



**Hewlett Packard
Enterprise**

New User Experiences with K3s and MetalLB on Managed Nodes



Alan Mutschelknaus: HPE Cray CSM User Team
Presented by: Jeremy Duckworth and Harold Longley

May 2023

Containerized Logins

User Access Instances on
Managed Hardware (e.g.
UANs)

Interactive Podman
Containers

K3s

Infrastructure Deployed
using K3s

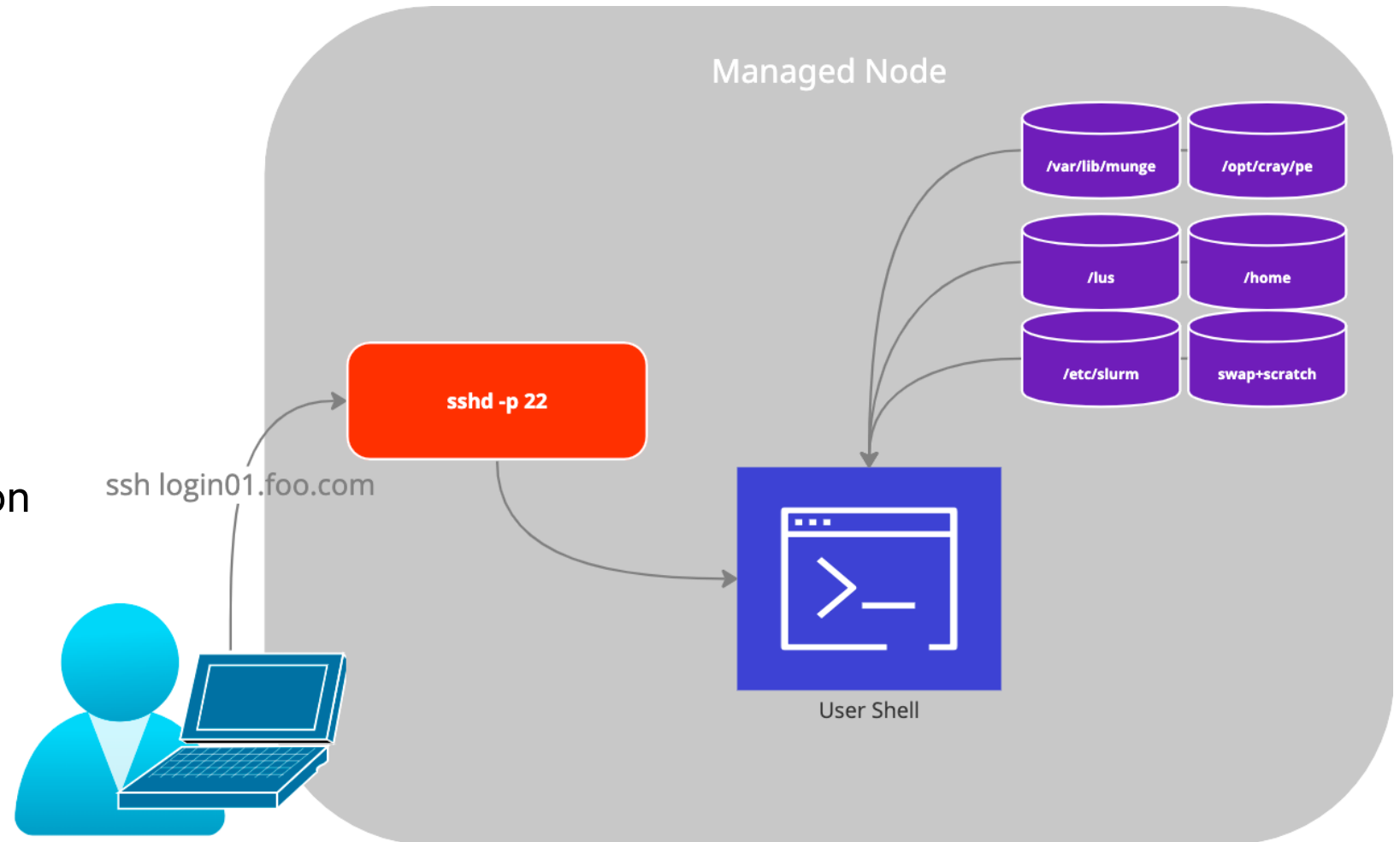


Helm and K3s on Managed
Hardware == Opportunity



Standard SSH Model

- Tried and true configuration for HPC compilation and job launch
- User uses SSH to initiate a session on the node
- Cray Programming Environment available
- Job launch via Workload Managers (Slurm, PBS, etc)



