Introduction to the Blue Planet Platform

Instructor-led training with hands-on lab exercises

blueplanet
a division of Ciena

blueplanet®
a division of Ciena

Blue Planet Platform Overview

Introduction to the Blue Planet Platform

PLF111ILT-A, Revision 1.0

Blue Planet Platform Introduction

blueplanet
a division of Ciena

Objectives

- Describe Microservice Architecture
- Examine Docker Containers in Blue Planet
- Discover Blue Planet Architecture Model
- Identify Layers of Blue Planet Common Platform

Blue Planet Platform Introduction

Agenda

1. **Blue Planet Platform Introduction**
2. Blue Planet Microservices
3. Blue Planet Platform Architecture

Blue Planet Platform

- **Blue Planet offers a variety of products that run on a common Blue Planet Platform.**
- **Blue Planet applications run as Docker-based containerized microservices.**
- **Common platform components are used to install, manage, and monitor Blue Planet Applications as well as the platform itself.**

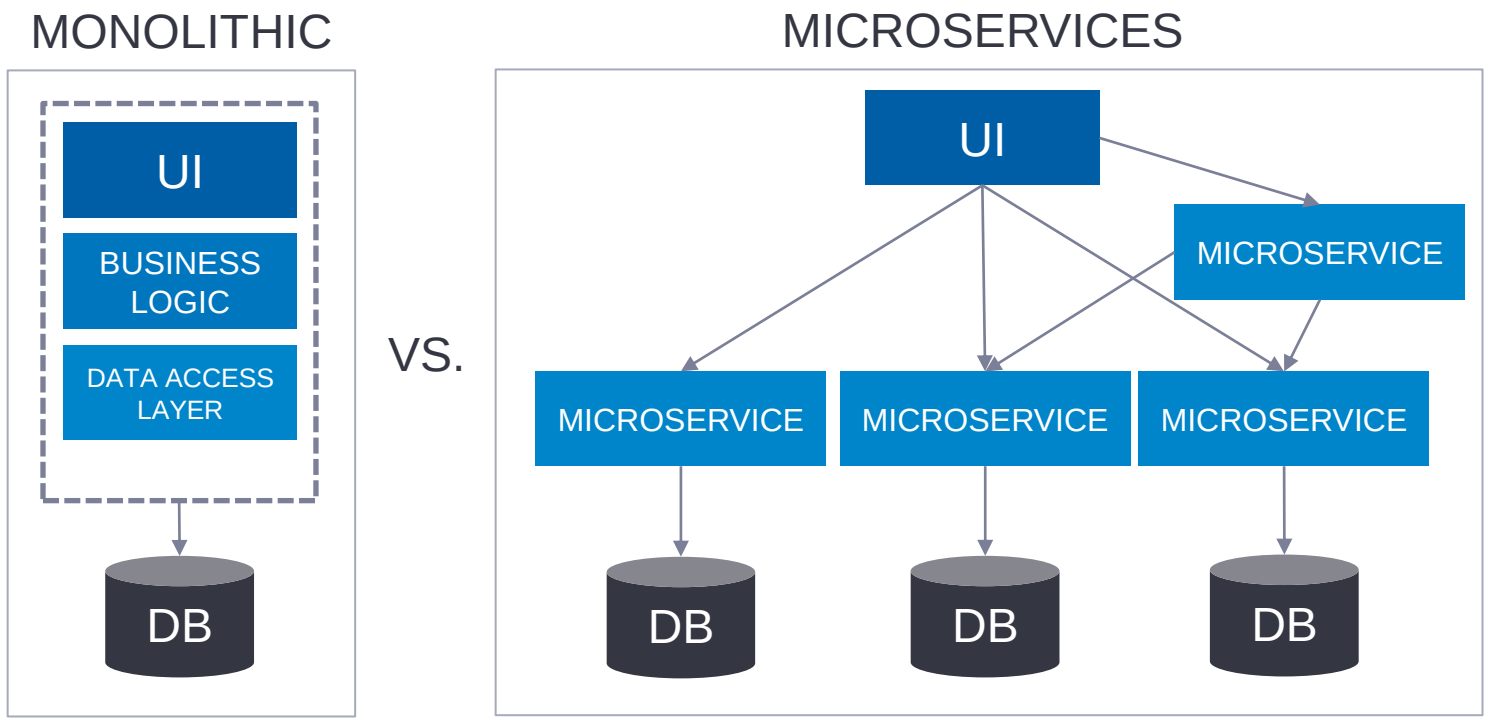
- **In the next slides, we will learn what are the components that make the Blue Planet Platform and how they will be used to operate Blue Planet solutions.**

Agenda

1. Blue Planet Platform Introduction
2. **Blue Planet Microservices**
3. Blue Planet Platform Architecture

Blue Planet Microservices

- **Blue Planet applications run as microservices that run in Docker containers.**
- **Applications in microservice architecture are structured as a collection of separate services that are connected by an internal, private network.**
- **This makes them easier to rebuild, develop, and deploy, and gives better scalability and fault isolation, but it brings some configuration and environment management challenges.**

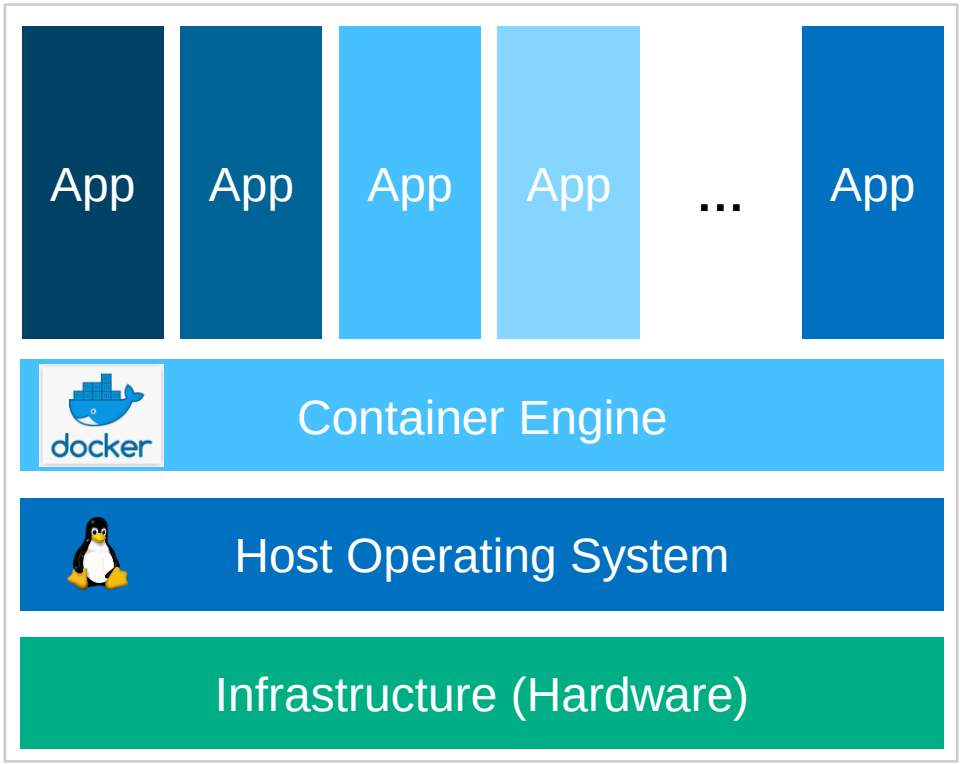


Note: Blue Planet Platform provides a set of applications to support easier microservices environment setup and Operations and Monitoring tasks.

blueplanet
a division of Ciena

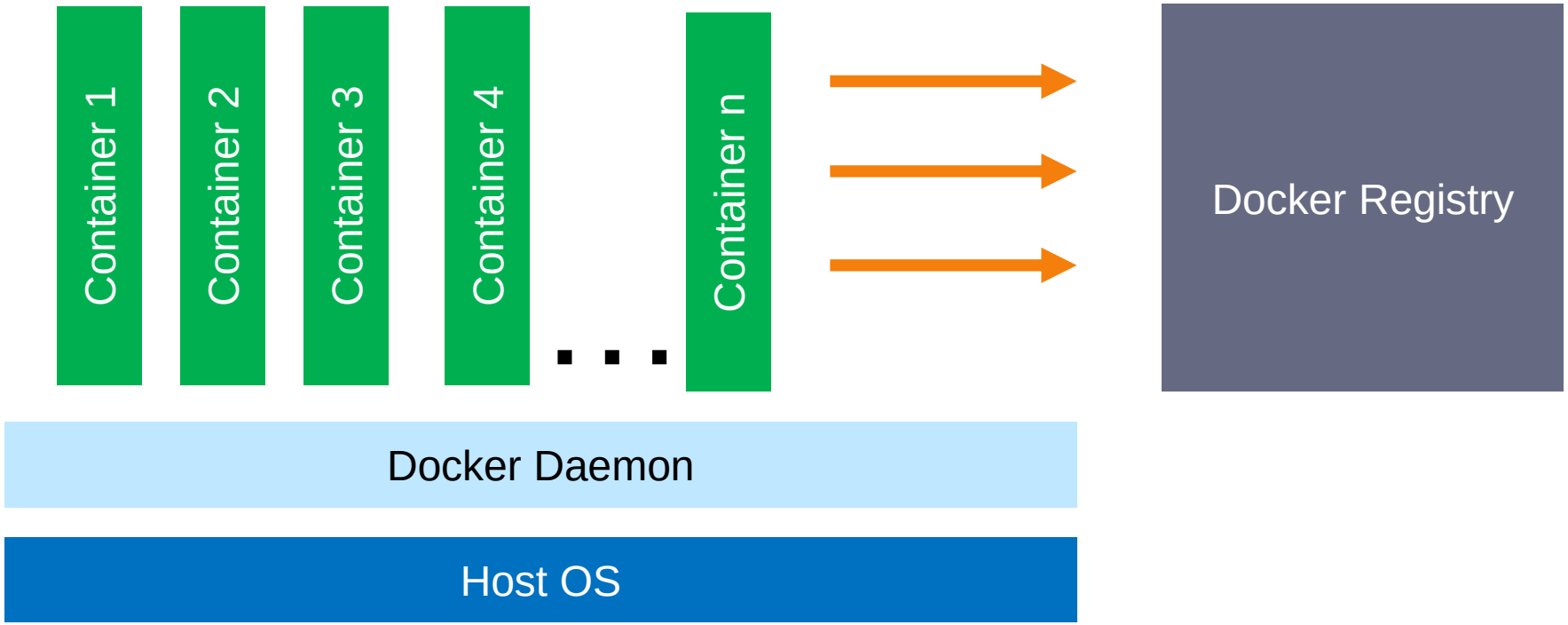
Containers

- **Container technology uses host OS features to provide an isolated environment for the applications to run on the same hardware server.**
  - Enables applications to be built as micro-services.
    - Using isolated, service-specific software containers.
  - Micro-services are largely viewed as the best way to deploy large-scale distributed application software.
  - Container engine is a crucial piece of software that manages containers.
    - Example: Docker is an open-source solution.
- **Container benefits:**
  - Applications are easier to enhance, maintain, and scale.
  - Isolated testing and fault detection.
  - Deploy enhancements to production faster and at lower costs.



Docker in Blue Planet Platform

Docker containers are self-contained software Packages.
All applications in Blue Planet are published as Docker images.
Docker images are uploaded to the Docker registry.
A Docker container is a running instance of the docker image.



Agenda

1. Blue Planet Platform Introduction
2. Blue Planet Microservices
3. **Blue Planet Platform Architecture**

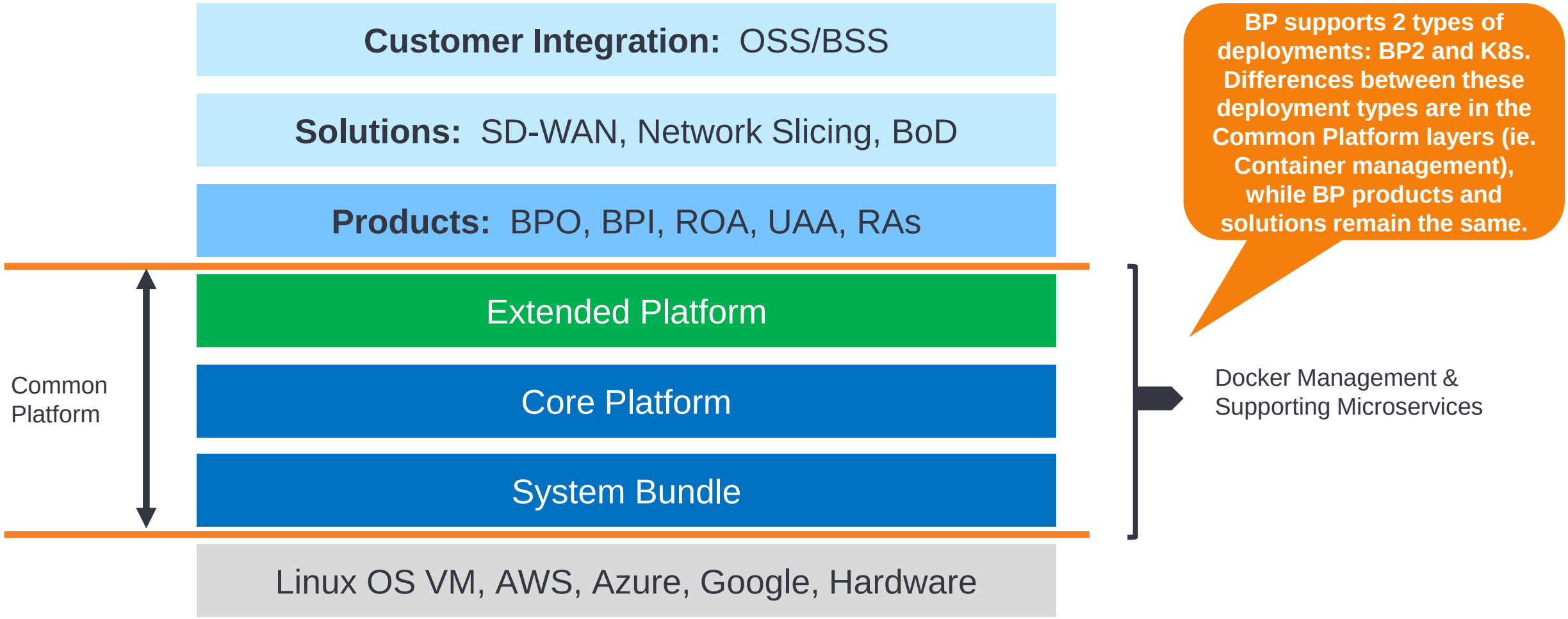
Blue Planet Platform

The Blue Planet Platform is intended to support and facilitate the following:

- **Deploying and managing containerized services (micro or otherwise).**
- **Facilitates Highly Available (HA) and Horizontally Scalable (HS) deployments and applications.**
- **Intended to support deployments with fixed size or "minimally sized”, clusters and Operations with constrained resources.**
- **Enables Monitoring, Performance and Logging collection, and visualization capabilities.**

Blue Planet Platform

Architecture



Summary

Blue Planet Platform Introduction

In this section, you learned about Blue Planet Platform Architecture.

- Blue Planet applications run as microservices that run in Docker containers.
- Applications in a microservice architecture are structured as a collection of separate services that are connected by an internal, private network.
- Blue Planet supports 2 types of deployments: BP2 and K8s. The difference between these deployments is in the Common Platform layers (in other words, Container management), while Blue Planet products and solutions remain the same.
- Blue Planet Common Platform’s intent is to facilitate the deployment and management of containerized services, along with Monitoring, Logging, and Performance visualization.

HW and SW Requirements

Objectives

- Identify types of Blue Planet deployments
- Examine installation requirements
- Show deployment requirements examples

blueplanet
a division of Ciena

HW and SW Requirements

Agenda

1. **Blue Planet SW Requirements**
2. Blue Planet Platform Sizing, Scaling, and Dimensioning

Blue Planet Platform Architectural Requirements

Blue Planet software can be deployed as a single host or a cluster configuration, using:

- Bare-metal servers.
- Virtual Machines (VMs):
  - VM performance must be comparable to bare-metal benchmarks.
  - VMs must have dedicated virtual CPUs (vCPUs), RAM, and storage.
- Cloud Native (AWS, Azure, Google).

Blue Planet supports two types of deployments: BP2 (Docker swarm) or K8s (Kubernetes) based.

Requirements also differ for the following two environments:

- **Production:** appropriate for horizontally-scaled high availability environments in enterprise business deployments and operations.
- **Lab development:** a full Blue Planet solution stack or a smaller number of representative services.
  - Typically used for development, training, testing, and proof of concept tasks.
  - Can be allowed to run with fewer hardware resources than production.

BP2 Operating System Requirements

- **BP2 requires one of the following operating systems:**
  - Red Hat Enterprise Linux (RHEL)
  - CentOS

- **During OS installation, select the Infrastructure Server option:**
  - Infrastructure Server package must be based on the baseline OS image for each release.
    - Installing an updated Infrastructure Server package is not supported.
  - Failing to choose the Infrastructure Server option will cause Blue Planet installation to fail.

- **Blue Planet apps for K8s Cloud Native deployment are agnostic to Host OS.**

Note: Blue Planet supports only Linux operating system deployments.
Refer to Product specific Installation and Engineering guides provided by Blue Planet.

HW and SW Requirements

Agenda

1. Blue Planet SW Requirements
2. **Blue Planet Platform Sizing, Scaling, and Dimensioning**

Sizing Guide

- **Server and VM sizing is dependent on the solution and products that will be installed on top of the Blue Planet Platform.**
- **Sizing and dimensioning also depend on the Deployment model type.**
- **We will show only a few examples.**

Note: For detailed sizing requirements, please contact your Blue Planet representative.

BP2 Lab Environment Example

Item	Requirement
Cluster Size	• Single-host, • 3 host, or • 6 host cluster deployments.
Server	Dell PowerEdge R430 or Equivalent
Disk Type	HDD or SDD
Disk I/O	150 IOPS
Swap Space	8 GB
Network Bandwidth	• Northbound bandwidth greater than or equal to 1 Gbps. • Machine to Machine (east-west bandwidth): 1 Gbps. • Southbound DCN to NEs bandwidth: greater than or equal to 1 Gbps.
Latency Between Cluster Nodes	<5 ms Average Latency

BP2 Production Environment Examples

Item	Requirement
Cluster Size	• Single-host • 3 Host • 6 Host Cluster Deployments
Server	• Dell PowerEdge R730 or Equivalent • Dell PowerEdge R830 or Equivalent • HP DL 560 G9 or Equivalent
Disk Type	• SSD only
Disk I/O	• 4K Random Read: 16,000 Input/output Operations Per Second (IOPS) or better. • 4K Random Write: 16,000 IOPS or better. • Sequential Read/Write: 500 MB or better.
Swap Space	• 16 GB (for hosts with up to 64 GB RAM). • 24 GB (for hosts with up to 96 GB). • 32 GB (for hosts with up to 128 GB RAM or more).

BP2 Production Environment Examples

Item	Requirement
Network Bandwidth	• Northbound bandwidth greater than or equal to 1 Gbps. • Machine to Machine (east-west bandwidth): 10 Gbps. • Southbound DCN to NEs bandwidth: greater than or equal to 1 Gbps.
Latency Between Cluster Nodes	<5 ms Average Latency

BP2 Storage Recommendations

Directory	Description	Production	Lab
/var/log	Logging from applications and services to syslog.	100 GB	100 GB
/opt/ciena/bp2	Scales based on number of inventory objects such as services, connections, and devices.	600 GB	200 GB
/opt/ciena/ (in addition to bp2 subdirectory)	Installation Files and Docker Metadata	100 GB	100 GB
/opt/ciena/data/docker	Docker Containers	200 GB	100 GB
Docker images (defaults to the unallocated disk space in the root volume group)	Docker Images	200 GB	200 GB

K8s Infra Dependencies Example

- **Variable Cluster size offers flexibility.**
- **Instance Type:**
  - Dependent on load and scale test.
  - Dependent on hosting limitations a customer may have within the environment.
- **Kubernetes Version:**
  - Minimum 1.18 and above required to install Blue Planet applications on K8s.
- **Host OS and Docker:**
  - Blue Planet apps are agnostic to Host OS and Docker version.
- **Persistent Storage Limit:**
  - Cloud providers like Google, Amazon, and Microsoft typically have a limit on how many volumes can be attached to a Node. It is important for Kubernetes to respect those limits. Otherwise, Pods scheduled on a Node could get stuck waiting for volumes to attach.

ITEM	Details
Cluster Size	At least 3-6 worker nodes (AWS). At least 6-9 worker nodes (Azure).
Instance Types AWS	m4.2xlarge (8 cpu/32GB) m4.4xlarge (16 cpu/64GB)
Instance Types Azure	Standard_D8s_v3 (8 CPUs/32GB) Standard_D16s_v3 (16 CPUs/64GB) Standard_D32s_v3 (32 CPUs/128GB)
Kubernetes Version	1.18+
Host OS	N/A -- Managed Service
Docker Version	N/A -- Managed Service
Azure Persistent Storage Limit AWS Persistent Storage Limit	16 Volume Mounts 39 Volume Mounts

Note: Sizing greatly depends on the application and use case and is not determined by the K8s platform.

Amazon EKS Worker Node sizing Example for BPO Installation

ITEM	PRODUCTION	LAB
Cluster Size	At least 3-6 worker nodes.	At least 3 worker nodes.
Instance Type	m4.4xlarge	m4.2xlarge
Worker Node vCPU	16 vCPU	8 vCPU
Worker Node RAM	64 GB	32 GB
Worker Node Disk Space	150 GB	150 GB
Persistent Storage (Estimate Usage)	~3 TB	~3 TB

Summary

HW and SW requirements

In this section, you have examined some examples of hardware and software requirements for different types of Blue Planet deployments.

- Blue Planet supports two types of deployments: BP2 (Docker swarm) or K8s (Kubernetes) based, and many deployment configurations regarding the HW and SW requirements.
- Blue Planet software can be deployed as a single host or a cluster configuration. Server requirements and cluster node/host requirements depend on the products and lineup/combo used.
- Various installation guides provided by Blue Planet are required for any strict requirements definition.

Deployment Models

Objectives

- Identify deployment models
- Describe differences between BP2 and K8s deployment types

blueplanet
a division of Ciena

Deployment Models

Agenda

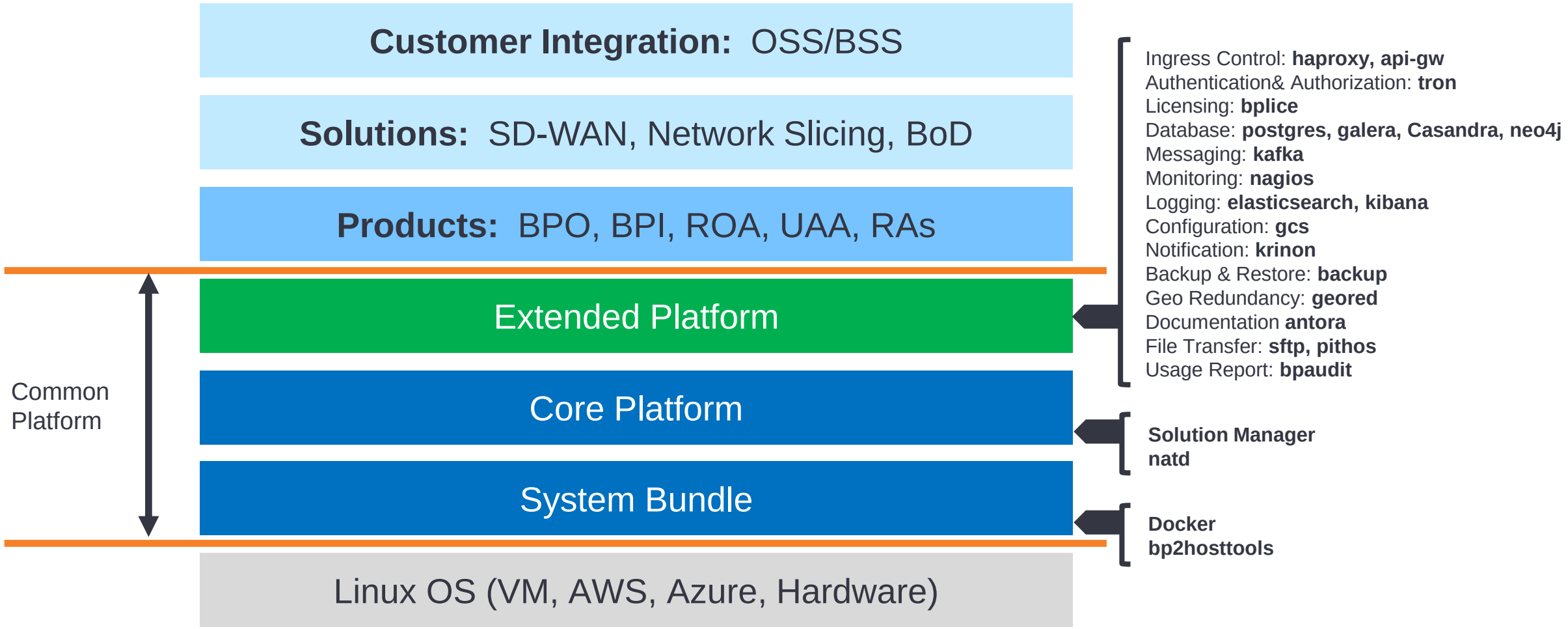
1. **Blue Planet Deployment Models – BP2 and Kubernetes**

Blue Planet Deployment Models

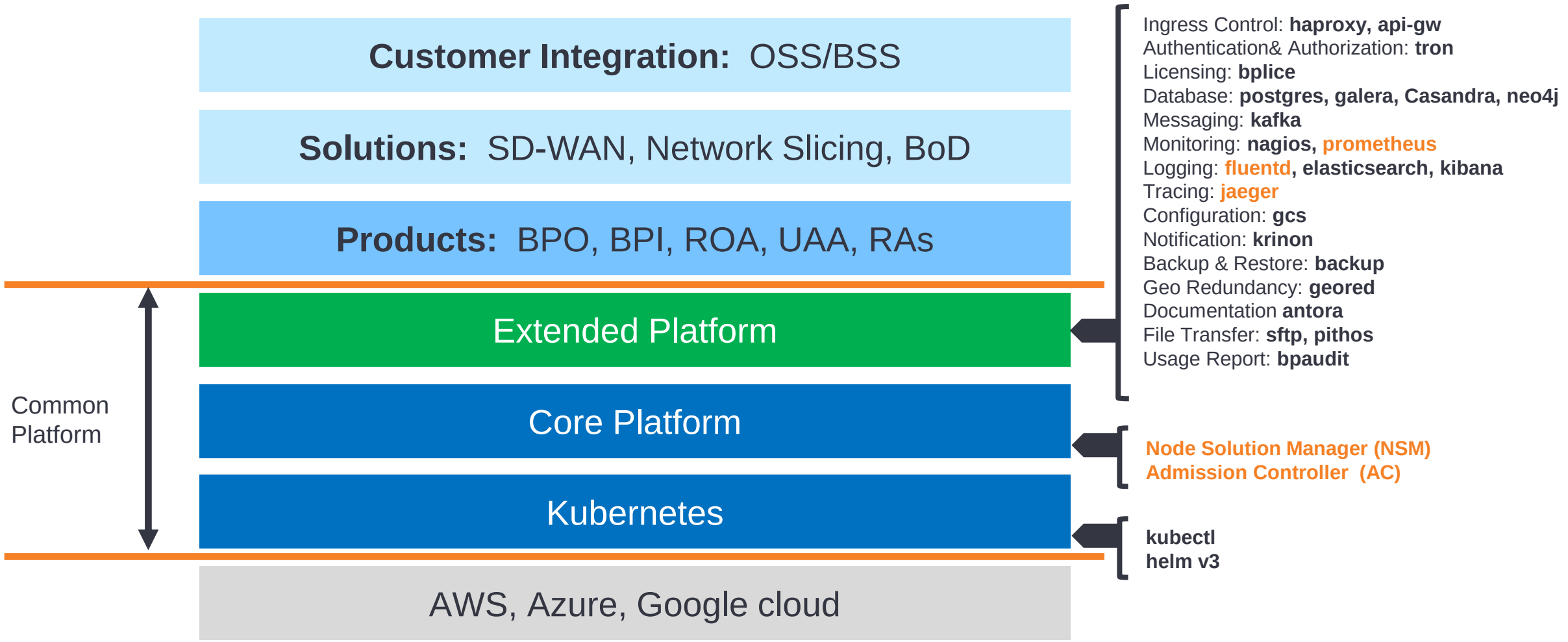
- **There are two types of supported deployment models for the Blue Planet Platform.**
  - **BP2**
    - Combination of Docker Swarm and proprietary solutions (Solman).
    - Service discovery is done via a proprietary solution.
    - Graphite for metric data collection.
  - **K8s**
    - Kubernetes-based container management solution.
    - Cloud native (AWS, Azure, Google).
    - NSM (Node Solution Manager) and AC (Admission controller)
    - Prometheus for metric data collection.

Note: These are only the basic differences between the two deployment models.

BP2 Deployment



Kubernetes Deployment



Summary

Deployment Models

In this section, you learned that there are two supported models of Blue Planet deployments.

- Common Platform can be deployed as BP2 (Docker swarm) or K8s (Kubernetes) type and the difference is in Platform applications used by each model as well as the container management solution.

Common Platform

Objectives

- Identify features of Blue Planet common platform
- Describe BP2 environment
- Examine K8s environment
- Discover Swagger and REST APIs
- Examine Solution Manager functions

blueplanet
a division of Ciena

Common Platform

Agenda

1. **Common Platform Components**
2. BP2 Environment
3. Kubernetes Environment
4. REST API
5. Solution Manager

Blue Planet Common Platform Feature Set

Common Platform

Container Orchestration & Deployment

Solution Manager

- Solution Abstraction
- Service Discovery
- Container Management (Hooks & Commands)
- App Clustering & Grouping

Bp2hosttools & natd

- Docker Engine Configuration
- Networking

BPInstaller

- Site/Host/User Configuration
- System Validation
- Solution Installation

Kubernetes

- Advanced Cluster Management
- Node Solution Manager

Access Control & Load Balancing

HAProxy

- Reverse Proxy
- Load Balancing
- ACL (Access Control List)

API-Gateway

- Authentication & Authorization for API Calls
- Caching Tokens & Permissions for Speed

Tron

- Centralized RBAC Service
- Support External Authentication with LDAP, RADIUS, etc.
- Support SAML SSO

Common Application Services

Databases

Messaging & Notification

Krinon

Configuration

GCS

SFTP & Distributed Store

Sftp

Management & Operational Tools

Monitoring

Centralized Logging

Log Archive

Geo Redundancy

Backup & Restore

Upgrade

Tracing

Documentation

Blue Planet Platform Microservices

Quick Reference

- **asset-manager: Version control software and interface to Git**
- **apigw: Used to create, publish and maintain APIs.**
- **BP-nagios: Alerting and monitoring tool used to monitor Blue Planet microservices.**
- **chronos: Scheduler that provides a means for other microservices to delay an action.**
- **datomic: Distributed database, data stored as immutable facts and good at modeling complex relationships.**
- **drools: Stateful Policy Management engine.**
- **elasticsearch: Distributed search and analytics engine.**
- **galera: SQL database, used by several microservices to store data that does not need to be scaled.**
- **fluentd: Data collector for unified logging layer.**
- **camunda: Business Process Modeling and Notation engine.**
- **kibana: Logs data Visualization plugin for Elasticsearch**

Blue Planet Platform Microservices

Quick Reference

- **heka: Stream processing s/w used for parsing log files.**
- **heroic: Time-series database/repository that consumes metrics via Kafka.**
- **kafka: Messaging bus.**
- **kibana: Data visualization plugin for elasticsearch.**
- **graphite: Contains graphite time-series database and Grafana dashboard tool.**
- **nrpe: Remotely executes Nagios plugin on other hosts.**
- **nagios: Monitoring of microservice, network, hosts and infrastructure.**
- **pm: Orchestrates collection of metrics from resources.**
- **postgresSQL: SQL database.**
- **scriptplan: App in a Python virtual environment for Remote scripts execution.**
- **solutionmanager: BP internally developed container manager.**
- **swagger-ui: Provides access to APIs for testing and modeling.**
- **tron: Centralized User Access Control system.**

Common Platform

Agenda

1. Common Platform Components
2. **BP2 Environment**
3. Kubernetes Environment
4. REST API
5. Solution Manager

BP2 Architecture

- **Container Environment**
  - Docker-based containerization.
  - Service discovery is done via a proprietary solution.
  - Blue Planet uses a combination of Docker Swarm and proprietary solutions.
  - HAProxy used for routing and load balancing.
- **Clustering**
  - N+1 clustering supported.
  - Active/active and leader-based clustering models.
  - Quorum based open-source technologies leveraged to maximize horizontal scale.
- **Communication**
  - Intra-service communication done via REST or message bus.
  - All REST APIs documented and exposed via Swagger.
  - Resource adaptor framework mediates to external models/devices/protocols.
- **Databases**
  - Two primary databases for persistent storage are PostgresSQL and Galera (mysql).
  - Elasticsearch is used as an indexing database.

