

E4S: The Extreme-scale Scientific Software Stack Release 25.11

Release 25.11 notes
November 14, 2025



HPSF
HIGH PERFORMANCE
SOFTWARE FOUNDATION

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



U.S. DEPARTMENT OF
ENERGY

Office of
Science

E4S 25.11: What's New?

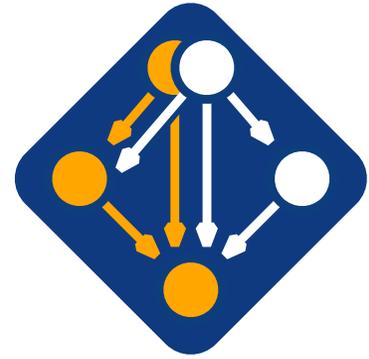


- E4S includes 125+ HPC-AI packages on aarch64, x86_64, and ppc64le platforms.
- All new website [<https://e4s.io>] with an OpenAI based chatbot to simplify access to E4S documentation.
- Support for NVIDIA Blackwell on x86_64 and aarch64 (Grace-Blackwell) architectures.
- Support for Rocky Linux 9.6 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, DeaII, 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.



- <https://adaptivecomputing.com/> and <https://paratoolspro.com>

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



HPSF
HIGH PERFORMANCE
SOFTWARE FOUNDATION

E4S: Extreme-scale Scientific Software Stack



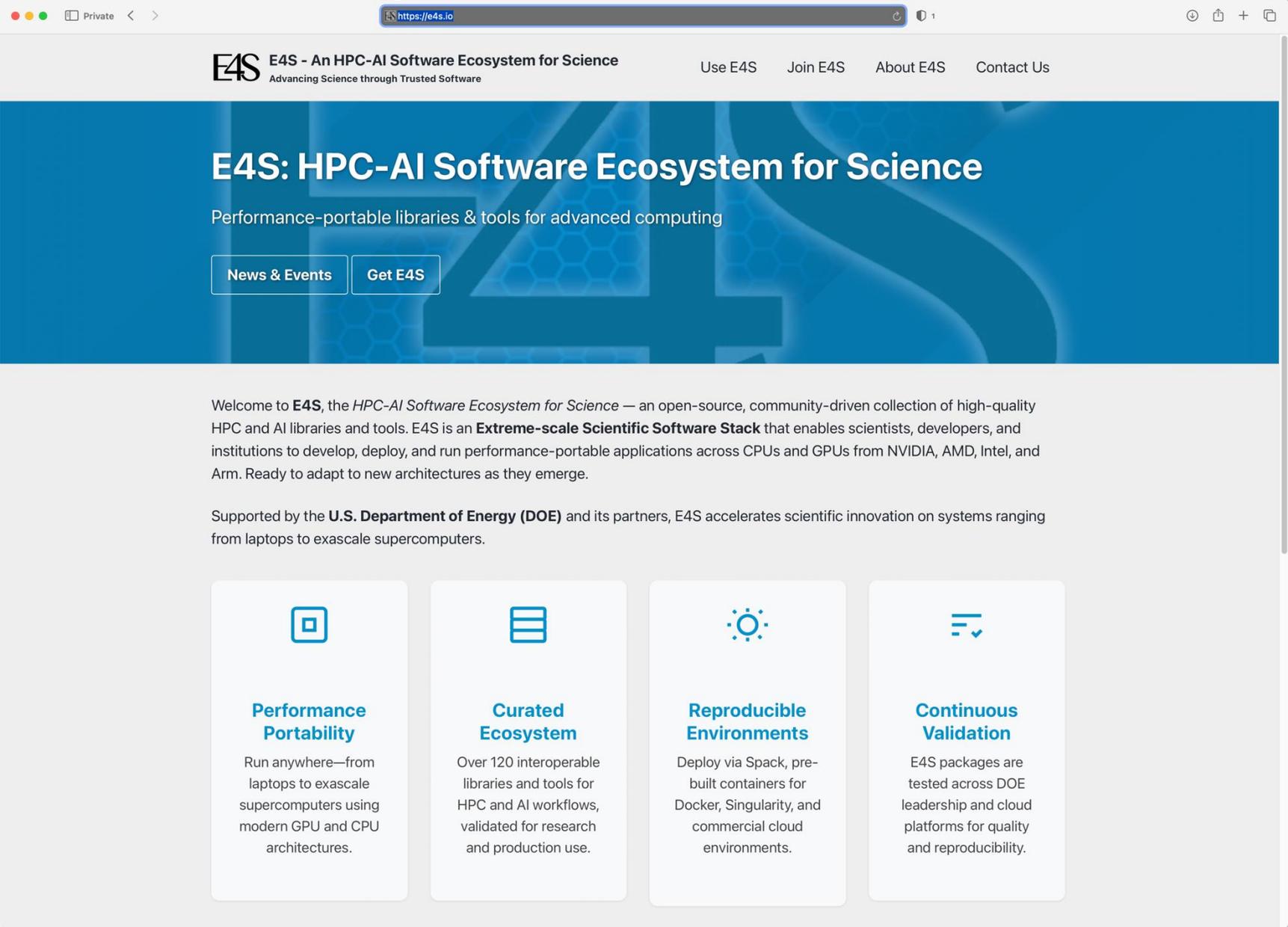
<https://e4s.io>

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 the homepage of the E4S website. At the top, there is a navigation bar with the E4S logo and the tagline "E4S - An HPC-AI Software Ecosystem for Science" and "Advancing Science through Trusted Software". Navigation links include "Use E4S", "Join E4S", "About E4S", and "Contact Us". The main header features a large blue banner with the text "E4S: HPC-AI Software Ecosystem for Science" and "Performance-portable libraries & tools for advanced computing". Below this are two buttons: "News & Events" and "Get E4S". The main content area contains a welcome message, a paragraph about the ecosystem, and a section supported by the U.S. Department of Energy (DOE). At the bottom, there are four feature cards: "Performance Portability", "Curated Ecosystem", "Reproducible Environments", and "Continuous Validation".

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 a web browser window with the URL <https://e4s.io>. The page features four main columns of content:

- 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.

Below these columns are two call-to-action boxes:

- Chat with the E4S Bot:** Ask questions, explore tools, or get recommendations from the E4S Guide Bot. Includes a button labeled "Chat with the E4S Guide Bot".
- Visit the E4S Product Catalog:** Browse member products, product families, with extensive links, via the E4S Product Catalog. Includes a button labeled "Browse the E4S Product Catalog".

At the bottom of the page, there is a section titled "Trusted by the HPC-AI Community" with the following text:

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.

The footer contains the following text:

© 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.

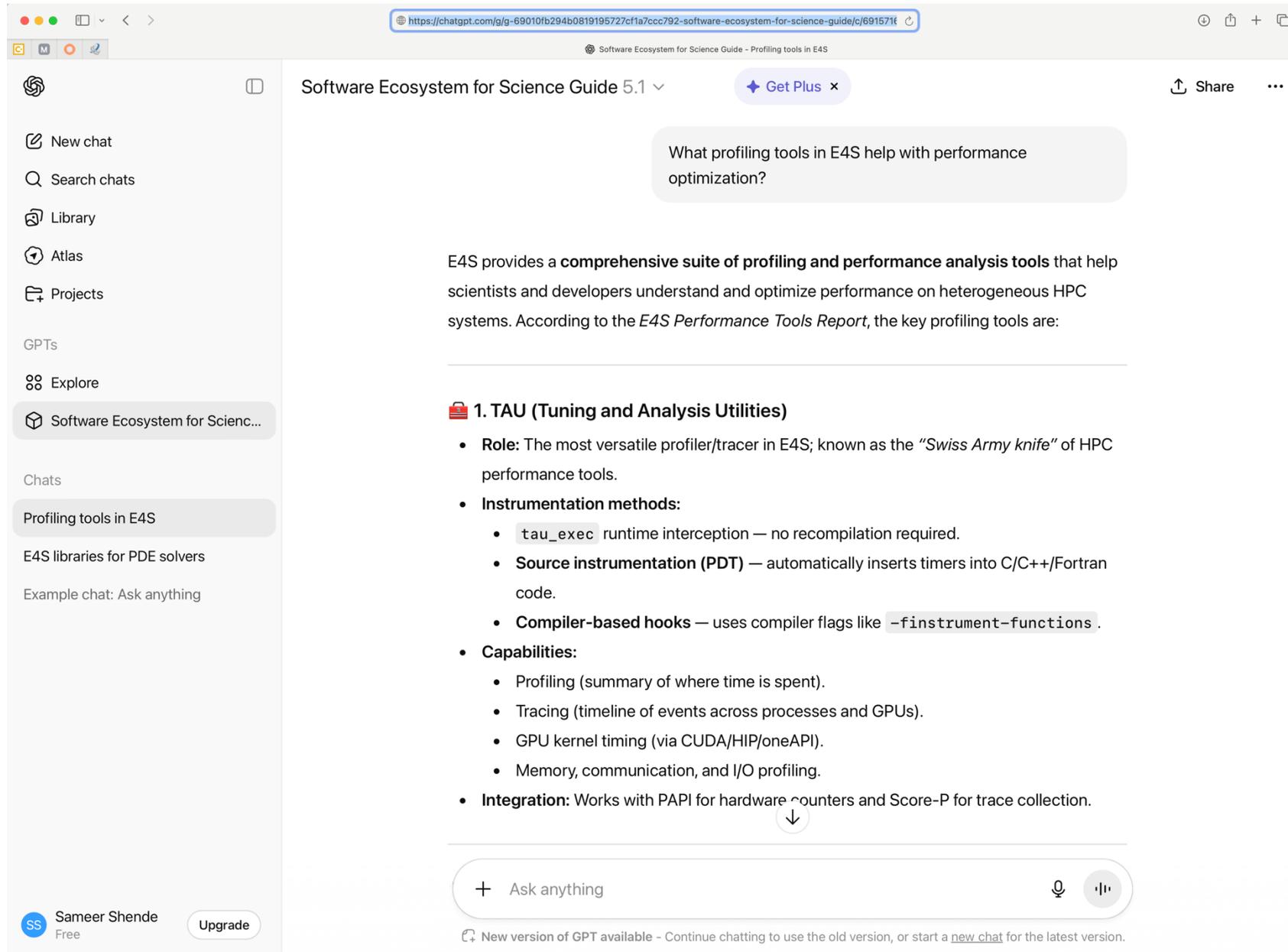


Chatbot integration in E4S Website

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 "ChatGPT - Software Ecosystem for Science Guide". The main content area displays the chatbot's name "Software Ecosystem for Science Guide 5.1" with a "Get Plus" button. Below the name is a cube icon and the title "Software Ecosystem for Science Guide" by Michael Heroux. The bot is described as an "Expert on scientific libraries and tools in the E4S (Ecosystem for Science) ecosystem." There are four suggested prompts in rounded rectangles: "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 text input field with a plus sign and the text "Ask anything", a microphone icon, and a voice icon. The left sidebar contains navigation options: "New chat", "Search chats", "Library", "Atlas", "Projects", "GPTs", "Explore", and "Software Ecosystem for Scienc...". At the bottom left of the sidebar, it says "Sameer Shende Free" with an "Upgrade" button.