CGU 2022 Best Paper

- This design evolves the concepts discussed by Eric Lund at CGU 2022
- Works towards a solution that does not host UAIs in management K8s and further isolates user processes and decouples the architecture towards user value

UAIs Come of Age: Hosting Multiple Custom Interactive Login Experiences Without Dedicated Hardware

Eric Lund
Hewlett Packard Enterprise
Bloomington, MN
eric.lund@hpe.com

Abstract— On HPE Cray EX systems, User Access Instances (UAIs) provide convenient lightweight temporary single-user interactive login environments. Access to a set of End-User UAIs is mediated by a Broker UAI which presents an SSH login to users. On successful login, the Broker UAI locates or creates an End-User UAI as needed for the user, then redirects the user to the End-User UAI. On reaching the End-User UAI the user sees a login environment that supports interactive work and launching of

matches the runtime environment in which the software will execute.

UAIs on the other hand appear on-demand and can run shorter- or longer-term tasks without requiring a permanent presence on the system. UAIs connect to external storage through Volume Mounts but contain no persistent state of their own. As a result, UAIs are disposable, and easily come and go

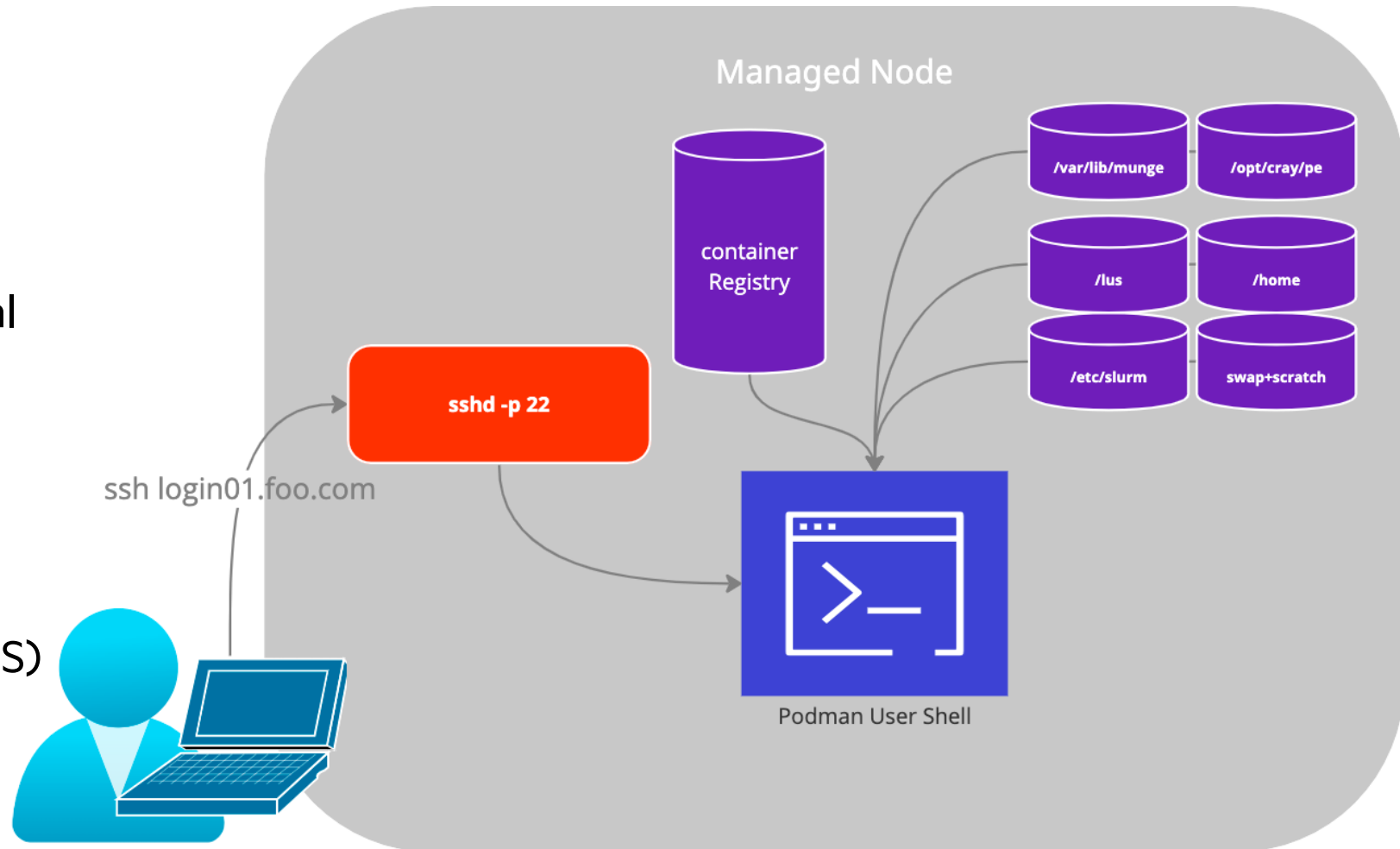
Shifting the Execution Model

Hosting on UANs; (rootless) Podman; K3s, MetalLB, and HAProxy for light-weight orchestration



SSH with Podman

- User uses SSH to initiate a terminal on the node
- Force the user into a Podman container with the SSHD configuration ForceCommand
- Use host network namespace to allow for job launch (e.g. Slurm, PBS)



