Hydra Infrastructure Management Guide

Student Container Platform Administration

Computer Science Department
SUNY New Paltz

Last Updated: January 2025

Contents

1 System Overview

Hydra is a containerized development platform providing persistent development environments for Computer Science students and faculty at SUNY New Paltz. The system uses SAML 2.0 Single Sign-On via Azure AD and Docker for container orchestration across a 3-node cluster.

1.1 Key Features

- **SSO Authentication:** Azure AD SAML 2.0 with automatic user provisioning
- **Persistent Containers:** One development environment per student with data persistence
- **Built-in Services:** VS Code (code-server), Jupyter Notebook, Docker-in-Docker
- **SSH Access:** Direct SSH access to containers via assigned ports
- **GPU Computing:** Access to NVIDIA GPUs on Chimera and Cerberus nodes
- **Dynamic Routing:** Traefik-based routing for custom web applications
- **Resource Management:** Time-limited resource allocations with automatic expiry
- **Integration:** OpenWebUI (GPT) and n8n account management

1.2 Access URLs

Service	URL	Description
Dashboard	`https://hydra.newpaltz.edu/dashboard`	Main user interface
OpenWebUI	`https://gpt.hydra.newpaltz.edu/`	AI chat interface
VS Code	`https://hydra.newpaltz.edu/students/{user}/vscode`	Browser IDE
Jupyter	`https://hydra.newpaltz.edu/students/{user}/jupyter`	Notebooks
Servers	`https://hydra.newpaltz.edu/servers`	Cluster status

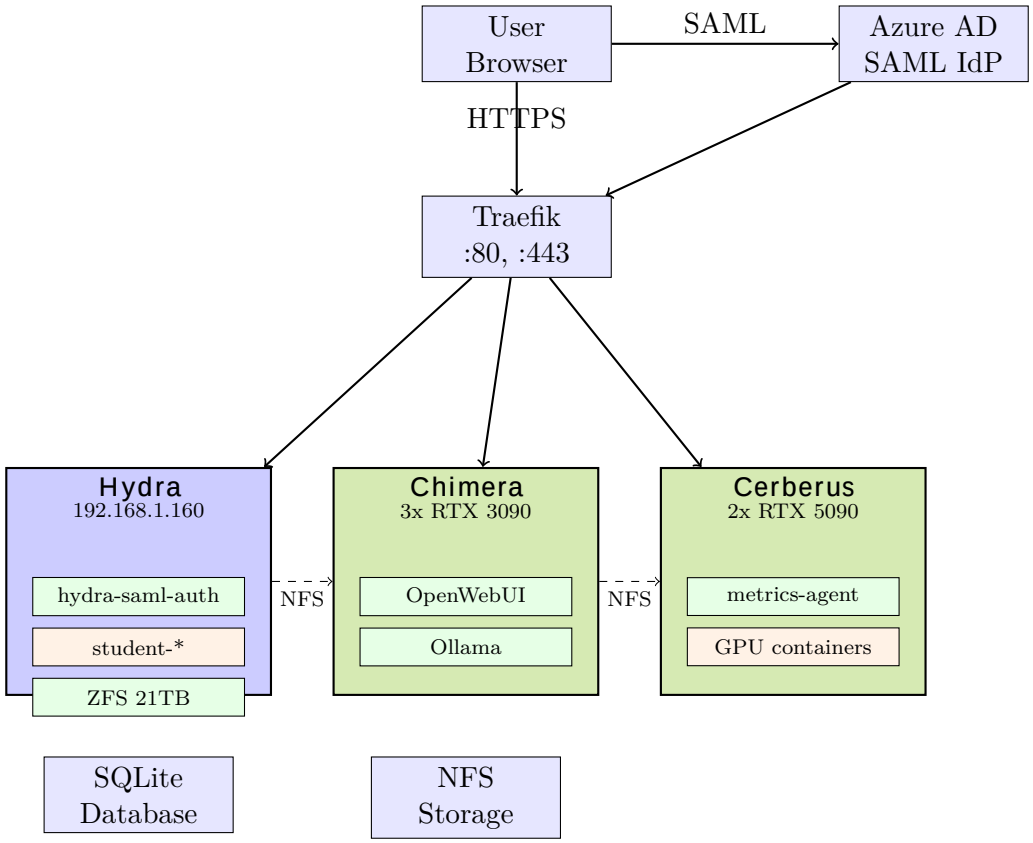
2 Cluster Architecture

The Hydra platform operates across a 3-node cluster, each with specialized roles.