Blue Planet Networking & High Availability

- **Intended to support deployments with fixed size or "minimally sized" clusters.**
- **Supports add/replace hosts and placement of containers.**
- **Stateful apps all require a quorum redundancy architecture.**
- **Stateless apps typically only require duplex redundancy.**
- **Horizonal Scalability (HS) apps may scale to many instances.**
- **GRE or vxlan tunnel-based internal network.**

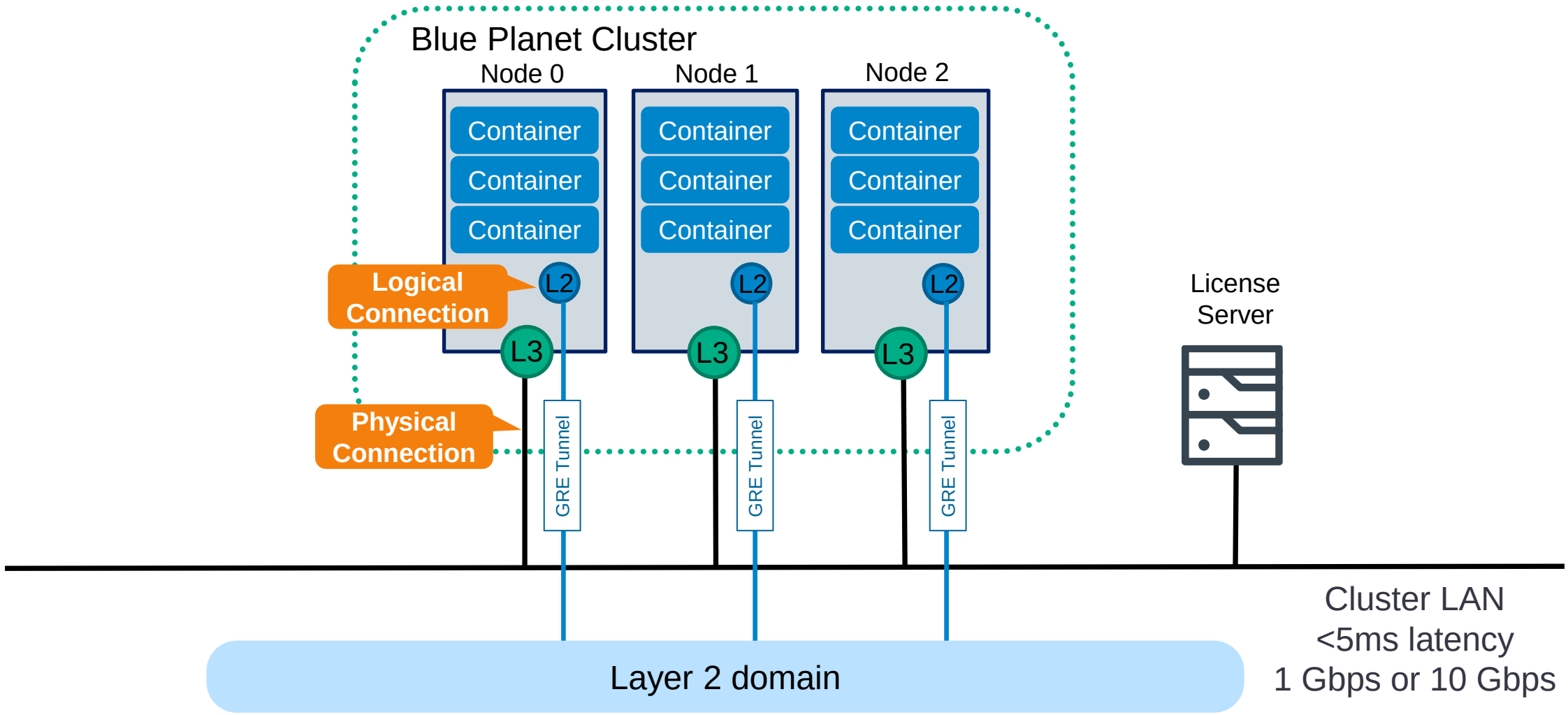
bp2hosttools

This component is installed directly on a host (Debian or rpm) and not in a container.
The bp2hosttools package is installed by the CienaBundle and its functionality is used by the bp-installer.

It facilitates:

- Site/Host Configuration.
- Docker engine configuration.
- ILAN (internal/container lan) Installation (GRE or vxlan Tunnel-based internal network).
- Registry configuration and registry-less support.
- Alternate configurations to support GoogleCloudPlatform, Azure, and AWS deployments.
- Provides helpers/utilities for debugging multi-host sites.
- Provides an installer for the Core Platform.

BP2 Three-Nodes Cluster with HA Configuration Example



Common Platform

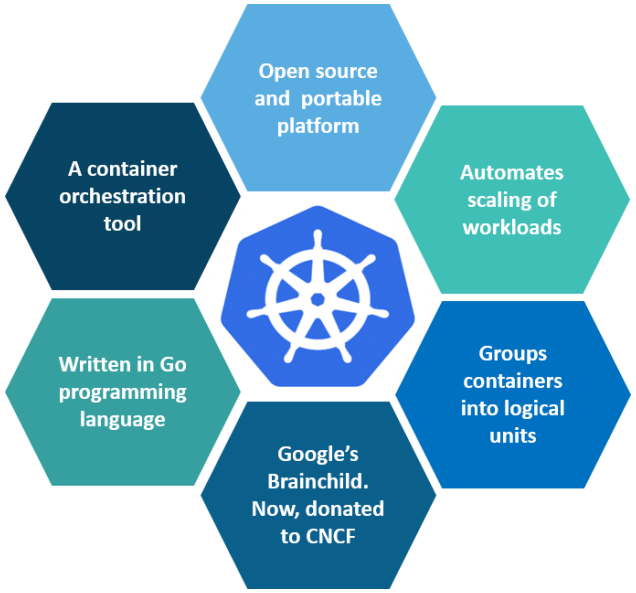
Agenda

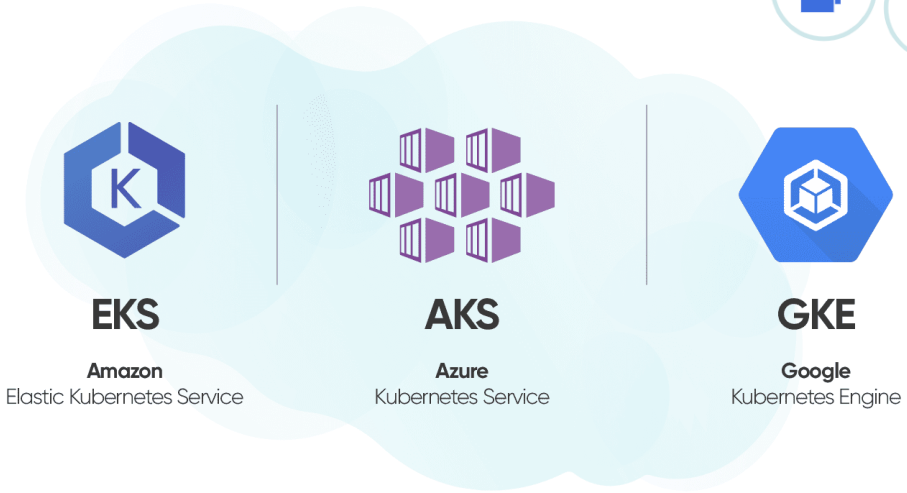
1. Common Platform Components
2. BP2 Environment
3. **Kubernetes Environment**
4. REST API
5. Solution Manager

What is Kubernetes?

Kubernetes, also known as K8s, is an open-source system for automating deployment, scaling, and management of containerized applications.

- It groups containers that make up an application into logical units for easy management and discovery. Kubernetes builds upon 15 years of experience of running production workloads at Google, combined with the best-of-breed ideas and practices from the community.





Why Kubernetes?

- **To become agnostic of the OS, Host, and Docker Dependencies.**
- **Imperative to move to the Cloud native architecture.**
- **Support for Automated Rollouts and Rollback.**
- **Support for Auto Scaling and Self Healing.**
- **Address the need to minimize hardware dependency.**
- **Minimize Installation dependencies.**

Blue Planet Deployment in Kubernetes Environment

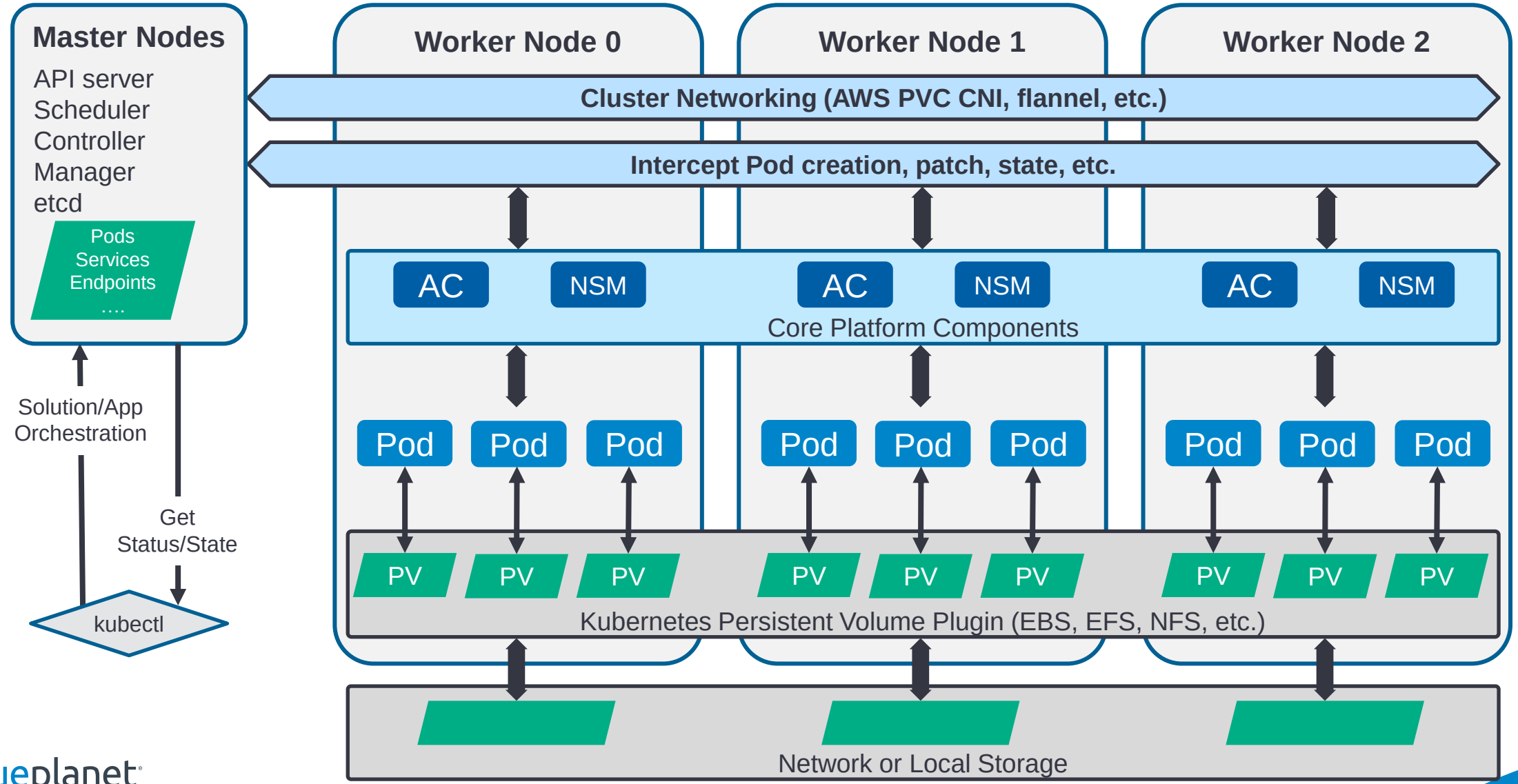
AWS EKS, Azure AKS, and GCP

AWS K8s Deployment Highlights:

- Deploying and managing Blue Planet containers using Kubernetes on AWS EKS.
- Support of the Helm package manager to deploy Blue Planet apps on Kubernetes.
- Support of the Backup/restore functionality for the Kubernetes deployment.
- Support for High Availability (HA) configuration on AWS EKS using Single or Multi-Availability zone deployment models.
- Support for in-release upgrades for Kubernetes-based deployments.
- Software delivery through Blue Planet online registry (bphub).

Blue Planet Three Node Cluster Deployment in AWS Kubernetes Environment

Architecture



blueplanet
a division of Ciena

NSM and AC

NSM (Node Solution Manager)

- Blue Planet specific container, deployed as daemonset.
- It provides a subset of the Solution Manager functionality to enable Blue Planet containers to run on K8s and still be able to communicate to each other through HTTP hooks to be backward compatible.
- Embeds the core platform to help deploy and communicate between the applications.
- Manages the apps within the Blue Planet environment.

AC (Admission Controller)

- Blue Planet specific container, deployed as daemonset.
- It intercepts K8s commands and adds additional information for Blue Planet containers.
- It may only be configured by the cluster administrator.
- Admission controllers limit requests to create, delete, modify objects, or connect to the proxy.
- Many advanced features in Kubernetes require an admission controller to be enabled in order to properly support features.

PersistentVolume

PersistentVolume (PV) is a piece of storage in the cluster that has been provisioned by an administrator or dynamically provisioned using Storage Classes.

- The Persistent Volume subsystem provides an API for users and administrators that abstracts details of how storage is provided from how it is consumed.
- It is a resource in the cluster just like a node is a cluster resource.
- PVs are volume plugins like Volumes but have a lifecycle independent of any individual Pod that uses the PV.

PersistentVolumeClaim (PVC) is a request for storage by a user.

- It is similar to a Pod. Pods consume node resources and PVCs consume PV resources.

StatefulSet

Blue Planet Kubernetes Common Platform utilizes StatefulSet, which is the workload API object used to manage stateful applications that require one or more of the following:

- Stable, unique network identifiers.
- Stable, persistent storage.
- Ordered, graceful deployment and scaling.
- Ordered, automated rolling updates.

StatefulSet manages the deployment and scaling of a set of Pods and provides guarantees about the ordering and uniqueness of these Pods.

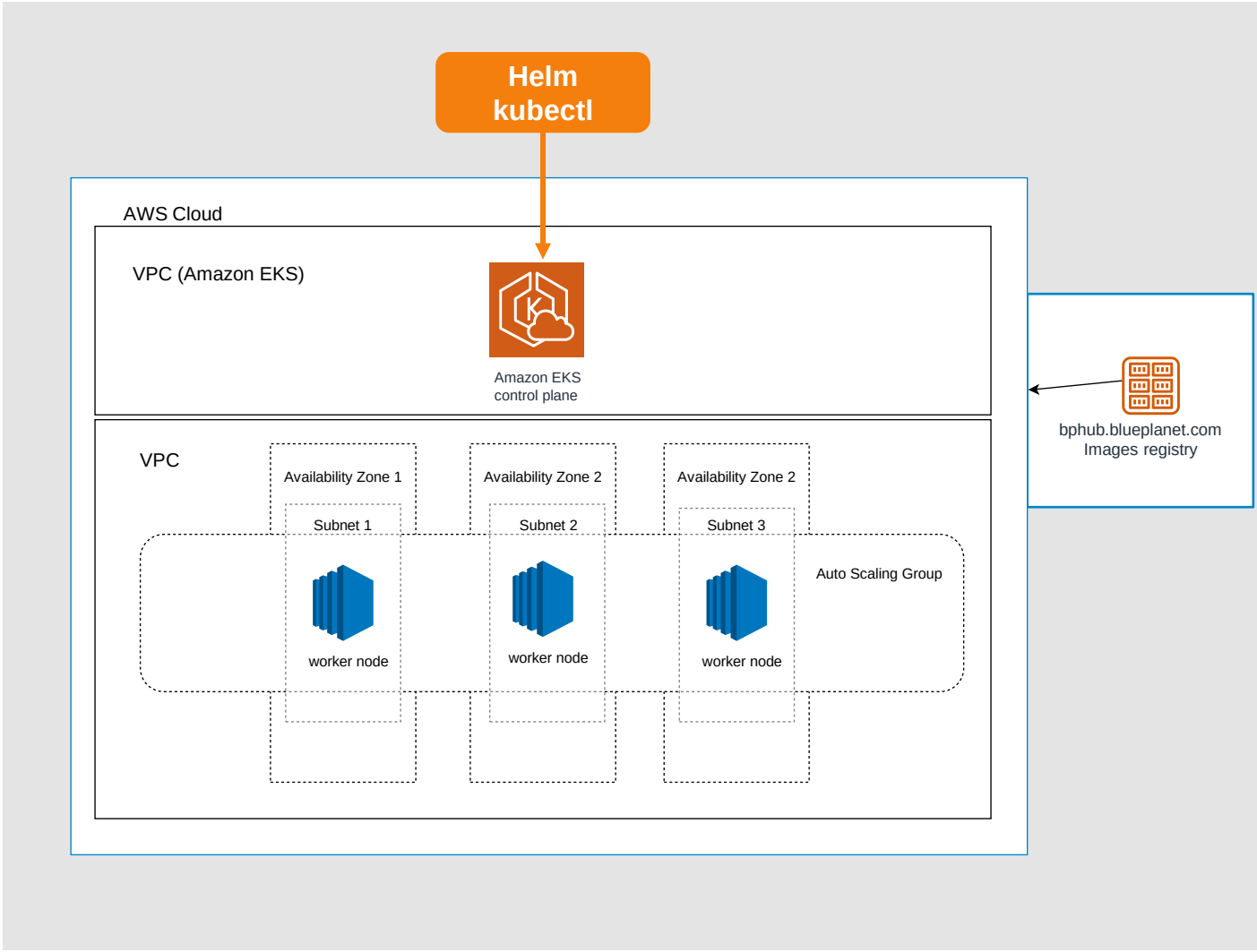
- Like a Deployment, a StatefulSet manages Pods that are based on an identical container spec.
- Unlike a Deployment, a StatefulSet maintains a sticky identity for each of their Pods.

AWS EKS Reference Architecture

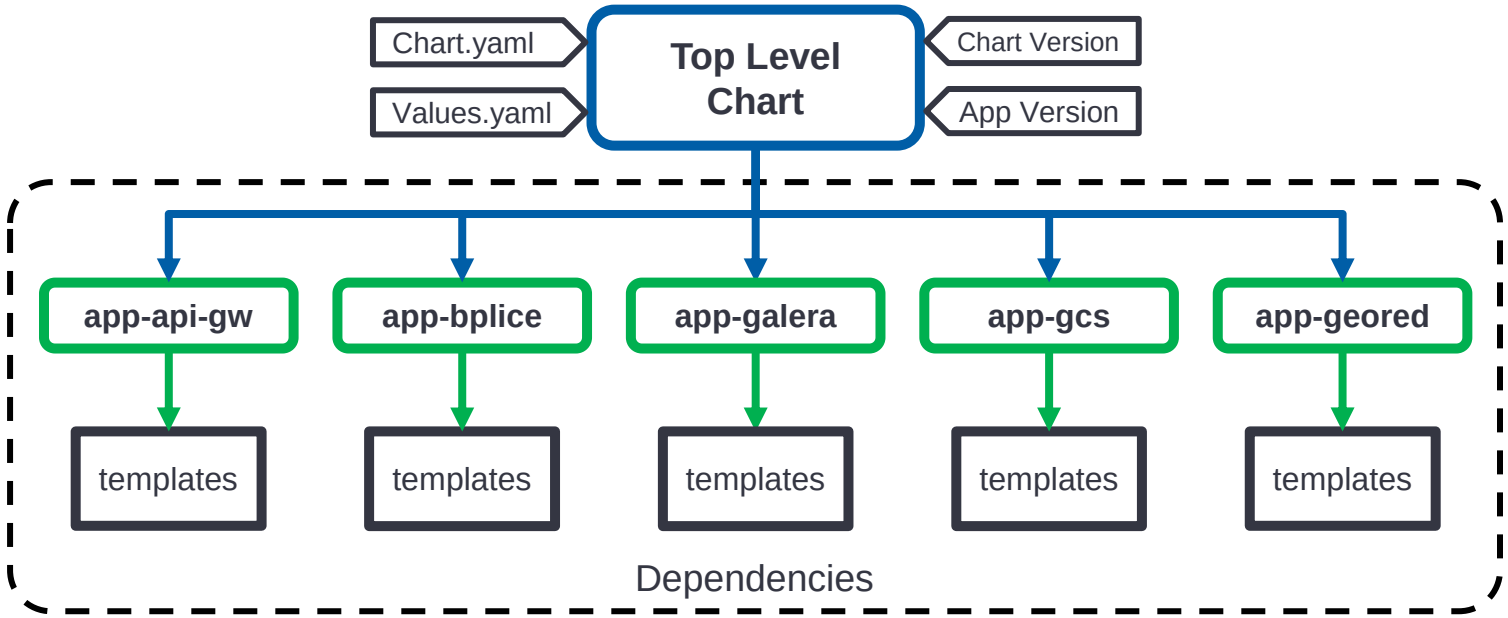
Blue Planet provides:

- Worker node sizing requirements.
- All components are included and deployed as microservices.
- Helm Charts and Installation scripts for Deployment.

- **Amazon Kubernetes EKS Service is used.**
- **Worker nodes are in different Availability Zones with separate Node Groups managed by their respective Auto Scaling Group (ASG).**
- **If worker nodes go down, ASG will launch the new worker node automatically with the help of Cluster Autoscaler running in the cluster.**



Helm Packaging

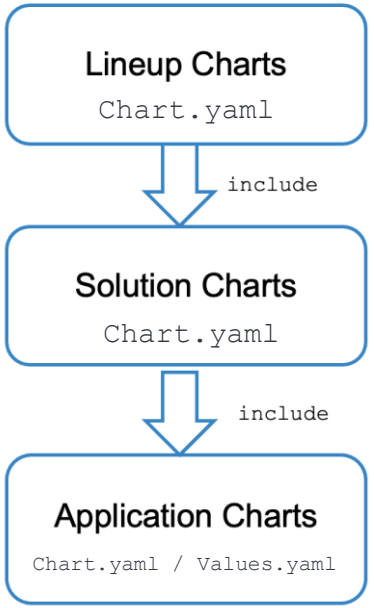


The Helm package manager for Kubernetes helps to install and manage applications on your Kubernetes cluster. Applications are deployed using helm chart lineups.

Helm Top Level chart hierarchy and dependencies

- The helm3 client directly interacts with Kubernetes API to deploy the charts.
- The Kubernetes YAML files are imported into the **templates** on each specific component.
- **Chart.yaml** contains the **Helm chart Version** and **Application Version**.
- The **Values.yaml** is used to override the values of all of the chart's component and sub-chart component values.

- **Helm Hierarchy as defined within Blue Planet.**



blueplanet
a division of Ciena

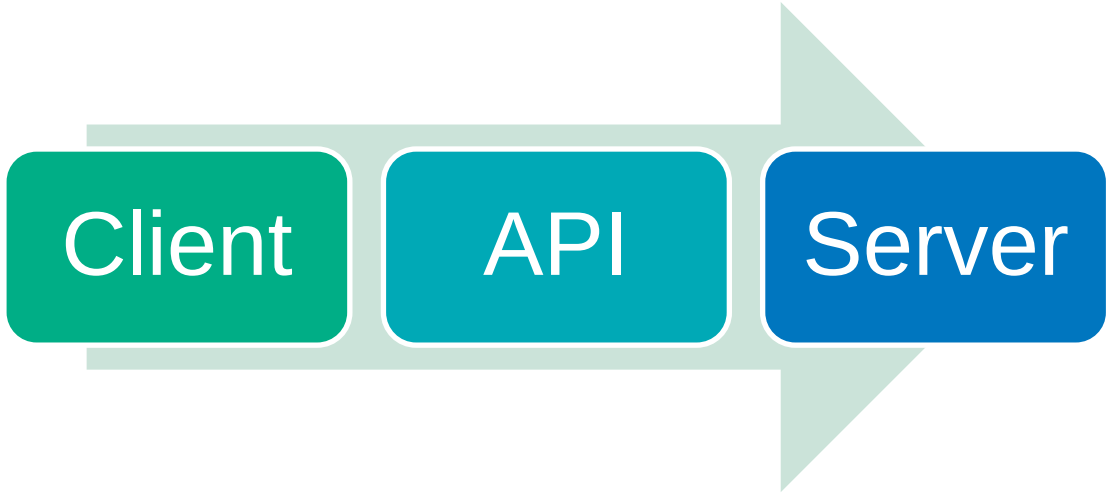
Common Platform

Agenda

1. Common Platform Components
2. BP2 Environment
3. Kubernetes Environment
4. **REST API**
5. Solution Manager

What is an API?

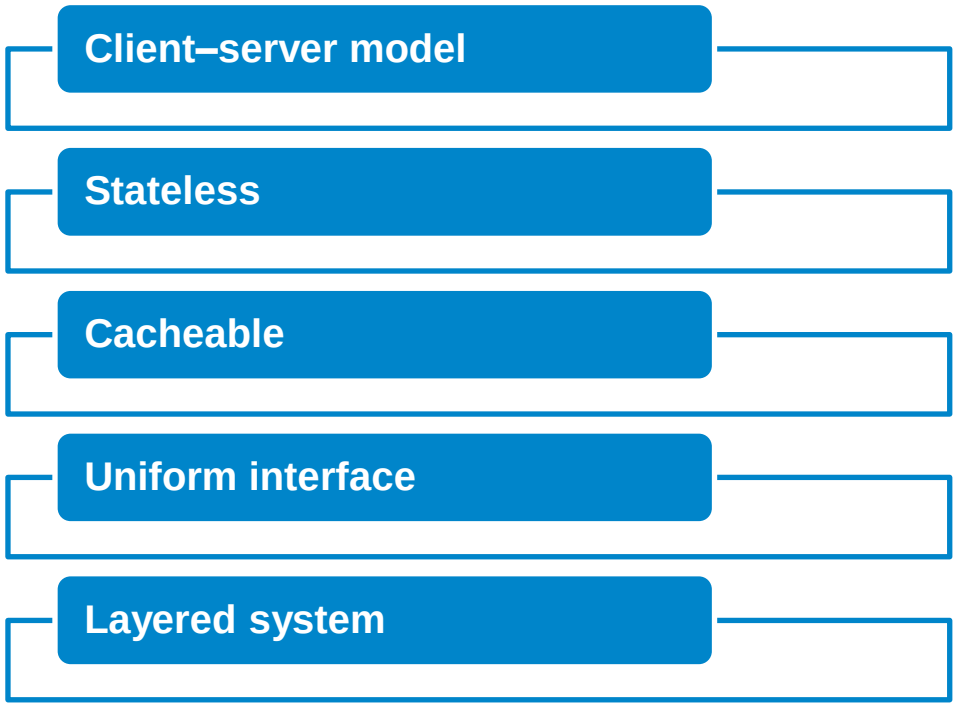
- **Microservices applications use APIs for internal and external communications.**
- **API = Application Programming Interface**
  - A software component that acts as an intermediary between a client and server:



- **The client does not need to know how the process is performed on the server.**
- **The client only needs to understand how to send the request via the API.**
- **This creates a standard interface, even if the server processes change.**
- **Common API: REST**

What is REST?

- **Stands for REpresentational State Transfer.**
- **Architectural style with six constraints:**



The operations that the REST server performs are not known to the REST client. This means that changes to these operations should not affect REST calls to the server.

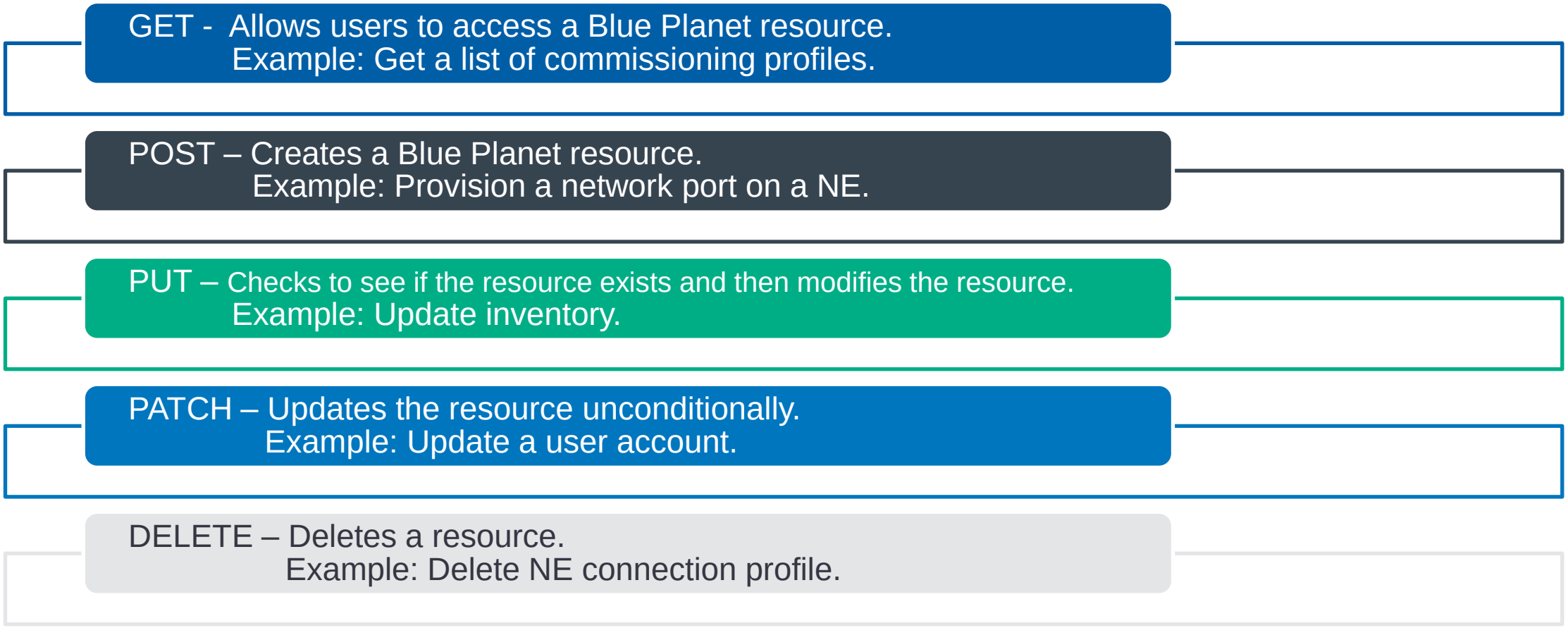
CRUD

- **REST Client calls are often associated with HTTP requests.**
- **While any HTTP request could be used for REST calls, the most common calls fall within the following categories:**
  - **C**reate – An HTTP **POST** Request
  - **R**etrieve/Read – An HTTP **GET** Request
  - **U**pdate – An HTTP **PUT** Request
  - **D**elete - An HTTP **DELETE** Request

- **As these are the four most common forms of REST Client calls, documentation often refers to the term CRUD when describing common REST calls.**

Permission Operation Types

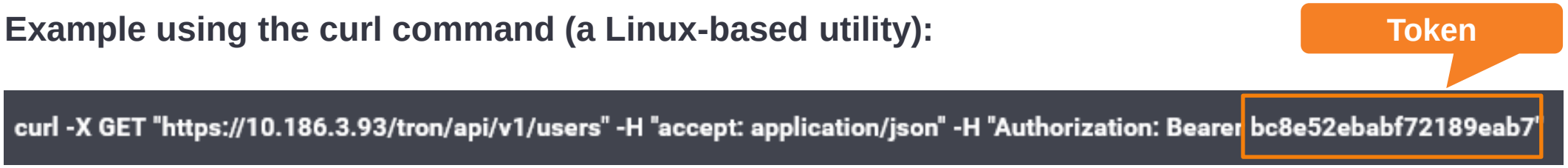
- **Permissions are grouped into five primary operation categories.**
- **Knowing the purpose of these categories can make it easier to find specific permission:**



API Tokens

- **In a production environment, API calls are also made from utilities outside of Blue Planet, including:**

- **These tools must provide authentication information to successfully run API calls.**
- **An API token can be generated to authenticate the API call.**
- **Example using the curl command (a Linux-based utility):**

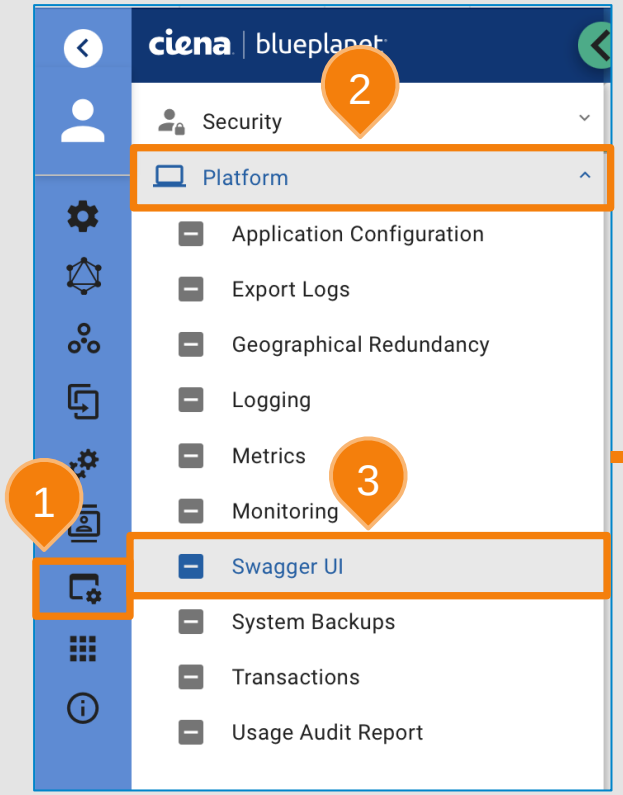


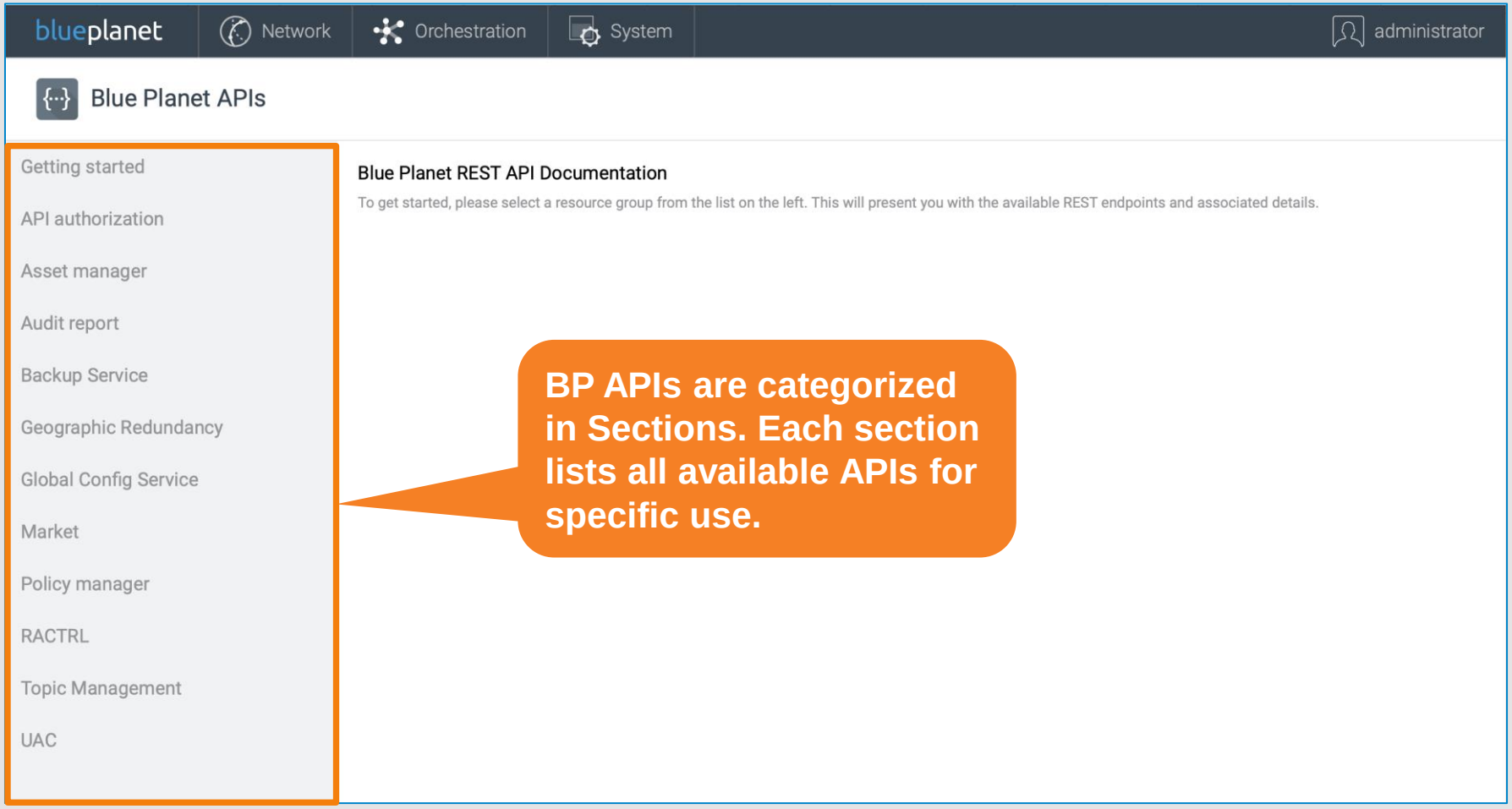
blueplanet
a division of Ciena

Swagger

- **Swagger is a server-side tool that can perform REST API calls on the server.**
- **Swagger is a tool to demonstrate and develop API calls.**
- **It is integrated within a server application and customized for that application.**
- **Swagger also provides a means of viewing available API details.**
- **Normally an API call would need to include some sort of authentication method.**
- **Because Swagger is accessed after logging into Blue Planet, authentication has already taken place.**
- **Swagger UI is accessed on a Blue Planet server by Clicking: System > Platform > Swagger UI.**