E4S Bot trained on E4S content on OpenAI's ChatGPT



The screenshot shows a web browser window with a ChatGPT chat interface. The URL in the address bar is <https://chatgpt.com/g/g-69010fb294b0819195727cf1a7ccc792-software-ecosystem-for-science-guide/c/691571f>. The chat title is "Software Ecosystem for Science Guide 5.1". A user has asked: "What profiling tools in E4S help with performance optimization?". The AI response 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:". Below this, a section titled "1. TAU (Tuning and Analysis Utilities)" is shown with a list of details:

- **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.

The interface also shows a sidebar with navigation options like "New chat", "Search chats", "Library", "Atlas", "Projects", "GPTs", "Explore", and "Software Ecosystem for Scienc...". At the bottom, there is a "Sameer Shende Free" badge and an "Upgrade" button. A footer note says: "New version of GPT available - Continue chatting to use the old version, or start a [new chat](#) for the latest version."

E4S Bot

chatgpt.com

Launch GPT-OSS 120B E4S

Get Plus x

Share ...

How can I launch OpenAI's 120B parameter OSS model using E4S?

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 HuggingFace 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

Sameer Shende Free Upgrade

ChatGPT can make mistakes. Check important info.



E4S Documentation

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

Use E4S Join E4S About E4S Contact Us

ABOUT E4S

- Overview
- E4S Introduction
- First-Time Users
- E4S Product Families
- E4S FAQs
- News & Events

About E4S

E4S provides a curated collection of scientific libraries and tools supporting scientific discovery through advanced computing technologies. The primary value E4S provides is bringing together a large collection of independently developed libraries and tools and enhancing the overall enterprise of developing and using them in aggregate. This portion of the E4S website focuses on topics that help understand what E4S is, how to contribute to it, and how it can be used.

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.

© 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.



E4S Product Catalog

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

Use E4S | Join E4S | About E4S | Contact Us

USE E4S

- 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
- ParaTools Pro for E4S™

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



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

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

Use E4S Join E4S About E4S

GET E4S

Get E4S

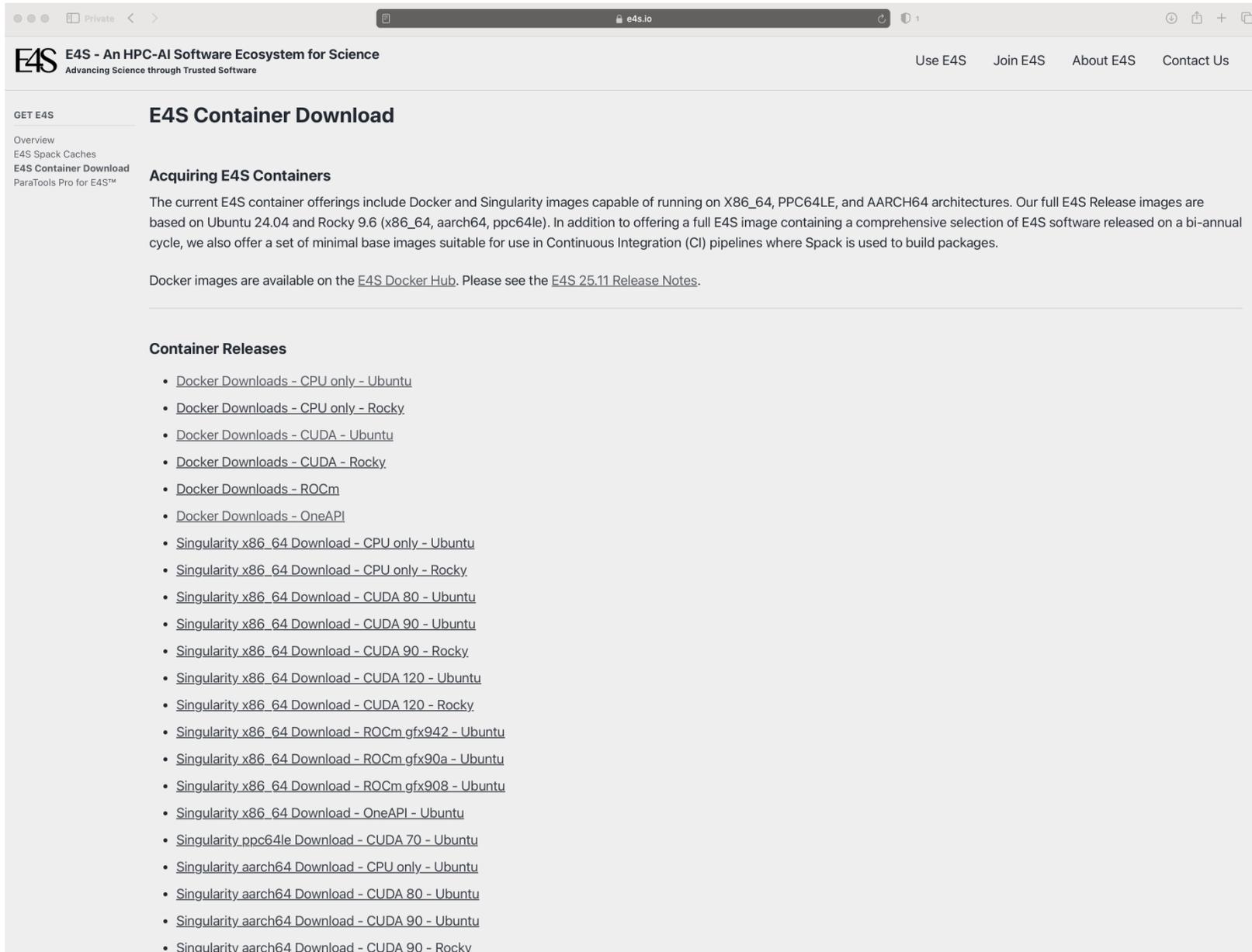
E4S offers multiple methods for obtaining its collection of HPC and AI software packages. Below is an overview of these deployment options. Additional documentation is available from the side menu.

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 with the following content:

- E4S - An HPC-AI Software Ecosystem for Science**
Advancing Science through Trusted Software
- Navigation: Use E4S, Join E4S, About E4S, Contact Us
- E4S Container Download**
- Acquiring E4S Containers**

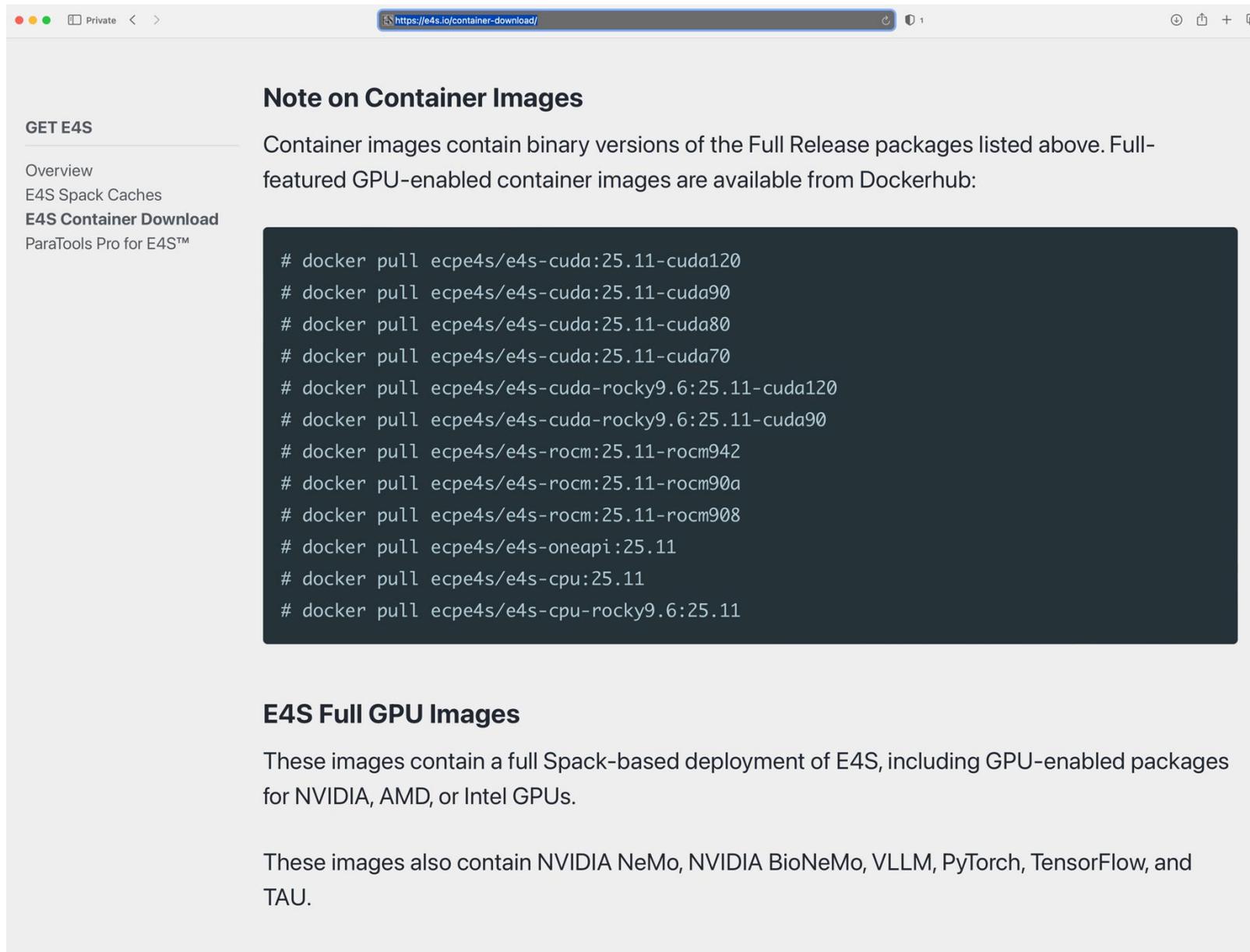
The current E4S container offerings include Docker and Singularity images capable of running on X86_64, PPC64LE, and AARCH64 architectures. Our full E4S Release images are based on Ubuntu 24.04 and Rocky 9.6 (x86_64, aarch64, ppc64le). In addition to offering a full E4S image containing a comprehensive 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



