.0-gfx942-openmp legion@25.03.0-gfx942  papi@7.2.0-gfx942      sundials@7.5.0-gfx942    umpire@2025.03.0-gfx942
Key:
loaded modulepath
Singularity> spack find -x
-- linux-ubuntu24.04-x86_64_v3 / %c,cxx,fortran=gcc@13.3.0 -----
amrex@25.10      heffte@2.4.1  libceed@0.12.0  papi@7.2.0    slepc@3.24.0    sundials@7.5.0      tau@2.35      umpire@2025.03.0
caliper@2.12.1   hypre@2.33.0  magma@2.9.0    petsc@3.24.0  strumpack@8.0.0  superlu-dist@9.1.0  trilinos@16.1.0

-- linux-ubuntu24.04-x86_64_v3 / %c,cxx=gcc@13.3.0 -----
cabana@0.7.0     chai@2025.03.0  fftx@1.2.0    gasnet@2025.8.0  ginkgo@1.10.0  hpctoolkit@2025.0.1  legion@25.03.0  raja@2025.03.0  upcxx@2023.9.0  vtk-m@2.3.0

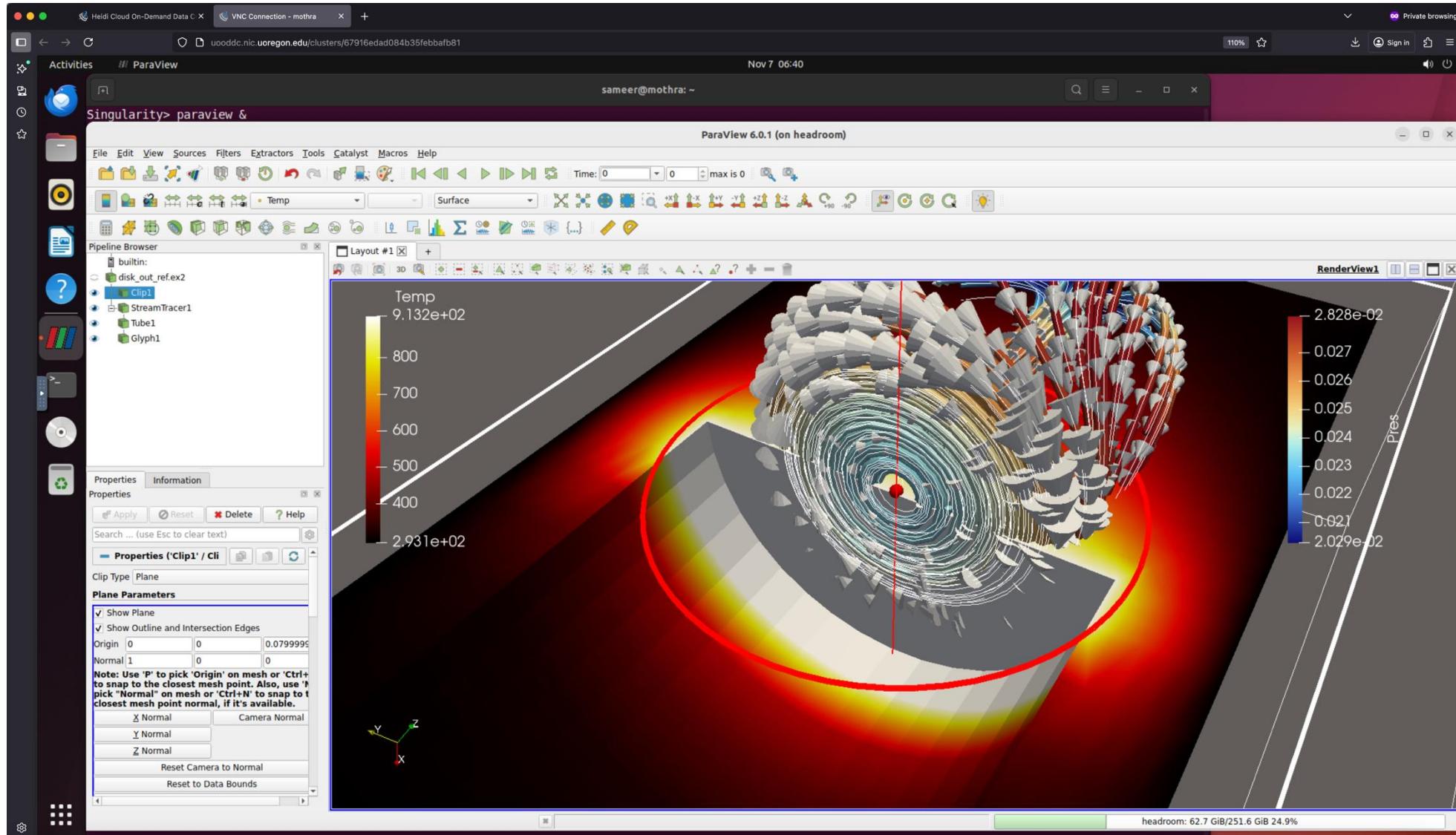
-- linux-ubuntu24.04-x86_64_v3 / %cxx,fortran=gcc@13.3.0 -----
tasmanian@8.1

-- linux-ubuntu24.04-x86_64_v3 / %cxx=gcc@13.3.0 -----
arborx@2.0.1     kokkos@4.7.01  mfem@4.8.0

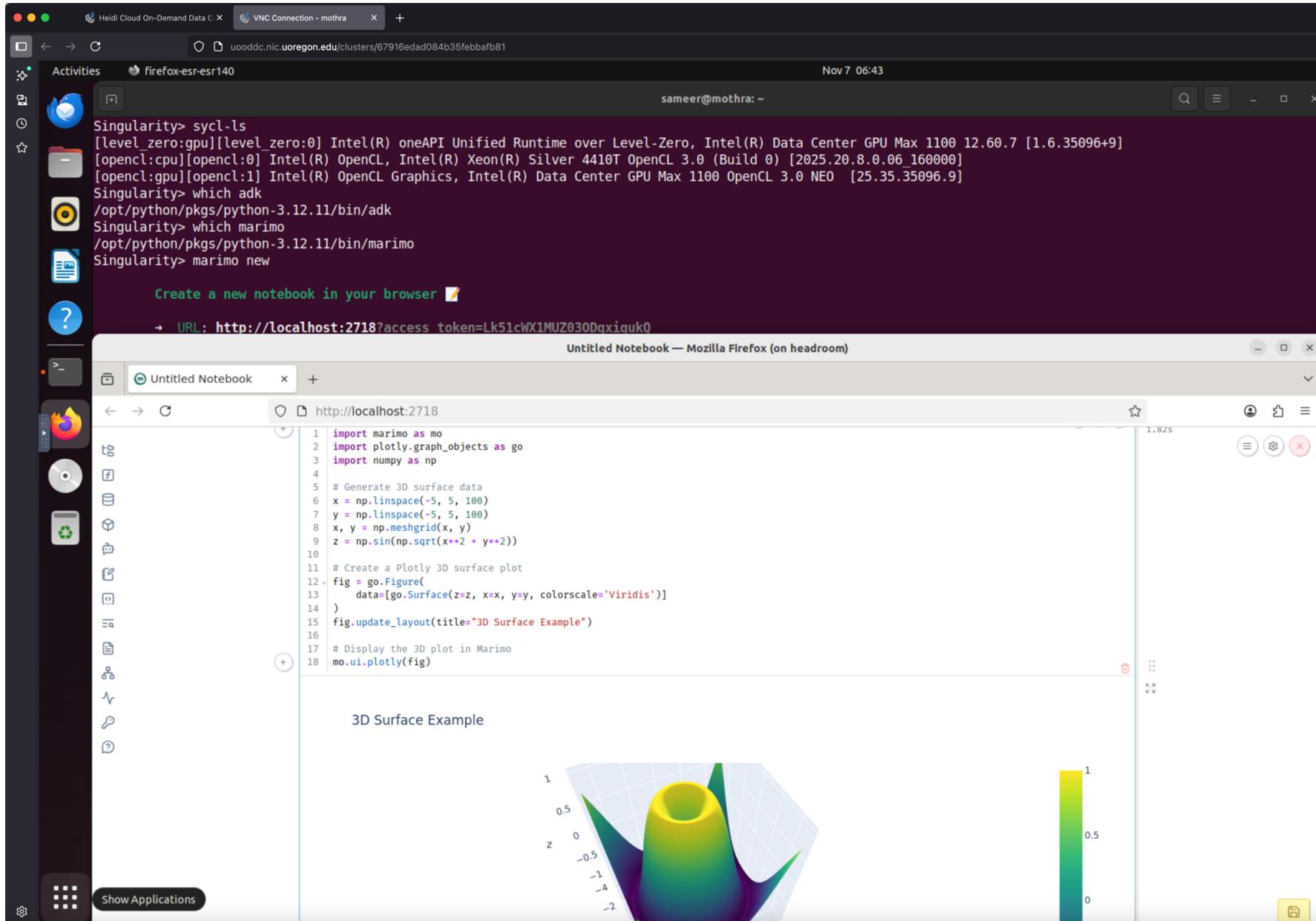
-- linux-ubuntu24.04-x86_64_v3 / no compilers -----
e4s-alc@1.0.3    e4s-cl@1.0.5  mpich@4.3.1