Swagger UI





Common Platform

Agenda

1. Common Platform Components
2. BP2 Environment
3. Kubernetes Environment
4. REST API
5. **Solution Manager**

Solution Manager

- **Solution is a set of applications that are deployed and managed as a single unit.**
- **Solution Manager facilitates the life-cycle collection of related microservices as a single solution.**
- **Part of the BP2 type deployments.**
- **Configured to run with a Docker registry.**
- **Provides an HTTP API and the "solman" CLI for user interaction.**
- **Provides service discovery to allow various microservices to work together.**
  - Connects service providers and consumers.
  - Uses the following abstractions:
    - NorthBound Interfaces (NBIs)
    - SouthBound Interfaces (SBIs)
  - App Clustering
- **Manages the solution life-cycle (deploy, un-deploy, upgrade, backup, restore, and so on).**
- **Provides system services for service discovery and clustering.**
- **Uses a "hook" based interface for communications to apps.**

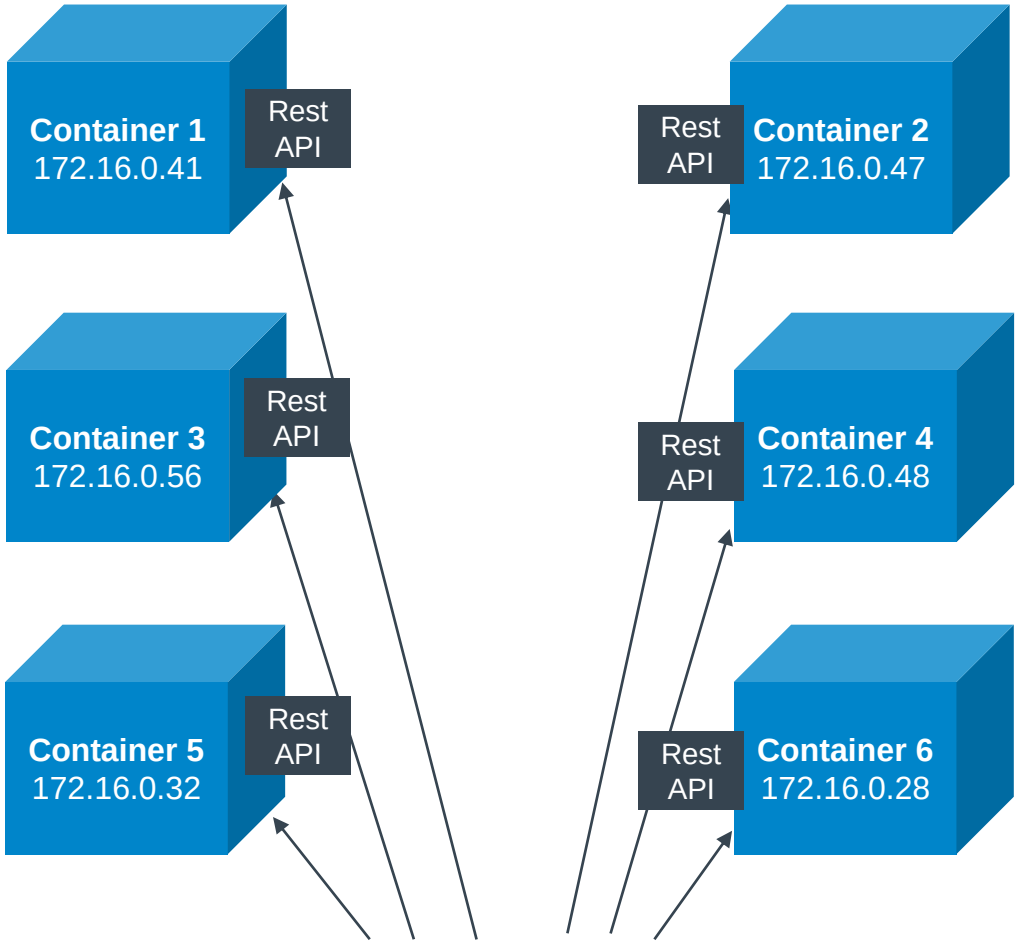
Solutions: Solution Data Image

Solution Metadata

Solution Apps

```
__version__: 1
docker_registry: dockerreg.cyanoptics.com
solution_name: logging
solution_version: 3.0.4
upgrade_order:
    - elasticsearch
    - heka
    - kibana
apps:
    elasticsearch:
        image: dockerreg.cyanoptics.com/cyan/elasticsearch:2.1.5-es2.2.1
        volumes:
            - /etc/hostname:/etc/physical_hostname:ro
        environment:
            - NBI_elasticsearch_checkauth=true
            - NBI_elasticsearch_checkrbac=true
    kibana:
        image: dockerreg.cyanoptics.com/cyan/kibana:2.2.8-4.4.2
        environment:
            - NBI_bplogging_checkauth=true
            - NBI_bplogging_checkrbac=true
    heka:
        image: dockerreg.cyanoptics.com/cyan/heka:2.4.1-h0.10.0
        volumes:
          - /var/log:/var/log
          - /etc/hostname:/etc/physical_hostname:ro
```

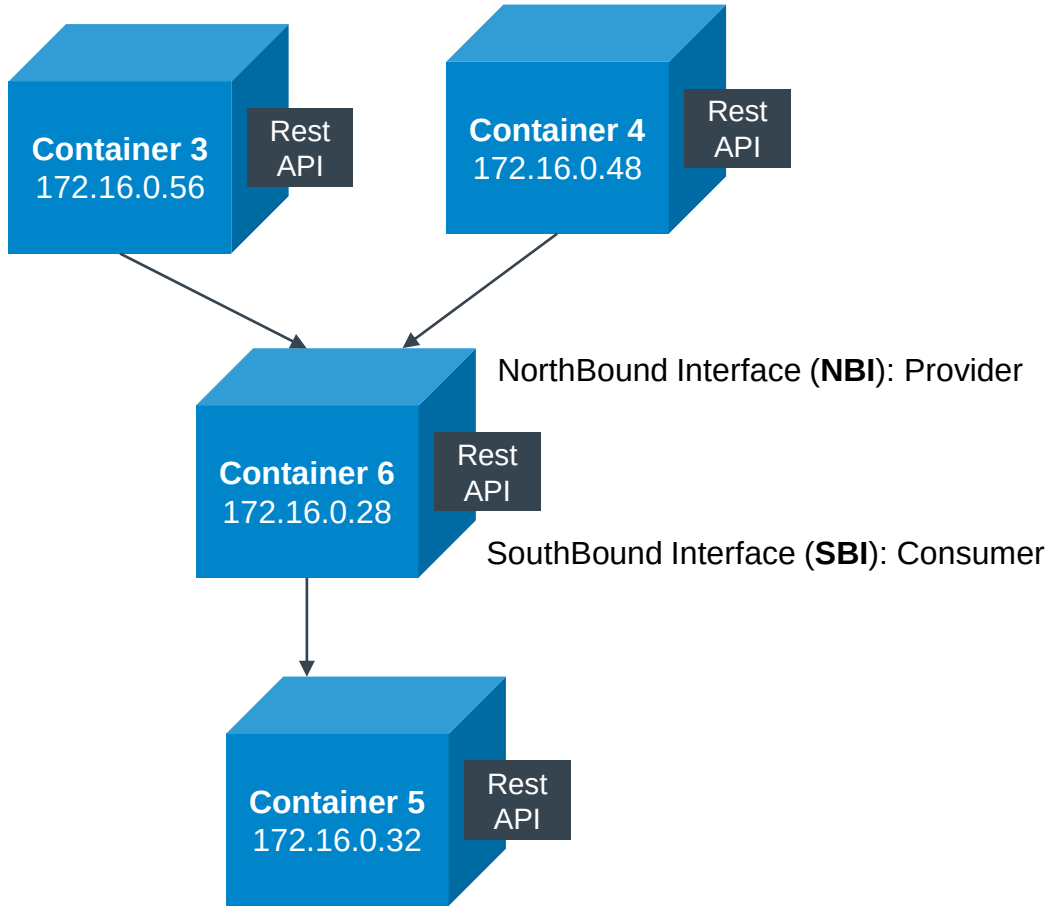
Service Discovery



Container	1	2	3	4	5	6
IPAddress	172.16.0.41	172.16.0.47	172.16.0.56	172.16.0.48	172.16.0.32	172.16.0.28
Capabilities	{ }	{ }	{ }	{ }	{ }	{ }
Dependencies	{ }	{ }	{ }	{ }	{ }	{ }

blueplanet
a division of Ciena

Service Discovery: NBI / SBI



Container	1	2	3	4	5	6
IPAddress	172.16.0.41	172.16.0.47	172.16.0.56	172.16.0.48	172.16.0.32	172.16.0.28
Capabilities	{ }	{ }	{ }	{ }	{ }	{ }
Dependencies	{ }	{ }	{ }	{ }	{ }	{ }

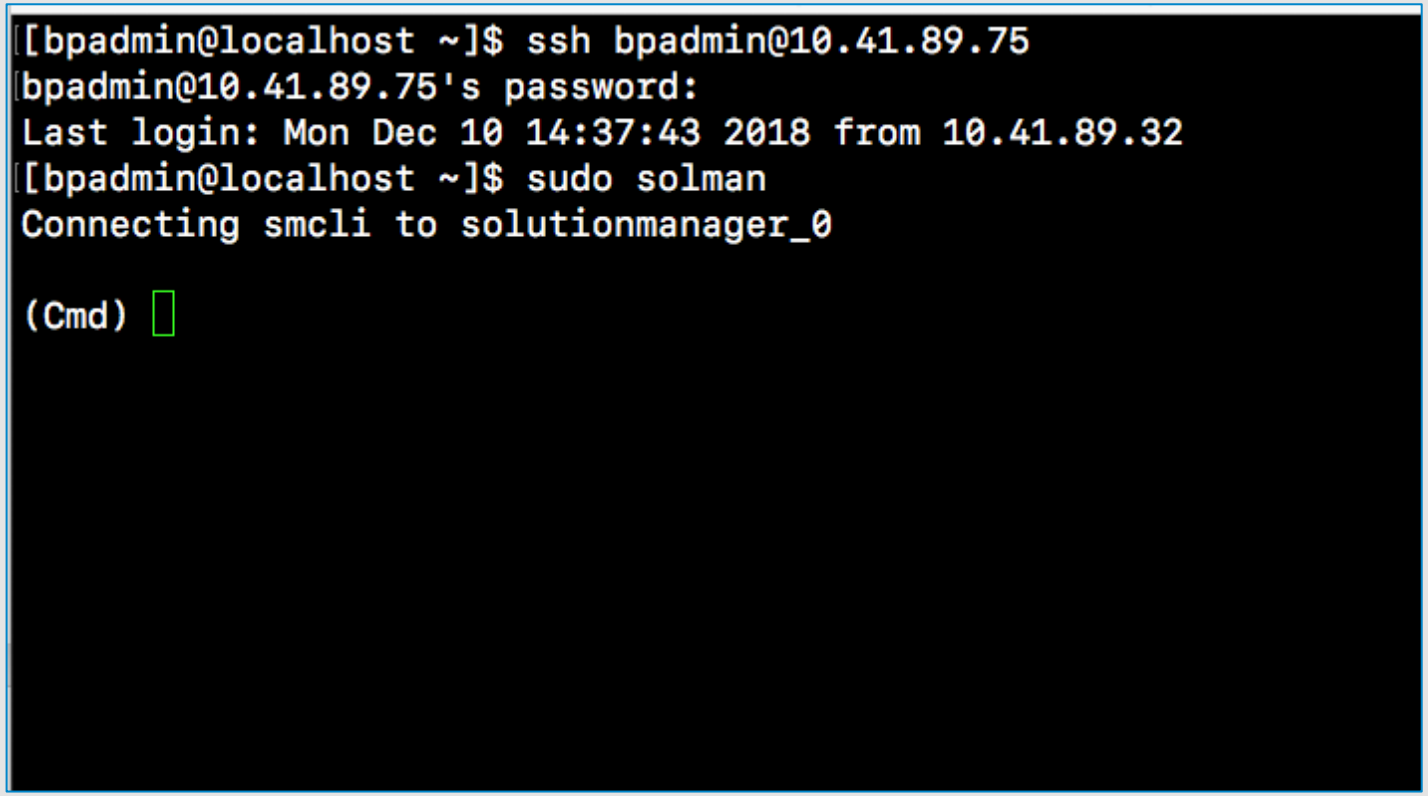
blueplanet
a division of Ciena

Access Solution Manager CLI

- **ssh into the server:**
  ssh bpadmin@<Host IP>

  <Host IP> is a placeholder for your specific Blue Planet instance.

- **Solman to Run Solution Manager:**
  sudo solman



Solution Manager – the Solman Utility

Show available commands:

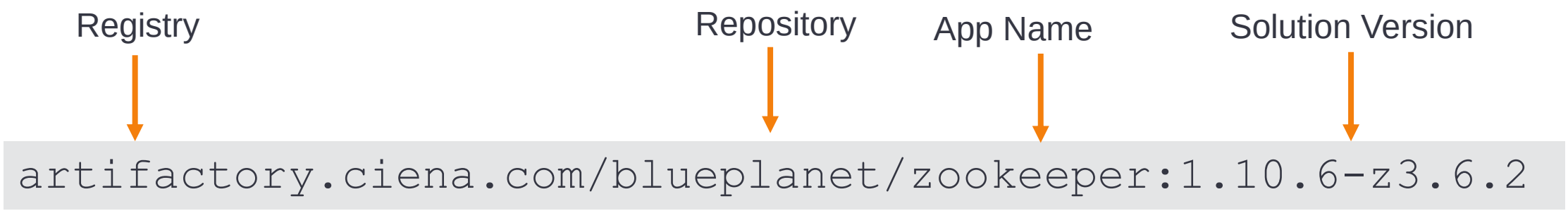
```
(cmd) help
```

Provides list of deployed solutions:

```
(cmd) sps
```

View specific status of specific container:

```
(cmd) sps | grep <container name>
```



Summary

Common Platform

In this section, you learned about Common Platform components in two different environment types.

- BP2 is intended to support deployments with fixed size or "minimally sized" clusters.
- K8s imperative is to move to the Cloud native architecture and support Automated Rollouts, Rollbacks, Auto Scaling, and Self Healing.
- Swagger is a server-side tool that can perform REST API calls on the server. It is used to demonstrate and develop API calls.
- Solution Manager is a part of BP2 deployment model types and facilitates the life-cycle collection (deploy, un-deploy, upgrade, backup, restore, and so on) of related microservices as a single solution.
- Solution Manager Provides an HTTP API and the "solman" CLI for user interaction.

Infrastructure Apps and Extended Platform

Objectives

- Identify and describe the core components of the extended Platform
- Explore system monitoring options
- Examine Performance Metrics Collection and Visualization
- Describe the logging process in Blue Planet

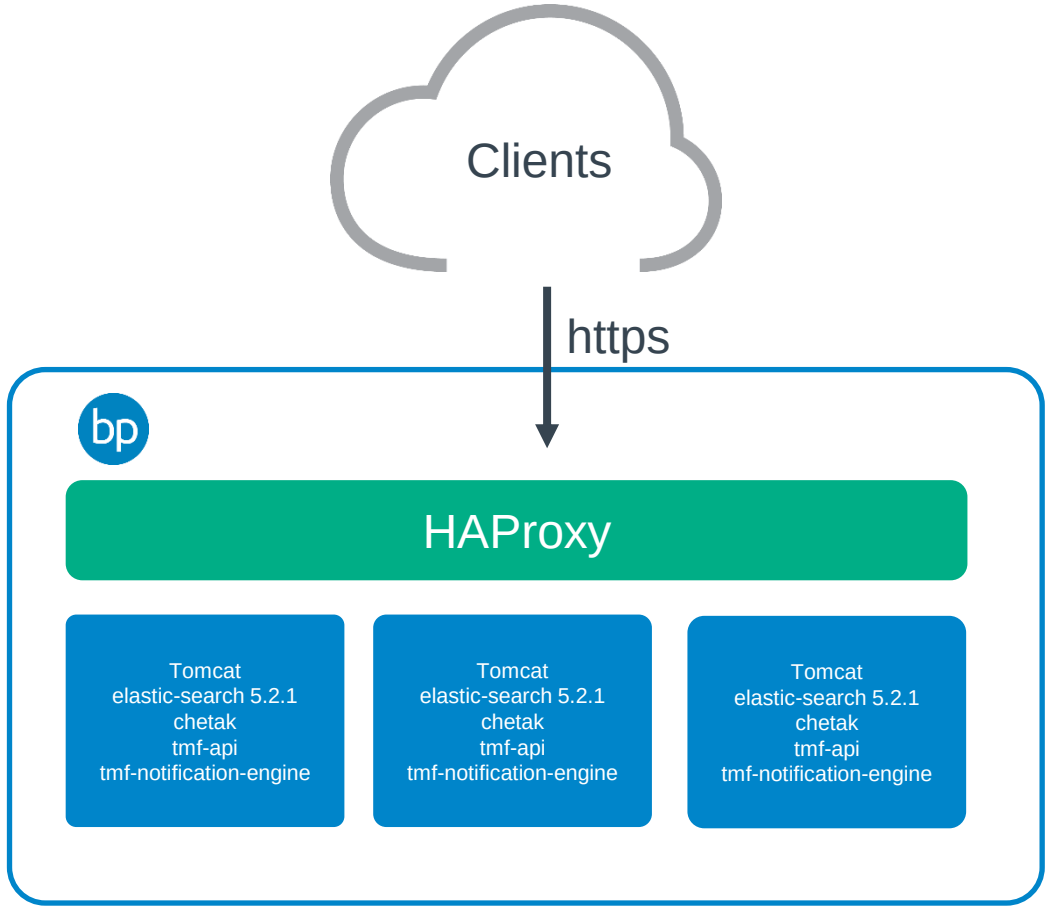
Infrastructure Apps and Extended Platform

Agenda

1. **Core Components**
2. System Monitoring
3. Performance Metrics
4. Logging

HAProxy Overview

- **HAProxy is one of the most widely used software load balancers and application delivery controllers.**
  - Built for Speed
  - Feature Rich
  - Open Source
- **Blue Planet Platform uses HAProxy to provide high availability and load balancing for Blue Planet applications.**
- **HAProxy exists between the clients and the services.**
  - Handles and directs HTTPS requests.
  - SSL Certificates are important in HTTPS communication.

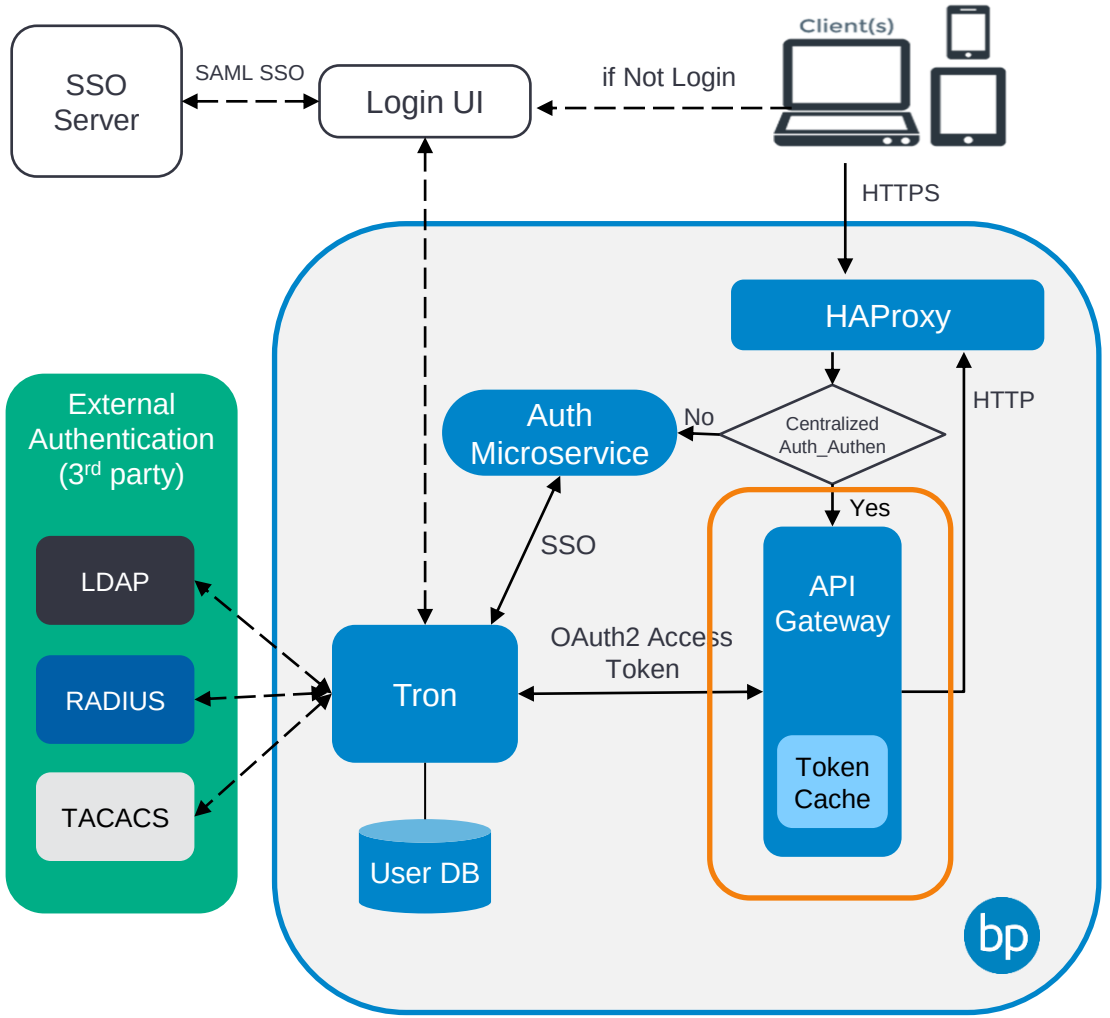


HAPROXY

API Gateway

- **API-GW is a separate microservice in the Blue Planet solution and is primarily responsible for AAA (authentication, authorization, and accounting) for any API call to any microservice that requires it.**
  - Authentication involves the api-gw to check if the user is known or not.
  - Authorization is to confirm if a successfully authenticated user is authorized to make the API call or not.

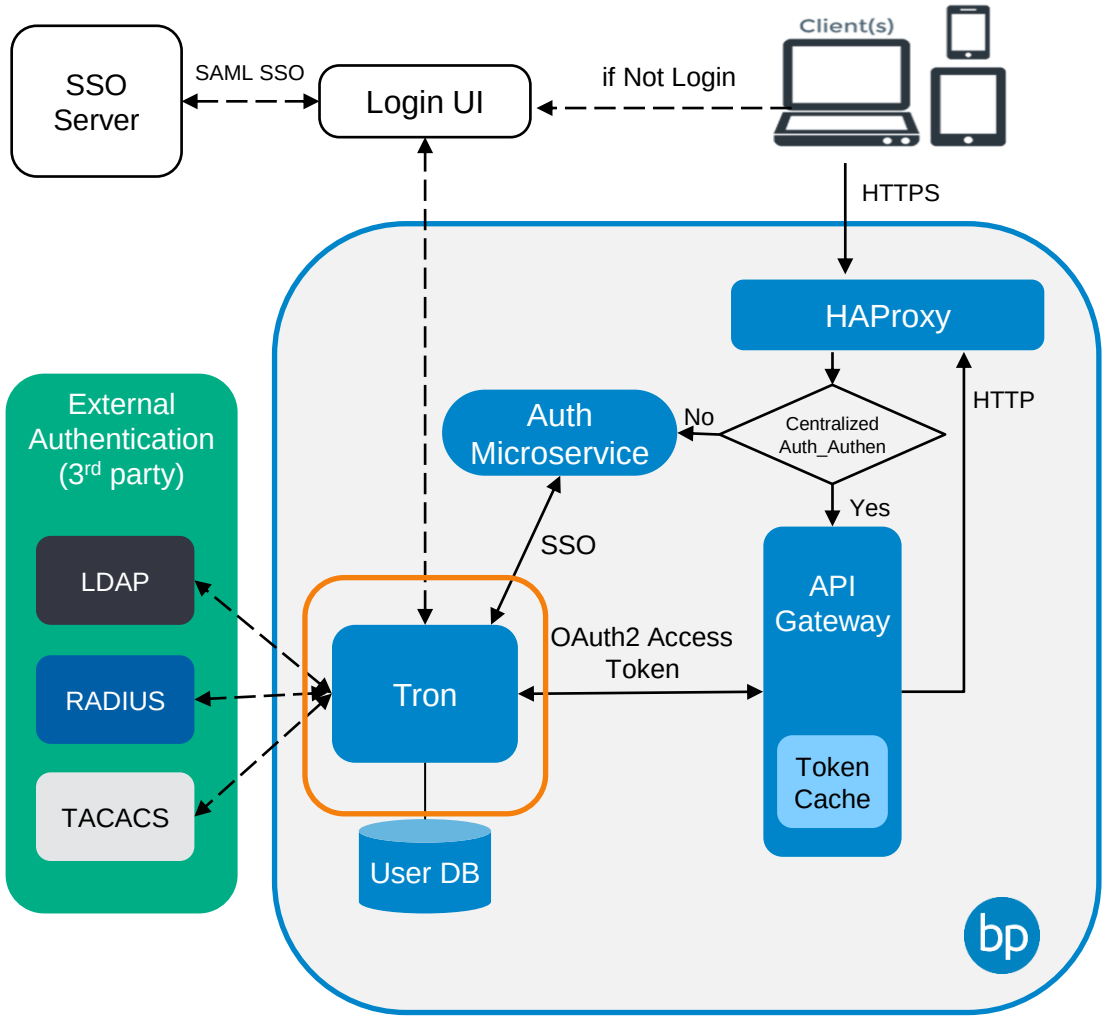
- **As part of the validation, it first verifies the license and returns with failure details in case it is not valid.**
- **It retrieves information about the user and its roles that require querying the Tron.**



Tron

Tron is the centralized User Access Control service for Blue Planet.

- It supports multiple external authentication systems:
  - LDAP
    - LDAP and LDAPS are supported.
    - RADIUS
  - TACACS
- Tron authenticates against the configured external authentication systems in order until either there is a successful authentication, or all methods have failed.



Apache Kafka

- **Apache Kafka is an open-source framework implementation of a software bus using stream-processing:**
  - The project aims to provide a unified, high-throughput, low-latency platform for handling real-time data feeds.
  - Based on the commit log, allows users to subscribe to it and publish data to any number of systems or real-time applications.
- **Kafka architecture involves:**
  - Storing key-value messages that come from producers in topics and partitions.
  - Reading or consuming those stored messages by consumer processes.
- **Blue Planet and Kafka:**
  - There are several Blue Planet apps producing and consuming messages (a.k.a. events) via a shared Apache Kafka cluster.
  - Each app includes a bit of metadata useful for versioning, tracing, debugging, and so on.
  - Kafka brokers are deployed as containers.
  - Kafka runs on the well-known port 9092.

Apache ZooKeeper

- **ZooKeeper is a centralized service for maintaining configuration information, naming, providing distributed synchronization, and providing group services.**
- **ZooKeeper is primarily used to track the status of nodes in the Kafka cluster and maintain a list of Kafka topics and messages.**
- **Even if all the nodes in a site go down, ZooKeeper persists the data. And while the node is coming up it fetches the data from ZooKeeper in case of restart.**
- **ZooKeeper is also leveraged for Geo-redundancy in BP2 type deployments.**

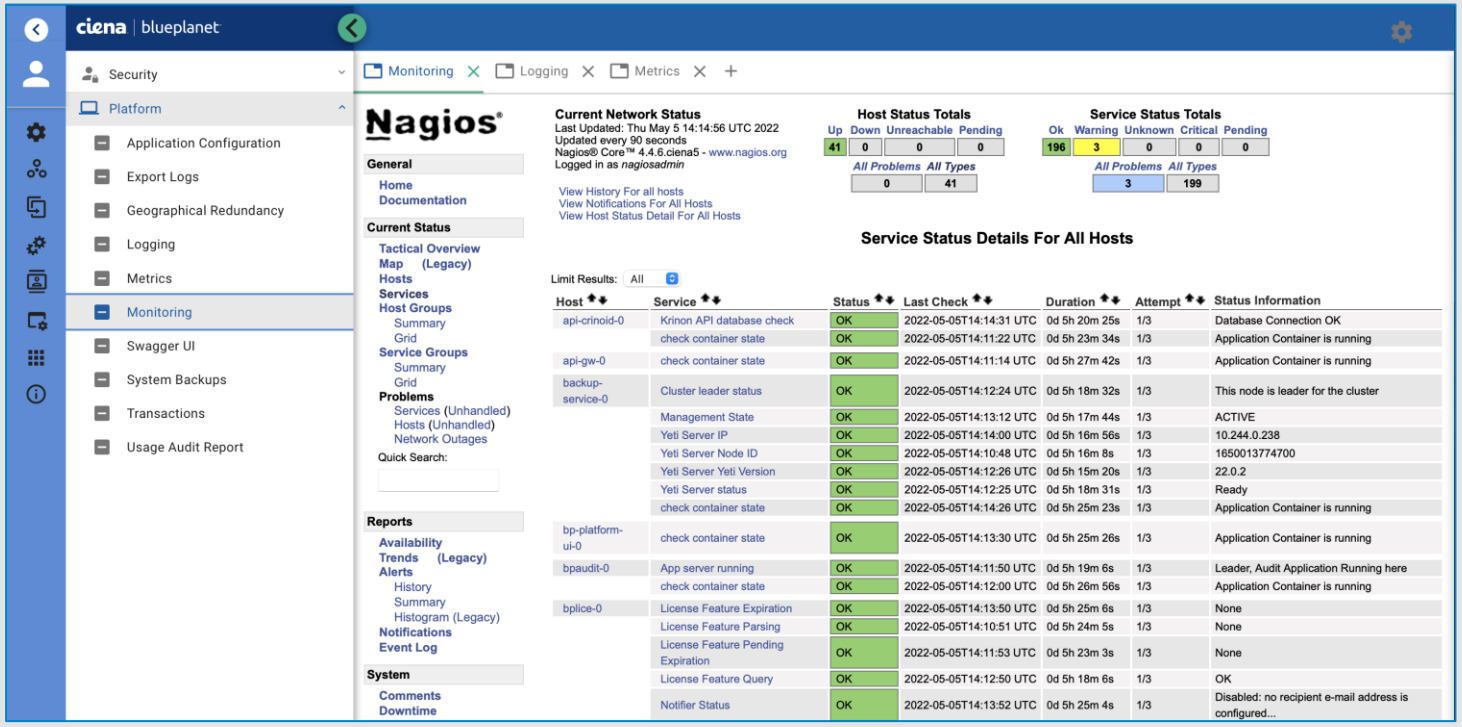
Infrastructure Apps and Extended Platform