2.1 Cluster Nodes

Node	IP	Role	GPU	Description
Hydra	192.168.1.160	Control Plane	None	Main server, ZFS storage, student containers
Chimera	192.168.1.150	Inference	3x RTX 3090	OpenWebUI, inference workloads
Cerberus	192.168.1.242	Training	2x RTX 5090	Student GPU training

2.2 Architecture Diagram



2.3 Storage Configuration

Node	Storage	Capacity	Purpose
Hydra	ZFS RAID-10	21 TB	Student volumes, primary data
Hydra	Seagate SSD	1.1 TB	Daily backups
Chimera	NFS mount	–	Student data access
Cerberus	NFS mount	–	Student data access

2.4 Component Overview

Component	Port	Description
Traefik	80, 443	Reverse proxy, TLS termination, routing
hydra-saml-auth	6969	SAML auth, dashboard, container management
OpenWebUI	3000	AI chat interface (Ollama frontend)
Ollama	11434	LLM inference engine
metrics-agent	9100	Node metrics collection (GPU nodes)
Student Containers	Dynamic	Per-user development environments

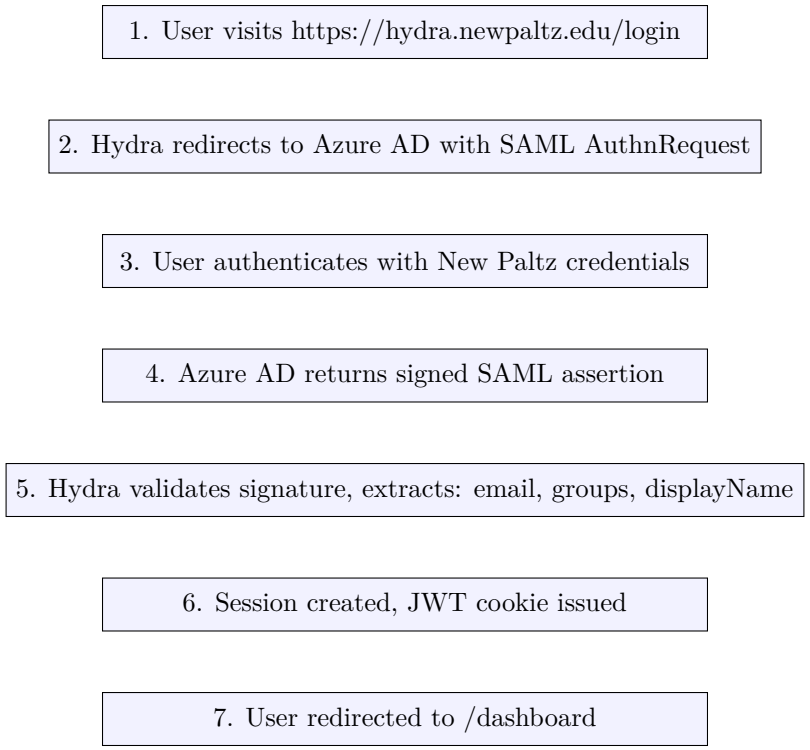
2.5 Network Architecture

Student containers operate on an isolated Docker network (`hydra_students_net`) with:

- No direct internet access (configurable)
- Internal DNS resolution
- Traefik-mediated external access via ForwardAuth
- Cross-node NFS access for GPU workloads

3 Authentication System

3.1 SAML 2.0 SSO Flow



3.2 Session Management

Sessions are managed via:

- **Express Session**: Server-side session storage in SQLite
- **JWT Cookie**: Site-wide authentication cookie for cross-service SSO
- **JWKS Endpoint**: Public key endpoint for JWT verification by other services

JWT Configuration:

- TTL: Configurable via `JWT_TTL_SECONDS` (default: 86400)
- Algorithm: RS256

- Cookie Domain: `.newpaltz.edu`

4 Container System

4.1 Student Container Features

Each student receives a single persistent container with:

Feature	Details
Node.js	Latest LTS via nvm
Python	3.11+ with pip, venv, Jupyter
Java	OpenJDK 21
Docker	Full Docker-in-Docker support (privileged mode)
VS Code	code-server browser IDE with extensions
Jupyter	Notebook and JupyterLab
SSH	Direct SSH access via assigned port
Tools	Git, curl, wget, build-essential, etc.

4.2 SSH Access

Students can access their containers via SSH from any terminal:

```
# Connect to your container
ssh -p <assigned_port> student@hydra.newpaltz.edu

# Example with port 2222
ssh -p 2222 student@hydra.newpaltz.edu
```

- SSH ports are dynamically assigned from range 2200-2299
- Password is displayed in the dashboard after container creation
- SSH supports key-based authentication (add keys to `~/.ssh/authorized_keys`)

SSH Setup: Each container runs an SSH server via supervisord. The SSH port and password are shown in the dashboard under "SSH Access" section.

4.3 Resource Presets

Preset	RAM	CPU	Storage	GPU	Approval
Minimal	256 MB	0.5	5 GB	0	Auto
Conservative	512 MB	1	10 GB	0	Auto
Standard	1 GB	1	20 GB	0	Auto
Enhanced	2 GB	2	40 GB	0	Required
GPU Inference	32 GB	8	100 GB	1	Required
GPU Training	48 GB	16	200 GB	2	Required

GPU Access: GPU presets run on Chimera (inference) or Cerberus (training). Chimera GPUs are shared with OpenWebUI; use Cerberus for dedicated GPU work.

5 Resource Management

5.1 Time-Limited Allocations

Resource allocations are time-limited to ensure fair sharing among students:

Duration	Approval	Use Case
1 Day (Default)	Auto	Quick testing
3 Days	Auto	Short assignment
1 Week	Auto	Short projects
2 Weeks	Auto	Standard projects
1 Month	Auto	Semester project
2 Months	Required	Extended project
3 Months	Required	Full semester

5.2 Resource Expiry

When a resource allocation expires:

1. Configuration resets to **minimal** preset
2. Container moves back to **Hydra** node
3. Container automatically **restarts** to apply new limits
4. Student receives notification (if email configured)

The expiry checker runs hourly via the `resource-expiry` service.

5.3 Requesting Additional Resources

1. Navigate to Dashboard → Configure Resources
2. Select desired preset, node, and duration
3. Submit request (auto-approved if within thresholds)
4. If approval required, admin reviews within 7 days

Auto-Approval Thresholds (Hydra only):

- Memory: up to 2 GB
- CPU: up to 2 cores
- Storage: up to 20 GB

6 GPU Computing

6.1 GPU Node Configuration

Node	GPUs	Model	VRAM	Primary Use
Chimera	3	RTX 3090	72 GB total	OpenWebUI/Inference
Cerberus	2	RTX 5090	64 GB total	Student training

6.2 GPU Access Guidelines

- **Cerberus (Recommended):** Use for GPU training and student projects. RTX 5090 offers newer architecture with 32GB VRAM per card.
- **Chimera:** Reserved for OpenWebUI inference. 1 GPU is reserved for Ollama. Only use if Cerberus is unavailable.

6.3 Requesting GPU Access

1. Select "GPU Training" preset in Configure Resources
2. Choose Cerberus as target node
3. Provide justification for GPU access
4. Wait for admin approval

7 Backup System

7.1 Daily Cluster Backups

All cluster nodes are backed up daily at 1:00 AM to the Seagate drive on Hydra.

