Hewlett Packard
Enterprise

UAIS COME OF AGE:

HOSTING MULTIPLE CUSTOM INTERACTIVE LOGIN EXPERIENCES WITHOUT DEDICATED HARDWARE

Eric Lund, HPE CSM Engineer

May 3, 2022

AGENDA

- Introduction and Overview
- UAS / UAI Relationships
- UAI Deeper Dive
- UAS / UAI Configuration
- Broker UAI Creation
- Some Possible Use Cases
- Resources

INTRODUCTION AND OVERVIEW

UANs and UAIs can both provide interactive login access to an HPE Cray EX system.
For long-running operations or memory-intensive activities, UANs are most suitable.
For flexible, cloud-like interactions, UAIs are most suitable.

User Access Node (UAN)

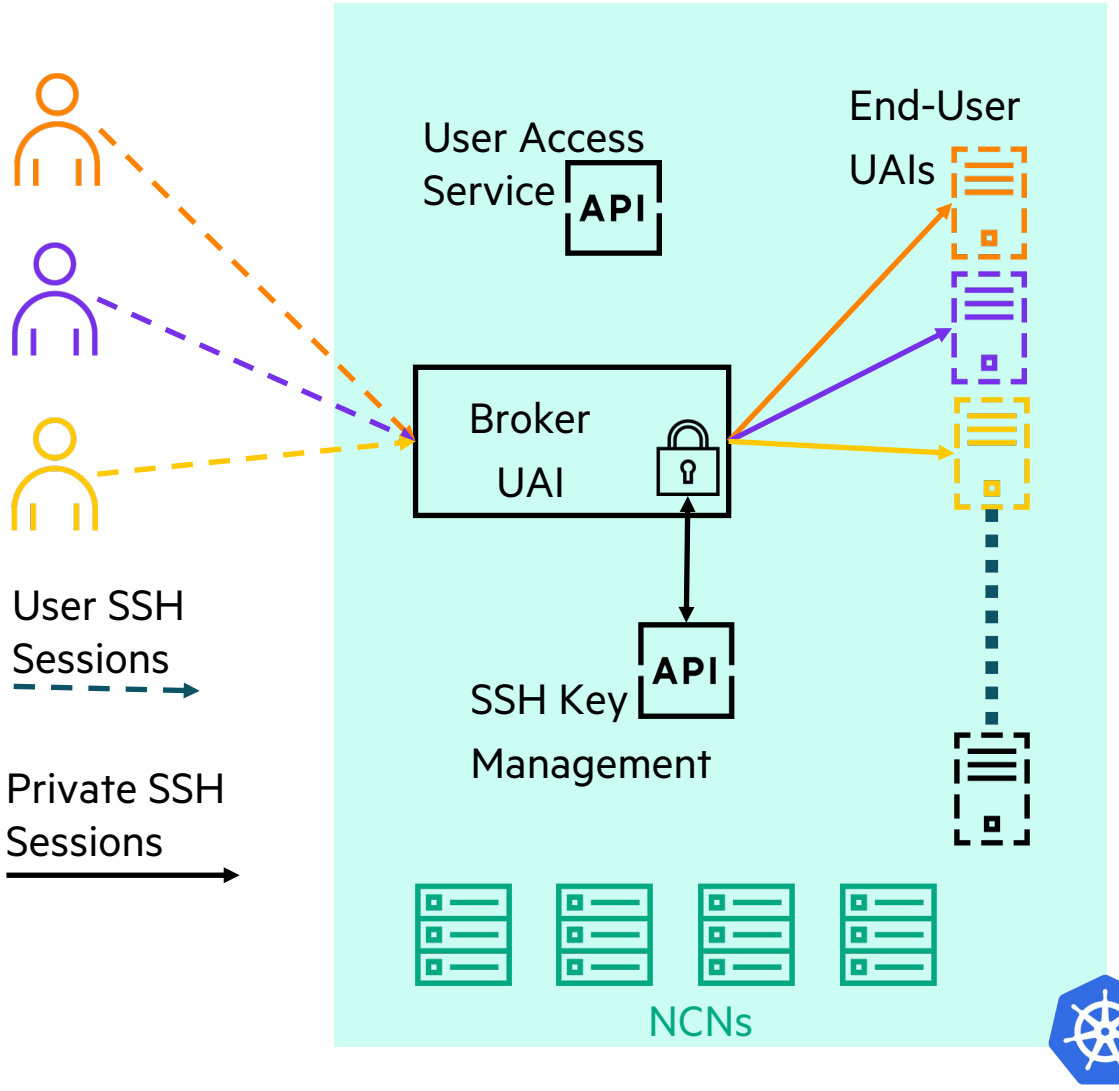
- Dedicated multi-user node (HW & SW) ideal for stable long term persistent tasks
- Can be configured to match compute nodes to ensure a common programming environment
- Can take full advantage of all hardware resources, for example GPUs, and swap for memory-intensive workloads