Agenda

1. Core Components
2. **System Monitoring**
3. Performance Metrics
4. Logging

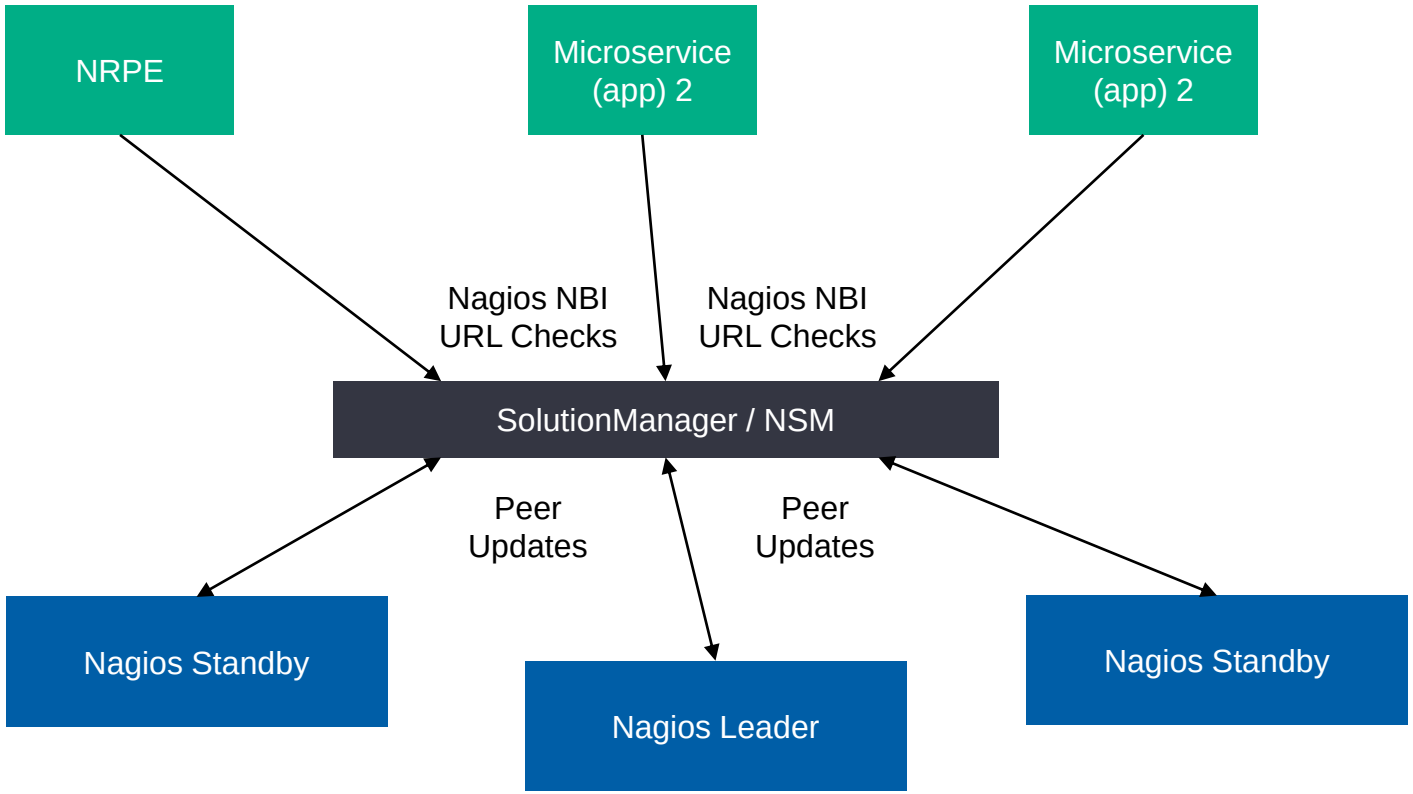
Nagios

- **Monitors Blue Planet Microservices.**
- **Checks are automatically created when solutions are deployed.**
- **Monitors the availability of each microservice.**
- **Provides multiple visual representations and reports.**
- **Accessed from System > Platform > Monitoring menu.**



blueplanet
a division of Ciena

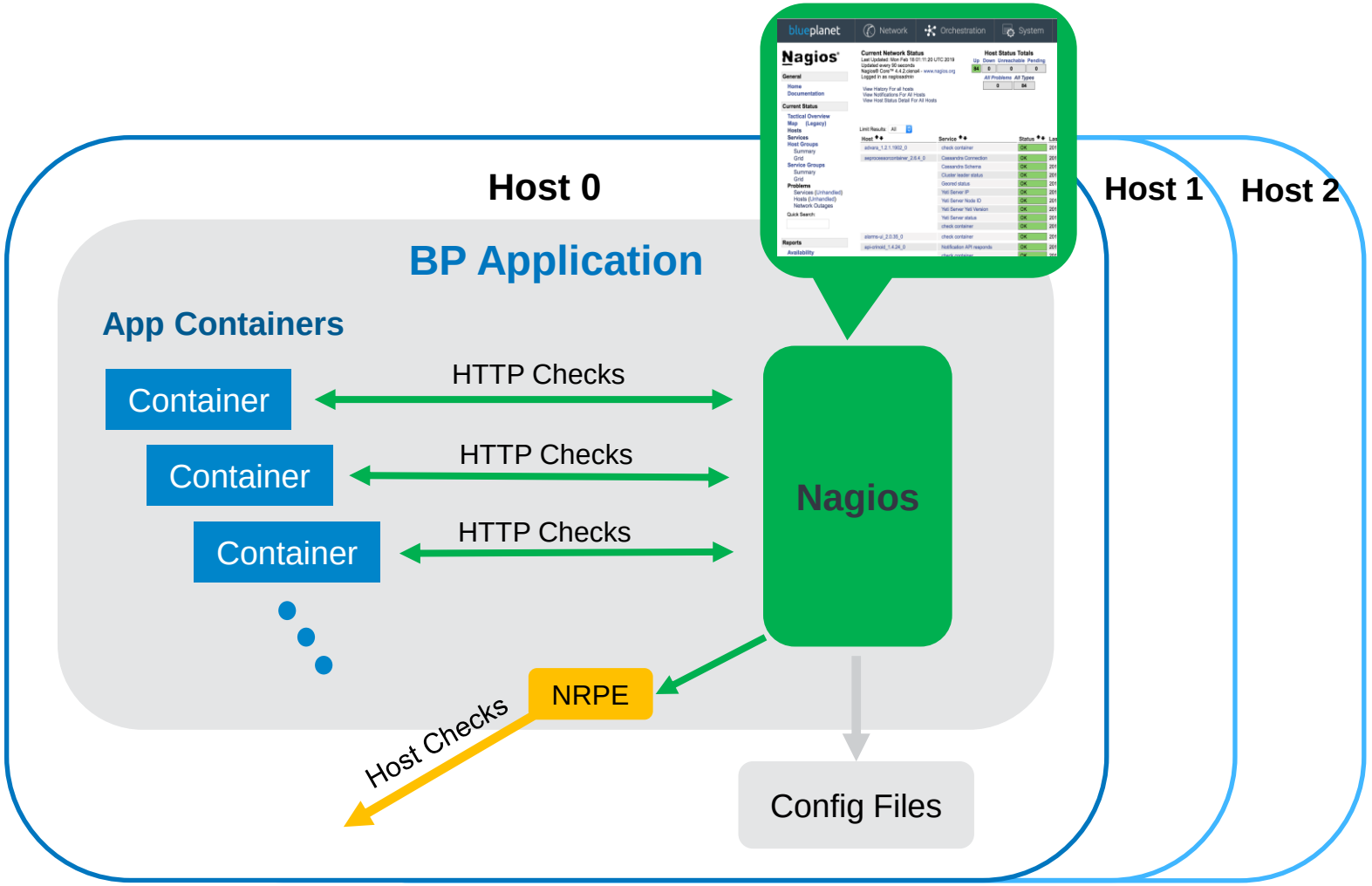
Nagios Implementation



1. Microservices register with Nagios by providing a URL and a description.
2. Nagios Remote Plugin Executor (NRPE) checks local and remote machines (CPU, Memory, storage checks).
3. Nagios performs an HTTP GET against the URL.

Blue Planet Nagios Monitoring

- **Nagios and NRPE containers are deployed on all hosts by default.**
- **Only one active Nagios server at any time (based on leader election mechanism).**
- **Nagios server runs checks every 5 minutes (configurable).**
- **HTTP checks for microservices.**
  - An “HTTP check” is an HTTP endpoint that Nagios calls periodically using check_http plugin.
  - HTTP response (200, 400, 500) is converted to Nagios status (ok, warning, critical).



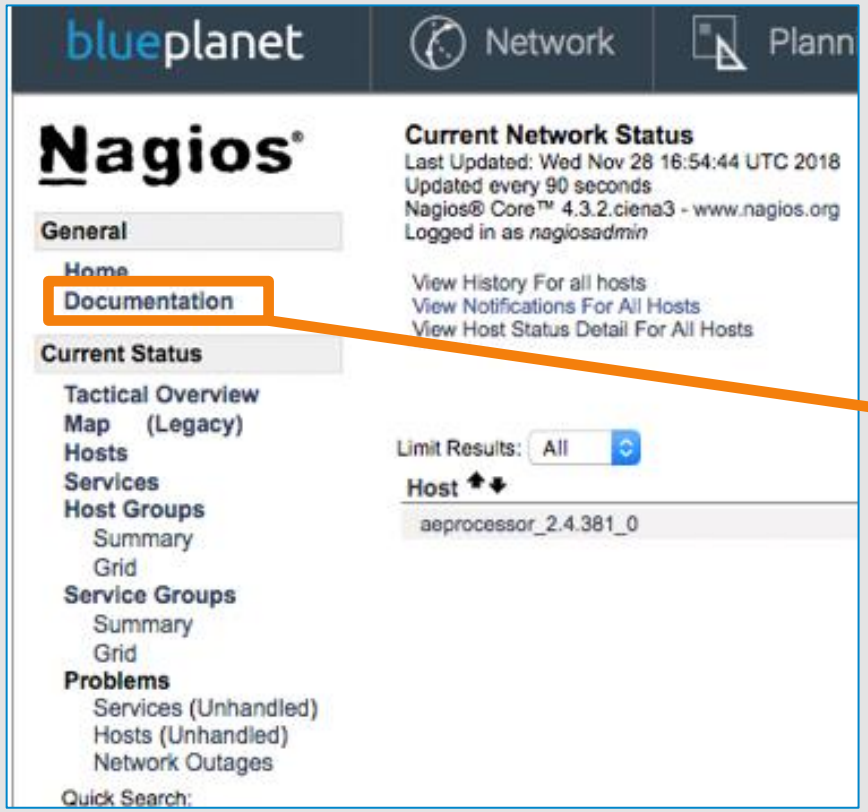
Nagios Remote Plugin Executer (NRPE)

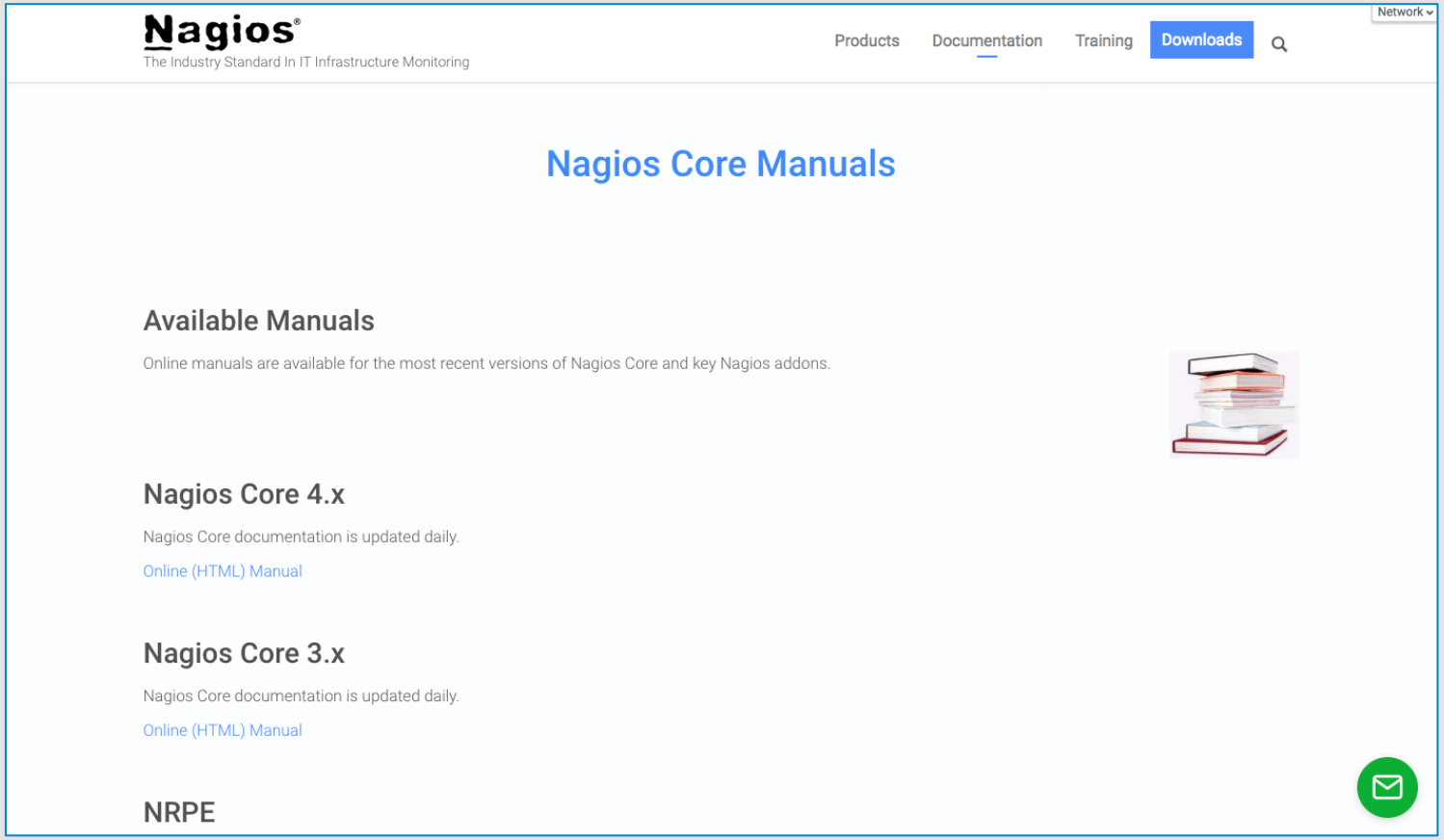
- **Component of Nagios.**
- **Monitors disk space, CPU, and memory.**
- **One NRPE per host in a cluster.**

nrpe_18.06-8.0.0_0	A Hostname	OK
	Current Load	OK
	Memory Usage	OK
	check clock drift	OK
	check container	OK
	check disk free space	OK
	check disk free space lvm	OK

Host: localhost.localdomain
OK - load average: 30.06, 31.98, 31.31
Memory: OK Total: 64265 MB - Used: 20537 MB - 31% used
TIME OK - 172.16.0.57 time is Sat Dec 22 00:50:12 2018 (0 seconds variance).
Container OK
DISK OK - free space: /rootfs/var/log 220911 MB (77.58% inode=99%): /rootfs/bp2-host-volumes 220911 MB (77.58% inode=99%):
OK - ALL LVM thinpools OK

Nagios: Documentation





Infrastructure Apps and Extended Platform

Agenda

1. Core Components
2. System Monitoring
3. **Performance Metrics**
4. Logging

System/Services Metrics Gathering

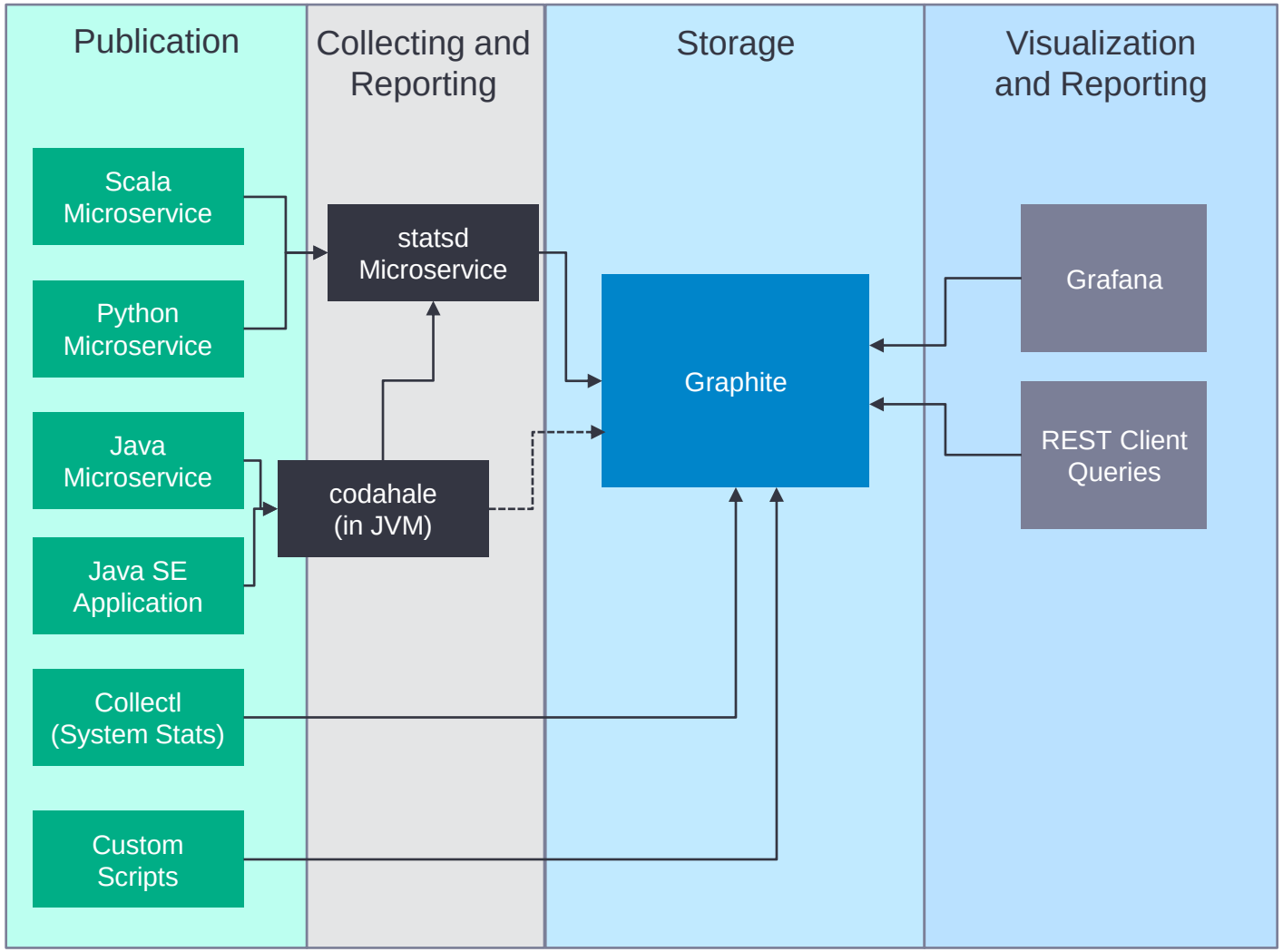
Graphite

In BP2 deployment *statsd* and *collectd* are the components used for metric data collection.

All metrics are then stored in Graphite.

Grafana is the visualization component.

REST queries directly into Graphite can be also used.



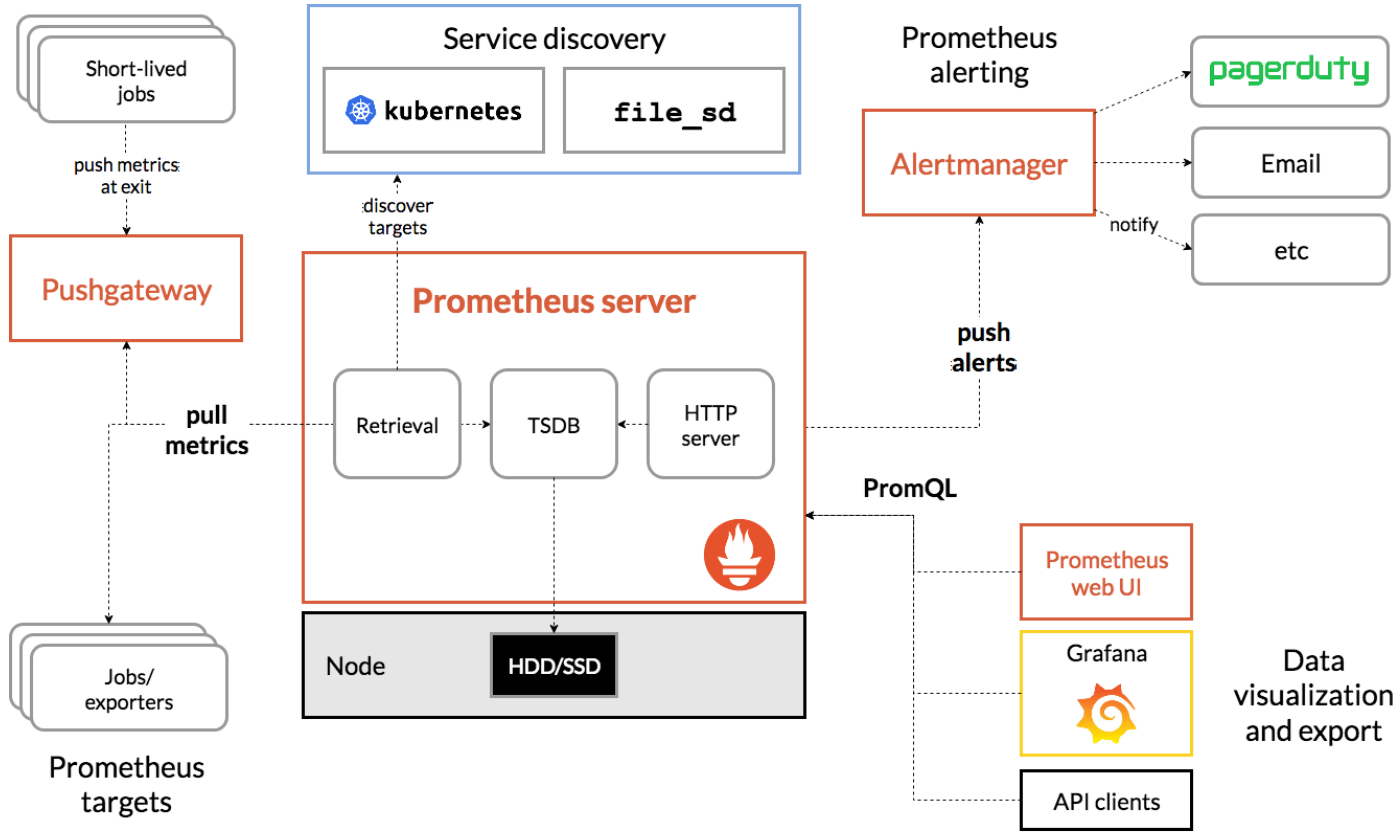
System/Services Metrics Gathering

Prometheus

For K8s Deployment, Prometheus is the component used for metric data collection.

Prometheus webUI is disabled and not used in Blue Planet.

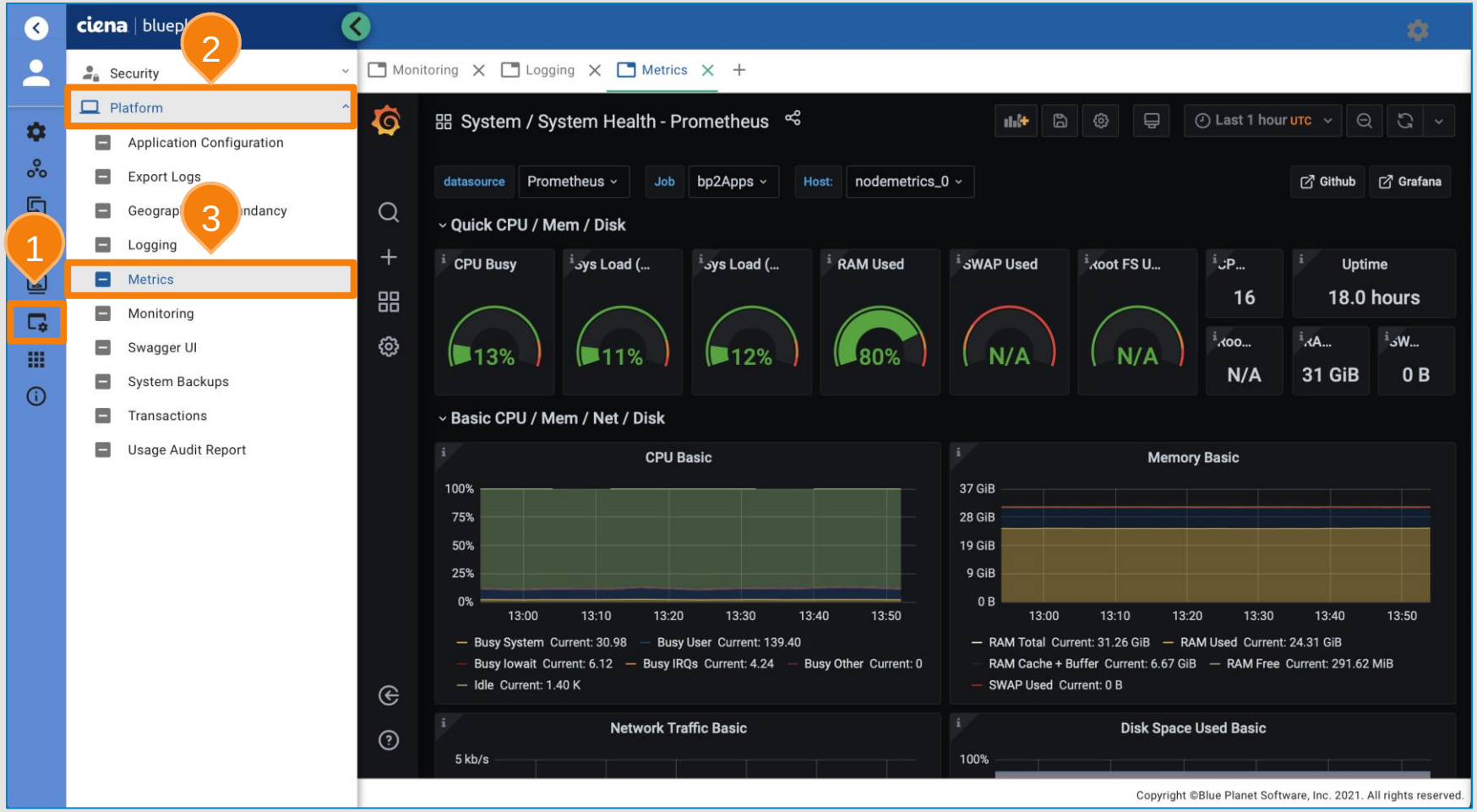
Grafana is the Blue Planet Platform component used for performance data visualization.



Grafana

Grafana is the metrics visualization component of the Blue Planet Platform.

It can be accessed from: System > Platform > Metrics menu.



Infrastructure Apps and Extended Platform

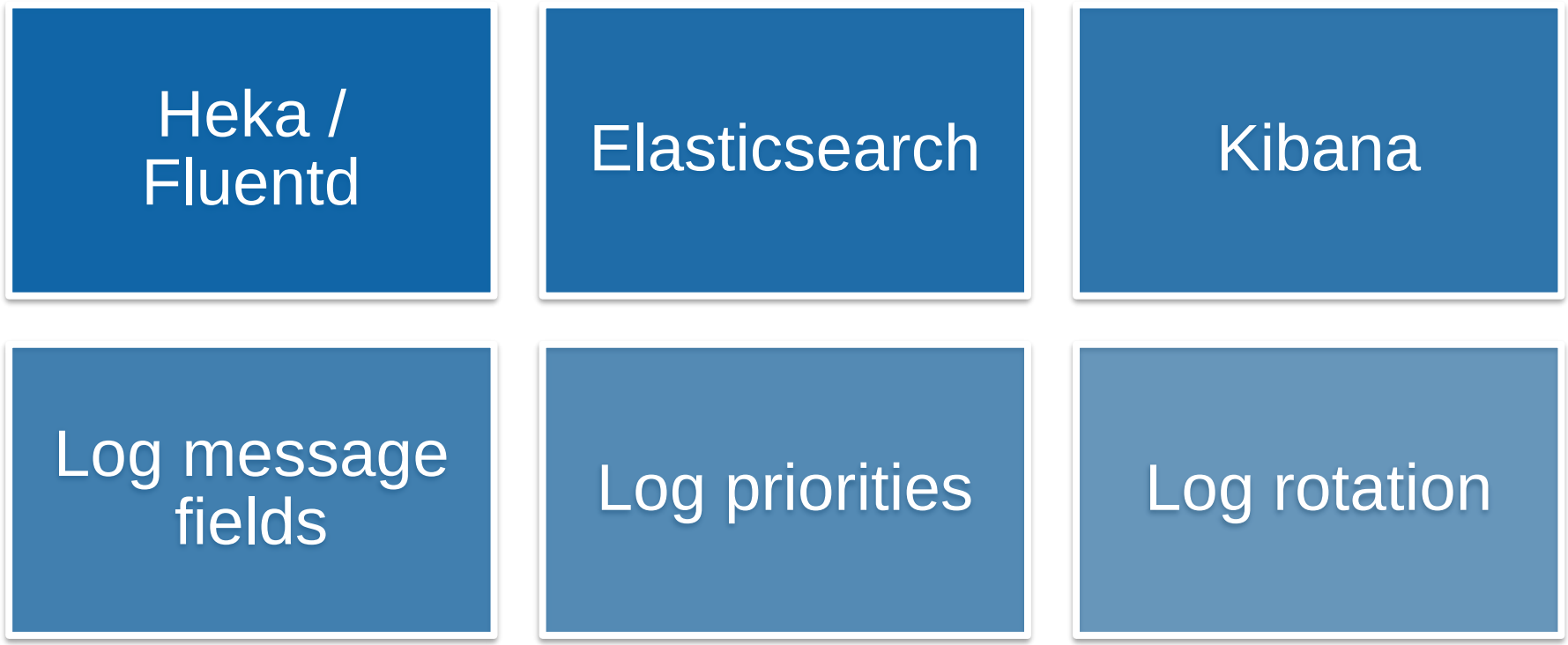
Agenda

1. Core Components
2. System Monitoring
3. Performance Metrics
4. **Logging**

Logging Components

- **Blue Planet Platform Administrators need to be able to view logs to determine the source of the issue.**

- **Key topics for logging:**



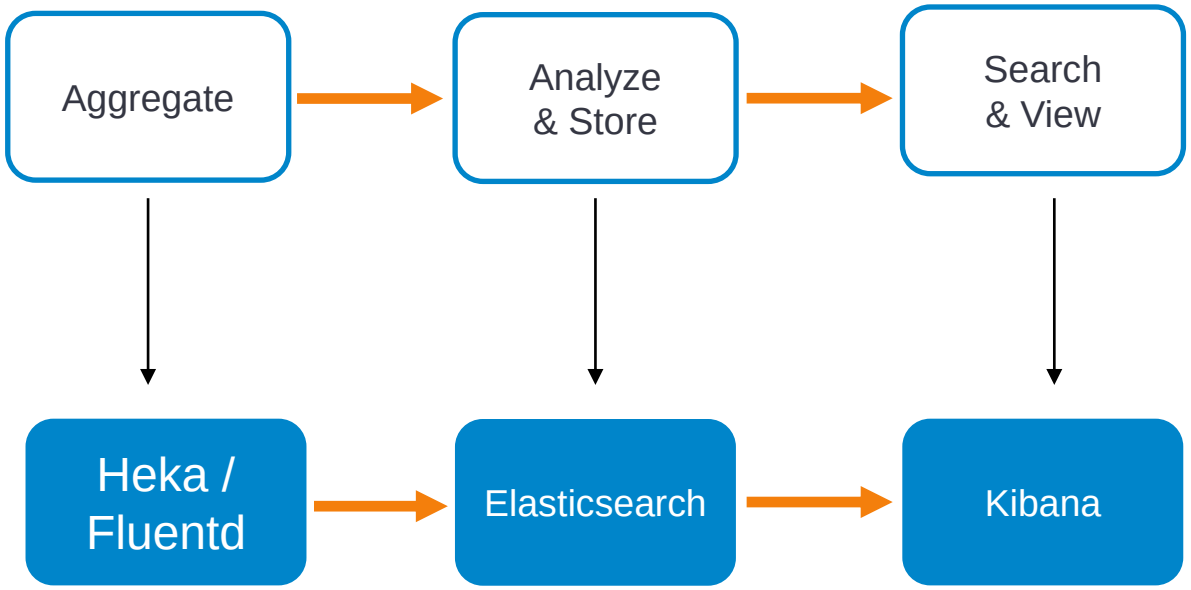
About Blue Planet Centralized Logging

Centralized logging is a system which:

- Aggregates all the logs from different hosts to a central location.
- Consolidates all the logs.
- Makes the data accessible through a single, easy-to-use interface.

Centralized logging in Blue Planet consists of three tools:

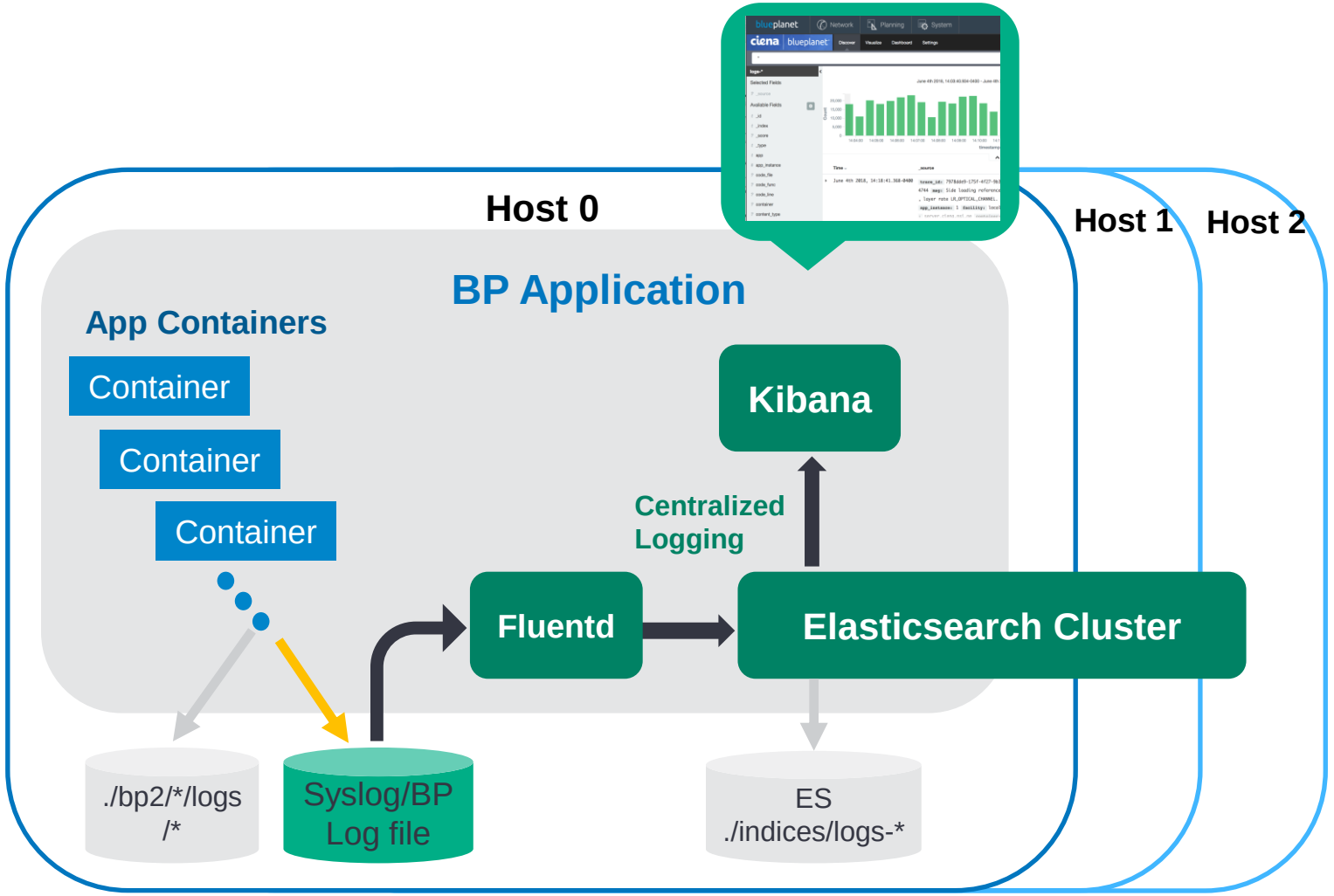
- Elasticsearch
- Fluentd / Heka
- Kibana



Blue Planet Centralized Logging

Elasticsearch & Kibana

- **Elasticsearch is an open-source, broadly-distributable, readily-scalable, enterprise-grade search engine.**
- **Kibana is an open-source data visualization plugin for Elasticsearch.**
- **Both are accessed via one UI.**
- **Blue Planet Platform Administrators need to be able to use the features provided by Elasticsearch & Kibana to view logs.**



Blue Planet Logging Conventions

Log Message Fields

Each log message contains one or more fields.