GET E4S

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

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:

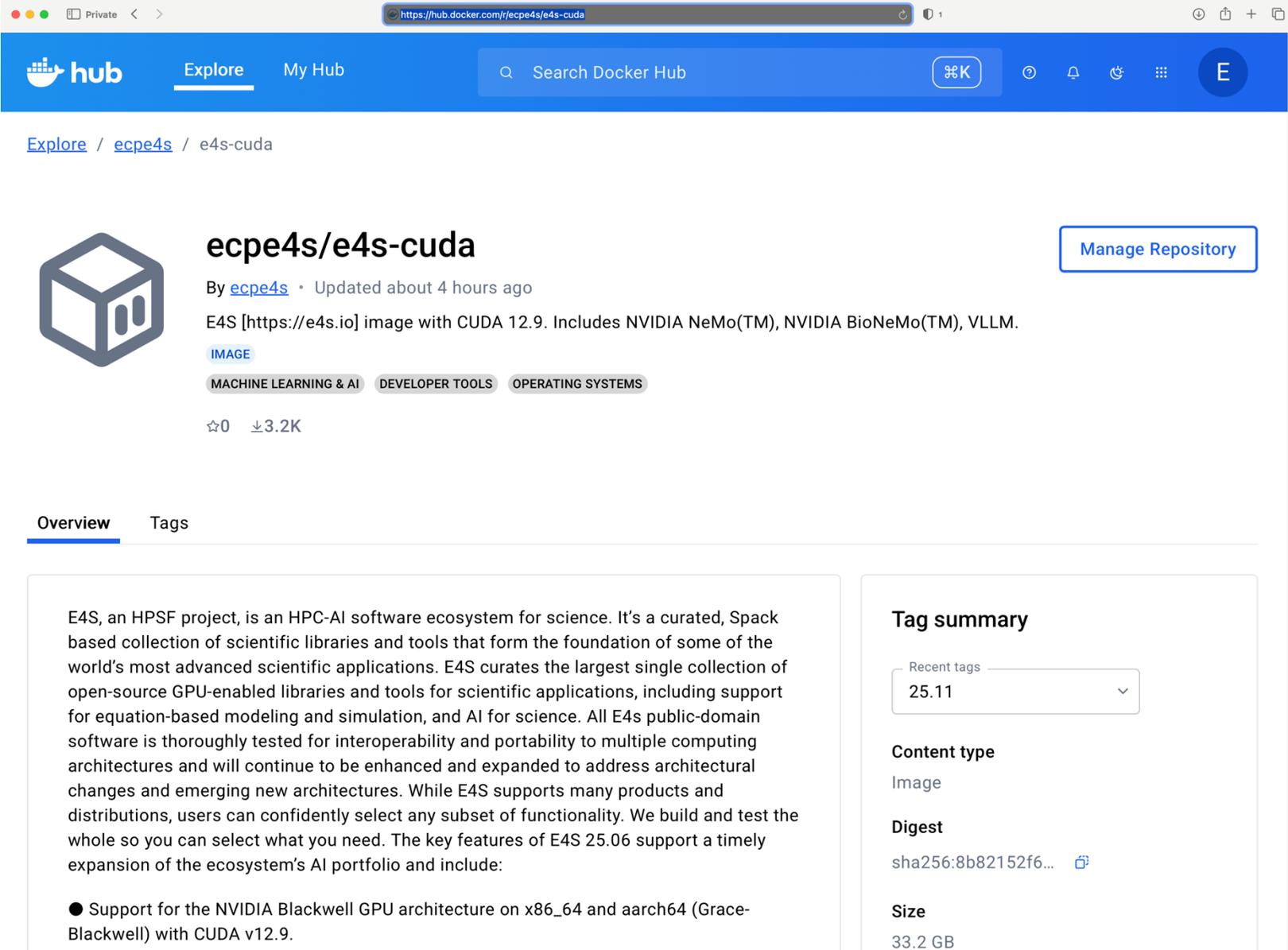
```
# docker pull ecpe4s/e4s-cuda:25.11-cuda120
# docker pull ecpe4s/e4s-cuda:25.11-cuda90
# docker pull ecpe4s/e4s-cuda:25.11-cuda80
# docker pull ecpe4s/e4s-cuda:25.11-cuda70
# docker pull ecpe4s/e4s-cuda-rocky9.6:25.11-cuda120
# docker pull ecpe4s/e4s-cuda-rocky9.6:25.11-cuda90
# docker pull ecpe4s/e4s-rocm:25.11-rocm942
# 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 the DockerHub page for the repository `ecpe4s/e4s-cuda`. The page includes a navigation bar with the DockerHub logo, search bar, and user profile. The repository name is `ecpe4s/e4s-cuda`, created by `ecpe4s` and updated about 4 hours ago. The description states it is an E4S image with CUDA 12.9, including NVIDIA NeMo, BioNeMo, and VLLM. The repository is categorized as an image and includes tags for Machine Learning & AI, Developer Tools, and Operating Systems. It has 0 stars and 3.2K downloads. The overview section describes E4S as an HPC-AI software ecosystem for science, and the tag summary shows the current tag is 25.11, which is an image with a size of 33.2 GB.

hub Explore My Hub Search Docker Hub

Explore / ecpe4s / e4s-cuda

 **ecpe4s/e4s-cuda** [Manage Repository](#)

By [ecpe4s](#) · Updated about 4 hours ago

E4S [<https://e4s.io>] image with CUDA 12.9. Includes NVIDIA NeMo(TM), NVIDIA BioNeMo(TM), VLLM.

IMAGE

MACHINE LEARNING & AI DEVELOPER TOOLS OPERATING SYSTEMS

☆0 ↓3.2K

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

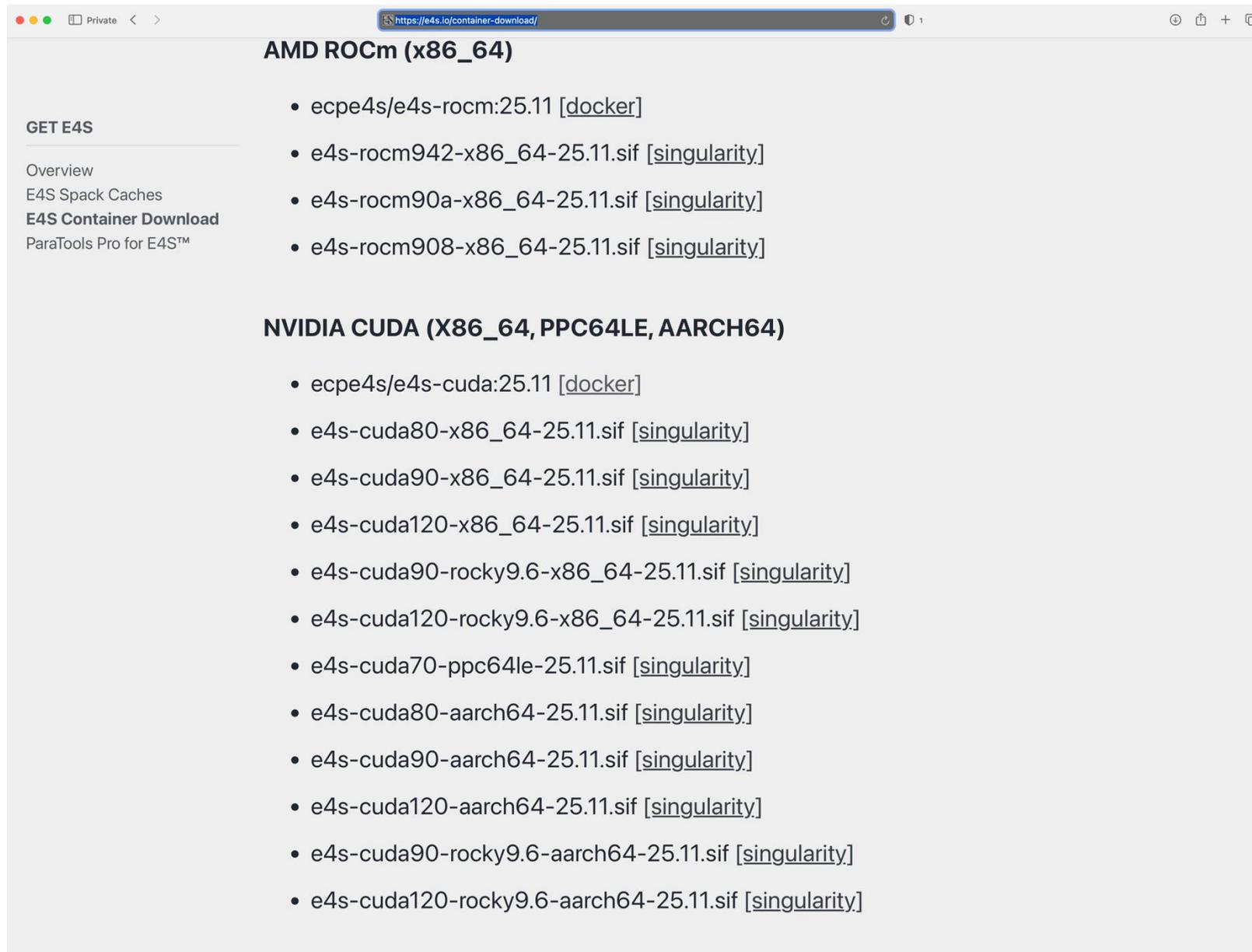
Content type
Image

Digest
sha256:8b82152f6...

Size
33.2 GB



Download E4S containers: NVIDIA and AMD GPUs

A screenshot of a web browser displaying the E4S container download page. The browser's address bar shows the URL "https://e4s.io/container-download/". The page content is organized into two main sections: "AMD ROCm (x86_64)" and "NVIDIA CUDA (X86_64, PPC64LE, AARCH64)". Each section contains a list of container images with links to their respective download pages. On the left side of the page, there is a sidebar with navigation links: "GET E4S", "Overview", "E4S Spack Caches", "E4S Container Download", and "ParaTools Pro for E4S™".

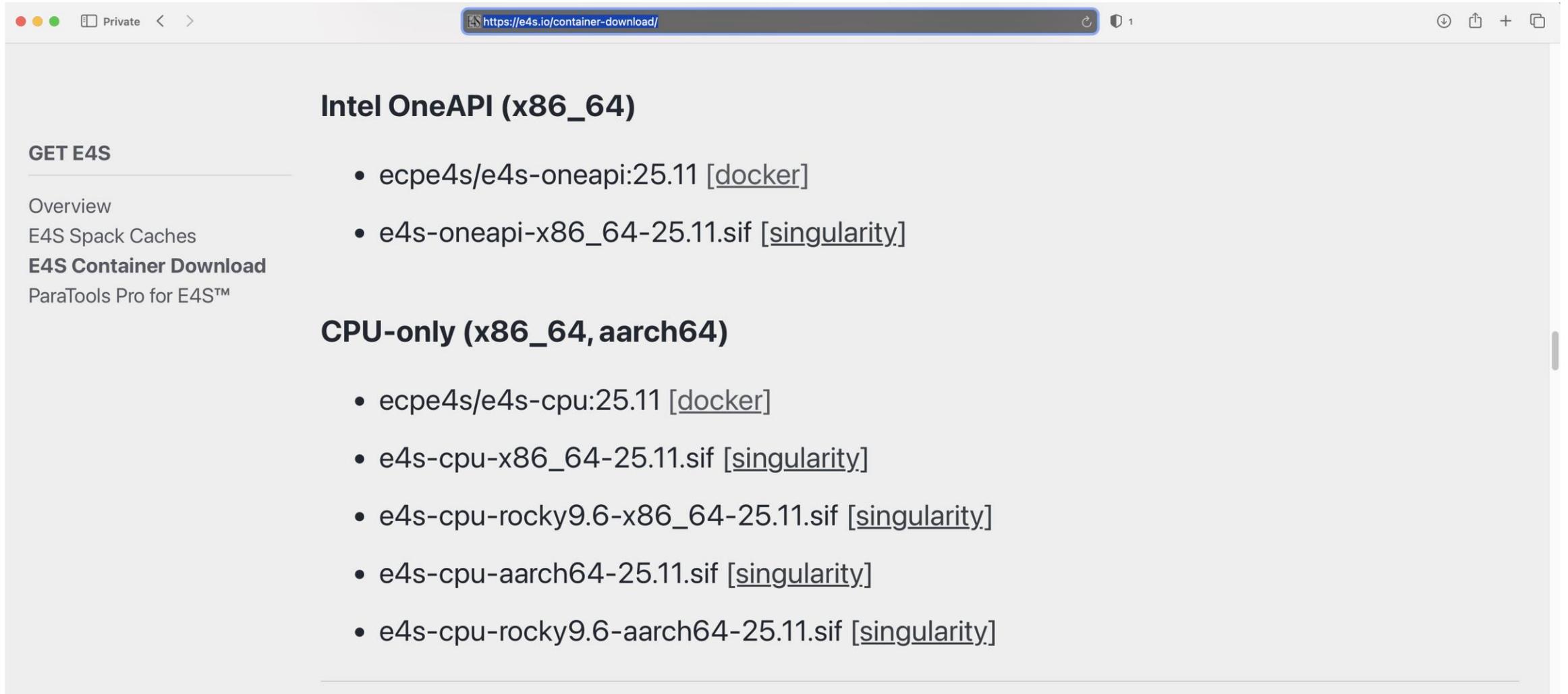