Setting	Value
Backup Location	`/mnt/sdh4/backups/`
Schedule	Daily at 1:00 AM
Method	rsync with compression
Log File	`/var/log/cluster-backup.log`

7.2 What Gets Backed Up

- **Hydra:** Full OS, configuration, application code (excludes Docker volumes)
- **Chimera:** Full OS, NVIDIA drivers, OpenWebUI config
- **Cerberus:** Full OS, NVIDIA drivers, metrics agent

7.3 Backup Exclusions

The following are excluded from backups:

- `/dev/*`, `/proc/*`, `/sys/*`, `/run/*`
- `/tmp/*`, `/var/tmp/*`, `/var/cache/*`

- /mnt/*, /media/*, /lost+found
- /var/lib/docker/* (Docker data)

7.4 Manual Backup

```
# Run backup manually
sudo /home/infra/backup-cluster.sh

# Check backup status
cat /var/log/cluster-backup.log

# View backup sizes
du -sh /mnt/sdh4/backups/*
```

8 File Structure

```
hydra-saml-auth/
|-- index.js              # Main entry: SAML, JWT/JWKS, routes,
   WebSocket
|-- routes/
|   |-- containers.js     # Container lifecycle, services, ports, logs
|   |-- resource-requests.js  # Resource allocation requests
|   |-- webui-api.js      # OpenWebUI account proxy
|   |-- n8n-api.js        # n8n account management
|   |-- servers-api.js    # Cluster status endpoints
|   |-- admin.js          # Admin panel routes
|-- services/
|   |-- db-init.js        # Database initialization and migrations
|   |-- resource-expiry.js    # Time-limited resource expiry checker
|   |-- activity-logger.js    # Activity tracking
|   |-- email-notifications.js # Email alerts
|   |-- metrics-collector.js  # Node metrics collection
|-- config/
|   |-- resources.js      # Resource presets and node configuration
|-- agents/
|   |-- metrics-agent.js  # Node.js metrics agent (Hydra)
|-- scripts/
|   |-- metrics-agent.py  # Python metrics agent (GPU nodes)
|-- views/                # EJS templates
|-- student-container/
|   |-- Dockerfile        # Ubuntu 22.04 + dev tools + SSH
|   |-- supervisord.conf  # Process manager config (incl. sshd)
|   |-- entrypoint.sh     # Container startup
|-- ansible/              # Cluster deployment playbooks
|   |-- inventory.yml     # Node definitions
|   |-- playbooks/        # Deployment scripts
|-- docker-compose.yaml   # Production stack
|-- docs/                 # Documentation
```

9 Common Operations

9.1 View Running Containers

```
docker ps --filter "name=student-"
```

9.2 Access Container Shell

```
docker exec -it student-<username> /bin/bash
```

9.3 View Container Logs

```
docker logs -f student-<username> --tail=100
```

9.4 Restart a Container

```
docker restart student-<username>
```

9.5 Remove a Stuck Container

```
docker rm -f student-<username>
```

9.6 Rebuild Student Container Image

```
cd student-container
docker build -t hydra-student-container:latest .
```

Note: Students with existing containers must recreate them to use updated images.

9.7 Check Cluster Node Status

```
# Check metrics from Chimera
curl http://192.168.1.150:9100/metrics

# Check metrics from Cerberus
curl http://192.168.1.242:9100/metrics
```

9.8 Trigger Resource Expiry Check

```
# From within the application
curl http://localhost:6969/api/admin/resource-expiry/check
```

10 Service Management

10.1 Restart Main Service

```
docker compose restart hydra-saml-auth
```

10.2 Rebuild and Redeploy

```
docker compose build hydra-saml-auth
docker compose up -d hydra-saml-auth
```

10.3 View Service Logs

```
docker compose logs -f hydra-saml-auth
```

10.4 Check Traefik Routing

```
docker compose logs traefik | grep -i error
curl -I https://hydra.newpaltz.edu/
```

10.5 Manage Metrics Agent (GPU Nodes)

```
# On Chimera or Cerberus
sudo systemctl status metrics-agent
sudo systemctl restart metrics-agent
sudo journalctl -u metrics-agent -f
```

11 Troubleshooting

11.1 Authentication Issues

Symptom	Solution
SAML assertion invalid	Verify `METADATA_URL` and `SAML_SP_ENTITY_ID` match Azure config exactly
Cookie not set	Check `COOKIE_DOMAIN`, ensure HTTPS, check browser settings
JWT verification fails	Verify JWKS endpoint accessible, check key rotation