- Standard fields (see table later in this section).
- Application specific fields (Custom fields).

Fields are named with snake_case strings and are considered case-insensitive.

- Applications must not emit fields named with an initial underscore.
- Applications must not emit fields named with a period.

Typically, field values should be:

- String
- Boolean
- Integer
- Double

Null values typically represent existing fields that have dropped their values.

Binary data is typically stored as an array of integers: [65, 32, 66].

Blue Planet Logging Conventions

Important Fields

Field Name	Description
app	Name of the Blue Planet application from which the message originates, for example, "stok" or "galera". The core platform sets the BP_APP environment variable when running your app. You should use this rather than a hard-coded value (the BP2 logging libraries do this automatically).
app_instance	Instance number of the container, as a string. This is a small number assigned by the platform, available through the BP_APP_INSTANCE environment variable set by the core platform.
container	Docker Container ID of the container which generated the message truncated to 12 characters. Within a container, this can be found in /etc/hostname or the environment variable HOSTNAME.
container_id	A Docker container ID, for example, 23bfeeb0a6a4. This should be truncated to 12 characters to match the Docker-assigned hostname of the container. Note that this field is for the subject of the message, not the source. Use the container field to indicate the source.
host	The hostname of the physical host on which the Docker container runs. This is typically obtained by bind-mounting the host's /etc/hostname file as /etc/physical_hostname. If the file contains an FQDN only the local part should be used (for example, if /etc/physical_hostname contains foo.example.com the "host" field should contain foo).
msg	Human-readable message. This should be present for messages of priority INFO and up. Note that "human readable" means "understandable to someone who hasn't read the code", such as other developers and customer service, thus these abbreviations should be avoided.

Blue Planet Logging Conventions

Important Fields

Field Name	Description
msg_id	A well-known ID string that marks the message has specific semantics. See the list below.
namespace	Category of the log message within the application, for example, the name of a Python logging. Logger, the namespace of a Twisted logger message, or the name of a Java Logger object in slf4j, log4j, or logback.
oplog	Boolean which identifies this message as an Operational Log.
priority	A syslog priority value from 0 to 7 (emerg, alert, crit, err, warning, notice, info, debug). See the Message Priorities section below. Also known as a log level or severity. Your language's logging framework likely defines a slightly different list and maps to these values. Custom priorities are not allowed.
timestamp	Syslog-style RFC 3339 timestamp. This MUST have millisecond precision and SHOULD have at least microsecond precision.
trace_id upstream_id hop_id	Tracing identifiers, each a UUID in 36-character hexadecimal form.
traceback	Exception stack trace as a multi-line string.

Blue Planet Logging Conventions

Message Priorities

A priority is a numeric value used to indicate how serious a log message is.

Blue Planet follows the Linux conventional log priorities:

Priority	#	Semantics	Python	Java	ASP.NET
emerg	0	Kernel panic, and so on. Not for application use.			
alert	1				
crit	2	Something very wrong: unavailability, data loss. If CS doesn't know what to do, some developer is going to be woken up.	CRITICAL		LogLevel.Critical
err	3	An unexpected error condition, e.g. a fallback exception handler. The developer should be notified on a next-business-day basis.	ERROR	ERROR	LogLevel.Error
warning	4	Something went wrong, but the condition was handled.	WARNING	WARN	LogLevel.Warning
notice	5	Administratively important lifecycle messages, such as completion of application startup, schema migrations, and backup and restore operations.			
info	6	Normal request processing status messages. Generally, every request handled should emit at least one message at this priority to enable traceability.	INFO	INFO	LogLevel.Information
debug	7	For developer use.	DEBUG	DEBUG	LogLevel.Debug

Linux Administration Feature

Log Rotation

- **System log files become very large over time.**
- **Logrotate is used to handle this potential issue:**
  - Executed nightly through a cron job.
  - Designed to rotate log files.
  - Configured automatically with "CienaBundle”.
  - Configuration files:
    - /etc/logrotate.conf
    - /etc/logrotate.d

logrotate.conf example: rotates every 30 days

```
# rotate log files daily
daily

# keep 30 days worth of backlogs
rotate 30
```

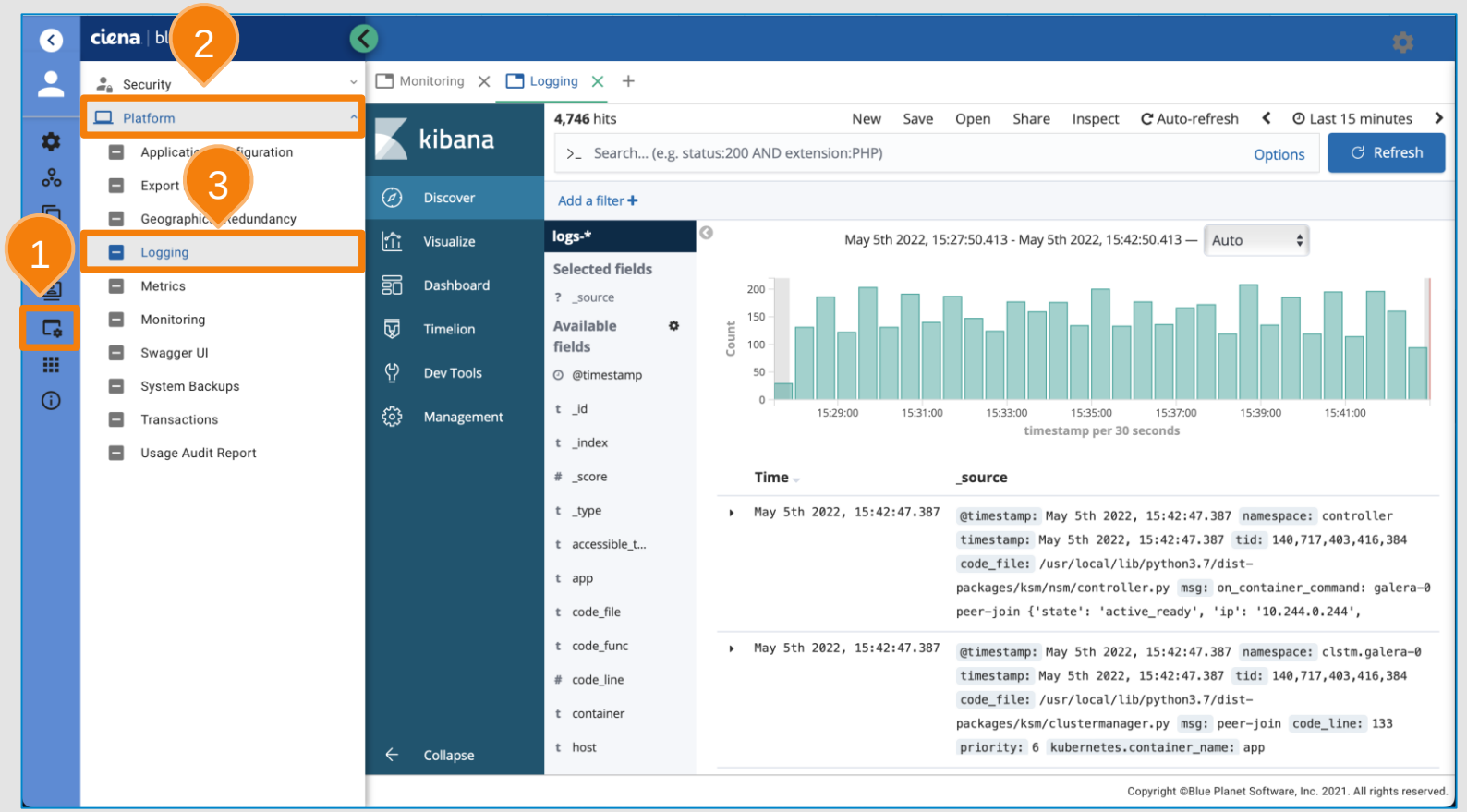
Using Elasticsearch & Kibana

Log tools are accessible by clicking System > Platform > Logging.

Dashboards provide visualizations to get an overall picture.

Prebuilt dashboards are available, but you can also create custom dashboards.

The search feature allows you to search by criteria like priority, app name, or host.



Summary

Infrastructure Apps and Extended Platform

In this section, you learned about Infrastructure Apps and Extended Platform components.

- Blue Planet Platform uses HAProxy to provide high availability and load balancing for Blue Planet Applications.
- Tron is the centralized User Access Control service for Blue Planet.
- API-GW is a separate microservice in Blue Planet solution primarily responsible for AAA (authentication, authorization, and accounting) of any API call to any microservice that requires it.
- Kafka is used as a software message bus.
- Nagios Monitors Blue Planet Microservices.
- Grafana is the metrics visualization component of the Blue Planet Platform.
- Elasticsearch is used for log aggregation to a central location and Kibana is used for log data visualization.

blueplanet®

a division of Ciena

blueplanet®
a division of Ciena

HA and GR Overview

Introduction to the Blue Planet Platform

PLF111ILT-A, Revision 1.0

Introduction to Blue Planet Platform HA and GR

Objectives

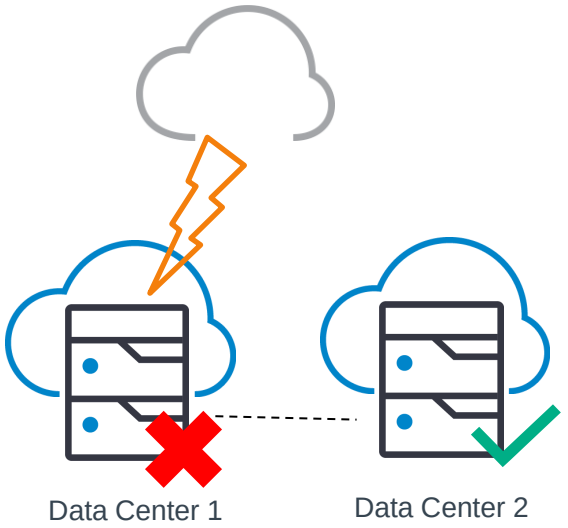
- Describe Blue Planet disaster recovery options
- Explain the high availability and geographical redundancy architecture in Blue Planet platform
- Describe the Blue Planet software component clustering concept
- Analyze the role of HAProxy in HA enabled Blue Planet deployments
- Describe intra cluster status communication and processes for joining and leaving the cluster
- Explain Blue Planet cluster synchronization

Agenda

1. **BP2 HA Overview**
2. Application Clustering
3. Intra-cluster Communication
4. Geo-Redundancy

Disaster Recovery

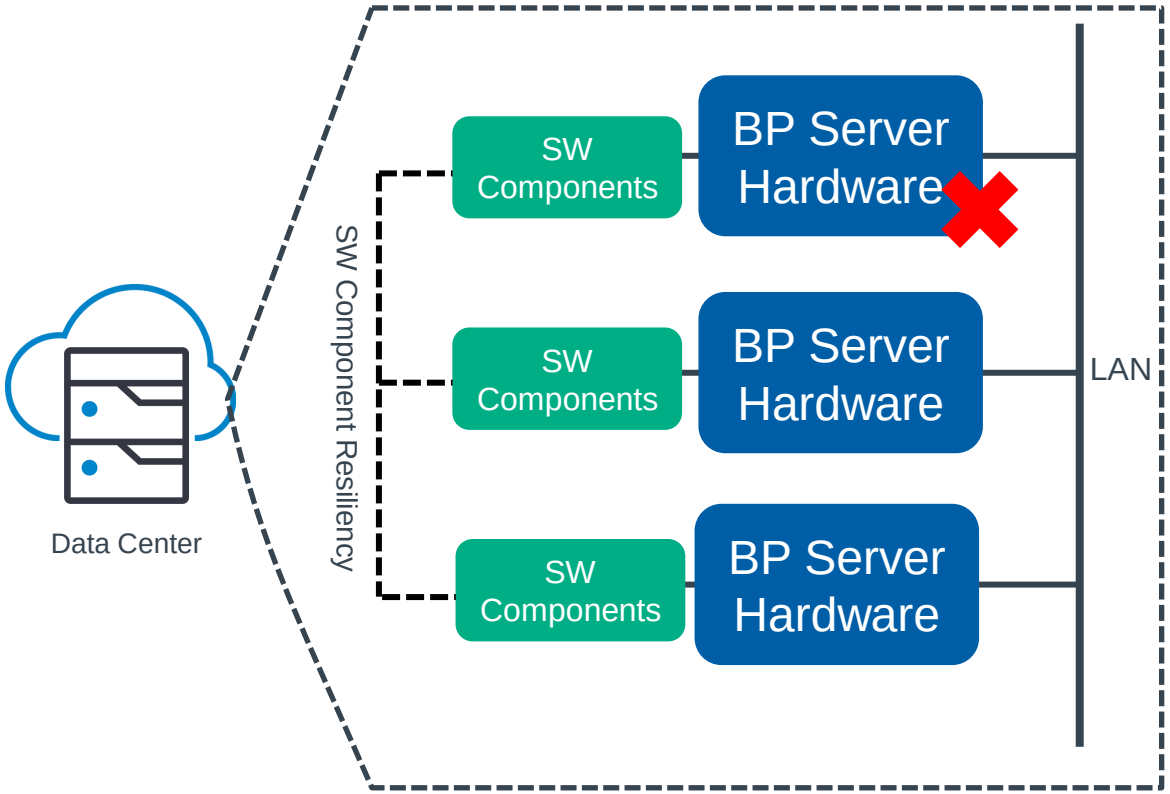
- **Disaster recovery plan helps to minimize the disruptions on the Blue Planet servers when things go wrong:**
  - Hardware and Software Defects
  - Network Outages
  - Power Outages
  - Natural Disasters
- **The Blue Planet solution supports two disaster recovery options:**
  - High Availability (HA)
  - Geo-Redundancy (GR).
- **The best time to implement HA and GR is during the initial installation process.**



Note: HA protects servers within a cluster. GR prevents the loss of an entire cluster.

BP2 HA Overview

- **HA provides capabilities that allow a site's primary functionality to remain operational in response to a failure of a component on that site.**
  - Accomplished by deploying a *server cluster*.
    - Redundant components distributed across independent hardware (independent hosts/servers).
    - Standard is the 3-server cluster.
  - An additional benefit of HA is *load balancing*.
    - Software components in a cluster share the processing load.
  - Blue Planet components use a combination of *quorum* and *duplex* redundancy to provide HA.

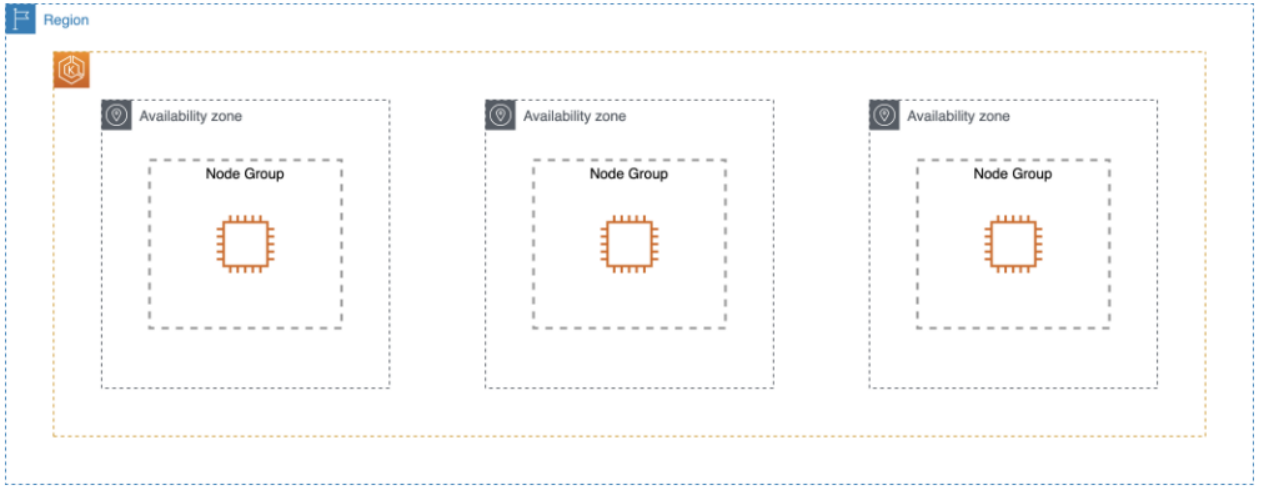


Note: The HA feature is designed to survive a single-host failure.

K8s HA Overview

HA refers to capability, internal to a site, that allows the site's primary functionality to remain operational in response to a failure of a component on the site.

- A Highly Available architecture that spans three Availability Zones.
- A virtual private cloud (VPC) configured with public and private subnets according to AWS best practices, to provide you with your own virtual network on AWS.
- In the public subnets, managed NAT gateways to allow outbound internet access for resources in the private subnets.
- In one public subnet, a Linux bastion host in an Auto Scaling group allows inbound Secure Shell (SSH) access to Amazon Elastic Compute Cloud (Amazon EC2) instances in private subnets. The bastion host is also configured with the Kubernetes kubectl command-line interface for managing the Kubernetes cluster.
- An Amazon EKS cluster, which provides the Kubernetes control plane.
- In the private subnets, a group of Kubernetes nodes.
- Kubernetes allocates a pod to a node based on several factors, such as resource availability, node availability, taints, tolerations, and affinity and anti-affinity rules. In the default setup, Kubernetes can schedule a pod on any node that meets these constraints.



Introduction to Blue Planet Platform HA and GR

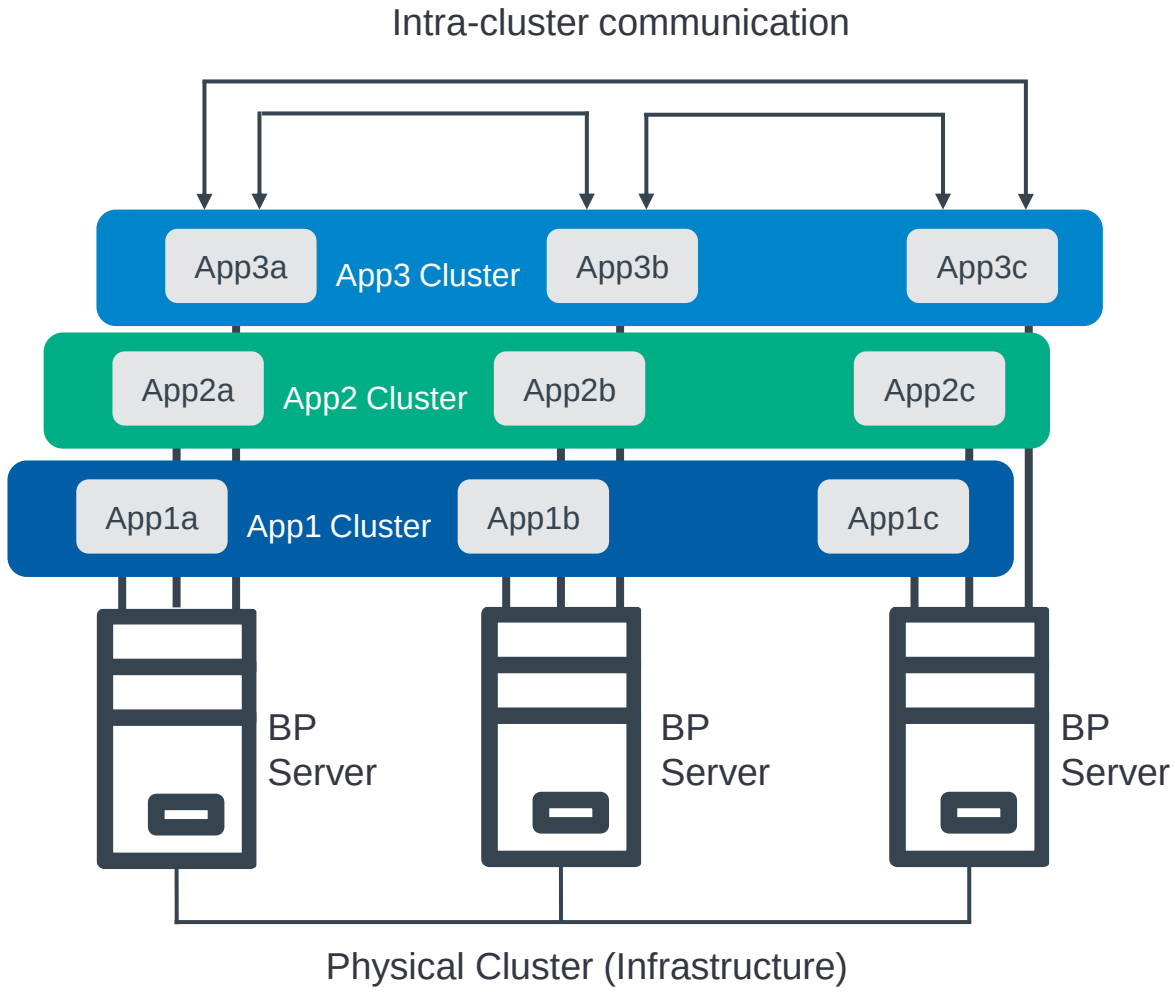
Agenda

1. BP2 HA Overview
2. **Application Clustering**
3. Intra-cluster Communication
4. Geo-Redundancy

Software Component Clustering

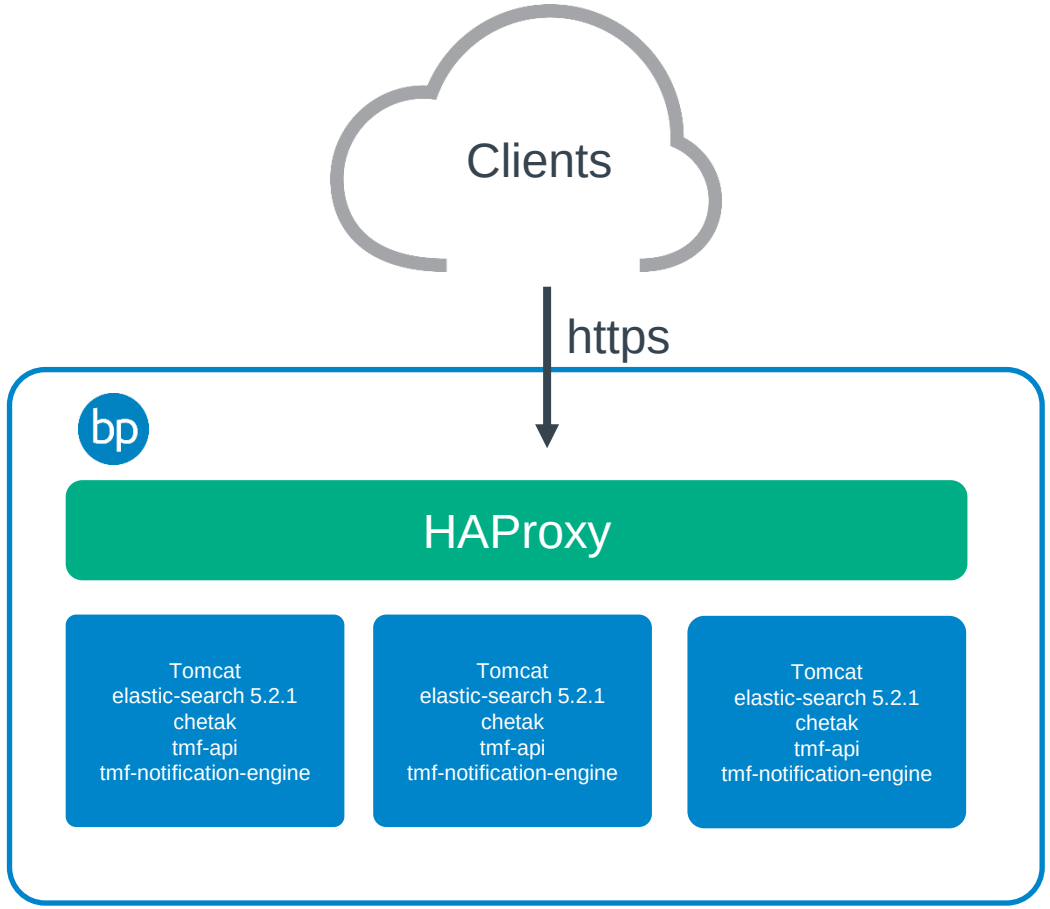
Blue Planet HA cluster functions at two levels:

- **Infrastructure Level**
  - Hardware that supports clustering and Blue Planet clustering software for managing clusters.
- **Application Level**
  - Individual applications form their own HA cluster.
  - Multiple application clusters running on top of a physical cluster infrastructure at the same time.
  - Example: App1, App2, and App3 - run on top of the servers in the physical cluster.



HAProxy Overview

- **HAProxy is one of the most widely used software load balancers and application delivery controllers.**
  - Built for Speed
  - Feature Rich
  - Open Source
- **Blue Planet Platform uses HAProxy to provide high availability and load balancing for Blue Planet applications.**
- **HAProxy exists between the clients and the services.**
  - Handles and directs https requests.
  - SSL Certificates are important in https communication.



HAPROXY

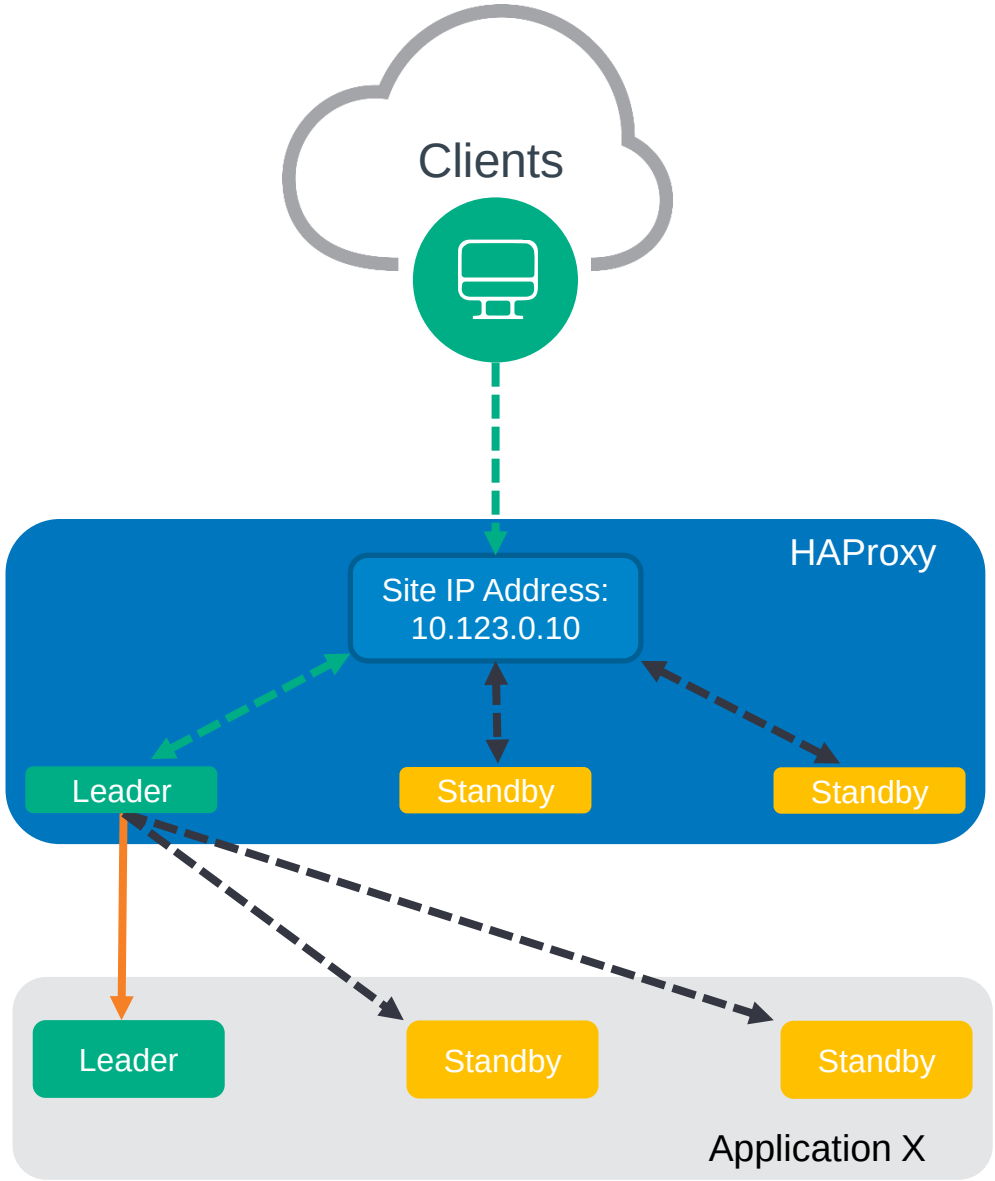
HAProxy Floating IP Address

HAProxy application cluster uses a single floating IP address exposed to the clients and managed resources.

- *Site IP address* is used to connect to the HA proxy leader.
- In the example, the users and managed resources connect to the site IP address 10.123.0.10.

Site IP address follows the HAProxy leader.

- In case the current leader fails:
  - One of the standbys transitions to Leader.
  - The new leader takes over the site IP address.



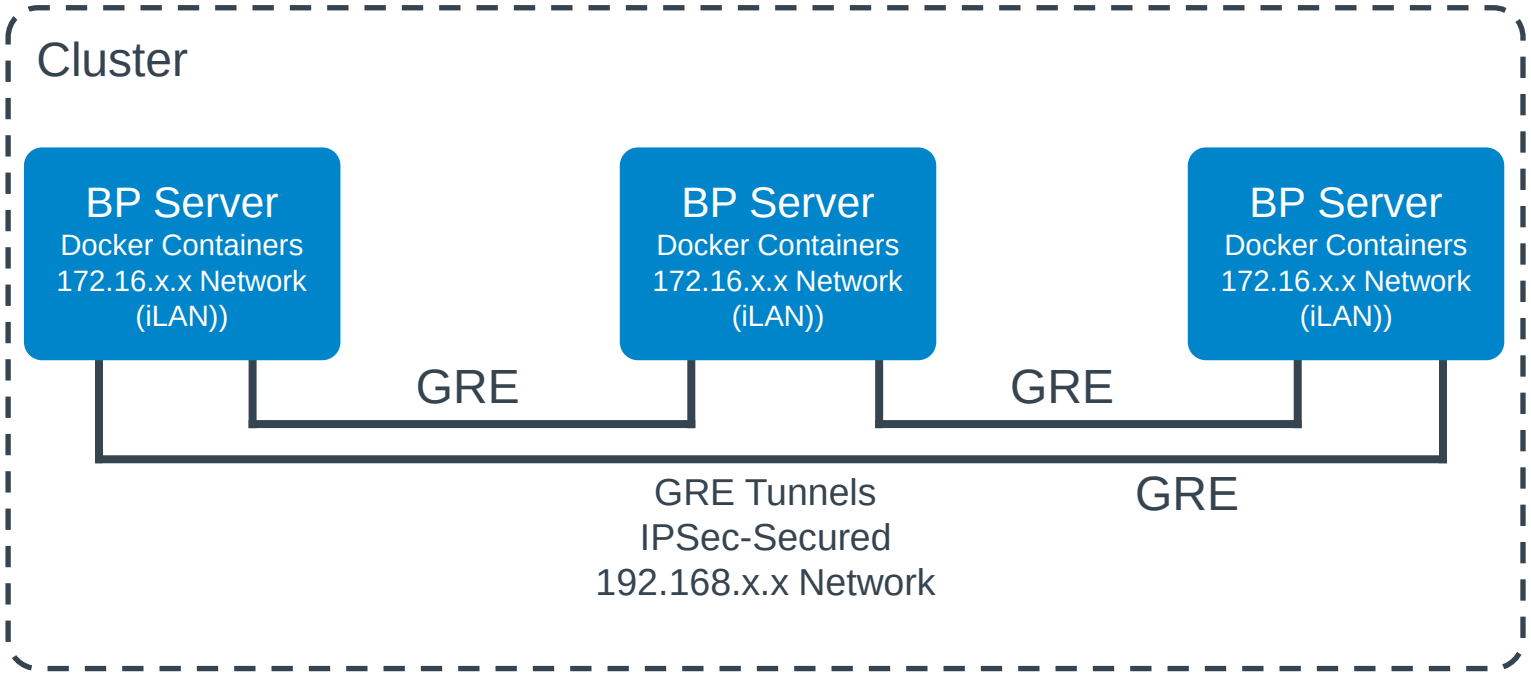
Introduction to Blue Planet Platform HA and GR

Agenda

1. BP2 HA Overview
2. Application Clustering
3. **Intra-cluster Communication**
4. Geo-Redundancy

Intra-cluster Connectivity

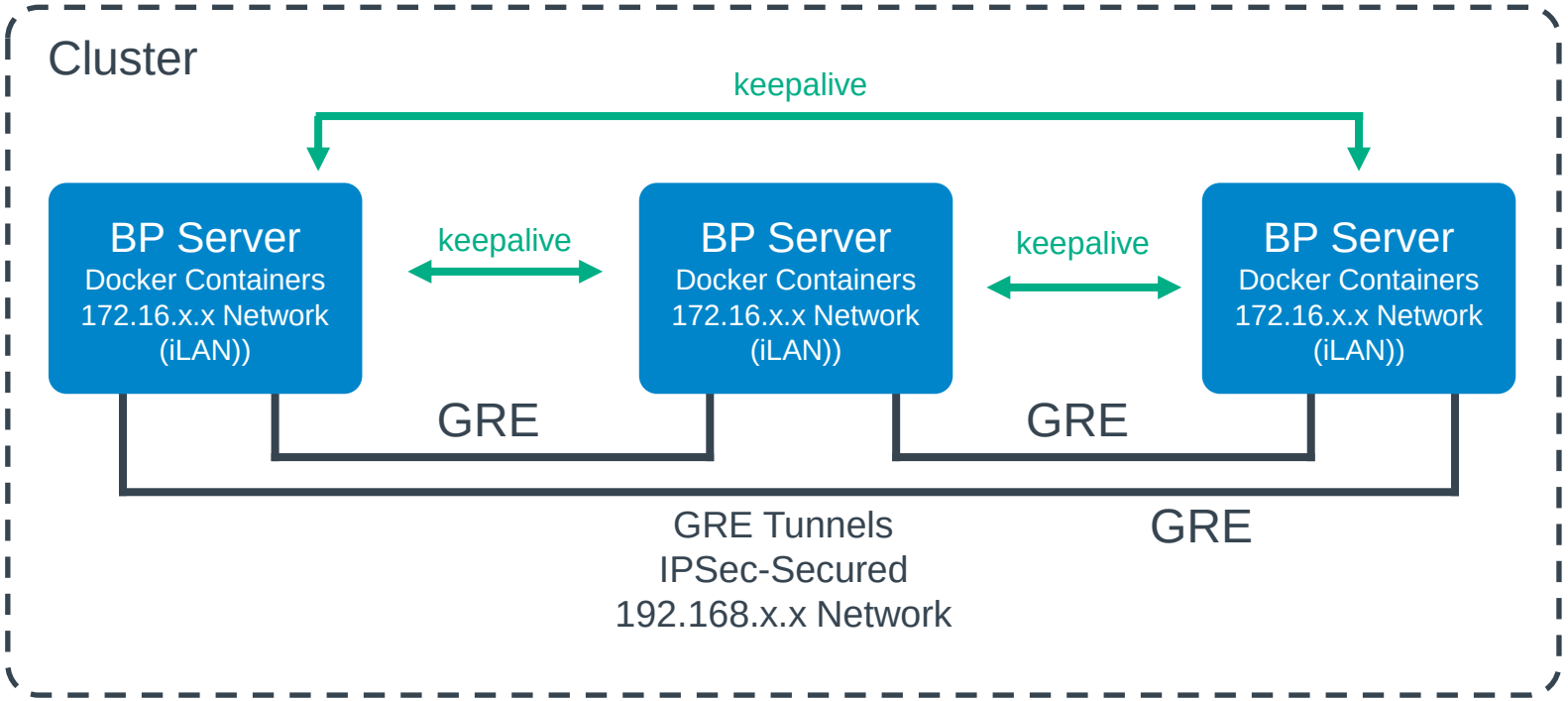
- **Internal LAN (iLAN)**
  - Docker IP Addresses
- **Valid options for iLAN tunnel type are**
  - GRE (default), or
  - VXLAN
- **Generic Routing Encapsulation (GRE) tunnels**
  - Node-to-Node
  - IPSec



Status Communications

Servers in the cluster exchange keepalives.