AMD ROCm (x86_64)

- 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 at the URL <https://e4s.io/container-download/>. The page content is as follows:

GET E4S

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

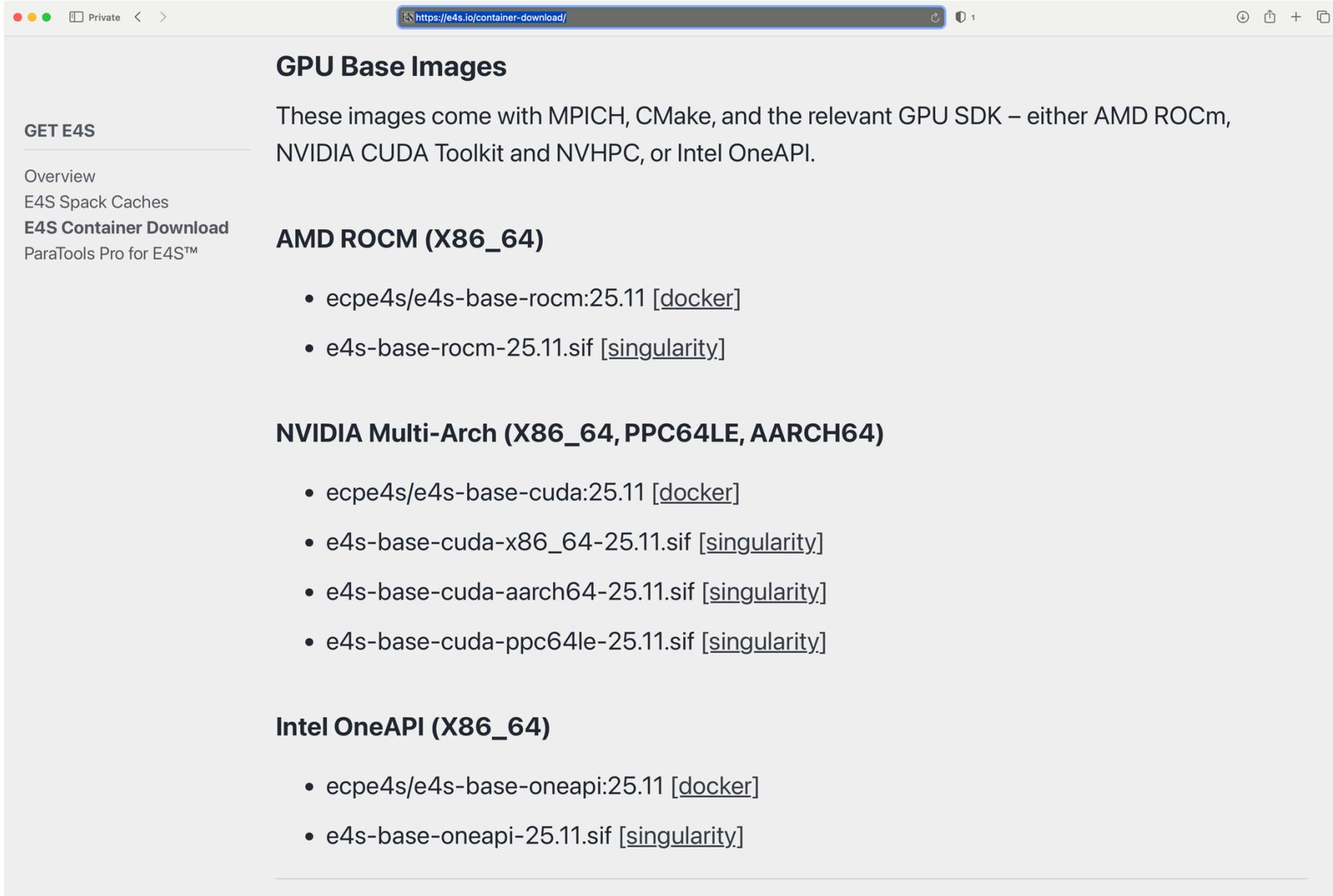
Intel OneAPI (x86_64)

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

CPU-only (x86_64, aarch64)

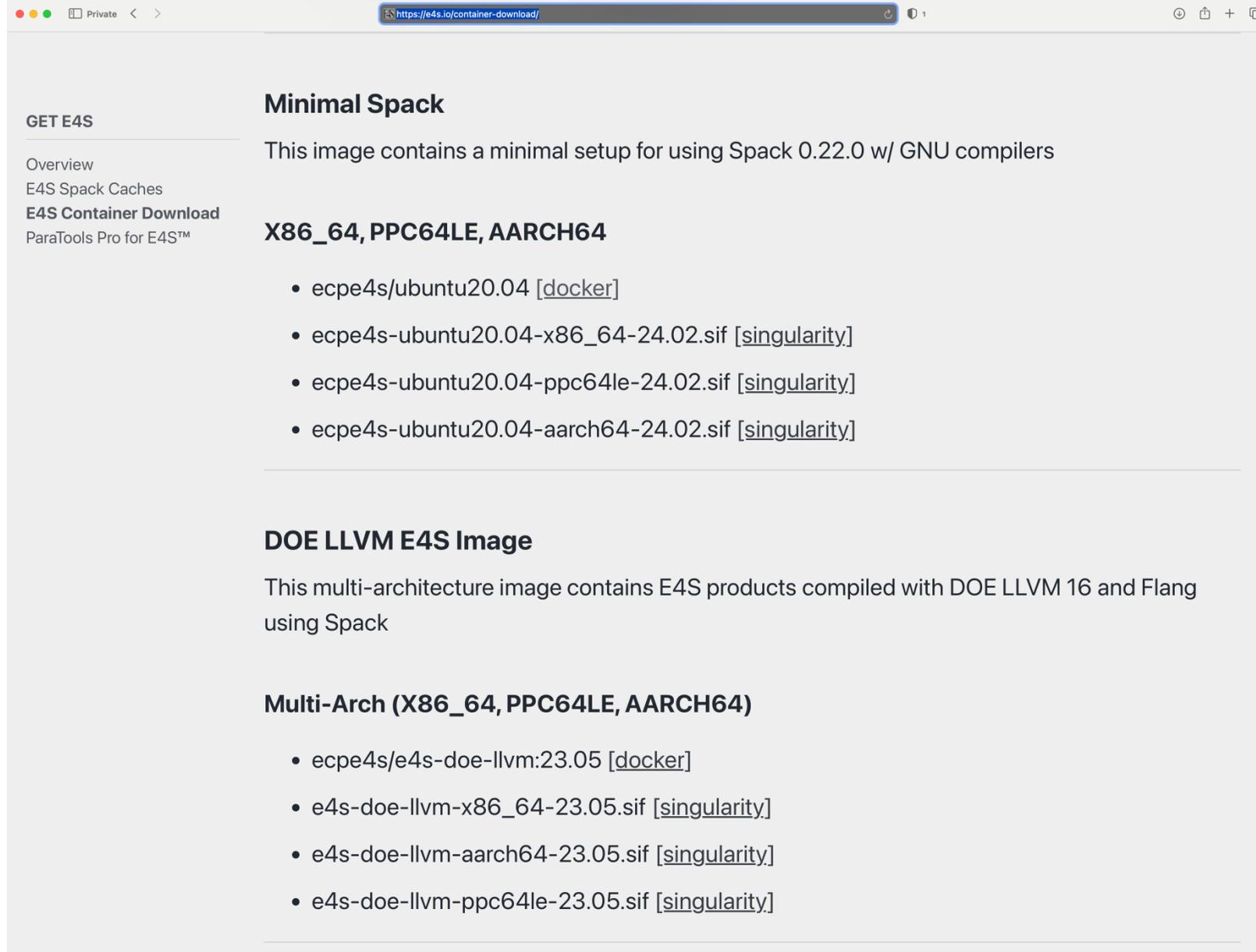
- `ecpe4s/e4s-cpu:25.11` [[docker](#)]
- `e4s-cpu-x86_64-25.11.sif` [[singularity](#)]
- `e4s-cpu-rocky9.6-x86_64-25.11.sif` [[singularity](#)]
- `e4s-cpu-aarch64-25.11.sif` [[singularity](#)]
- `e4s-cpu-rocky9.6-aarch64-25.11.sif` [[singularity](#)]

E4S Base Containers with GPU runtimes and MPI

A screenshot of a web browser showing the E4S Container Download page. The browser's address bar displays the URL https://e4s.io/container-download/. The page has a sidebar on the left with navigation links: GET E4S, Overview, E4S Spack Caches, E4S Container Download (highlighted), and ParaTools Pro for E4S™. The main content area is titled "GPU Base Images" and contains a paragraph: "These images come with MPICH, CMake, and the relevant GPU SDK – either AMD ROCm, NVIDIA CUDA Toolkit and NVHPC, or Intel OneAPI." Below this, there are three sections, each with a list of container images:

- 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



GET E4S

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

Minimal Spack

This image contains a minimal setup for using Spack 0.22.0 w/ GNU compilers

X86_64, PPC64LE, AARCH64

- 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](#)]

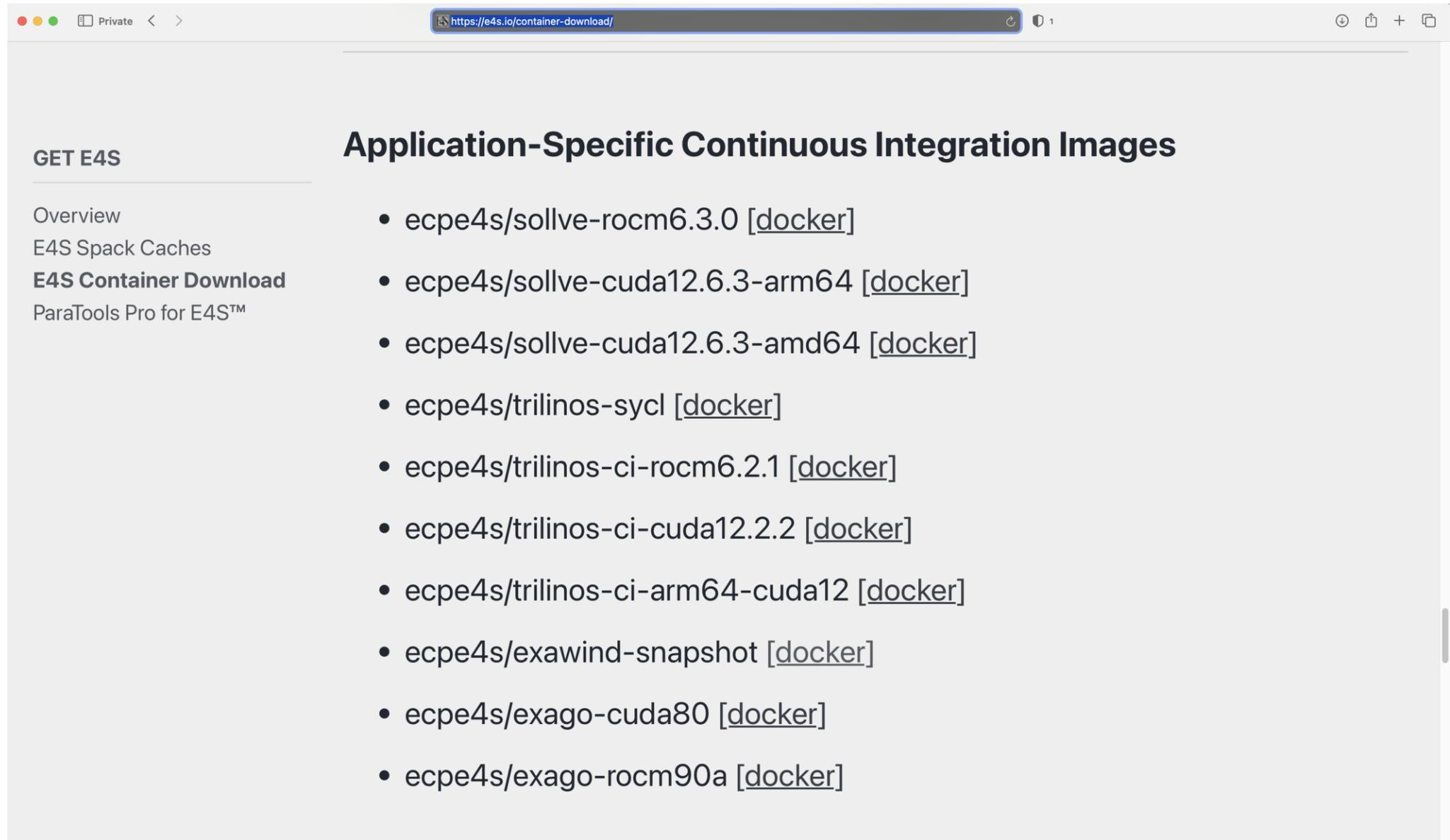
DOE LLVM E4S Image

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



Private < > https://e4s.io/container-download/ 1

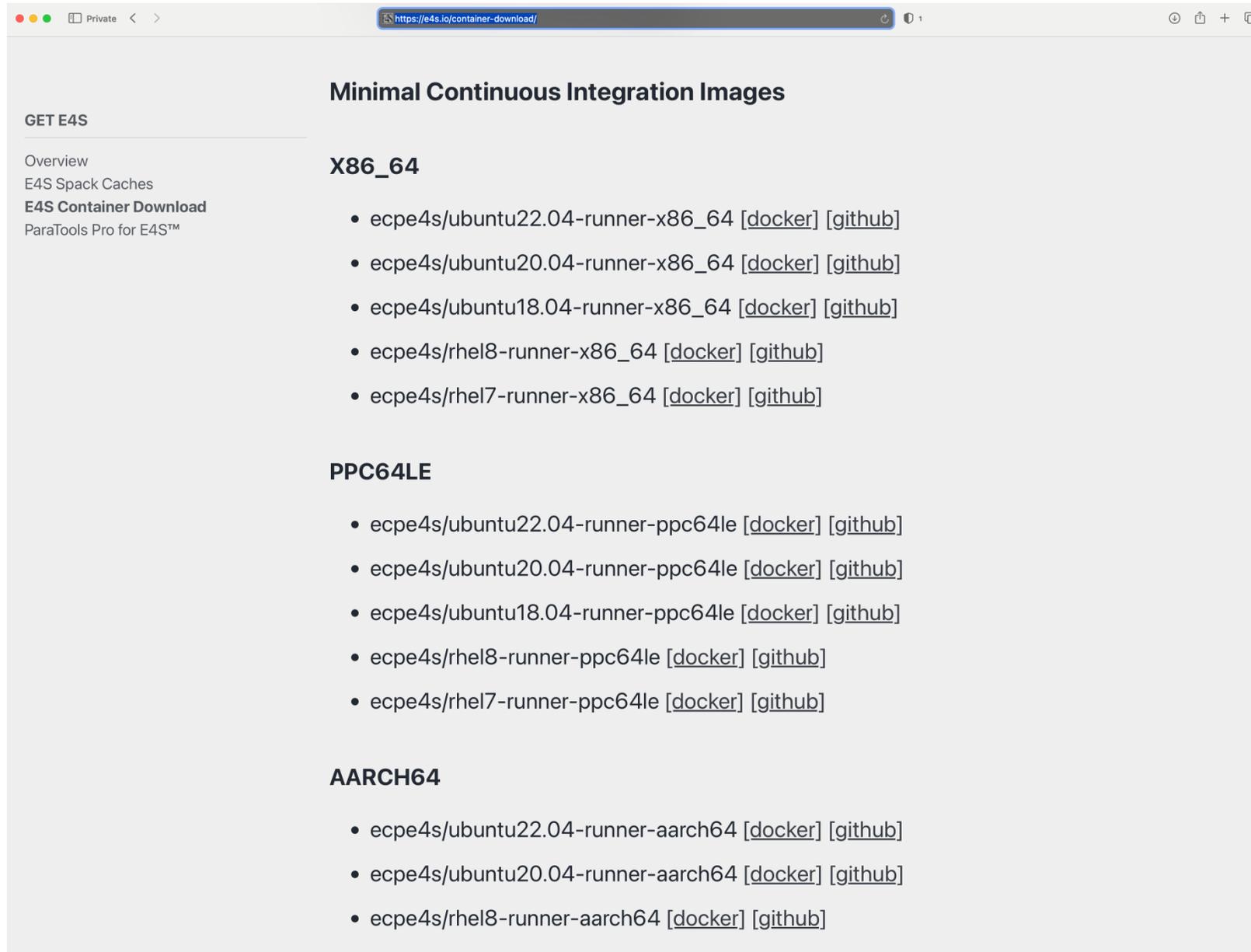
GET E4S

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

Application-Specific Continuous Integration Images

- [ecpe4s/solve-rocm6.3.0 \[docker\]](#)
- [ecpe4s/solve-cuda12.6.3-arm64 \[docker\]](#)
- [ecpe4s/solve-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