User Access Instances (UAI)

- On-demand: ideal for short-term interactive tasks
- Disposable: environments come and go without loss of user data
- Content easily customized for specific activities using both images and volume mounts
- Resource requirements easily customized for specific activities using resource specifications

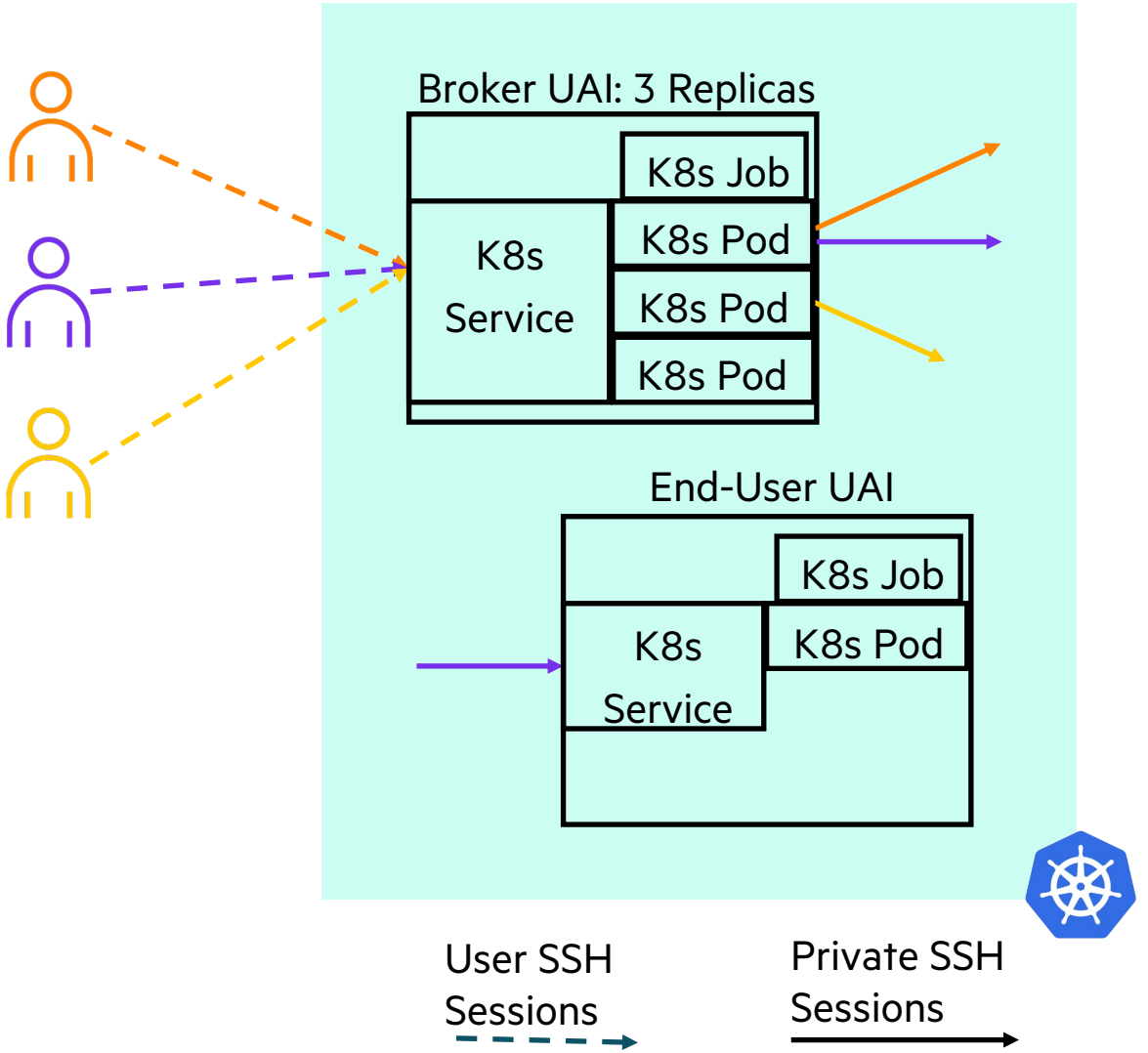
UAS / UAI RELATIONSHIPS

- The User Access Service (UAS) manages UAIs and UAI Configuration
- Two broad kinds of UAIs
  - Broker UAIs
  - End-User UAIs
- Broker UAIs
  - Face multiple users on external IP
  - Select or Create End-User UAIs on demand
  - Forward SSH sessions to End-User UAIs over private SSH sessions
  - Share private session keys among replicas using key management
- End-User UAIs each face a single user on internal Kubernetes IP
- All UAIs are Orchestrated by Kubernetes on NCNs