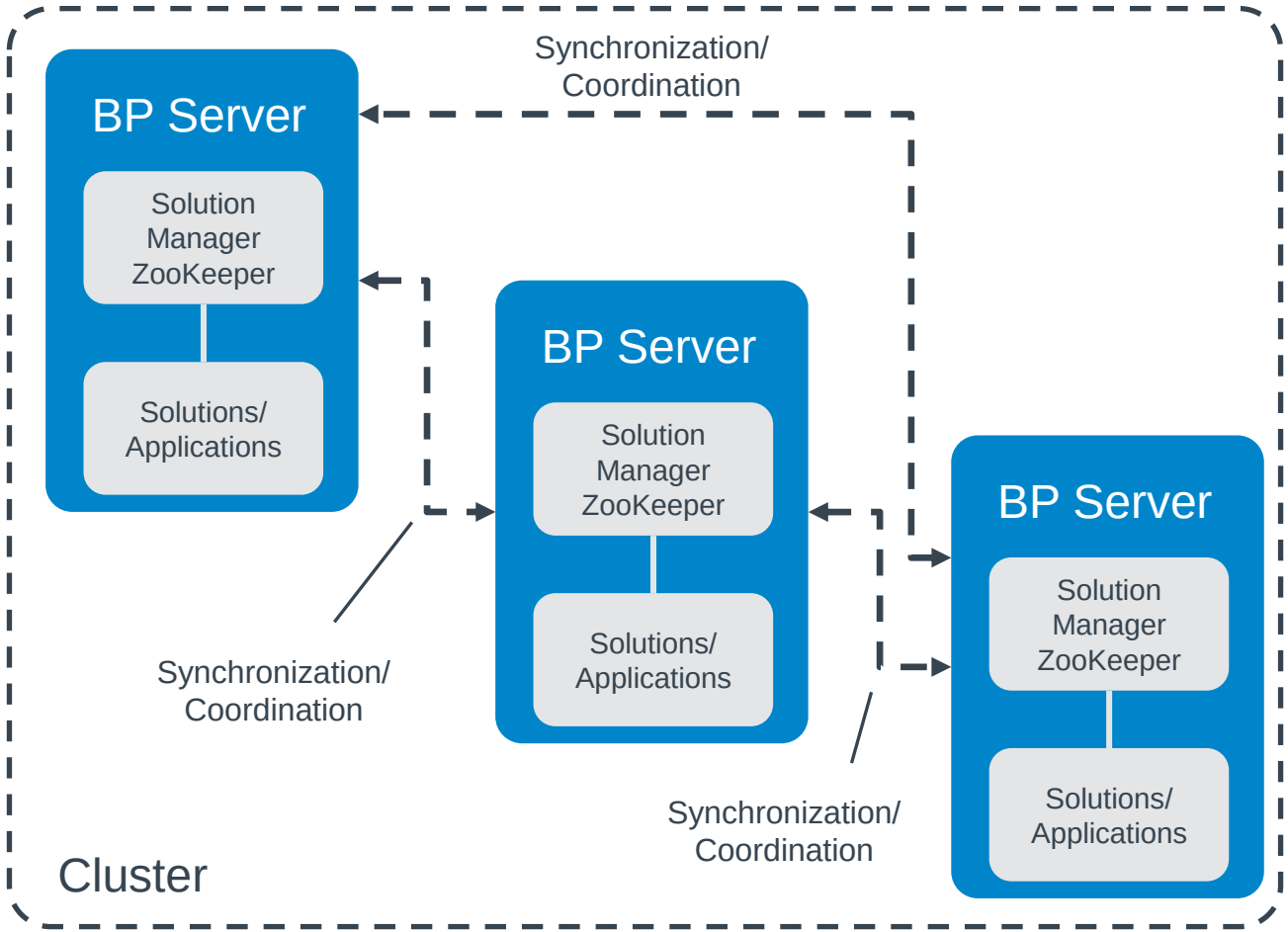
When one server in the cluster is not heard from for a while, the other servers assume it is down.



Cluster Synchronization

Components responsible for coordinating and synchronizing all the software components in the cluster:

- **Solution Manager**
  - Deploys Docker containers to servers.
  - Coordinates membership and leadership for applications.
  - Communicates membership states between application instances.
- **ZooKeeper**
  - Centralized service for maintaining configuration information, naming, providing distributed synchronization, and providing group services.
  - Primarily used to facilitate inter-app coordination.



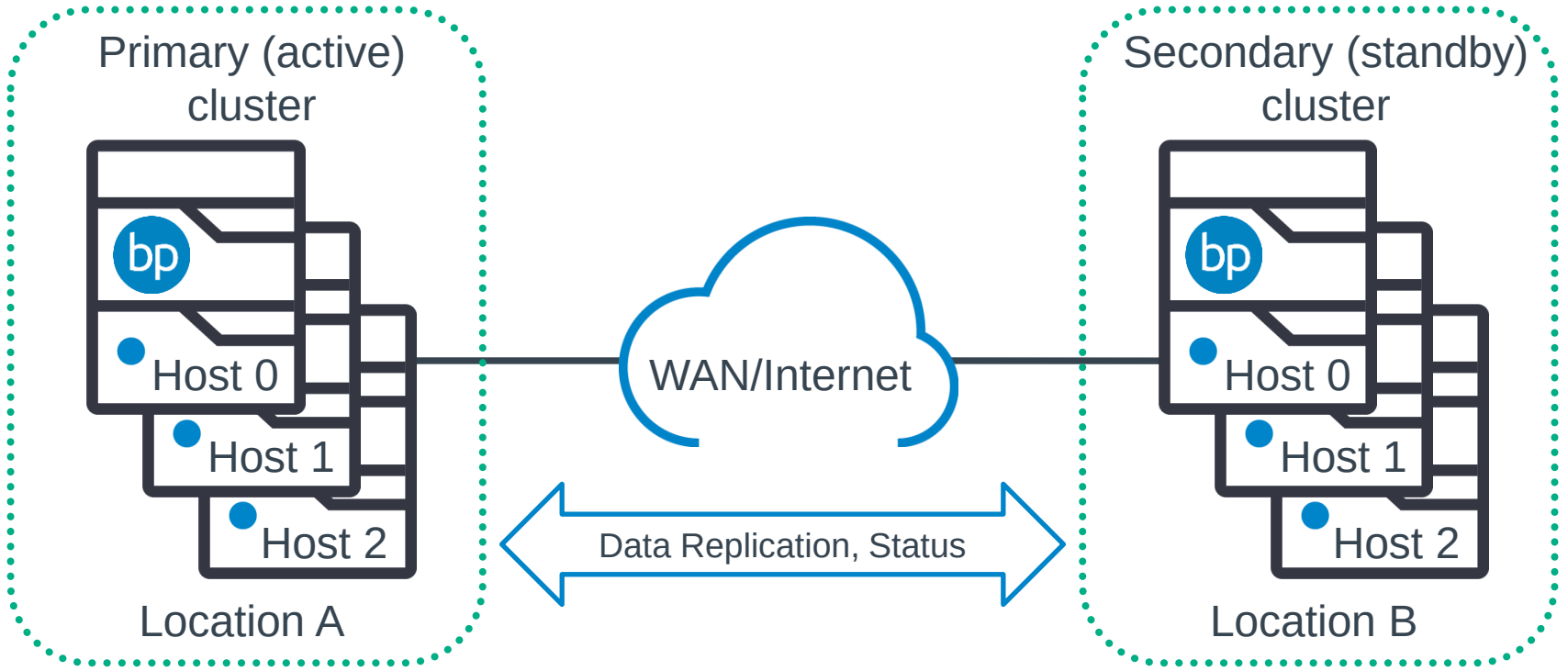
Introduction to Blue Planet Platform HA and GR

Agenda

1. BP2 HA Overview
2. Application Clustering
3. Intra-cluster Communication
4. **Geo-Redundancy**

Geo-Redundancy

- **HA protects against a server failure within a cluster, but what if the whole cluster fails?**
- **Geo-Redundancy (GR) provides a backup for an entire cluster.**
  - Two GR modes: *cold* standby and *warm* standby.



GR Deployment Options

BPI supports geographically redundant deployments on two geographical locations:

- Single host deployments:
  - Active and standby host spanning two sites.
  - The standby system for a single host must also be a single host.
- Cluster deployments:
  - Active and standby cluster on two sites.
- Warm standby redundancy between two geographically separate locations:
  - Active Site
  - Standby Site
  - Only two sites are supported.
- A tunnel is created between the sites for communication.
- HA systems require that 51% (2/3, 3/5, and so on) of the servers must be up for the site to be functional.
- In a warm standby geo-redundant environment, replication is managed individually by each application.

Summary

Introduction to Blue Planet Platform HA and GR

In this section, you learned the basics of the BP2 high availability features.

- Blue Planet solution supports two disaster recovery options, High Availability (HA) and Geo-Redundancy (GR).
- HA provides capabilities that allow a site's primary functionality to remain operational after a component failure on that site.
- Blue Planet HA cluster functions on two levels, infrastructure and application.
- Servers in the cluster exchange keepalives. When one server in the cluster is not heard from, it is assumed down.
- Solution Manager and ZooKeeper are responsible for synchronizing software components in a cluster.
- Geo-Redundancy (GR) provides a backup for an entire cluster, utilizing primary and secondary locations.

blueplanet®

a division of Ciena

blueplanet®
a division of Ciena

Security Overview and User Access Control

Introduction to the Blue Planet Platform

PLF111ILT-A, Revision 1.0

Introduction to Blue Planet Security

Objectives

- Describe Blue Planet Portfolio security objectives
- Examine the OS hardening best practices
- Discover the security architecture of Blue Planet products
- Discover the security of microservices and containerized applications
- Describe the security considerations in GR deployments
- Describe the security of Blue Planet interfaces
- Explain REST client, network element and user authorization and authentication

blueplanet
a division of Ciena

Introduction to Blue Planet Security

Agenda

1. **Blue Planet Platform Security Overview**
2. Blue Planet Platform Authentication Options

Security Objectives

Build consistent Security Policies across Blue Planet Portfolio.

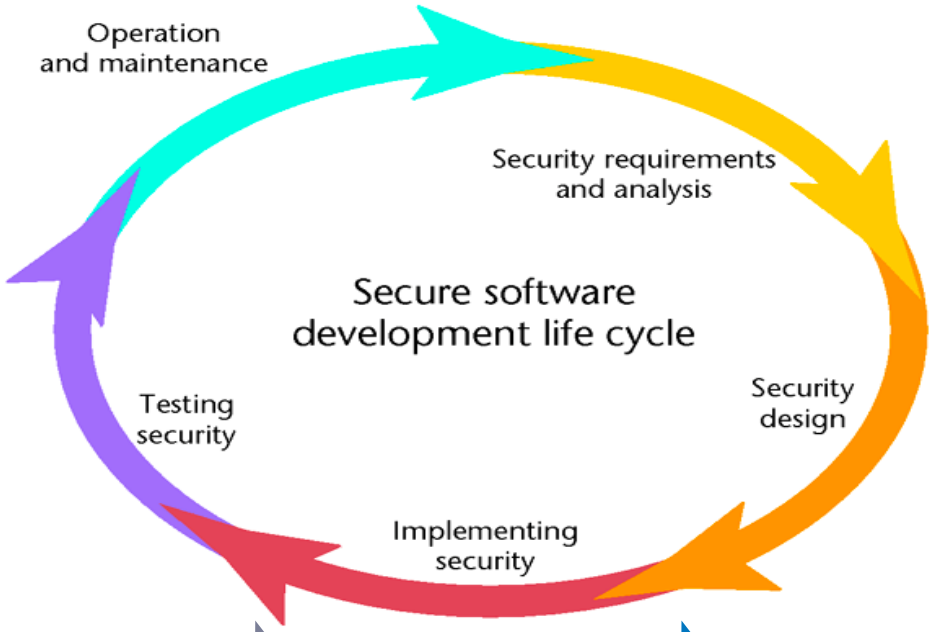
Identify areas that will help build a secure platform.

Build a security process that needs to be followed across product lines.

Create Security awareness by introducing a secure software development lifecycle.

Enhance existing capabilities of the platform by releasing Security Whitepaper/Posture Document.

Secure – Software Development



Requirements → Design → Development → Testing → Deployment

- Security Classification
- Risk Assessment
- Security Requirements

- Threat Modelling
- Design Review

- Secure Coding Guidelines
- Secure Code Spanning (e.g. in Build Pipeline)

- Security Testing
- Code Review

- Pentesting
- Security Hardening

blueplanet
a division of Ciena

Security in the Containerized Blue Planet Applications

Dedicated security authorization and access management system.

Network communication between docker container services.

- Use ILAN private IP addresses automatically assigned to hosts.
- Managed through an L2 virtual switch (open-v-switch).

Communication from/to external network traverse one specific proxy service (HAProxy).

- External northbound communication uses a single port (443).

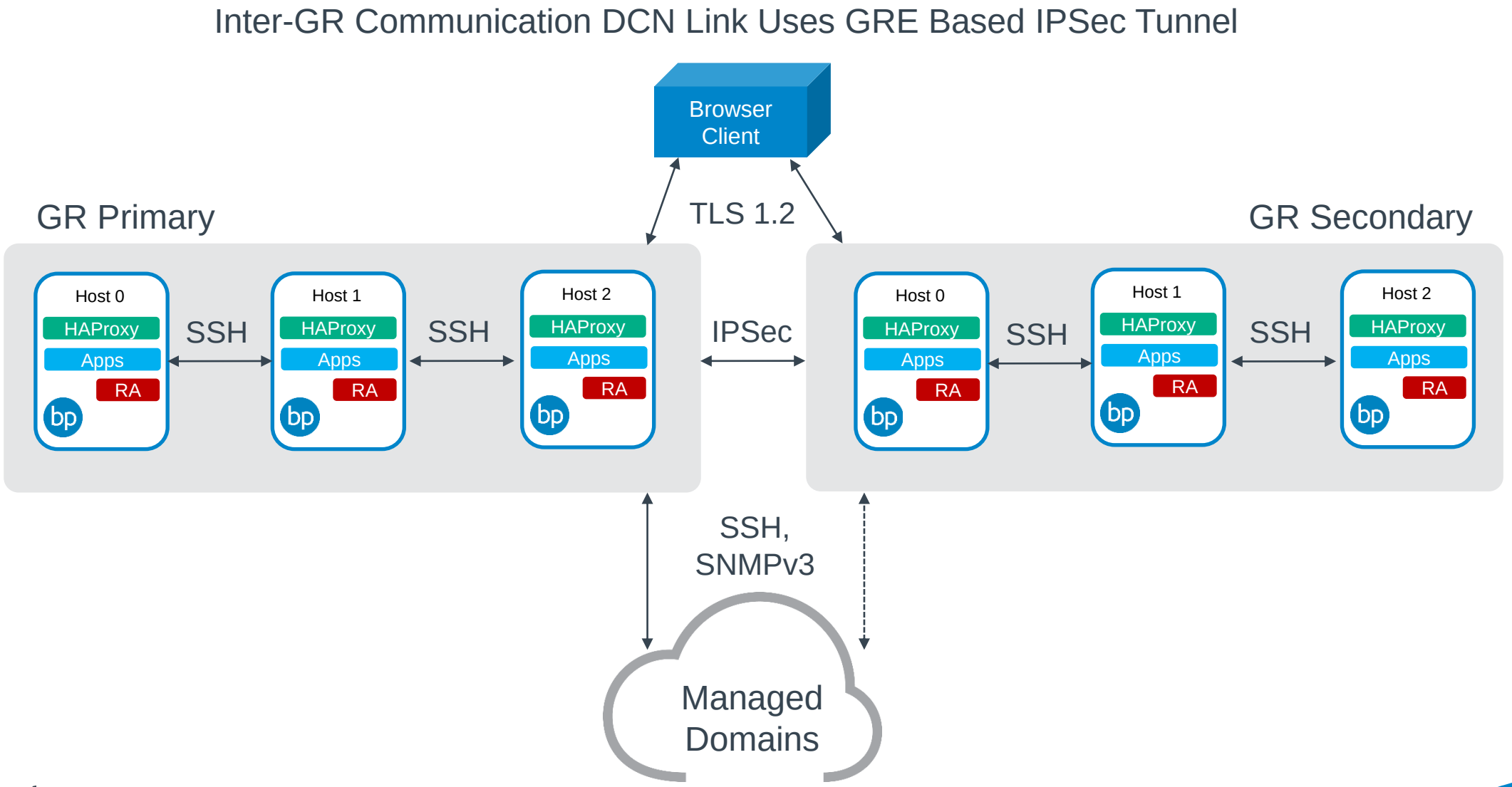
SSH communication use the SSH daemon running on the host server.

Data Communication Encryption

- TLS 1.2 (northbound)
  - Blue Planet products ship with a Ciena self-signed certificate (for TLS 1.2).
  - Certificate can be replaced by an SHA-2 signed certificate, purchased by the customer, emitted by a CA-trusted authority.
- SSH, and SNMPv3 (southbound)
- East-West communication within a multi-host site cluster uses SSH protocol.

User credentials are stored with encryption using AES-128+.

Security in GR Deployments



Blue Planet Interface Security Summary

- **Northbound Clients (REST and UI)**
  - Blue Planet enforces authentication and authorization.
    - Can be local or external (for example, RADIUS, LDAP).
  - Clients can use the OAuth2 token.
  - Communication is protected by TLS.
    - All REST and UI calls to Blue Planet use HTTPS.
- **Southbound Communication**
  - Blue Planet itself must be able to log in to all its domains.
  - Managed domains/devices enforce their own authentication and authorization.
    - Local or External
  - Communication is protected (encrypted) by SSH.

REST Client Authentication

- **Clients exchange credentials for a token.**
- **Token is then used as the authentication credential for subsequent REST calls.**
- **Pattern based on OAuth 2.**
  - RFCs 6749 and 6750
  - More information at *https://oauth.net/2*.
- **Credential authentication occurs at the Blue Planet server.**
  - The Blue Planet server may delegate to external authentication (RADIUS or LDAP).
- **Blue Planet UI authenticates in the same manner as any other REST client.**

Southbound Authentication to Network Elements

- **Network elements (NE) enforce their own authentication and authorization schemes.**
  - NEs can be configured for local or centralized authentication.
- **Blue Planet itself must be able to login to each NE with full privileges.**
  - Blue Planet is configured with NE credentials via connection profiles which are created by REST API or in the UI.
    - Connection profiles contain protocols and credentials.
  - A connection profile is specified for an NE when it is enrolled.
  - Connection profiles are editable after an NE is enrolled to allow bulk modification of credentials.

User Authorization

- **Blue Planet users can be assigned a set of predefined roles which define role-based access control (RBAC) permissions:**
  - **Application admin:** administration of Blue Planet, application-level security, basic visibility of network elements and services.
  - **Network admin:** full control of network elements and services.
  - **Planning admin:** full control of planning for the network.
  - **Observer:** basic visibility of network elements and services.
- **Additional roles with custom permissions can be created.**
  - Admin users are able to create new roles and assign permissions to them.
  - Once created, the new role can be assigned to users.
- **Blue Planet enforces permissions on both the UI and the REST API levels.**

Kubernetes Security

Cloud Provider Security

- Managed by the Public Cloud Providers
  - Infrastructure Security Best Practices

Cluster Security

- Securing the components in the application.

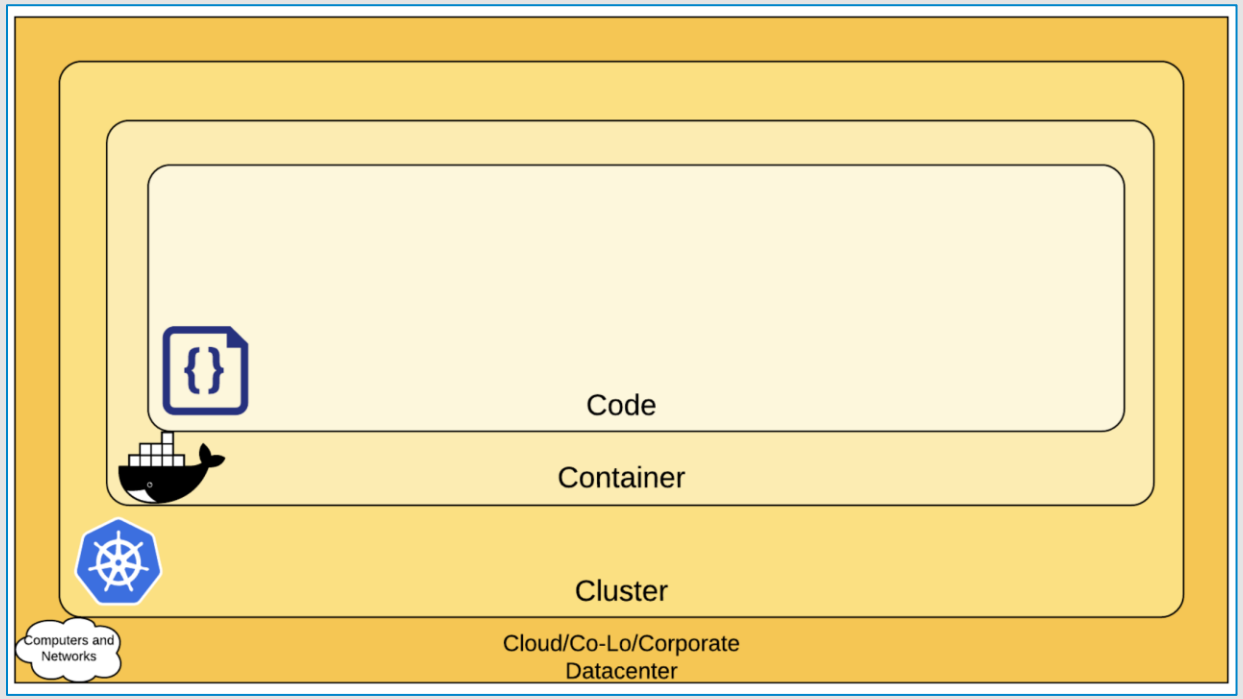
Container Security

- Base OS Updates
- Package Updates

Code Security

- Access Over TLS Only
- SAST and DAST

- Some of the POD Security Best Practices



- No host volume mounts.
- No privileged containers.
- No root user in the container.
- No sysctl to host kernel.
- Utilize readiness and liveness probes.
- RO container root file system.

Agenda

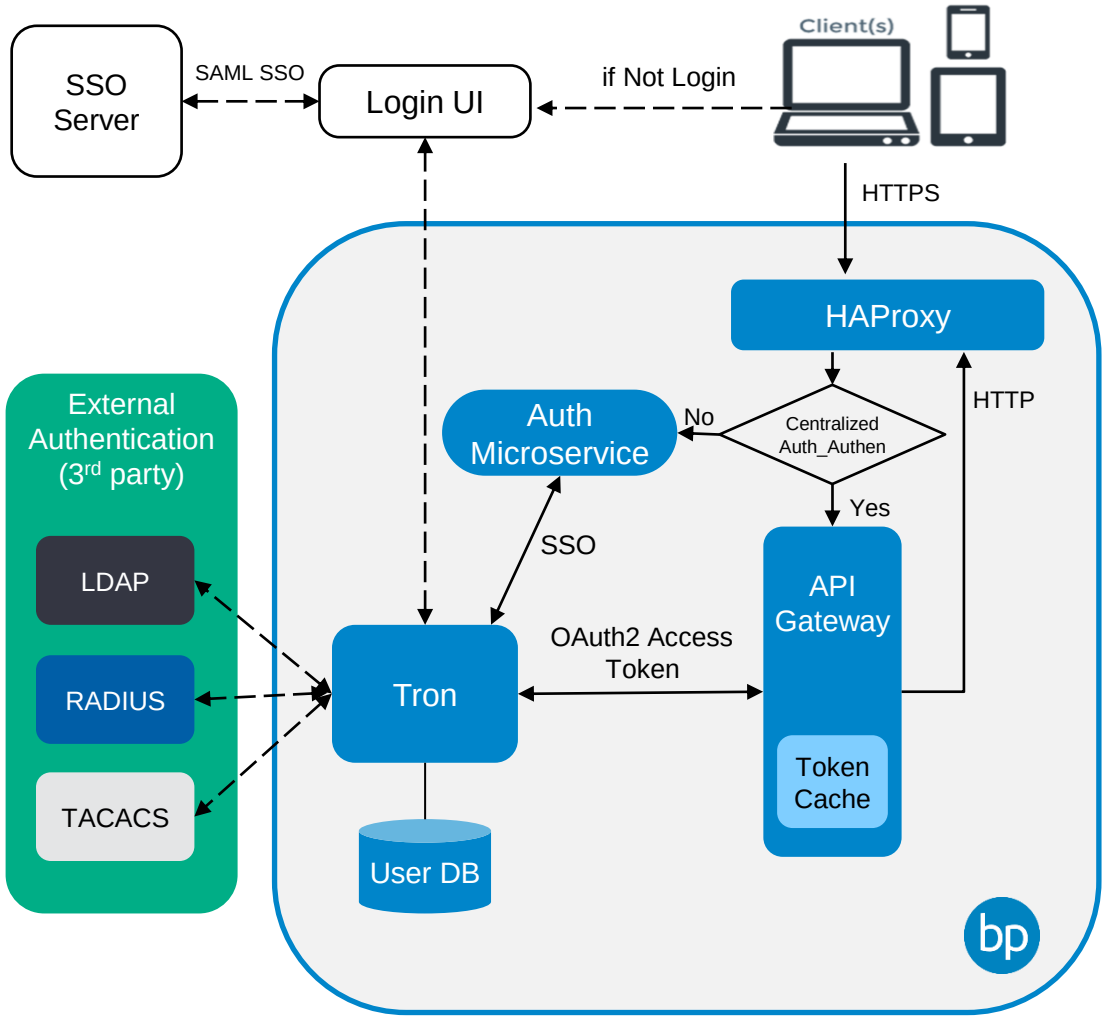
1 Blue Planet Platform Security Overview

2 **Blue Planet Platform Authentication Options**

Blue Planet Platform UI Authentication

Tron is the centralized User Access Control service for Blue Planet.

- Supports multiple external authentication systems:
  - LDAP
    - LDAP and LDAPS are supported.
  - RADIUS
  - TACACS
- Tron authenticates against the configured external authentication systems in order until either there is a successful authentication, or all methods have failed.



Remote Authentication Considerations

- **When remote authentication using RADIUS/LDAP is enabled:**
  - Local users are created but passwords can’t be changed except in RADIUS/LDAP.
  - Local and remote users can both exist in the same UAC instance, but not with the same usernames.
  - Local (native) users are not allowed to login except UAC admin/sysadmin user.
  - The **forgot password** link is solely for local users, users can not change remote passwords from BP.
- **Blue Planet users with *admin* and *sysadmin* roles bypasses LDAP and RADIUS authentication**
  - Authentication always succeeds based on local authentication, regardless if LDAP or RADIUS authentication is configured (for single remote server setups).

Remote Authentication and Multi-Tenancy

- **Multi-tenancy support in 22.x:**
  - RADIUS authentication supports single tenants only.
  - LDAP authentication supports multi-tenancy.
- **Role mapping support:**
  - RADIUS - role mapping not supported.
  - LDAP - role mapping supported.
- **Tenant name is configured in the radius-config or ldap-configs definition.**
  - Use the same tenant name to authenticate users.
  - If you do not include a tenant name, the system defaults to the **master** tenant.

Data Synchronization and Authentication Order

The data synchronization procedure:

- 1. User logs in to Blue Planet with RADIUS or LDAP-enabled password.
- 2. Blue Planet updates the user and provides a 24-hour token.
- 3. If the user is deleted from RADIUS/LDAP server, changes are reflected in Blue Planet on the next login.
  - The current token is valid until expiration.

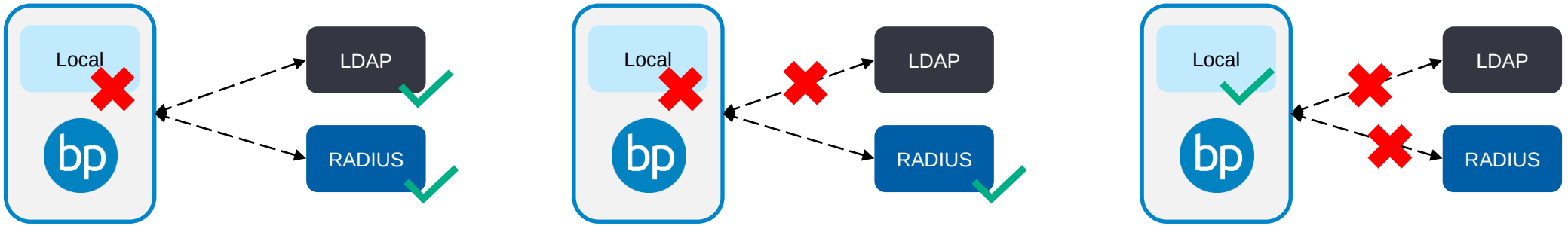
Authentication order:

- The order of authentication is fixed:
  - LDAP Primary > LDAP Secondary > RADIUS Primary > RADIUS Secondary > TACACS Primary > TACACS Secondary > Local database (Blue Planet).

Fallback to Local Authentication

Fallback is an optional setting when all external authentication (LDAP/RADIUS/TACACS) communication fails.

- If set, all local users will authenticate locally upon all remote authentication failure.
- **fallback_to_local_auth** parameter is configured at the tenant level.
- If remote servers are **reachable** and actively reject a user, this does **not** initiate fallback. Be careful not to lock yourself out of BP.



Caution: when using two or more remote servers, fallback must be enabled for admin/sysadmin to be able to login locally.

Summary

Introduction to Blue Planet Security

In this section, you learned about the security aspects of Blue Planet products.

- One of the Blue Planet security objectives is to build a security process that needs to be followed across product lines.
- Blue Planet Platform microservice architecture simplifies security maintenance.
- Blue Planet products use a pro-active approach to protection from security vulnerabilities and denial of service attacks.
- Blue Planet users can be assigned a set of predefined roles which define role-based access control (RBAC) permissions. Additional roles with custom permissions can be created.
- In REST, clients exchange credentials for a token. The token is then used as an authentication credential for subsequent REST calls.