A screenshot of a web browser displaying the 'E4S Container Download' page. The browser's address bar shows 'https://e4s.io/container-download/'. The page has a sidebar on the left with navigation links: 'GET E4S', 'Overview', 'E4S Spack Caches', 'E4S Container Download' (which is highlighted), and 'ParaTools Pro for E4S™'. The main content area is titled 'Minimal Continuous Integration Images' and is organized into three sections: 'X86_64', 'PPC64LE', and 'AARCH64'. Each section contains a list of container images with links to Docker and GitHub.

Minimal Continuous Integration Images

GET E4S

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

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](#)]

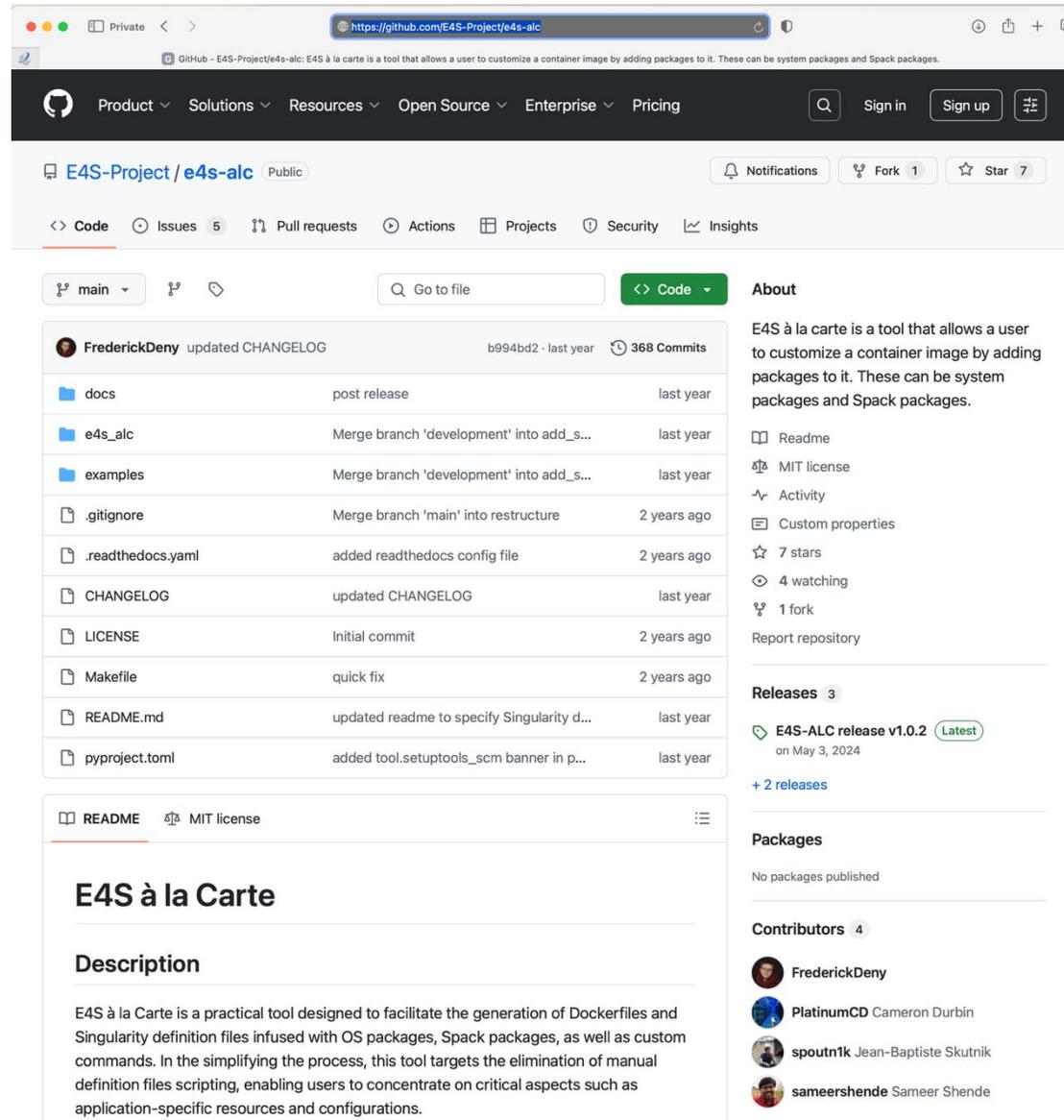
PPC64LE

- [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



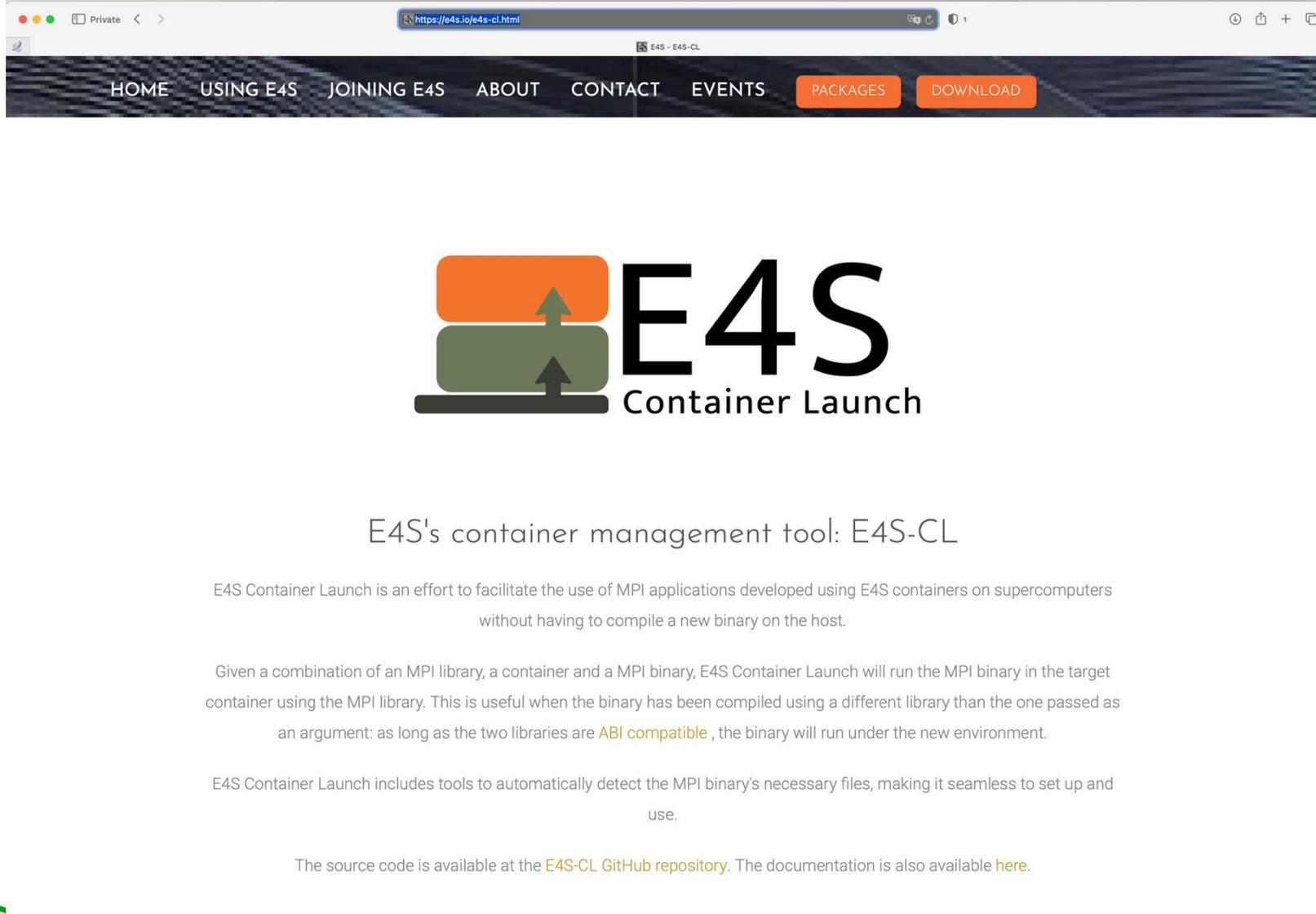
The screenshot displays the GitHub repository for `E4S-Project/e4s-alc`. The repository is public and has 7 stars and 1 fork. The main branch is selected. The repository structure includes folders `docs`, `e4s_alc`, and `examples`, and files `.gitignore`, `.readthedocs.yaml`, `CHANGELOG`, `LICENSE`, `Makefile`, `README.md`, and `pyproject.toml`. The README is titled "E4S à la Carte" and uses the MIT license. The description explains that the tool is designed to facilitate the generation of Dockerfiles and Singularity definition files by infusing them with OS packages, Spack packages, and custom commands, aiming to simplify the process and reduce manual scripting.

- 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!



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

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



The screenshot shows a web browser window displaying the E4S Container Launch website. The browser's address bar shows the URL <https://e4s.io/e4s-cl.html>. The website has a dark blue header with navigation links: HOME, USING E4S, JOINING E4S, ABOUT, CONTACT, EVENTS, PACKAGES, and DOWNLOAD. The main content area features the E4S logo, which consists of a stylized container icon with an orange top and green bottom, and the text "E4S Container Launch". Below the logo, the text reads "E4S's container management tool: E4S-CL". The page contains several paragraphs of text explaining the tool's purpose and usage, and a link to the source code repository.

HOME USING E4S JOINING E4S ABOUT CONTACT EVENTS PACKAGES DOWNLOAD



E4S Container Launch

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
```