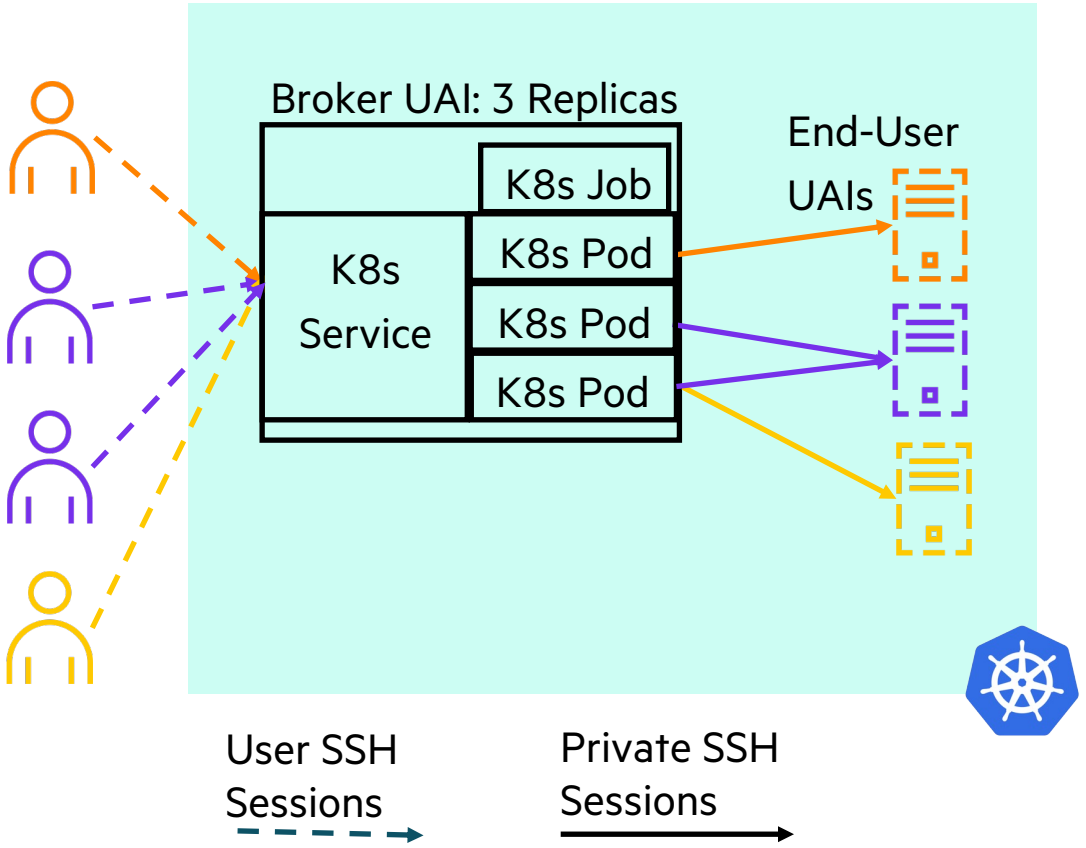
UAI DEEPER DIVE: STRUCTURE OF UAIS

- All UAIs composed of
  - Kubernetes service for network ingress
  - Kubernetes job to orchestrate pod lifecycles
  - At least 1 Kubernetes pod to run the UAI container
  - UAI terminates when pod hits successful completion
- End-User UAIs
  - Created by Broker UAIs
  - Accept forwarded SSH from Broker UAIs
  - Authenticate private user connections in pod based on a private session SSH key-pair known only to the Broker UAI