User Access Control (UAC)

Objectives

- Describe Role-Based Access Control
- Compare Permissions, Roles, and User Accounts
- Describe Tron
- Explore Tron APIs
- Describe Swagger and the Swagger UI
- Examine the bulk users' creation process through REST API

User Access Control (UAC)

Agenda

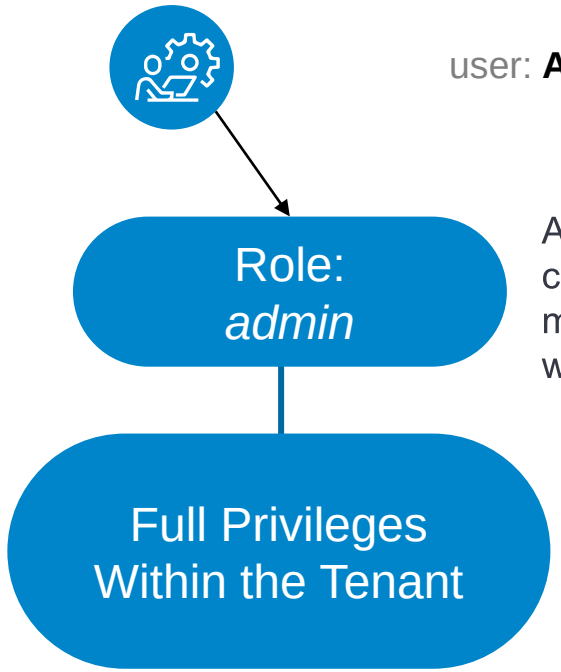
1. **Role-Based Access Control**
2. Tron
3. Swagger UI

Role-Based Access Control (RBAC)

- **Blue Planet uses RBAC to allow administrators to set user permissions, roles, and tenants.**
- **Each role has specific permissions attached to it, so the permissions are evaluated to determine whether an action on a resource is permitted.**
- **In addition to the User Administration app that manages user and tenant tasks, each app sets its own roles for task completion.**
- **Each user should have both a Security application role and a Blue Planet application role. For example, (admin, Application admin) or (user, Provisioner).**

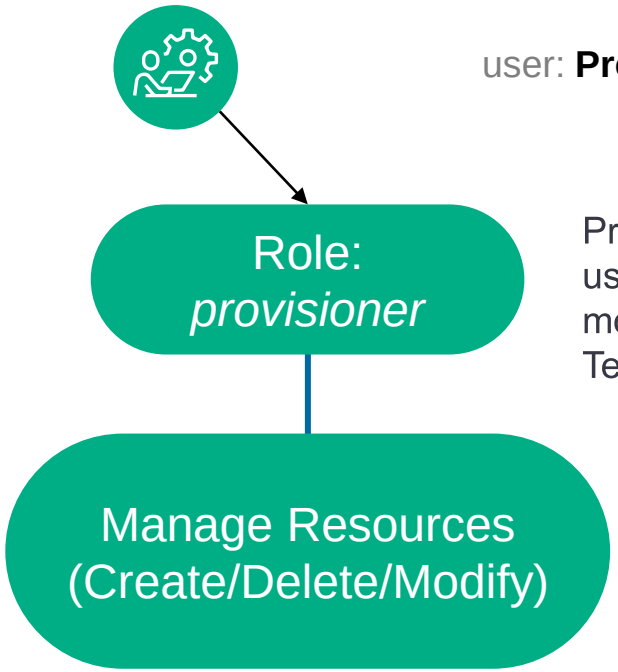
Roles

- **Each user within Blue Planet will be assigned one or more roles.**
- **Each Role has a set of specific permissions which control specific administrative and resource management actions that a user can perform.**



user: **Admin**

Admin role allows the user to create accounts, create domains, manage resources, and so on within the Tenant.



user: **Prov**

Provisioner role allows the user to create, delete, and modify resources within the Tenant.

Pre-defined Roles

The following default roles are provided as part of Blue Planet systems to assist with establishing main user roles to access different areas of the solution.

You cannot modify these default roles but you can define custom security roles.

- **Application admin:** This includes administration of Orchestrate, application-level security, and basic visibility of network elements and services.
- **Observer:** Read-only access with basic visibility of network elements and services.
- **Provisioner:** Create, modify, or delete resources. Cannot create or delete domains and products.
- **sysadmin:** Master tenant sysadmin can create or deactivate a tenant. It works with the Application Admin role.
- **admin:** All privileges except create tenants, deactivate tenants, create roles, and delete roles.
- **user:** Privileges on an identified tenant, can change user password.

Role Privileges within Blue Planet

TASK/ACTION	USER	ADMIN	SYSADMIN
Modify email address and password (self)	x	x	x
Modify API keys	x	x	x
Modify others' username, email address, or password		x	x
Create a new user		x	x
Enable (or disable) a user		x	x
Create roles			x (with Application admin role)
Assign or modify user roles		x	x
Create and delete API key assignments		x	x
Assign or modify user roles		x	x
Create, edit, and deactivate tenants or sub-tenants			x
Edit password configuration policy		x	x

User Application Role Examples

APPLICATION	ROLE	DESCRIPTION
Security	admin	All privileges for the user (except operations on sysadmin user).
Security	sysadmin	All privileges for UAC operations (It works along with the Application admin role only).
Security	user	Privileges on the identified tenant; can change user password.
Alerts Performance Monitor Policy Manager	watcher-admin	Resource discrepancy notifications for events, alerts, and polling.
Blue Planet Orchestration	Application admin	Author all content. This includes administration of Orchestrate, application-level security, and basic visibility of network resources and services.
Blue Planet Orchestration	SMGR admin	Slice Manager administrator. This includes privileges to create, delete, and modify subdomains, slices, and service specifications.
Blue Planet Orchestration	Provisioner	Create, modify, or delete resources. Cannot create or delete domains and products.
Blue Planet Orchestration	Observer	Read only access.

Viewing Permissions

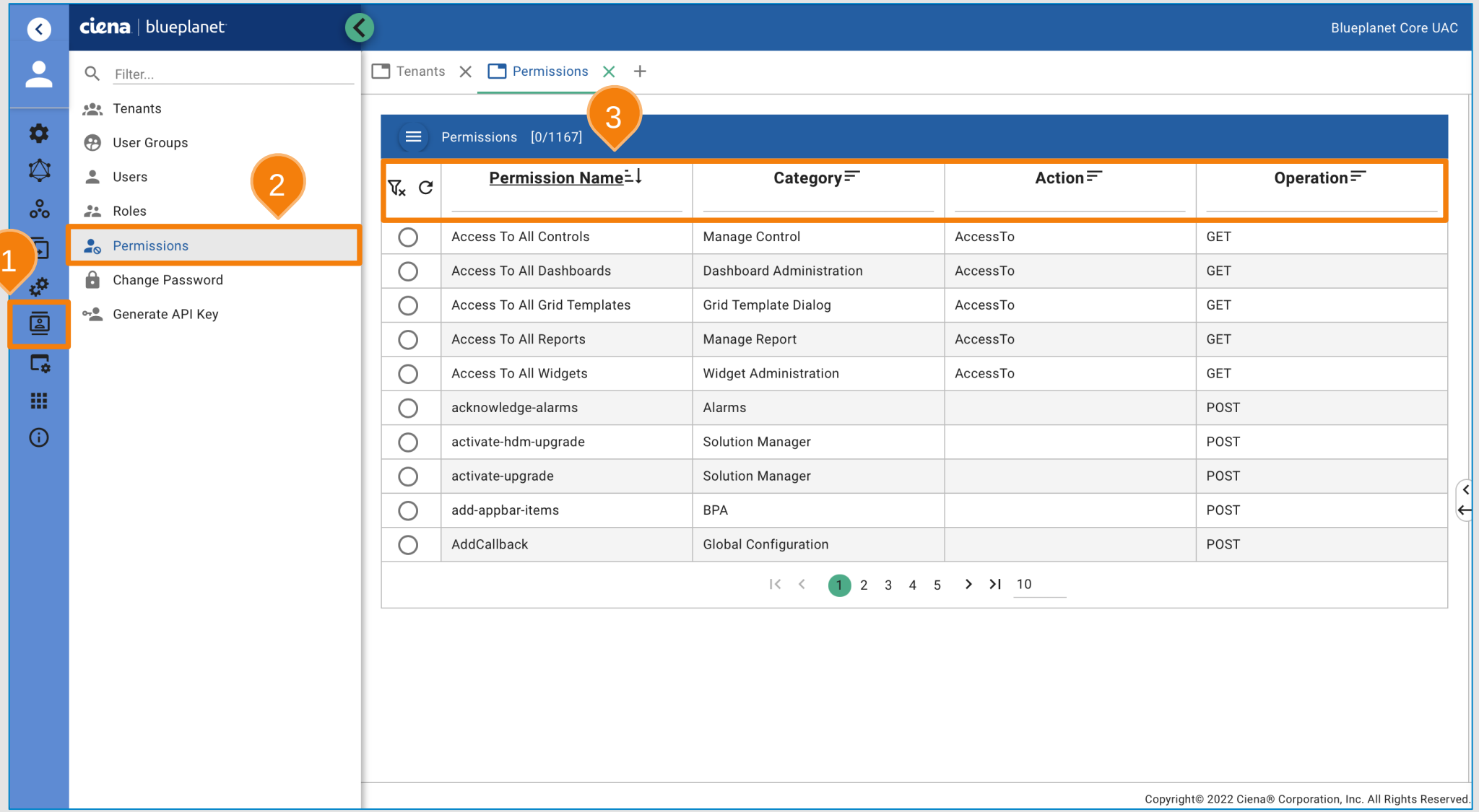
Access permissions configuration from the App Bar UI, click User Access Control > Permissions.

Each permission includes these fields:

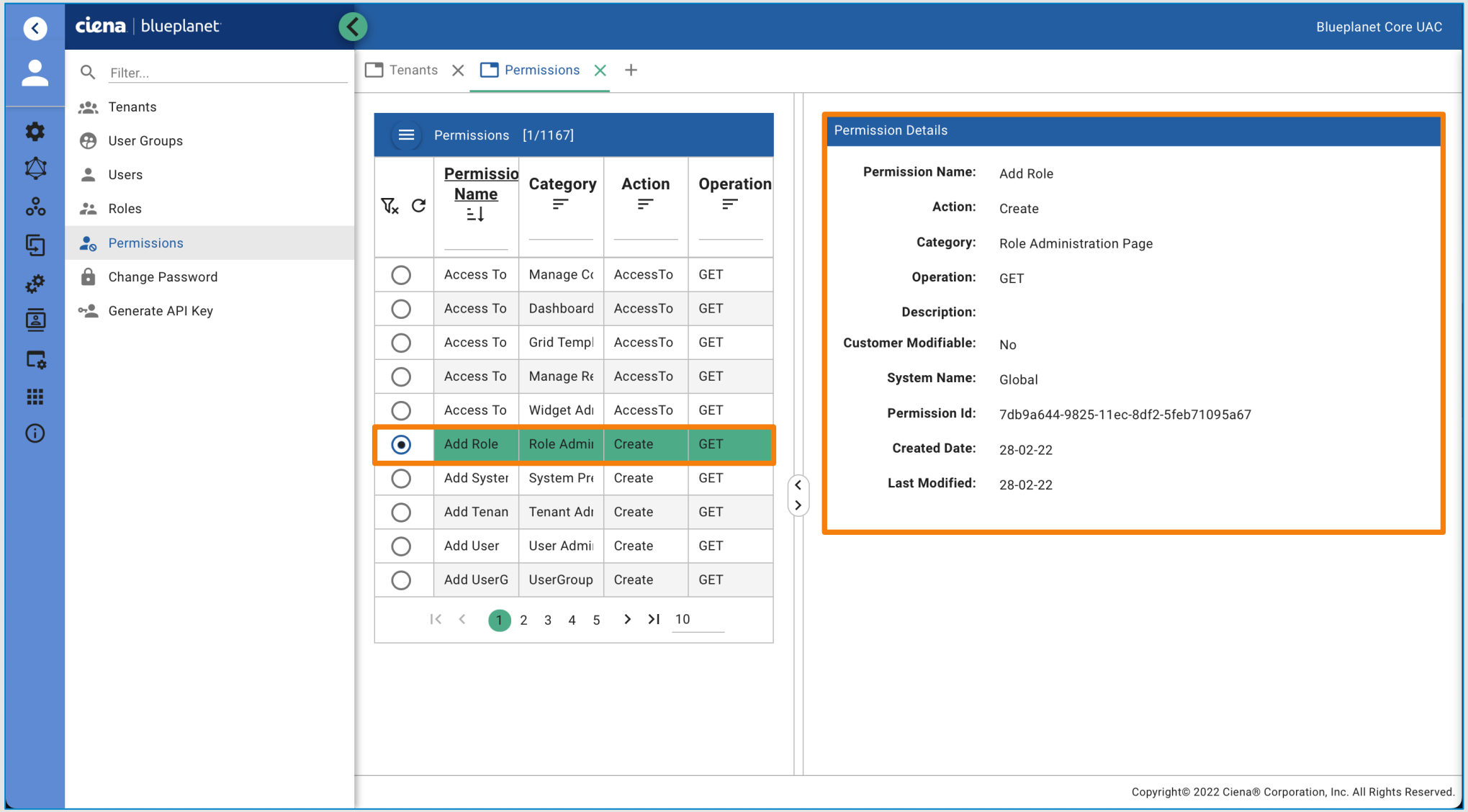
- Name
- Category
- Action
- Operation
- Endpoints (in expanded view)

Note: *You can filter permissions in the fields row.*

Viewing Permissions



Viewing Permission Details



Creating a Custom Role

Due to a large number of permissions, it is always recommended to:

- Assign a default role as a base role for a user, and then create a custom role with additional permissions and assign also the custom role to the user.

- Clone a default role, and then add or remove permissions. To clone a role, see the Editing a role section.

IMPORTANT:

- When assigning a custom role to a user with security admin privilege, you must assign the additional role of sysadmin, if this user will be performing security operations such as creating a user account and creating a role.

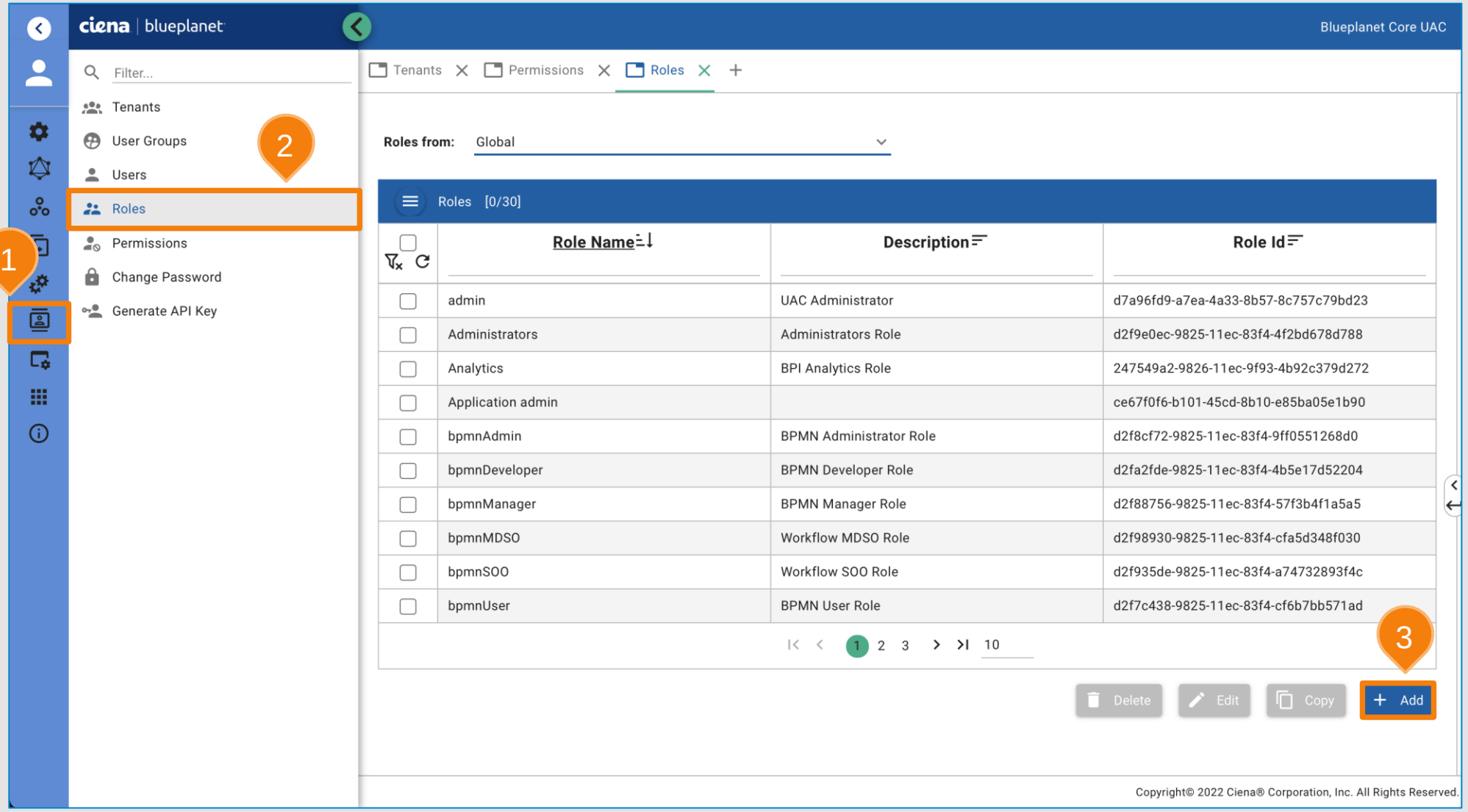
- Removing permission can impact access to Blue Planet products. For example, removing the view alarms permission will prevent the user from accessing alarms.

Creating a Custom Role

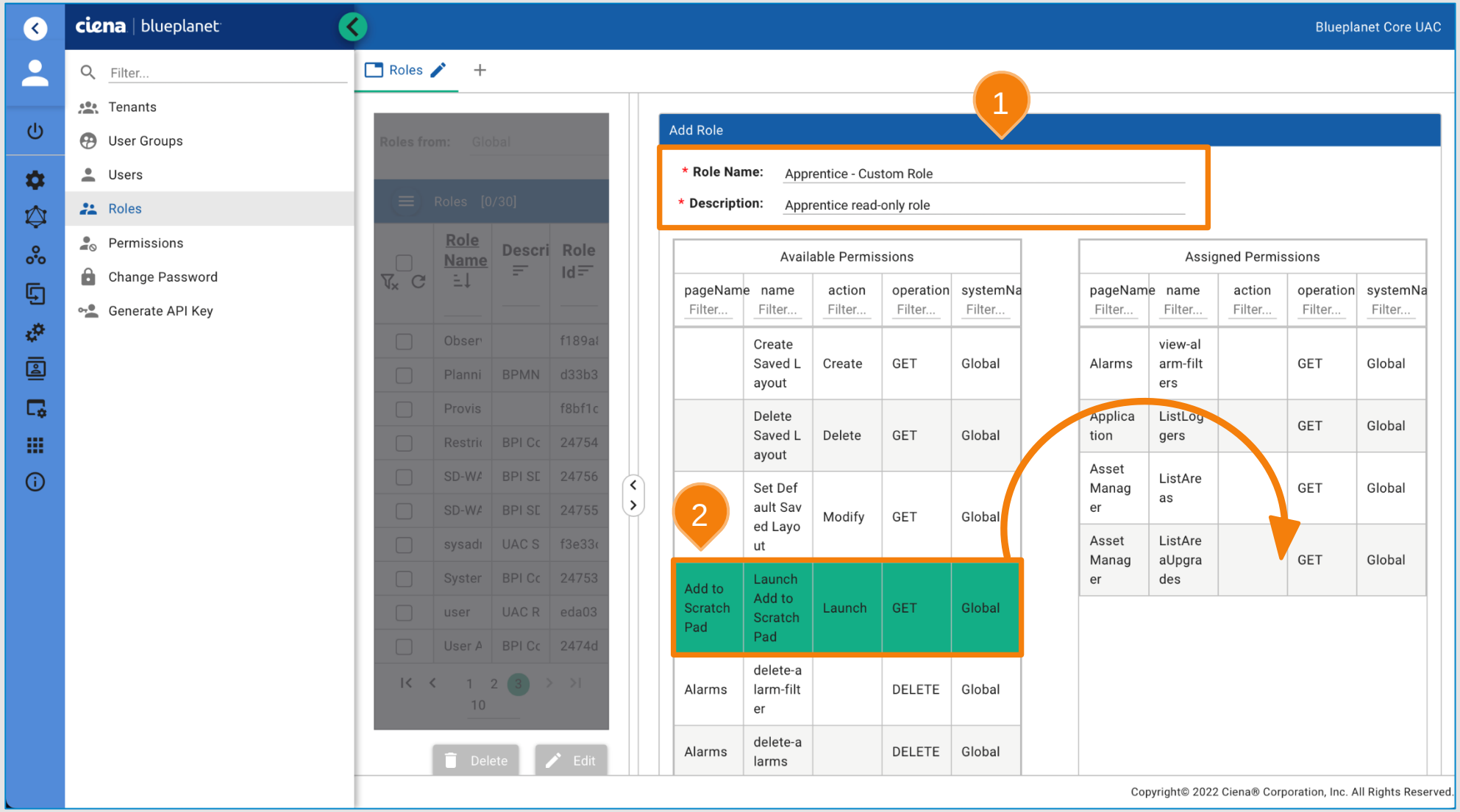
To create a role:

- Log in to the Blue Planet Platform as a user with the **Application admin** and **User Access Control sysadmin** roles.
- From the **App Bar UI**, navigate to **User Access Control > Roles**. The Roles page appears with a list of existing roles.
- Click **+ Add**.
- The **Add Role** dialog box opens.

Creating a Custom Role



Creating a Custom Role



Managing User Accounts

Blue Planet systems come preconfigured with the admin user that is assigned all default roles automatically:

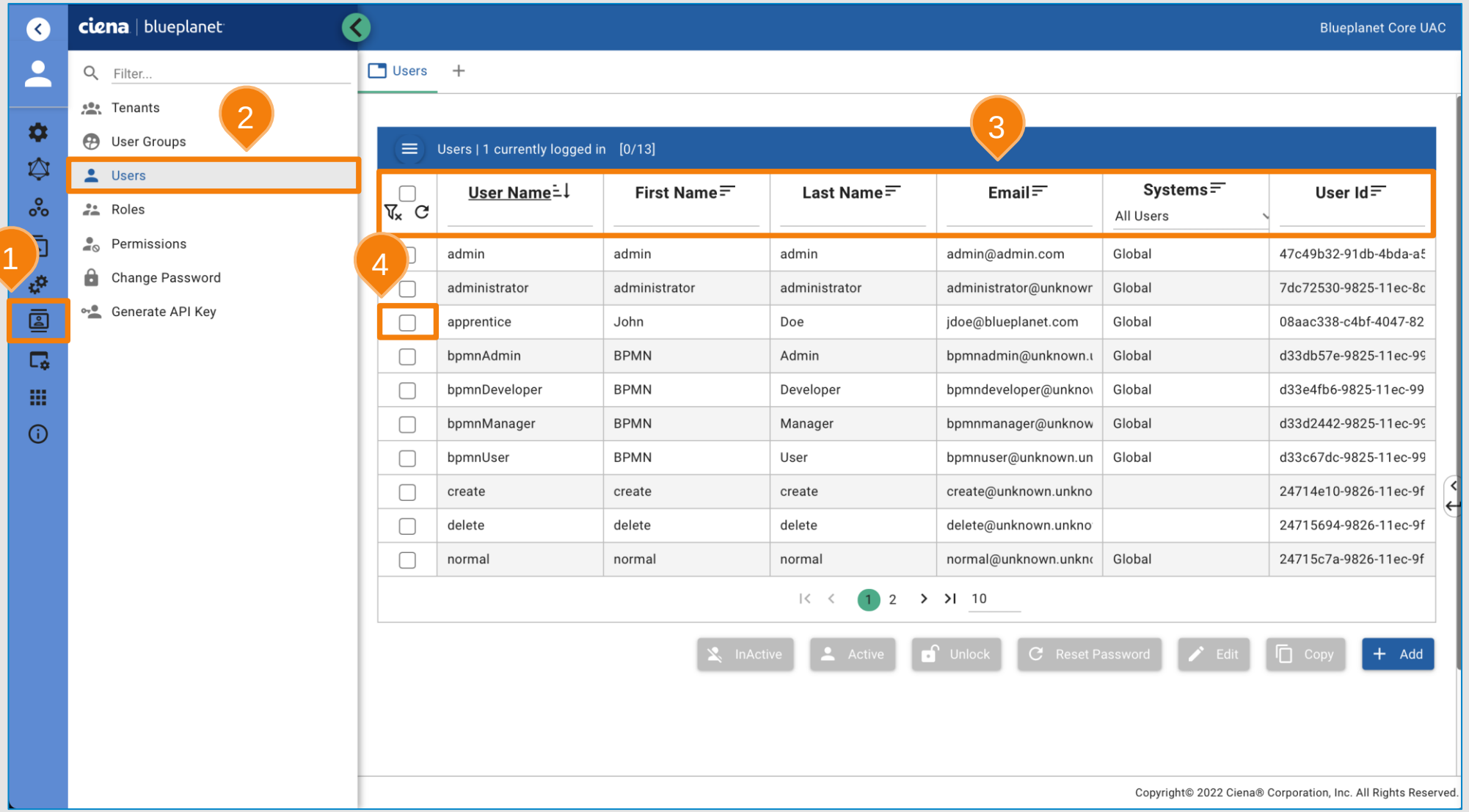
Application admin	This includes administration of Orchestrate, application-level security, and basic visibility of network elements and services.	
Observer	Read-only access with basic visibility of network elements and services.	
Provisioner	Create, modify, or delete resources. Cannot create or delete domains and products.	
sysadmin	Master tenant sysadmin can create or deactivate a tenant. It works with Application Admin role.	
admin	All privileges except create tenants, deactivate tenants, create roles, and delete roles.	
user	Privileges on identified tenant, can change user password.	

Admin users with Application admin and sysadmin roles can also create new users and assign roles as required.

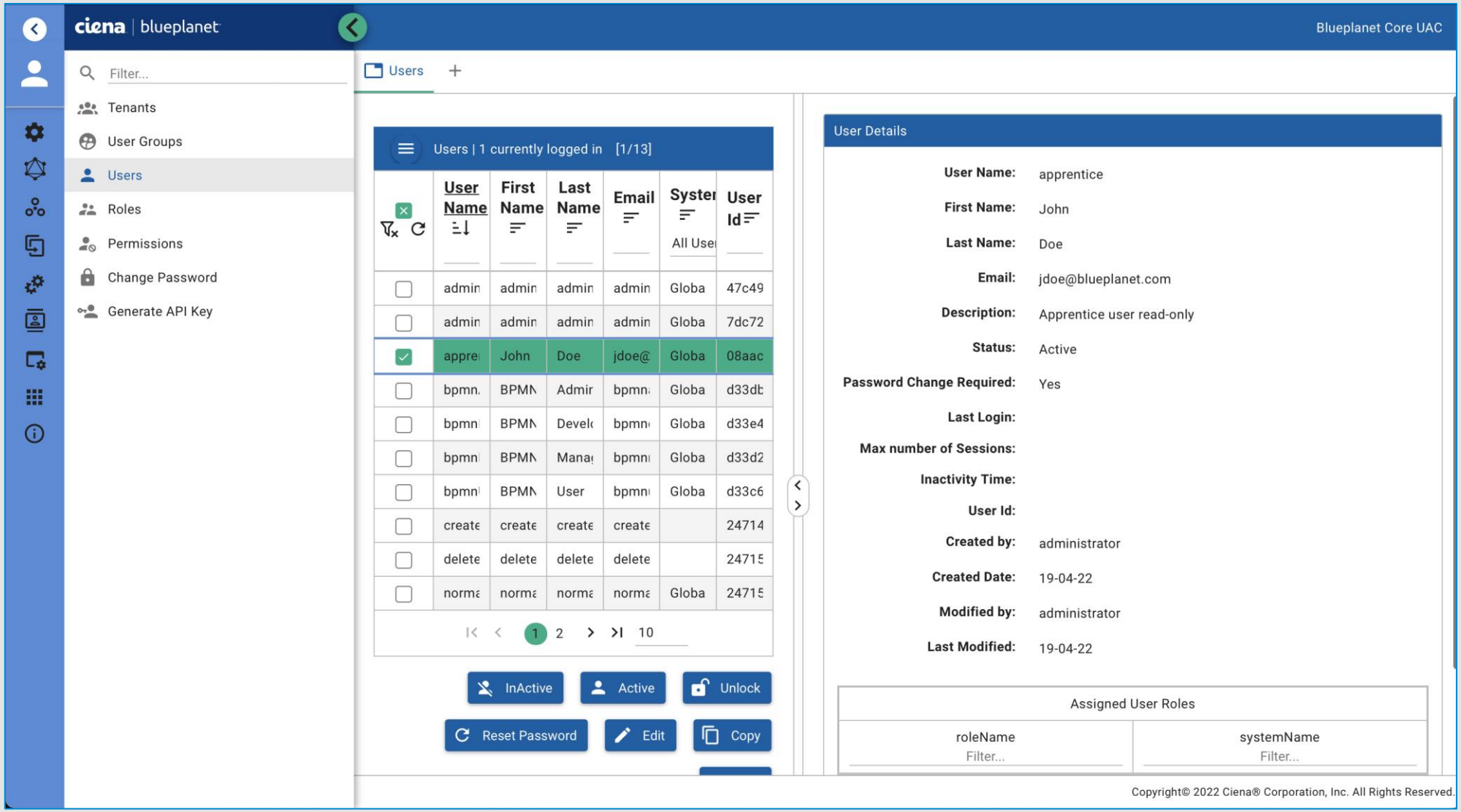
User Configuration

- **Blue Planet installs a single administrative user called admin that belongs to a master tenant within the User Access Control (UAC) app.**
- **The admin user then sets up multiple users as well as multiple tenants.**
- **Multi-tenancy permits the creation of separate customer tenants who can then set up associated users with access to only selected apps and resources.**

Viewing User Accounts



Viewing User Accounts – User Details

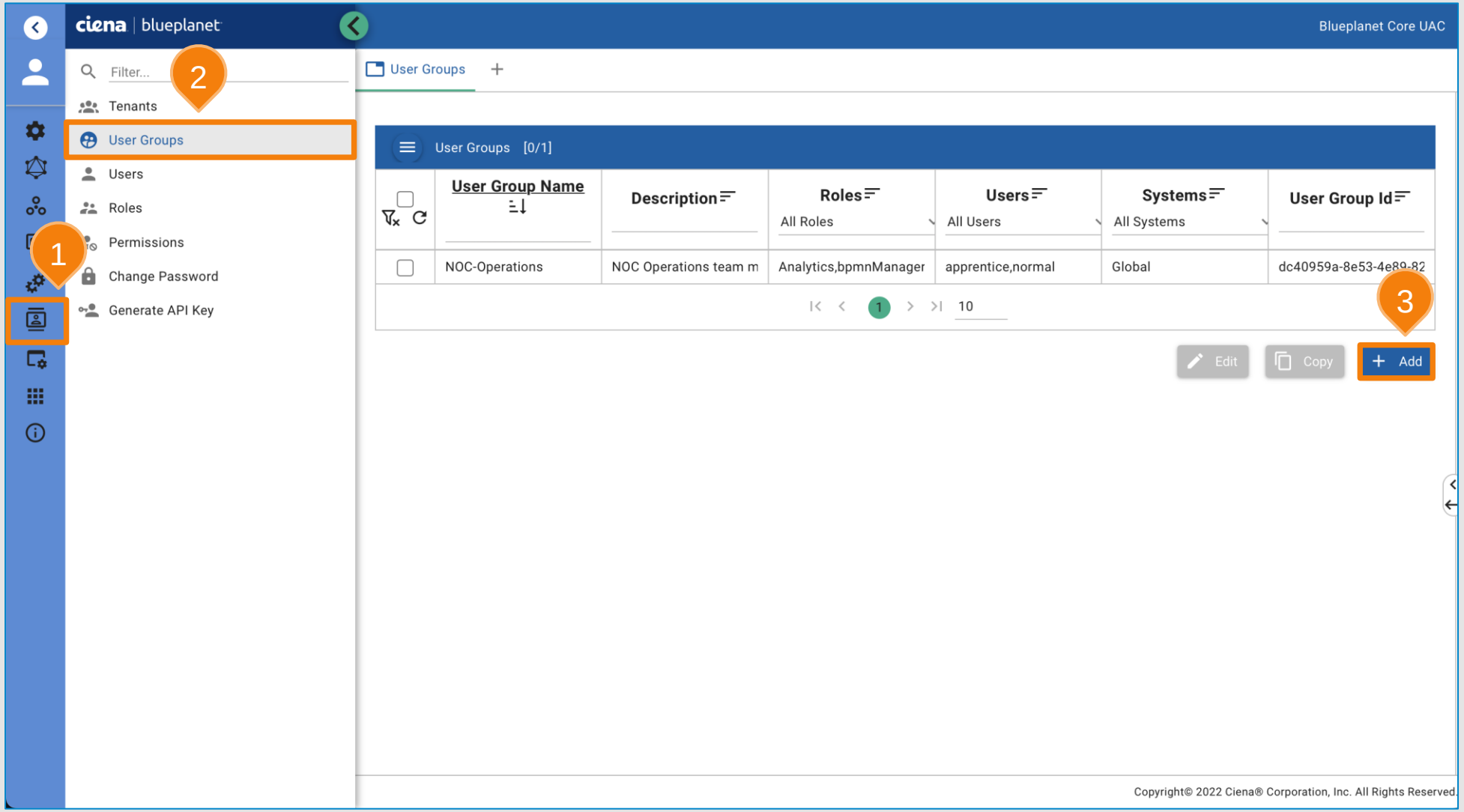


Managing User Groups

- **User groups are a set of users that are assigned roles to access specific resources and applications.**
- **You can assign users to groups using the User Group functionality.**
- **This allows you to assign similar roles to multiple users.**
- **You can also perform operations such as adding user groups, copying user groups, and editing user groups on the User Groups page.**

- **Examples of user groups: NOC Operations team members, Alarm viewers, Configuration managers**

Creating a User Group



User Access Control (UAC)

Agenda

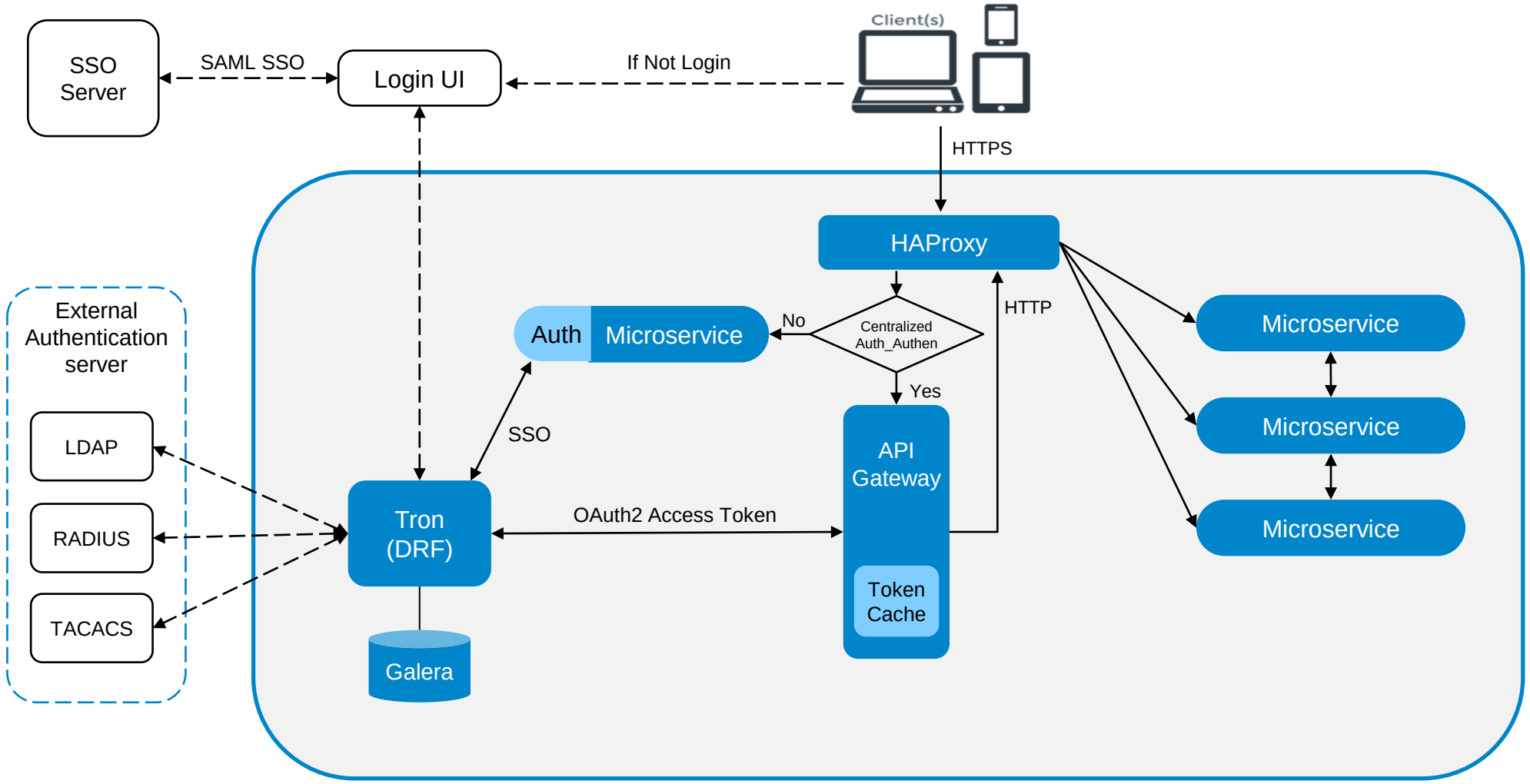
1. Role-Based Access Control
2. **Tron**
3. Swagger UI

What Is Tron?