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-4.2.3-47excoypwhfmx57rfs6reouvvninugcf)
[+] /usr (external glibc-2.35-a7drdl4tlx4bu3mzhor75pskvd3pdot6)
[+] /spack/opt/spack/linux-ubuntu22.04-x86_64_v3/gcc-11.4.0/gcc-runtime-11.4.0-f63c77kavzjtpmnhucd2oyfaxagwjzla
[+] /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-qp5blvcyuzgzhqsrp2ew6gq2nlos34b2
==> 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 [999999999]
                       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

The screenshot displays a Jupyter Notebook interface with the following components:

- Code Editor:** Contains Python code for predicting variant effects. Key sections include:
 - Path definitions for reference and variant predictions.
 - GPU support and info checks using `check_fp8_support()`.
 - Model execution commands for `predict_ref_command` and `predict_var_command`, including options for model size, tensor parallelism, and context parallelism.
- Terminal:** Shows the output of `cat /etc/os-release`, identifying the system as Rocky Linux 9.6 (Blue Onyx) with an NVIDIA GB10 GPU. It also shows the execution of `python` and `import nemo` / `import bionemo`.
- Extensions:** A sidebar on the left lists installed extensions such as Jupyter, Python, and Jupyter Cell Tags.
- Output:** A cell execution result shows `FP8 Support: True` and `Device: NVIDIA GB10, Compute Capability: 12.1`.