  - Execute user sessions in the UAI container
  - Pod will complete successfully based on timeouts
- Broker UAIs
  - Created by administrator
  - Accept SSH connections through service
  - Authenticate connections in pods using administratively configured Auth / Auth rules / domains.
  - Connections can be load balanced on multiple pods
  - Pods never complete successfully (broker runs indefinitely)



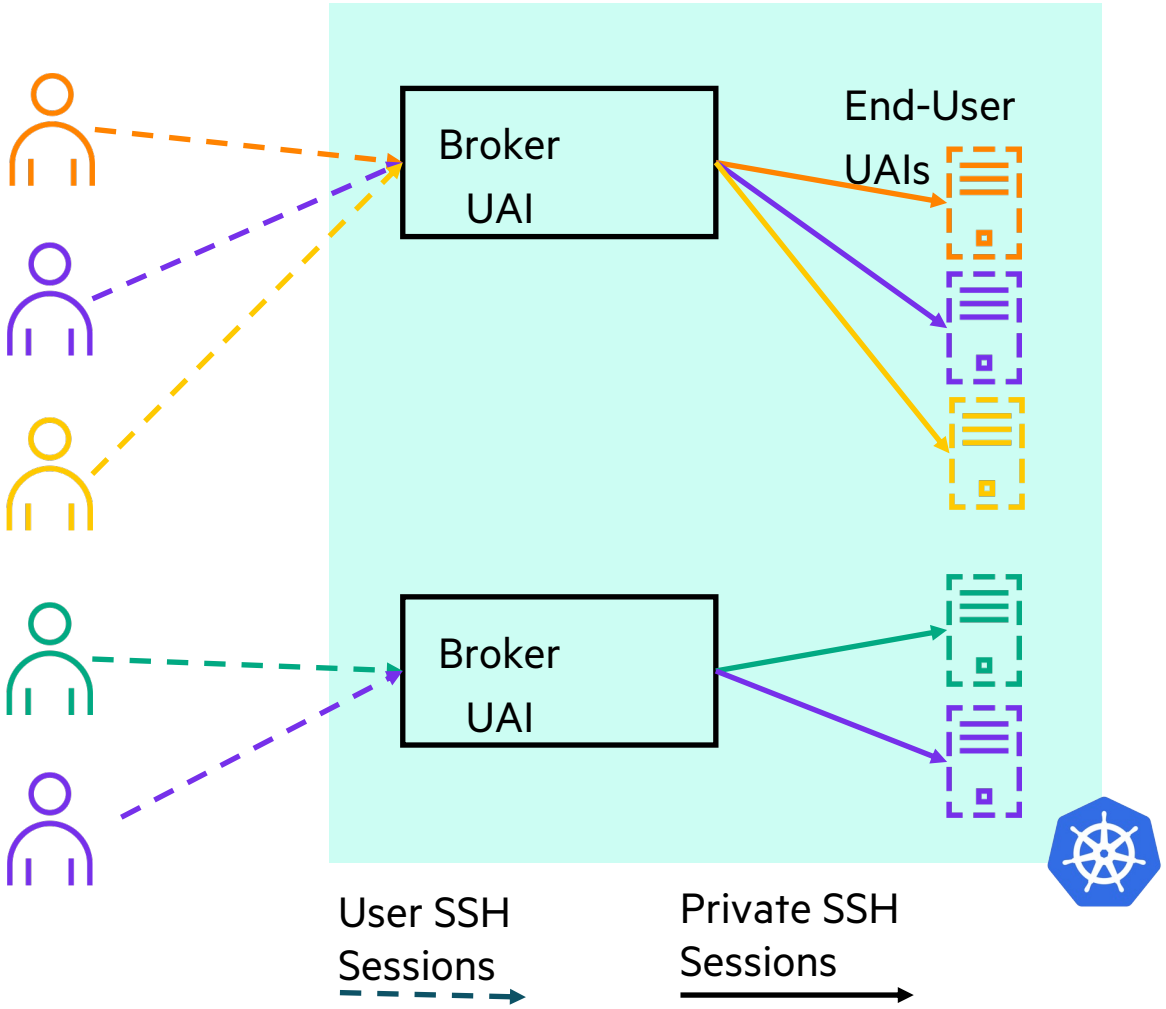
UAI DEEPER DIVE: LOAD BALANCING AND UAI SHARING