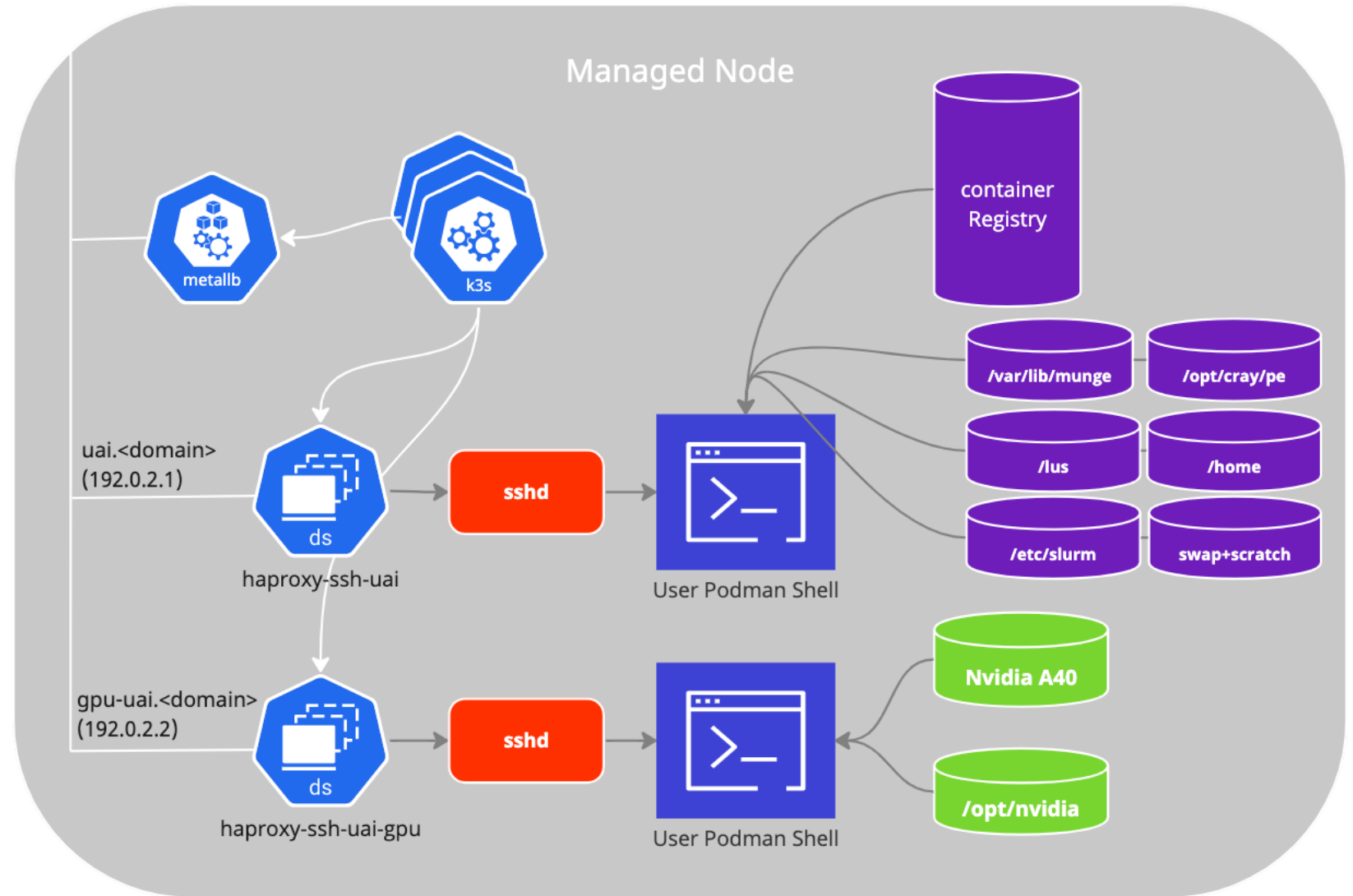
SSH with Podman

- Pros
 - No extra steps for the user
 - Container image flexibility, like UAIs
 - Containers run as user (rootless)
 - Entry point can be the shell (no SSHD in the container like UAIs)
- Cons
 - Doesn't load balance across managed nodes
 - Requires port 22 on the host for SSH ForceCommand

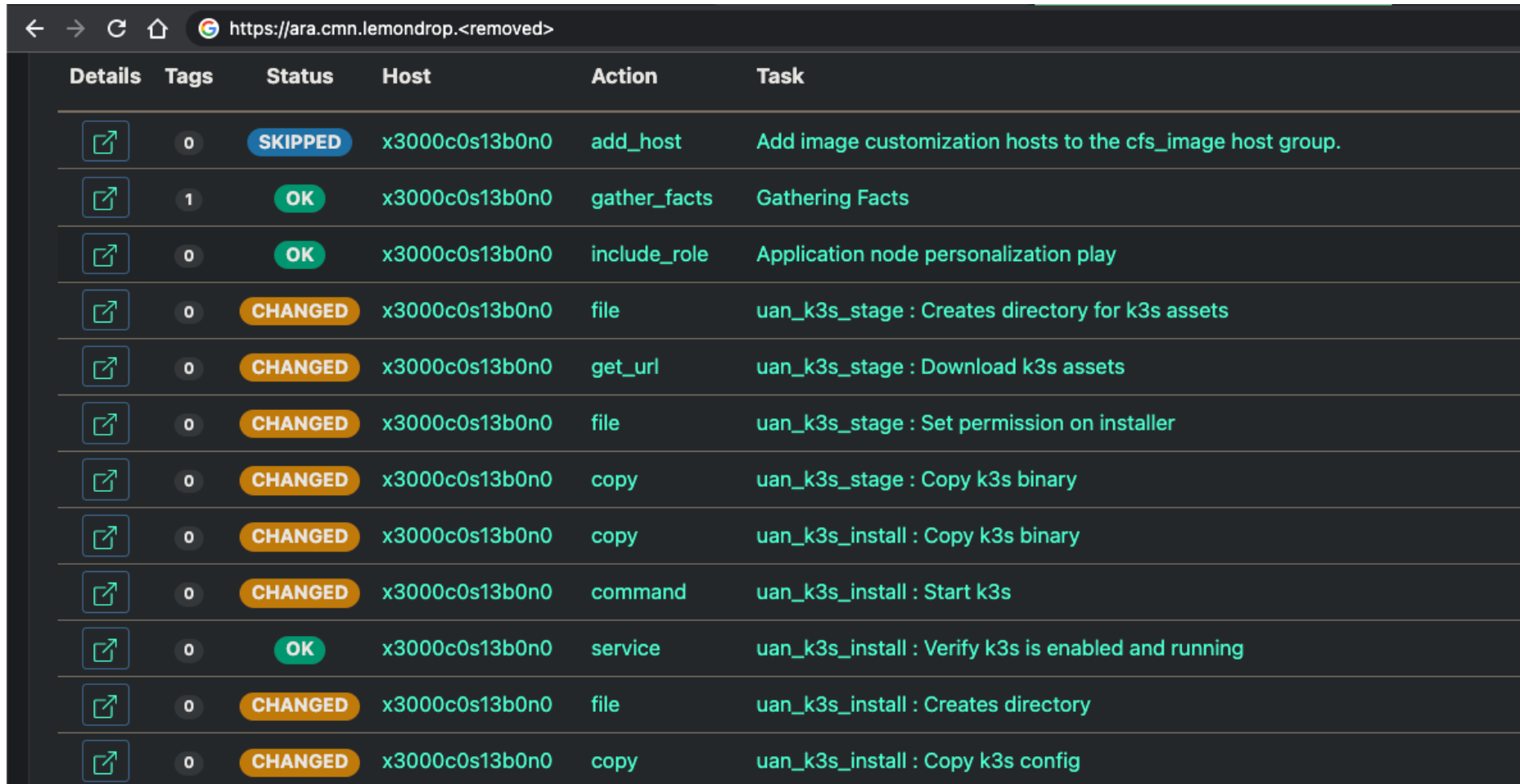
```
uan01:~ # cat /etc/ssh/uan/sshd_uai_config
...
# uan_sshd managed changes:
Port                22
Match User *
    AcceptEnv DISPLAY
    X11Forwarding yes
    AllowTcpForwarding yes
    PermitTTY yes
    ForceCommand podman --root /scratch/containers/$USER
run -it -h uai --cgroup-manager=cgroupfs --userns=keep-
id -v /home/users:/home/users -v
/var/run/slurm/conf/:/etc/slurm ... --network=host -e
DISPLAY=$DISPLAY registry.local/cray/uai:1.0
...
```