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 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
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 hypre/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 slepc/3.24.0-cuda120 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 vtk-m/2.3.0-cuda120-openmp
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 / %c,cxx,fortran=gcc@13.3.1 -----
tasmanian@8.1

-- linux-rocky9-x86_64_v3 / %cxx=gcc@13.3.1 -----
arborx@1.5 hp@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 tasmanian/8.1-cuda120
arborx/1.5-cuda120 cusz/0.14.0-cuda120 heffte/2.4.1-cuda120 legion/25.03.0-cuda70 parsec/4.0.2411-cuda120 tau/2.35-cuda
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 umpire/2025.03.0-cuda120
cabana/0.7.0-cuda120 e4s-cl/1.0.5 hp@1.11.0-cuda120 magma/2.9.0-cuda120 strumpack/8.0.0-cuda120-openmp vtk-m/2.3.0-cuda120-openmp
caliper/2.12.1-cuda120 fftx/1.2.0-cuda120 hypre/2.33.0-cuda120 mgard/compat-2023-12-09-cuda120-openmp sundials/7.5.0-cuda120 zfp/1.0.1-cuda120
chai/2025.03.0-cuda120 flecsi/2.4.1-cuda120 kokkos-kernels/4.7.01-cuda120 mpich/4.3.1 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  papi/7.2.0-cuda  tasmanian/8.1-cuda120
arborx/1.5-cuda120    cusz/0.14.0-cuda120  heffte/2.4.1-cuda120  legion/25.03.0-cuda70  parsec/4.0.2411-cuda120  tau/2.35-cuda
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  umpire/2025.03.0-cuda120
cabana/0.7.0-cuda120  e4s-cl/1.0.5  hpx/1.11.0-cuda120  magma/2.9.0-cuda120  strumpack/8.0.0-cuda120-openmp  vtk-m/2.3.0-cuda120-openmp
caliper/2.12.1-cuda120  fftx/1.2.0-cuda120  hypre/2.33.0-cuda120  mgard/compat-2023-12-09-cuda120-openmp  sundials/7.5.0-cuda120  zfp/1.0.1-cuda120
chai/2025.03.0-cuda120  flecsi/2.4.1-cuda120  kokkos-kernels/4.7.01-cuda120  mpich/4.3.1  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,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 / %c,fortran=gcc@13.3.1 -----
tasmanian@8.1

-- linux-rocky9-x86_64_v3 / %c,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             imghdr.py           nturl2path.py       __pycache__         _sitebuiltins.py    tabnanny.py          urllib
_aix_support.py     concurrent          filecmp.py          importlib            numbers.py           pycldr.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         io.py               operator.py         _pydatetime.py     smtpd.py            tempfile.py          venv
ast.py              contextlib.py       fractions.py        ipaddress.py        optparse.py         _pydecimal.py      sndhdr.py           test                 warnings.py
asyncio             contextvars.py      ftplib.py          json                os.py               pydoc_data          socket.py            textwrap.py         wave.py
base64.py           copy.py             functools.py       keyword.py          _osx_support.py    pydoc.py            socketserver.py     this.py              weakref.py
bdb.py              copyreg.py          __future__.py     lib2to3             pathlib.py          _pyio.py            sqlite3              _threading_local.py _weakrefset.py
bisect.py           cProfile.py        genericpath.py    lib-dynload         pdb.py              _pylong.py         sre_compile.py      threading.py        webbrowser.py
bz2.py              crypt.py            getopt.py          LICENSE.txt         pickle.py            queue.py            sre_constants.py   timeit.py           wsgiref
calendar.py         csv.py              getpass.py        linecache.py        pickletools.py     quopri.py           sre_parse.py        tkinter             xdrlib.py
cgi.py              ctypes              gettext.py        locale.py           pipes.py             random.py            ssl.py               tokenize.py         xml
cgitb.py            curses              glob.py           logging             pkgutil.py          re                   statistics.py       token.py             xmlrpc
chunk.py            dataclasses.py     graphlib.py       lzma.py             platform.py         reprlib.py          stat.py              tomllib             zipapp.py
cmd.py              datetime.py        gzip.py           mailbox.py          plistlib.py         rlcompleter.py     stringprep.py      traceback.py        zipfile
codecs.py           dbm                 hashlib.py        mailcap.py          poplib.py           runpy.py            string.py            tracemalloc.py     zipimport.py
codeop.py           decimal.py         heapq.py          _markupbase.py     posixpath.py       sched.py            _strptime.py       trace.py            zoneinfo
code.py             difflib.py        __hello__.py     mimetypes.py       pprint.py           secrets.py          subprocess.py      tty.py              turtledemo
collections         _collections_abc.py doctest.py       modulefinder.py    multiprocessing     selectors.py        sunau.py            types.py            turtle.py
colorsys.py         email              idlelib          nntplib.py         pstats.py           shelve.py           symtable.py        typing.py
_compat_pickle.py   encodings         nntplib.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/pkgs/python-3.12.11/bin/adk
Singularity> which jupyter
/opt/python/pkgs/python-3.12.11/bin/jupyter
Singularity> which vllm
/opt/python/pkgs/python-3.12.11/bin/vllm
Singularity> which marimo
/opt/python/pkgs/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 -----
arborx@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 butterflypack@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 libcatalyst@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 globalarrays@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 mpifileutils@0.12 openpmd-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-libensemble@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 -----
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-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        veloc@1.7
chai@2025.03.0    flecsi@2.4.1       gotcha@1.0.8          lammmps@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          mpi4py@1.5.1         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

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

-- linux-rocky9-x86_64_v3 / %c,cxx=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      libquoo/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
arborx/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       mgard/compat-2023-12-09-openmp  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      hpctoolkit/2025.0.1  park-variant/1.4.0   scr/3.1.0           variorum/0.8.0
boost/1.88.0     fftx/1.2.0          hpx/1.11.0        hypre/2.33.0      mpi/4.3.1            slate/2025.05.28-openmp  veloc/1.7
bricks/2023.08.25  flit/2.1.0         kokkos-kernels/4.7.01-openmp  nccmp/1.9.1.0    mpiutils/0.12        slepc/3.24.0       vtk-m/2.3.0-openmp
butterflypack/3.2.0-openmp  fpm/0.10.0-openmp  kokkos/4.7.01-openmp  nco/5.3.4-openmp  mpich/4.3.1          stc/0.9.0           wannier90/3.1.0
cabana/0.7.0     gasnet/2025.8.0     laghos/3.1        lammps/20250722-openmp  nekbone/17.0         strumpack/8.0.0-openmp  warpx/25.04
caliper/2.12.1   ginkgo/1.10.0-openmp  lbann/0.104      globalarrays/5.8.2  netcdf-fortran/4.6.2  sundials/7.5.0      wps/4.5
chai/2025.03.0   glvis/4.4           legion/25.03.0   gmp/6.3.0         netlib-scalapack/2.2.2  superlu-dist/9.1.0  wrf/4.6.1
chapel/2.6.0     gromacs/2025.3-openmp  libbann/0.104    h5py/3.10.0       nek5000/19.0         superlu/7.0.0       xyce/7.10.0
charliecloud/0.40  hdf5/1.14.6       libbann/0.104    hdf5/1.14.6       netcdf/4.7.4         swig/4.0.2-fortran  zfp/1.0.1
conduit/0.9.5    hdf5-vol-async/1.7  libbann/0.104    hdf5-vol-cache/v1.1  netlib-scalapack/2.2.2  pumi/2.2.9         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           libcatapult@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  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  mpifileutils@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-libensemble@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 / %c,cxx,fortran=gcc@13.3.0 -----
tasmanian@8.1

-- linux-ubuntu24.04-x86_64_v3 / %c,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 / %c,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 / %c,cxx,fortran=gcc@9.4.0 -----
tasmanian@8.1

-- linux-ubuntu20.04-ppc64le / %c,cxx=gcc@9.4.0 -----
blaspp@2025.05.28 hpx@1.11.0 kokkos@4.6.02 kokkos@4.7.01 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 petsc@3.24.0 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 xyce@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 mpiutils@0.12 papyrus@1.0.2 raja@2025.03.0 upcxx@2023.9.0
cabana@0.7.0 faodel@1.2108.1 ginkgo@1.10.0 hpctoolkit@2024.01.1 libpressio@0.99.4 nccomp@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 / %c=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 / %c,cxx,fortran=gcc@9.4.0 -----
fortrilinos@2.3.0 tasmanian@8.1 tasmanian@8.1

-- linux-ubuntu20.04-ppc64le / %c,cxx=gcc@9.4.0 -----
arborx@2.0.1 flit@2.1.0 hpx@1.11.0 hpx@1.11.0 kokkos@4.7.01 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 d366fa84f3fdbcd4b10847ebd5db572ae12a34fb)
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-rocm  fftx/1.2.0-gfx942  hypre/2.33.0-gfx942    mfem/4.8.0-gfx942     slepc/3.24.0-gfx942   tau/2.35-rocm              trilinos/16.1.0-gfx942
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  umpire/2025.03.0-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
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

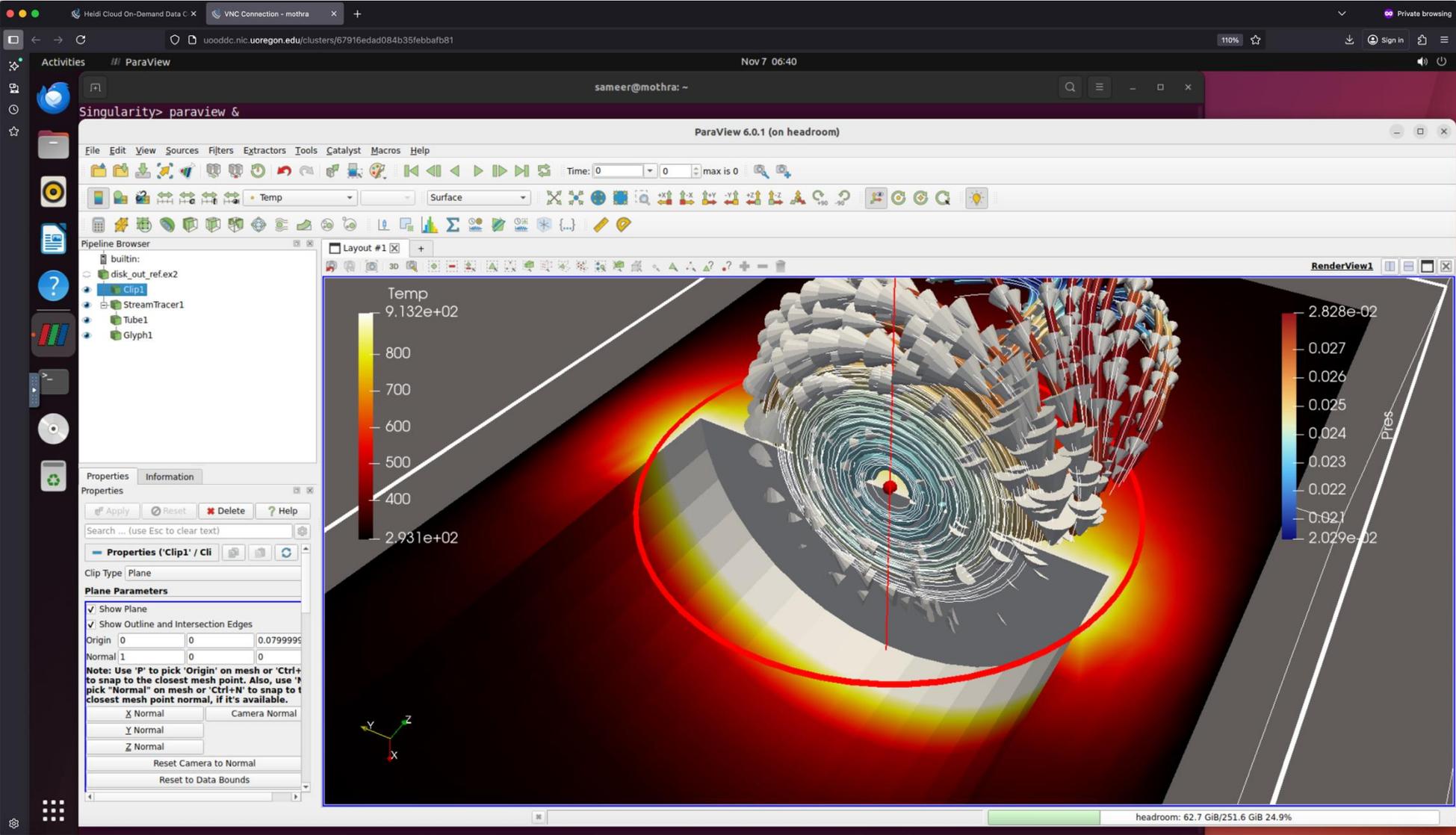
-- linux-ubuntu24.04-x86_64_v3 / %cxx=gcc@13.3.0 -----
arborx@2.0.1  kokkos@4.7.01  mfem@4.8.0