=> 32 installed packages
Singularity> rocminfo | grep "gfx942:"
  Name:      amdgn-amd-amdhsa--gfx942:sramecc+:xnack-
  Name:      amdgn-amd-amdhsa--gfx942:sramecc+:xnack-
  Name:      amdgn-amd-amdhsa--gfx942:sramecc+:xnack-
  Name:      amdgn-amd-amdhsa--gfx942:sramecc+:xnack-
Singularity>
```

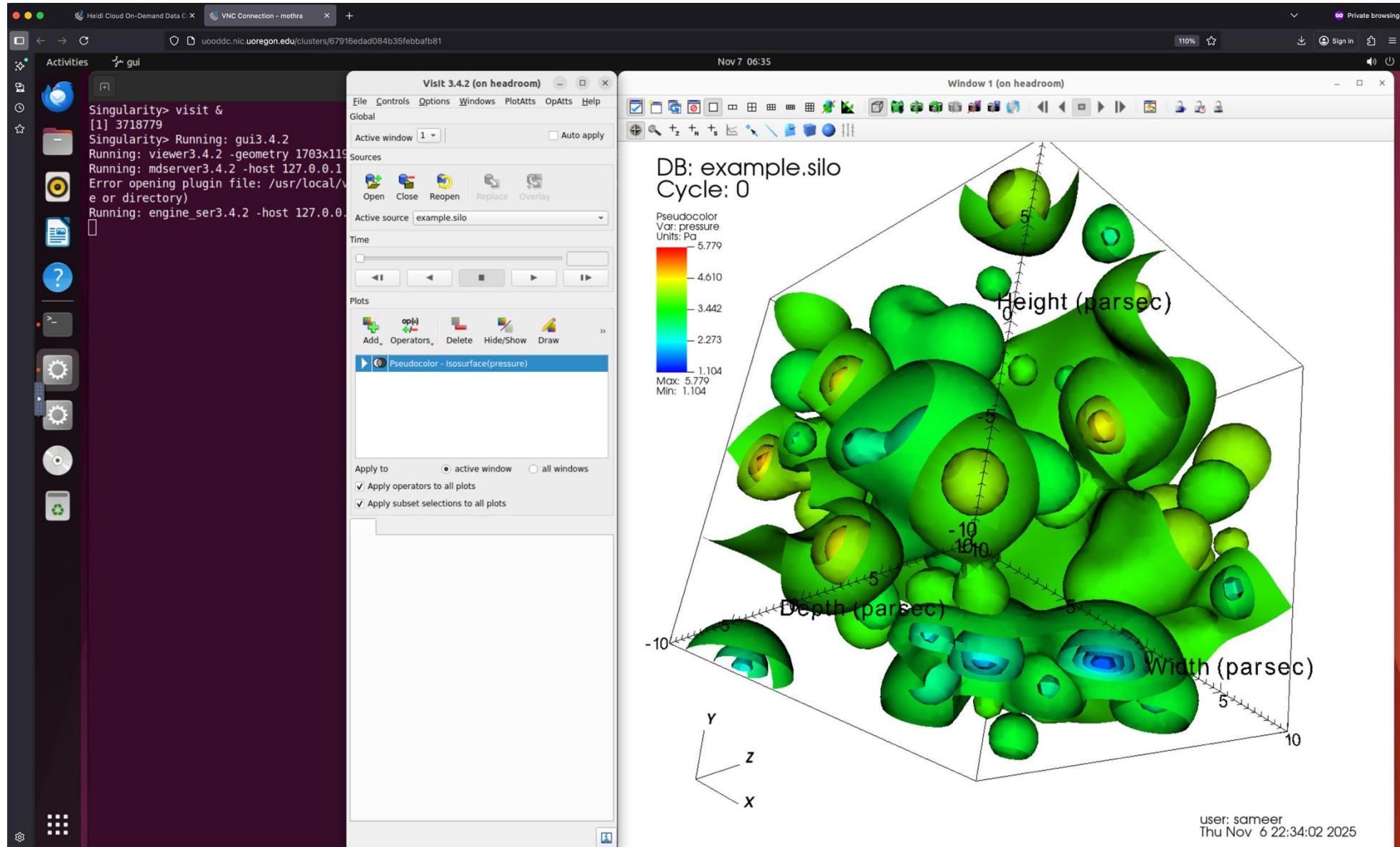
E4S: Visualization Tools: ParaView



E4S: Marimo Reactive Notebooks



E4S: VisIt



E4S: VS Codium Integrated Development Environment