11.2 Container Issues

Symptom	Solution
Container won't initialize	Verify `hydra-student-container:latest` image exists
Container 404	Check container is on `hydra_students_net`, Traefik running
Service won't start	Check supervisord logs inside container
Port routing fails	Verify port not reserved (8443, 8888) and not in use
SSH not working	Check sshd process in container, verify port assignment

11.3 GPU Issues

Symptom	Solution
GPU not detected	Run `nvidia-smi` on host, check NVIDIA drivers
GPU container fails	Verify nvidia-container-toolkit installed
Metrics not showing	Check metrics-agent service, firewall port 9100

11.4 Service-Specific Issues

- **VS Code not loading:** Check code-server process, ForwardAuth working
- **Jupyter issues:** Verify `NotebookApp.base_url` setting
- **Docker-in-Docker fails:** Container must have privileged mode
- **Files not persisting:** Only `/home/student/` is persisted
- **Resource expiry not working:** Check resource-expiry service logs

12 Ansible Deployment

12.1 Cluster Setup Overview

The cluster can be deployed using Ansible playbooks in `ansible/` directory:

```
# Full cluster deployment
cd ansible
ansible-playbook -i inventory.yml playbooks/site.yml
```

12.2 Playbook Execution Order

1. `00-preflight-backup.yml` - Create backups before changes
2. `01-prepare-nodes.yml` - Install packages, configure kernel
3. `02-rke2-server.yml` - Setup RKE2 control plane
4. `03-rke2-agents.yml` - Join GPU nodes to cluster
5. `04-gpu-setup.yml` - Configure NVIDIA drivers and GPU Operator
6. `05-deploy-hydra.yml` - Deploy Hydra application stack

12.3 Inventory Configuration

The cluster inventory is defined in `ansible/inventory.yml`:

```
# Key variables
rke2_version: "v1.28.4+rke2r1"
cluster_domain: hydra.newpaltz.edu
nfs_server: "192.168.1.160"
nfs_path: "/srv/hydra-nfs"
```

13 Environment Configuration

13.1 Required Variables

Variable	Description
BASE_URL	External URL (https://hydra.newpaltz.edu)
METADATA_URL	Azure AD federation metadata URL
SAML_SP_ENTITY_ID	SP Entity ID (must match Azure exactly)
COOKIE_DOMAIN	Cookie scope (.newpaltz.edu)
PORT	Service port (default: 6969)
DB_PATH	Database path (/app/data/webui.db)

13.2 Optional Variables

Variable	Description
PUBLIC_STUDENTS_BASE	Student URL base
JWT_TTL_SECONDS	JWT token lifetime
CHIMERA_HOST	Chimera IP (default: 192.168.1.150)
CERBERUS_HOST	Cerberus IP (default: 192.168.1.242)
STUDENT_IMAGE	Default container image
GPU_STUDENT_IMAGE	GPU container image

14 Monitoring

14.1 Servers Dashboard

The `/servers` page displays real-time metrics for all cluster nodes:

- CPU usage and load average
- Memory usage
- Disk usage and ZFS pool status
- Container count
- GPU utilization (Chimera/Cerberus)
- GPU temperature and VRAM usage

14.2 Metrics Collection

Node	Agent	Port
Hydra	metrics-collector.js (internal)	N/A
Chimera	metrics-agent.py	9100
Cerberus	metrics-agent.py	9100

15 References

- Docker Documentation: https://docs.docker.com/

- Traefik Documentation: https://doc.traefik.io/traefik/
- SAML 2.0 Specification: https://docs.oasis-open.org/security/saml/v2.0/
- Azure AD SAML: https://docs.microsoft.com/en-us/azure/active-directory/develop/single-sign-on-saml-protocol
- code-server: https://coder.com/docs/code-server/latest
- Jupyter: https://jupyter.org/documentation
- RKE2 Documentation: https://docs.rke2.io/
- NVIDIA GPU Operator: https://docs.nvidia.com/datacenter/cloud-native/gpu-operator/