- Broker UAI connections
  - Are load balanced across replica pods by the service
  - Are directed to the owner's End-User UAI if present
  - May travel through different replica pods to reach the same End-User UAI
- In this example Purple has two sessions
  - Each is directed by a different replica pod
  - Both reach the same (Purple) End-User UAI
  - Both use the same Private SSH credentials



UAI DEEPER DIVE: MULTIPLE BROKER UAIS

- Each Broker UAI
  - Has its own IP address and External DNS Hostname
  - Can have its own Auth/Auth rules or domain
  - Manages End-User UAIs of the same class
- End-User UAIs under a given Broker UAI
  - Are tailored to a specific set of use-cases
  - Share tailored storage and other external data
  - Run a potentially tailored UAI image
  - Have tailored resource limits and requirements
- In this example
  - Green connects through lower Broker UAI to green End-User UAI
  - Purple connects through both Broker UAIs
    - Purple reaches two distinct End-User UAIs
    - Each Purple session is tailored to a different purpose



UAS CONFIGURATION: MAJOR ELEMENTS

- UAI Image Registration
  - Tells UAS what UAI Images (container Images) are available for use to create UAIs
  - Assigns an identifier to each UAI Image for use in UAI classes
- UAI Volume
  - Describe external data or storage resources and where they should be mounted inside a running UAI container
  - Can describe any volume type that Kubernetes supports
  - Assigns an identifier to each description for use in UAI classes
- UAI Resource Specification
  - Describes resource limits and requests that can be assigned to UAI pods on creation
  - Guides Kubernetes scheduling of UAI pods based on resource availability
  - Can describe any resource limit or request supported by Kubernetes
  - Assigns an identifier to each description for use in UAI classes
- UAI Class
  - Provides a configuration template for creating UAIs
  - Assigns an identifier to the configuration template for use in creating UAIs
  - Used by Broker UAIs to create End-User UAIs (creation class found in the Broker UAI class)
  - Used by administrators to create Broker UAIs (UAI class provided on the command line)

UAS CONFIGURATION: FOCUS ON UAI CLASSES

- In addition to UAI image ID, a UAI volume list, and UAI resource specification a UAI class contains
  - End-User UAI creation class (for Broker UAI classes only)
  - Replica count (generally > 1 only on Broker UAIs, generally == 1 on End-User UAIs)
  - Internal or external IP network ingress choice (usually external for Broker, internal for End-User)
  - End-User UAI timeouts
    - Soft timeouts cause the End-User UAI to terminate after a certain amount of time if the UAI is or becomes idle
    - Hard timeouts cause the End-User UAI to terminate after a certain amount of time whether the UAI is idle or not
    - Warning issues a message to all sessions on the End-User UAI a configured amount of time before a hard timeout
  - Kubernetes tolerations to inform Kubernetes host node placement of UAI pods
  - Other settings are described in more detail in documentation

BROKER UAI CREATION

- Minimally requires UAI class ID for the Broker UAI
- Optionally specify
  - UAI Name for use in External DNS
  - UAI Owner string (arbitrary) for easier management
- In this example
  - UAI class ID is the UUID shown
  - UAI name used by External DNS to compose hostname is “workload-monitoring-uai”
  - UAI owner string is “workload-monitoring-broker”
- Providing a UAI owner facilitates
  - Filtering UAI Lists by owner to examine UAI status
  - Deleting the Broker UAI by its owner

```
$ cray uas admin uais create \
  --class-id \
  a9d61724-976a-4a07-85f8-00e422bff3ce \
  --uai-name workload-monitoring-uai \
  --owner workload-monitoring-broker
```