The screenshot shows the VS Codium IDE running on a Linux desktop environment. The terminal window displays a Singularity session output:

```
Singularity> codium f4.py
Singularity> cat /etc/os-release | head -1
PRETTY_NAME="Ubuntu 24.04.3 LTS"
Singularity>
```

The code editor window shows a Python script named `f4.py`:

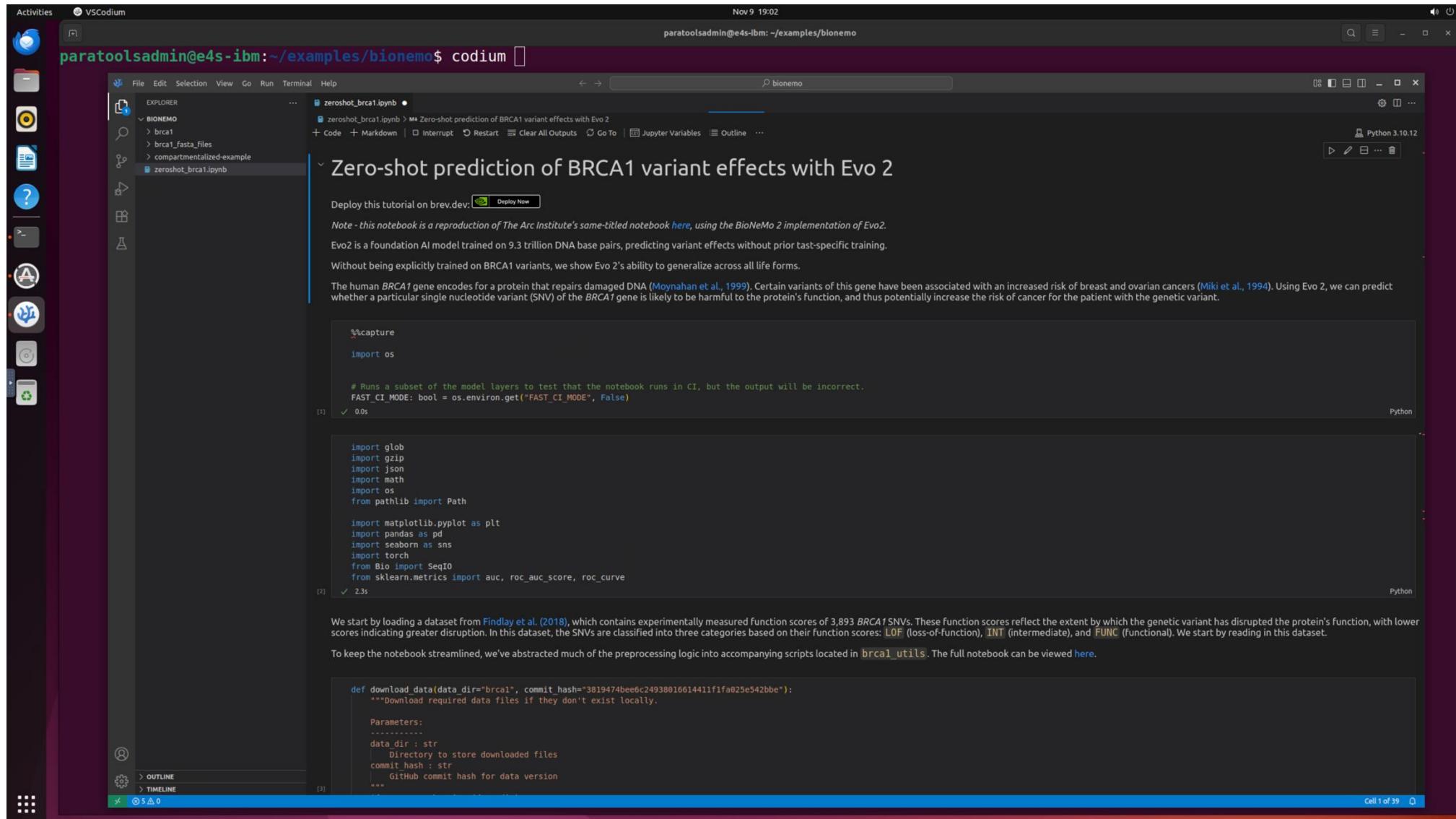
```
3 import torch
4
5
6 dtype = torch.float
7 device = torch.device("xpu")
8 print("PyTorch on xpu")
9
10 # N is batch size; D_in is input dimension;
11 # H is hidden dimension; D_out is output dimension.