-- 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:          amdgc-amd-amdhsa--gfx942:sramecc+:xnack-
Name:          amdgc-amd-amdhsa--gfx942:sramecc+:xnack-
Name:          amdgc-amd-amdhsa--gfx942:sramecc+:xnack-
Name:          amdgc-amd-amdhsa--gfx942:sramecc+:xnack-
Singularity>

```



E4S: Visualization Tools: ParaView



E4S: Marimo Reactive Notebooks

The screenshot shows a VNC connection to a remote system. The terminal window displays the following commands and output:

```
Singularity> sycl-ls
[level_zero:gpu][level_zero:0] Intel(R) oneAPI Unified Runtime over Level-Zero, Intel(R) Data Center GPU Max 1100 12.60.7 [1.6.35096+9]
[opencl:cpu][opencl:0] Intel(R) OpenCL, Intel(R) Xeon(R) Silver 4410T OpenCL 3.0 (Build 0) [2025.20.8.0.06 160000]
[opencl:gpu][opencl:1] Intel(R) OpenCL Graphics, Intel(R) Data Center GPU Max 1100 OpenCL 3.0 NEO [25.35.35096.9]
Singularity> which adk
/opt/python/pkgs/python-3.12.11/bin/adk
Singularity> which marimo
/opt/python/pkgs/python-3.12.11/bin/marimo
Singularity> marimo new
```

Below the terminal, a message prompts the user to create a new notebook in their browser with the URL: http://localhost:2718?access_token=Lk51cWX1MUZ030Dqxiquk0

The web browser window shows an "Untitled Notebook" with the following code:

```
1 import marimo as mo
2 import plotly.graph_objects as go
3 import numpy as np
4
5 # Generate 3D surface data
6 x = np.linspace(-5, 5, 100)
7 y = np.linspace(-5, 5, 100)
8 x, y = np.meshgrid(x, y)
9 z = np.sin(np.sqrt(x**2 + y**2))
10
11 # Create a Plotly 3D surface plot
12 fig = go.Figure(
13     data=[go.Surface(z=z, x=x, y=y, colorscale='Viridis')]
14 )
15 fig.update_layout(title="3D Surface Example")
16
17 # Display the 3D plot in Marimo
18 mo.ui.plotly(fig)
```

The notebook displays a 3D surface plot titled "3D Surface Example". The plot shows a bell-shaped surface with a color scale ranging from 0 (blue) to 1 (yellow). The axes are labeled x, y, and z.



E4S: VisIt

The screenshot displays a VNC connection to a headroom environment. On the left, a terminal window shows the following commands and output:

```
Singularity> visit &
[1] 3718779
Singularity> Running: gui3.4.2
Running: viewer3.4.2 -geometry 1703x1150
Running: mdserver3.4.2 -host 127.0.0.1
Error opening plugin file: /usr/local/...
Running: engine_ser3.4.2 -host 127.0.0.1
```

The central panel shows the VisIt 3.4.2 GUI with the following settings:

- Active window: 1
- Auto apply:
- Active source: example.silo
- Time: [Slider]
- Plots: Pseudocolor - Isosurface(pressure)
- Apply to: active window all windows
- Apply operators to all plots
- Apply subset selections to all plots

The right panel shows a 3D visualization of the data. The axes are labeled: Height (parsec), Depth (parsec), and Width (parsec). The data is represented as a complex, multi-colored isosurface structure. A color bar on the left indicates the pressure values:

- Var: pressure
- Units: Pa
- Max: 5.779
- Min: 1.104

The visualization shows a complex, multi-colored structure with a color scale ranging from 1.104 (blue) to 5.779 (red). The structure is composed of numerous interconnected, rounded shapes, resembling a cluster of particles or a complex molecular structure. The axes are labeled: Height (parsec), Depth (parsec), and Width (parsec). The axes range from -10 to 10 parsecs. A coordinate system with X, Y, and Z axes is shown at the bottom left.

user: sameer
Thu Nov 6 22:34:02 2025



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

The screenshot shows a VSCode window with a terminal at the top displaying the command `codium` in a shell prompt `paratoolsadmin@e4s-ibm:~/examples/bionemo$`. The main editor area displays a Jupyter notebook with the following content:

Zero-shot prediction of BRCA1 variant effects with Evo 2

Deploy this tutorial on brev.dev. [Deploy Now](#)

Note - this notebook is a reproduction of The Arc Institute's same-titled notebook [here](#), using the BioNeMo 2 implementation of Evo2.

Evo2 is a foundation AI model trained on 9.3 trillion DNA base pairs, predicting variant effects without prior task-specific training.

Without being explicitly trained on BRCA1 variants, we show Evo 2's ability to generalize across all life forms.

The human *BRCA1* gene encodes for a protein that repairs damaged DNA ([Moynahan et al., 1999](#)). Certain variants of this gene have been associated with an increased risk of breast and ovarian cancers ([Miki et al., 1994](#)). Using Evo 2, we can predict whether a particular single nucleotide variant (SNV) of the *BRCA1* gene is likely to be harmful to the protein's function, and thus potentially increase the risk of cancer for the patient with the genetic variant.

```
%capture
import os

# Runs a subset of the model layers to test that the notebook runs in CI, but the output will be incorrect.
FAST_CI_MODE = bool = os.environ.get("FAST_CI_MODE", False)
[1] ✓ 0.0s Python
```

```
import glob
import gzip
import json
import math
import os
from pathlib import Path

import matplotlib.pyplot as plt
import pandas as pd
import seaborn as sns
import torch
from Bio import SeqIO
from sklearn.metrics import auc, roc_auc_score, roc_curve
[2] ✓ 2.3s Python
```

We start by loading a dataset from [Findlay et al. \(2018\)](#), which contains experimentally measured function scores of 3,893 *BRCA1* SNVs. These function scores reflect the extent by which the genetic variant has disrupted the protein's function, with lower scores indicating greater disruption. In this dataset, the SNVs are classified into three categories based on their function scores: *LOF* (loss-of-function), *INT* (intermediate), and *FUNC* (functional). We start by reading in this dataset.

To keep the notebook streamlined, we've abstracted much of the preprocessing logic into accompanying scripts located in `brca1_utils`. The full notebook can be viewed [here](#).

```
def download_data(data_dir="brca1", commit_hash="3819474bee6c24938016614411f1fa025e542bbe"):
    """Download required data files if they don't exist locally.

    Parameters:
    -----
    data_dir : str
        Directory to store downloaded files
    commit_hash : str
        GitHub commit hash for data version
    """
```

At the bottom of the notebook, the status bar shows "Cell 1 of 39".

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

The screenshot shows a Linux desktop environment with several windows open. On the left, a terminal window displays the following commands and output:

```
paratoolsadmin@e4s-ibm:~$ glxinfo | grep NVIDIA
client glx vendor string: NVIDIA Corporation
OpenGL vendor string: NVIDIA Corporation
OpenGL renderer string: NVIDIA L4/PCIe/SSE2
OpenGL core profile version string: 4.6.0 NVIDIA 570.195.03
OpenGL core profile shading language version string: 4.60 NVIDIA 570.195.03
OpenGL shading language version string: 4.60 NVIDIA 570.195.03
OpenGL ES profile version string: OpenGL ES 3.2 NVIDIA 570.195.03
paratoolsadmin@e4s-ibm:~$ which paraview
/usr/local/paraview-5.13.3/bin/paraview
paratoolsadmin@e4s-ibm:~$ paraview &
paratoolsadmin@e4s-ibm:~$ glxgears &
[2] 57076
paratoolsadmin@e4s-ibm:~$ Running synchronized to
approximately the same as the monitor refresh rate
28913 frames in 5.0 seconds = 5782.519 FPS
25233 frames in 5.0 seconds = 5046.367 FPS
25470 frames in 5.0 seconds = 5093.550 FPS
24694 frames in 5.0 seconds = 4938.712 FPS
26629 frames in 5.0 seconds = 5325.770 FPS
31803 frames in 5.0 seconds = 6360.508 FPS
22328 frames in 5.0 seconds = 4465.600 FPS
```

The ParaView 5.13.3 window shows a 3D visualization of a complex structure, likely a turbine or engine component, rendered with a color scale. The color scale ranges from 2.029e (blue) to 9.132e (red). The visualization includes a central red core, surrounded by yellow and orange layers, and a large white structure on the left. The interface includes a Pipeline Browser, Properties panel, and RenderView1 window.



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

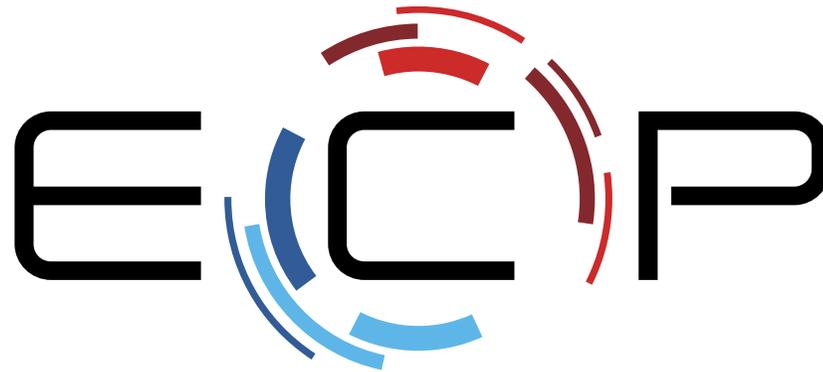
Office of
Science

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

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.



EXASCALE COMPUTING PROJECT

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.