EXAMPLE UAI USE CASES: BACKGROUND

- Project Isolation
  - Distinct Auth / Auth rules or domains allow different user sets to log into different Broker UAIs
  - Distinct storage configuration (volumes) in End-User UAI Classes facilitate isolation of data to groups of UAIs
  - Kubernetes Taints on nodes and Tolerations in UAI Classes allow control of UAI placement on host nodes
- UAI Specialization
  - Volumes and UAI Image contents tailor mission specific UAIs for content
  - Volumes tailor storage access to mission specific datasets
  - Resource specifications tailor UAI resources to mission needs permitting better host node utilization
  - Kubernetes taints and tolerations in UAI classes target nodes with specialized resources as needed
  - Timeouts tailor End-User UAI lifecycles to maximize UAI response or minimize idle UAIs

EXAMPLE UAI USE CASES: WORKLOAD DEVELOPMENT AND TESTING

- UAIs for workload development and testing activities
  - Likely long running user sessions
  - Can be memory and CPU intensive when compiling code or analyzing results
  - Long idle periods between long interactive sessions
- UAI characteristics
  - Resource limits – need to support memory and CPU intensive compilation tasks
    - Memory: large or very large
    - CPU: large
  - Taints and Tolerations
    - direct scheduling to higher capacity host nodes
    - Keep development activity away from production host nodes
  - Timeouts
    - Hard: long or none to permit extended interactive use
    - Soft: short for quick cleanup after interactive sessions
  - Programming environment tools in either the image or mounted volumes
  - Development storage access through volumes
  - Workload Manager access for launching and monitoring test runs
  - Authorized logins: developers

EXAMPLE UAI USE CASES: PROUCTION WORKLOAD LAUNCH

- UAIs for production workload launch
  - Users or scripts create sessions to initiate workloads
  - Infrequent low intensity user activity
  - Want minimal resource usage and fast cleanup after sessions
- UAI characteristics
  - Resource limits – should only need to run workload manager commands
    - Memory: small
    - CPU: small
  - Taints and tolerations direct scheduling to production only host nodes
  - Timeouts
    - Hard: modest to encourage quick launch or status gathering
    - Soft: short for quick cleanup after interactive sessions
  - Production software either in the image or accessed through mounted volumes
  - Production storage accessed through volumes
  - Workload Manager access for launching / monitoring production runs
  - Authorized logins: production staff

EXAMPLE UAI USE CASES: PRODUCTION WORKLOAD MONITORING

- UAIs for production workload monitoring
  - Most likely scripted frequent periodic gathering of workload status
  - Invoked as SSH commands
  - Want low UAI startup overhead
- UAI characteristics
  - Resource limits – should only need to run workload manager commands
    - Memory: small
    - CPU: small
  - Taints and Tolerations direct scheduling to production only host nodes
  - Timeouts
    - Hard: none to avoid session failure
    - Soft: long to minimize UAI startup overhead over multiple quick sessions
  - No access to production software or development tools – not needed for monitoring
  - No access to production or development storage – not needed for monitoring
  - Workload Manager access for monitoring production runs
  - Authorized logins: production staff

RESOURCES

- UAS and UAI Documentation
  - https://github.com/Cray-HPE/docs-csm/blob/main/operations/UAS_user_and_admin_topics/index.md
- UAS and UAI related code is public and open source
  - UAS Manager
    - https://github.com/Cray-HPE/uas-mgr
  - UAS Configuration Installer / Updater and HPE Supplied UAI Images
    - https://github.com/Cray-HPE/uai-images
  - Switchboard Tool (the heart of the Broker UAI)
    - https://github.com/Cray-HPE/switchboard
  - UAI Utilities (including the End-User UAI entry point script)
    - https://github.com/Cray-HPE/uai-util

THANK YOU

Eric Lund – eric.lund@hpe.com