12 N, D_in, H, D_out = 64, 1000, 100, 10
13
14 # Create random input and output data
15 x = torch.randn(N, D_in, device=device, dtype=dtype)
16 y = torch.randn(N, D_out, device=device, dtype=dtype)
17
18 # Randomly initialize weights
19 w1 = torch.randn(D_in, H, device=device, dtype=dtype)
20 w2 = torch.randn(H, D_out, device=device, dtype=dtype)
21
22 learning_rate = 1e-6
23 for t in range(500):
```

The terminal tab at the bottom shows the command being run and its output:

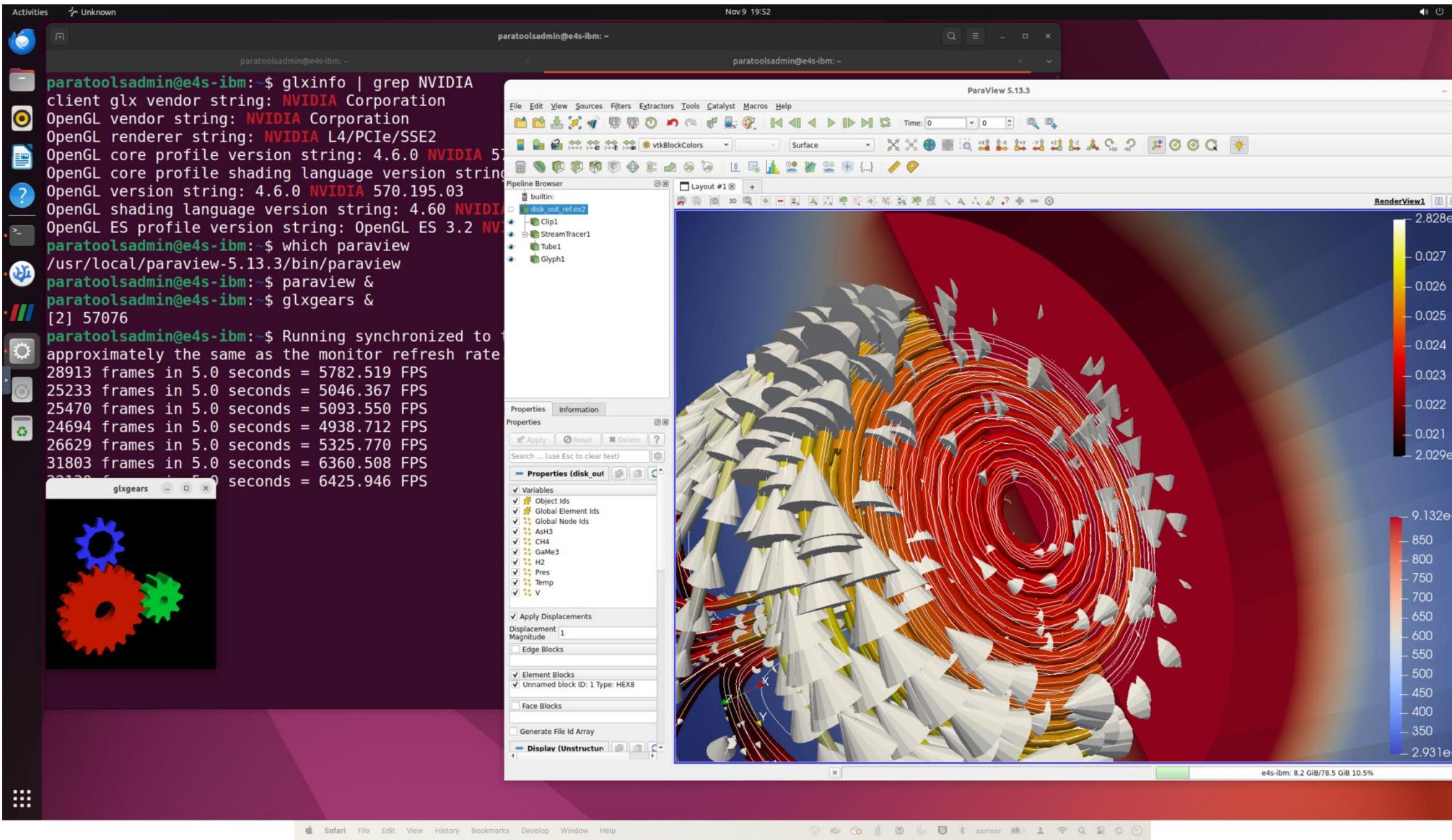
```
/opt/python/pkgs/python-3.12.11/bin/python /home/users/sameer/samples/e4s-cloud-examples/pytorch/f4.py
sameer@headroom:~$ /opt/python/pkgs/python-3.12.11/bin/python /home/users/sameer/samples/e4s-cloud-examples/pytorch/f4.py
PyTorch on xpu
99 555.7371215820312
199 2.236342191696167
299 0.018694469705224037
399 0.0004442477519623935
499 5.661723844241351e-05
sameer@headroom:~$
```

The status bar at the bottom indicates the file is `main*`, has 4 lines and 1 column, 4 spaces, and is in Python mode.

ParaTools Pro for E4S™: NVIDIA BioNeMo™ on IBM Cloud



ParaTools Pro for E4S™: HPC-AI Software Ecosystem on Clouds



Acknowledgment