K3s for SSH Load Balancing

- Add MetalLB to hand out external IP addresses
- Add HAProxy for SSH load-balancing
 - Each HAProxy represents a “type” of SSH ingress that initiates a rootless podman container
 - HAProxy can scale out across the managed nodes in the K3s cluster
- Podman containers still run on the Managed Node outside the purview of K3s
 - This may change in the future if orchestration of the user environment is required



Ansible Deployment



The screenshot shows a web interface for an Ansible deployment. At the top, there is a browser address bar with the URL `https://ara.cmn.lemondrop.<removed>`. Below the address bar is a navigation bar with tabs for 'Details', 'Tags', 'Status', 'Host', 'Action', and 'Task'. The main content area is a table with columns for 'Details', 'Tags', 'Status', 'Host', 'Action', and 'Task'. Each row represents a task execution. The 'Status' column contains colored labels: 'SKIPPED' (blue), 'OK' (green), and 'CHANGED' (orange). The 'Host' column shows the host ID 'x3000c0s13b0n0'. The 'Action' column lists the Ansible action type, and the 'Task' column provides a description of the task.

Details	Tags	Status	Host	Action	Task
	0	SKIPPED	x3000c0s13b0n0	add_host	Add image customization hosts to the cfs_image host group.
	1	OK	x3000c0s13b0n0	gather_facts	Gathering Facts
	0	OK	x3000c0s13b0n0	include_role	Application node personalization play
	0	CHANGED	x3000c0s13b0n0	file	uan_k3s_stage : Creates directory for k3s assets
	0	CHANGED	x3000c0s13b0n0	get_url	uan_k3s_stage : Download k3s assets
	0	CHANGED	x3000c0s13b0n0	file	uan_k3s_stage : Set permission on installer
	0	CHANGED	x3000c0s13b0n0	copy	uan_k3s_stage : Copy k3s binary
	0	CHANGED	x3000c0s13b0n0	copy	uan_k3s_install : Copy k3s binary
	0	CHANGED	x3000c0s13b0n0	command	uan_k3s_install : Start k3s
	0	OK	x3000c0s13b0n0	service	uan_k3s_install : Verify k3s is enabled and running
	0	CHANGED	x3000c0s13b0n0	file	uan_k3s_install : Creates directory
	0	CHANGED	x3000c0s13b0n0	copy	uan_k3s_install : Copy k3s config

User Experience

- Users SSH to DNS alias of the HAProxy IP Address assigned by MetalLB
- Each HAProxy instance has a configurable list of Managed Nodes to forward connections to
- SSHD instance on the Managed Node initiates the configurable Podman environment

```
# SSH connection is routed to HAProxy and then SSHD
→ ~ ssh uai.can.lemondrop.$DOMAIN
```

```
# SSHD starts the podman container and drops the user into a shell
alanm@uai:~> sinfo
PARTITION AVAIL  TIMELIMIT  NODES  STATE NODELIST
workq*      up    infinite     18   idle nid[000001-000018]
```

```
# Running a simple job from the container
alanm@uai:~> srun -N4 /lus/snx11010/alanm/mpi_hello
Hello world!  I am rank 3 of 4
Hello world!  I am rank 0 of 4
Hello world!  I am rank 1 of 4
Hello world!  I am rank 2 of 4
```

```
# Exiting the shell triggers the podman container to exit
alanm@uai:~> exit
exit
Connection to uai.can.lemondrop.$DOMAIN closed
```

Admin Experience

- HAProxy K3s ConfigMap controls the SSH load balancing
- Each HAProxy represents an ingress into a configurable environment
- SSHD instance on the Managed Node initiates the configurable Podman environment with ForceCommand

```
uan01:~ # kubectl get services -A
NAMESPACE          NAME                TYPE           CLUSTER-IP      EXTERNAL-IP      PORT(S)
default            kubernetes         ClusterIP      x.x.x.x         <none>           443/TCP
metallb-system     metallb-webhook-service ClusterIP      x.x.x.x         <none>           443/TCP
haproxy-uai        haproxy-uai        LoadBalancer   x.x.x.x.        x.x.x.224       22:31865/TCP
haproxy-gpu        haproxy-gpu        LoadBalancer   x.x.x.x         x.x.x.225       22:31865/TCP
```

```
uan01:~ # kubectl describe -n haproxy-uai cm/haproxy-uai
```

```
...
haproxy.cfg:
----

global
  log stdout format raw local0
  maxconn 1024
defaults
  log      global
  mode    tcp
  timeout connect 10s
  timeout client 36h
  timeout server 36h
  option dontlognull
listen ssh
  bind *:22
  balance leastconn
  mode tcp
  option tcp-check
  tcp-check expect rstring SSH-2.0-OpenSSH.*
  server uan01 uan01.can.lemondrop.<domain>:9000 check inter 10s fall 2 rise 1
```

Current Limitations

- Rootless Podman Containers and Lustre
 - The user namespace for the container shifts UIDs and GIDs outside the range that Lustre currently understands
 - While the underlying permissions are still correct, files will be shown as being owned by "nobody" and may represent a usability issue
- Secure Copy (SCP)
 - As the UAN Host SSHD is ultimately configured to use SSH Force directives, there are some use cases where SCP does not behave as intended



Future Opportunities

- **Explore load-balancing across systems**

```
listen ssh
bind *:22
balance leastconn
mode tcp
server hermod_uan01 uan01.chn.hermod.<domain>:9001 check inter 10s fall 2 rise 1
server hermod_uan02 uan02.chn.hermod.<domain>:9001 check inter 10s fall 2 rise 1
server loki_uan03 uan03.chn.loki.<domain>:9001 check inter 10s fall 2 rise 1
```

- **Deploy additional services to K3s**

```
helm repo add jupyterhub https://jupyterhub.github.io/helm-chart/
helm upgrade --install jupyterhub jupyterhub/jupyterhub --namespace jupyter --
create-namespace --values config.yaml
```

- **Consider alternatives to rootless Podman as the container runtime**

- Virtual Machines
- K3s pods (similar to existing UAIs)



Questions

<https://github.com/Cray-HPE/uan>

Feature is available as a technical preview in UAN 2.5.13 and 2.6.0

alanm@hpe.com