- *This material is based upon work supported by the U.S. Department of Energy, Office of Science, Office of Advanced Scientific Computing Research, Next-Generation Scientific Software Technologies program, under contract numbers DE-AC02-AC05-00OR22725 and DOE SBIR DE-SC0022502.*



U.S. DEPARTMENT OF
ENERGY

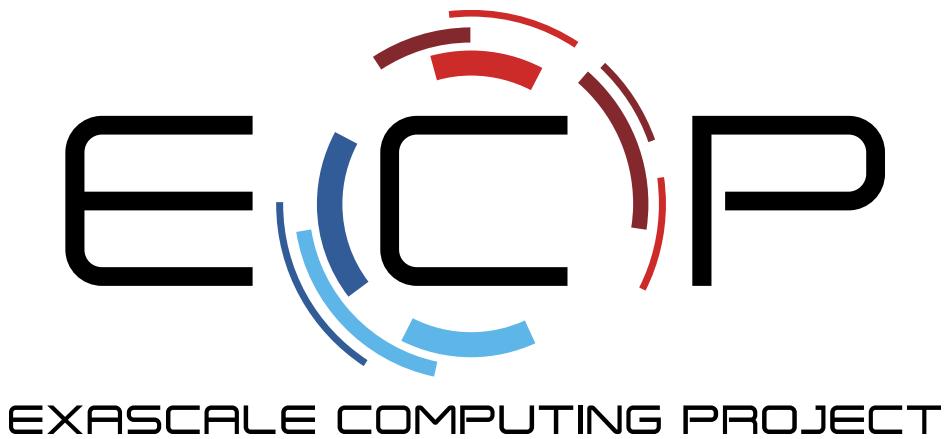
Office of
Science

- <https://science.osti.gov/ascr>
- <https://pesoproject.org>
- <https://ascr-step.org>
- <https://hpsf.io>
- <https://www.energy.gov/technologytransitions/sbirstr>

Thank you

<https://www.exascaleproject.org>

This research was supported by the Exascale Computing Project (17-SC-20-SC), a joint project of the U.S. Department of Energy's Office of Science and National Nuclear Security Administration, responsible for delivering a capable exascale ecosystem, including software, applications, and hardware technology, to support the nation's exascale computing imperative.



Thank you to all collaborators in the ECP and broader computational science communities. The work discussed in this presentation represents creative contributions of many people who are passionately working toward next-generation computational science.