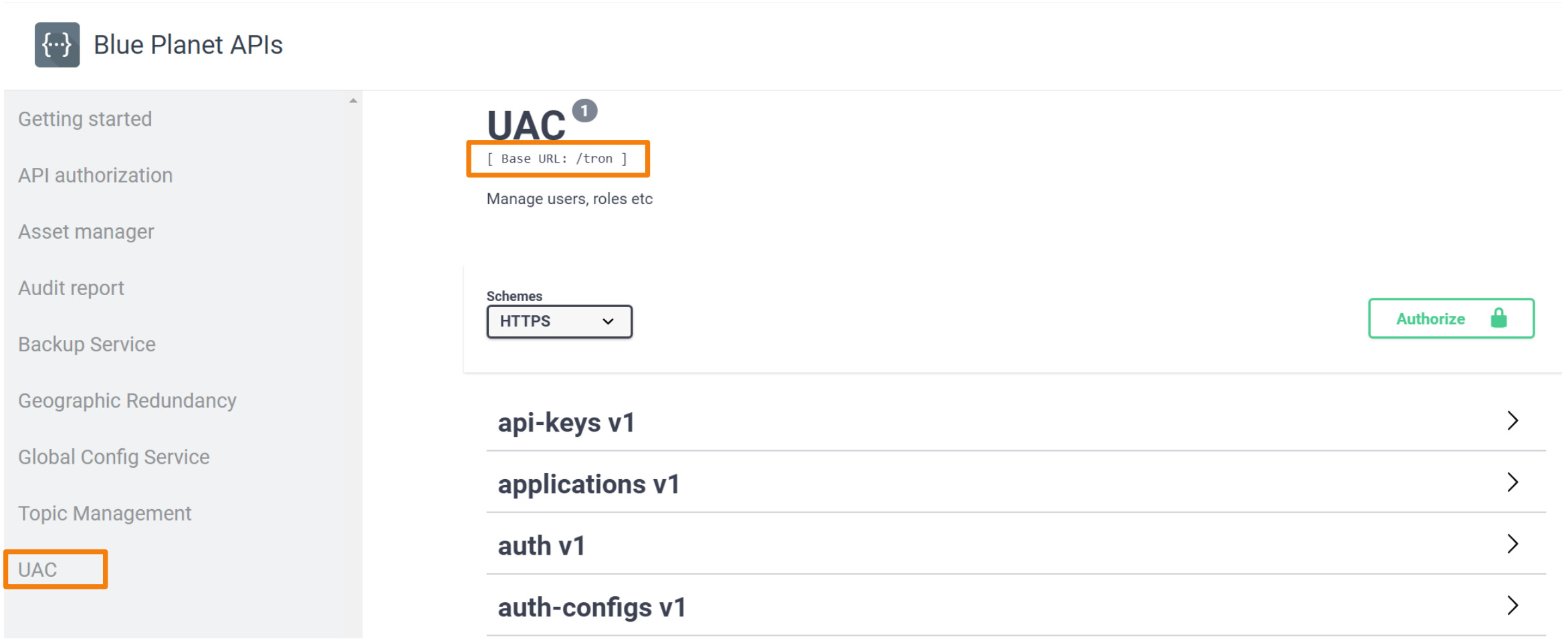
- **Tron is the centralized User Access Control service for Blue Planet.**
- **It is a Django REST Framework-based web application backend that provides an HTTP API.**
- **Tron takes advantage of the Django REST framework to perform Authentication and Authorization with its custom implementation.**

- **In the next slides we will cover Tron Authentication, Authorization, and how User access management is done using tenants, subtenants, users, user groups, roles, and permission.**

Blue Planet Role Based Access Control (RBAC)



Tron Swagger UI



User Access Control (UAC)

Agenda

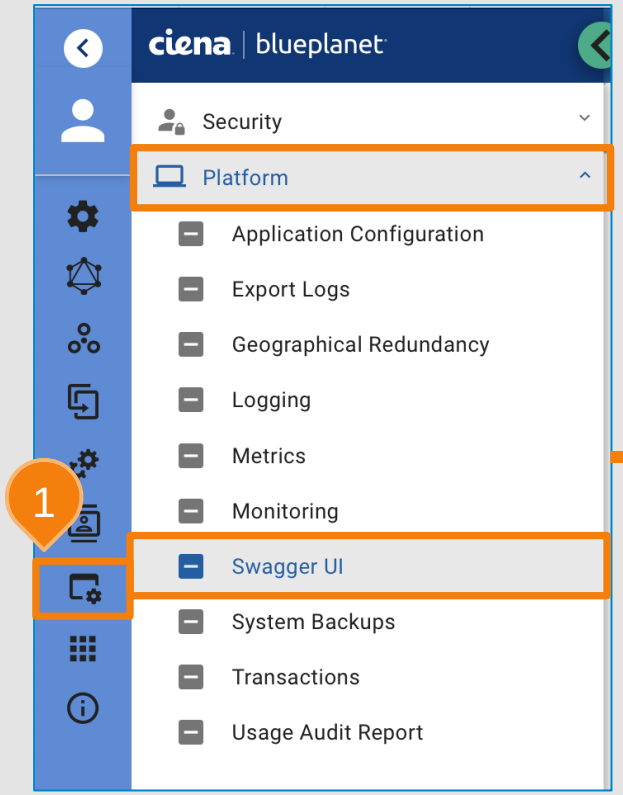
1. Role-Based Access Control
2. Tron
3. **Swagger UI**

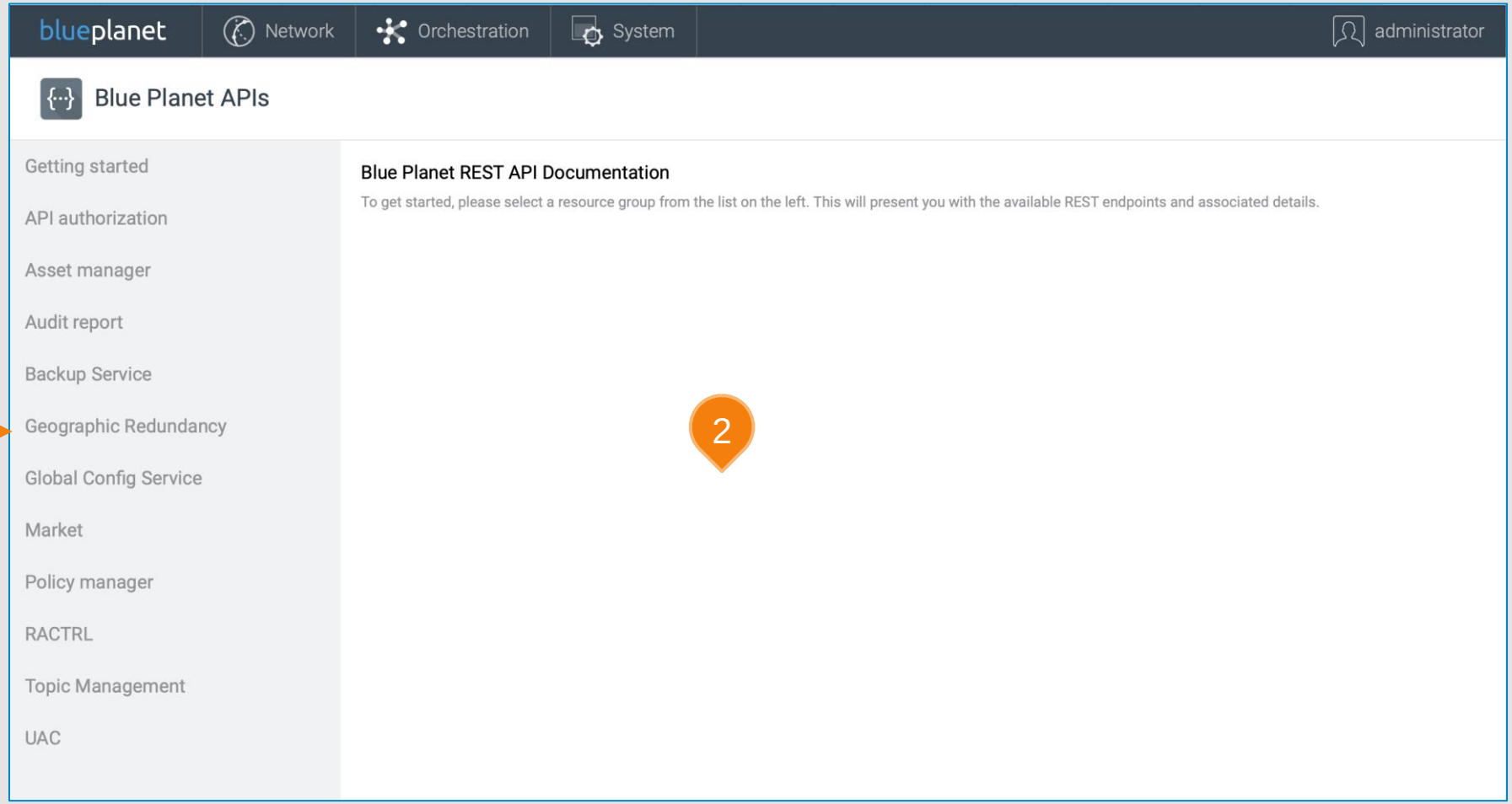
Swagger UI

- **If you are a client-side developer interested in integrating additional functionality using the open REST APIs, you can use the Swagger UI to view the list of resources for the Blue Planet Platform components and installed products.**
- **Your list of available components in the REST API depends on your installed solutions.**
- **The Blue Planet Swagger API documentation is available from the Blue Planet dashboard:**
  - Includes online specifications for available Blue Planet components and their resources.
- **To access the interactive APIs from the dashboard go to App Bar UI > System > Platform > Swagger UI.**

***For additional developer support, visit our online developer community* at *developer.blueplanet.com*.**

Swagger UI





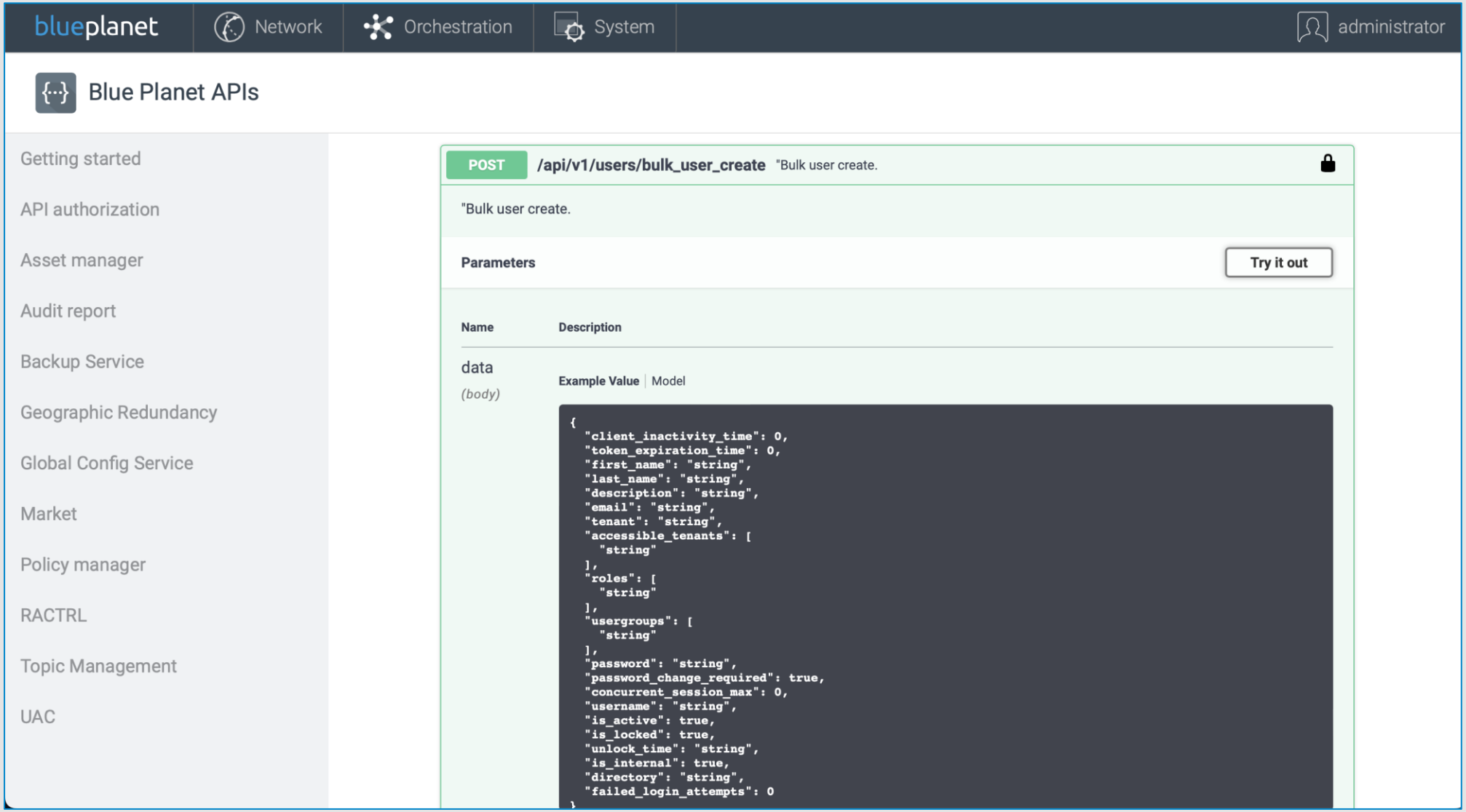
Creating Bulk Users Using the API and Swagger UI

To create multiple users:

- Log in to the Blue Planet Platform as a user having **security admin** or **sysadmin** privileges.
- From the **App Bar UI**, navigate to **System > Platform**.
- Select **Swagger UI**.
- The **Blue Planet APIs** page appears.
- Select **UAC > users v1 > POST /api/v1/users/bulk_user_create**.
- Click **Try it out** and provide attributes.
- Click **Execute**.

NOTE: *You can copy the code with the results.*

Creating Bulk Users Using the API



Summary

User Access Control

In this section, you have explored UAC on the Blue Planet Platform.

- Blue Planet uses RBAC (Role Based Access Control) to allow administrators to set user roles and permissions.
- Each user within Blue Planet will be assigned one or more roles and each Role has a set of specific permissions which control administrative and resource management actions that a user can perform.
- Creating and editing roles and permissions is done from the App Bar UI UAC menu.
- Tron is the centralized User Access Control service for Blue Planet.
- Tron takes advantage of the Django REST framework to perform Authentication and Authorization with its custom implementation.
- Swagger UI is available within the Blue Planet Platform and it is used to view the list of available API operations or when functionalities need to be integrated using Open REST APIs.

Lab 1: User Access Control (UAC)

In this Lab you will learn how to:

- Log in to BP master Tenant.
- Create a new Tenant.
- Create a Custom Role.
- Log in to a customer Tenant.
- Modify User information and change password.
- Create a new User and assign Roles.
- Use REST API and Swagger UI to create bulk users.
- Verify User Privileges.

blueplanet®

a division of Ciena

blueplanet®
a division of Ciena

Operations and Maintenance

Introduction to the Blue Planet Platform

PLF111ILT-A, Revision 1.0

Kubernetes Overview

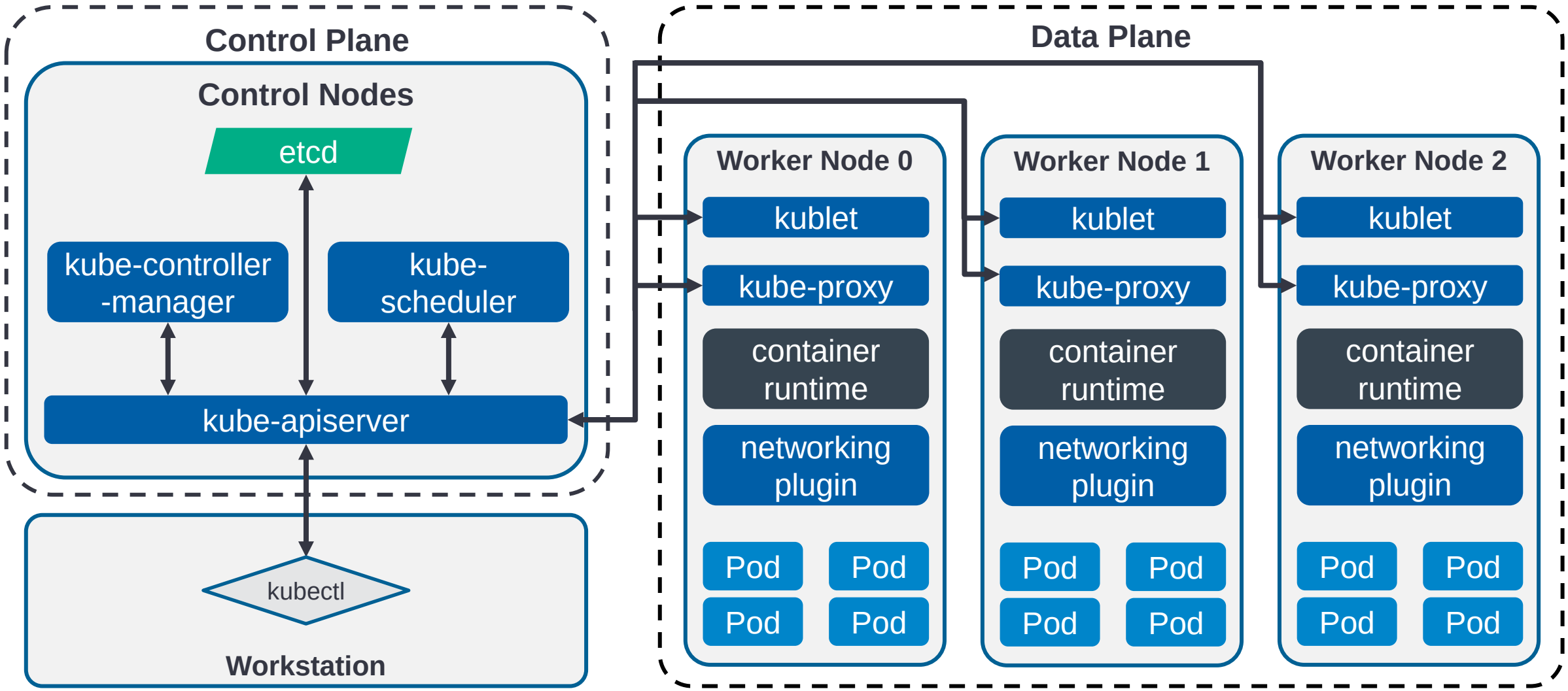
Objectives

- Describe Kubernetes architecture
- Distinguish Control and Data Plane functions
- Demonstrate K8s API interaction
- Examine YAML files
- Examine bphub Registry
- Explore Helm Charts

Agenda

1. **Kubernetes Architecture and Operations**
2. Software Delivery and Helm

K8s Architecture



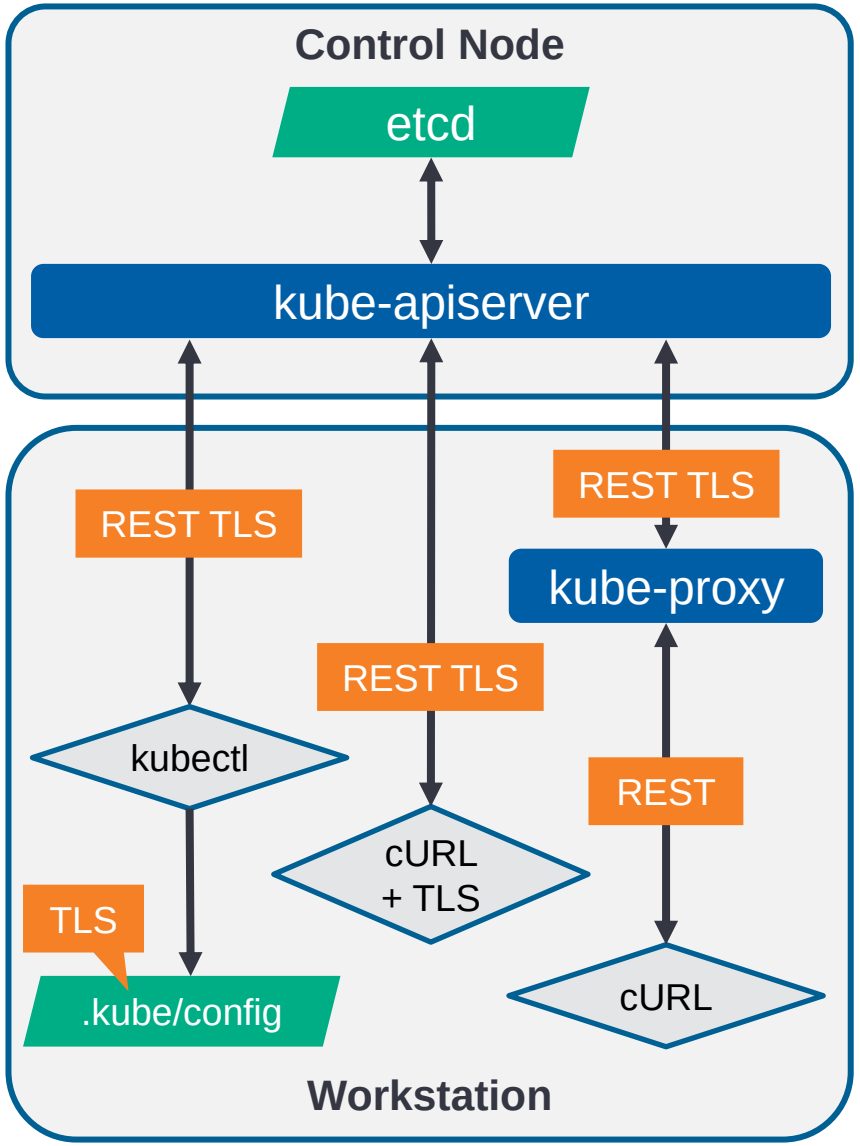
Interacting with kube-apiserver

kubectl is the default command to interface with K8s API.

- Install the **bash-completion,** to enable the **kubectl** auto-completion, execute **kubectl completion bash >> ~/.bashrc**.
- The configuration is stored in **~/.kube/config**.
- **kubectl -h** to get the list of available commands.
- **kubectl config -h** to get the list of available command options.
- **kubectl config view** to view the current configuration.
- **kubectl cluster-info** to get the information about the K8s cluster.
- **kubectl get nodes** to get the information about control and worker nodes.

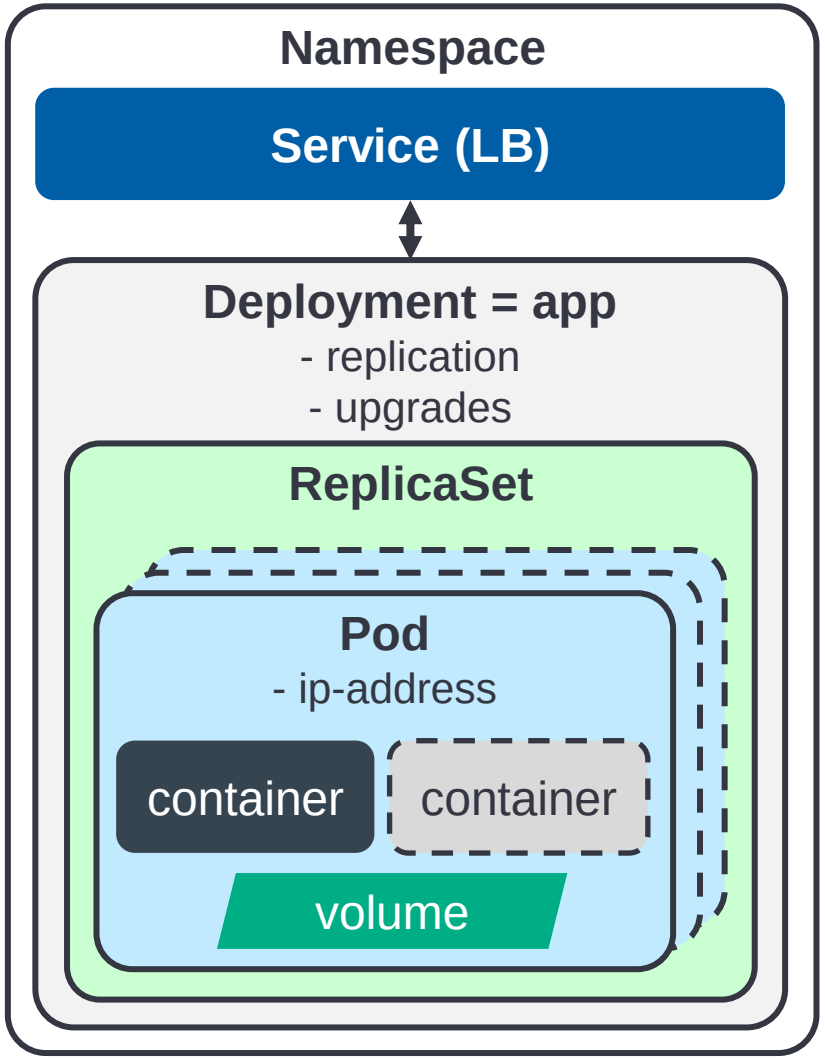
cURL with TLS can be used for the communication with K8s api-server.

Alternatively, you can use the kube-proxy which will accept the unencrypted communication and take care of the encrypted part.



K8s Basic API Object

- **Namespace (ns): Linux kernel feature.**
  - **default:** K8s resources are created in it by default.
  - **kube-system:** contains all K8s infrastructure pods.
- **Pod: the minimal object which is managed by K8s.**
  - Describes the IP address.
- **Container: one or more containers (typically one).**
  - Managed by the container runtime engine.
- **Volume: temporal storage for the container.**
  - Permanent storage is realized by **persistent volume (PV)** and requested by **persistent volume claim (PVC)**.
- **Deployment: equal to the application (app).**
  - Replication is realized by **replicaset**.
  - Defines upgrades.
  - **Pod (template)** is a part of the deployment definition.
- **ReplicaSet (rs): creates a defined number of pods in the Deployment.**
- **Service: enables external communication to the Pods.**
  - Acts as a load balancer (LB).



Documentation

- https://kubernetes.io/docs/home **official Kubernetes documentation.**
- https://kubernetes.io/docs/reference/generated/kubectl/kubectl-commands **kubectl reference documentation.**
- **kubectl explain pods**
  - Get the documentation of the resource and its fields.
- **kubectl explain pods.spec.containers**
  - Get the documentation of a specific field of resources.

Agenda

1 Kubernetes Architecture and Operations

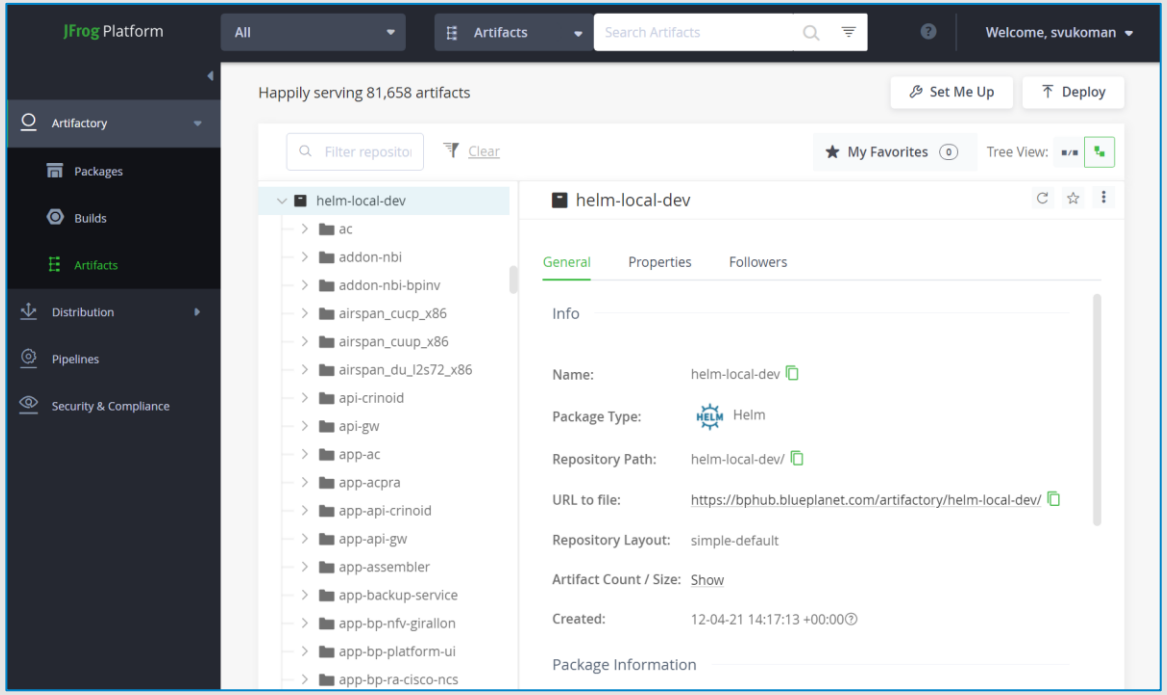
2 **Software Delivery and Helm**

The bphub Registry

https://bphub.blueplanet.com

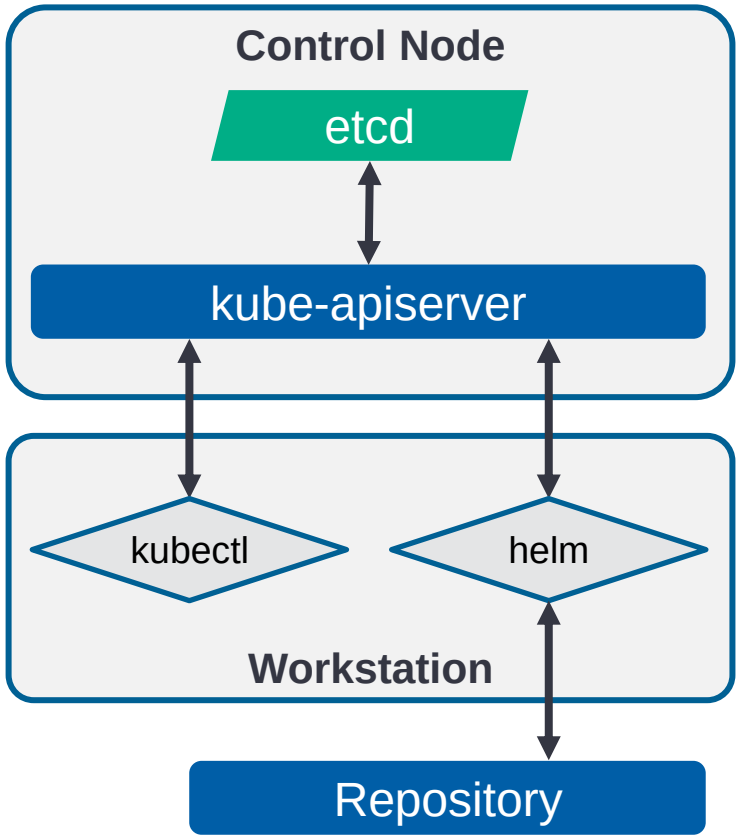
- Public-facing registry for software delivery.
- Hosts Docker images, installer scripts, Helm charts.

- **Previously, solutions were bundled in a tar file and distributed through the *my.ciena.com* portal.**
- **With bphub, customers can download and deploy just what they need for their solution, without creating and transferring large packages.**
- **Customers can receive specific logins and API Tokens.**
- **Customer specific locations are accessible by Ciena and that specific customer.**



Helm v3 Introduction

- **Helm v3 (client-only) is a package manager for K8s.**
- **Chart is a Helm package in a .tar archive and it bundles several manifests (Deployments, Services, ..) for a K8s application.**
- **Kubernetes applications have several manifests (Deployments, Services, ..), with Helm they can be bundled in a Helm Chart.**
- **In a Repository, Charts can be collected and shared:**
  - Public Repositories
  - Private Repositories (Blue Planet)
- **Release is an instance of a Chart running in K8s.**
- **helm is a Helm client to interact with repositories and work with the Charts.**

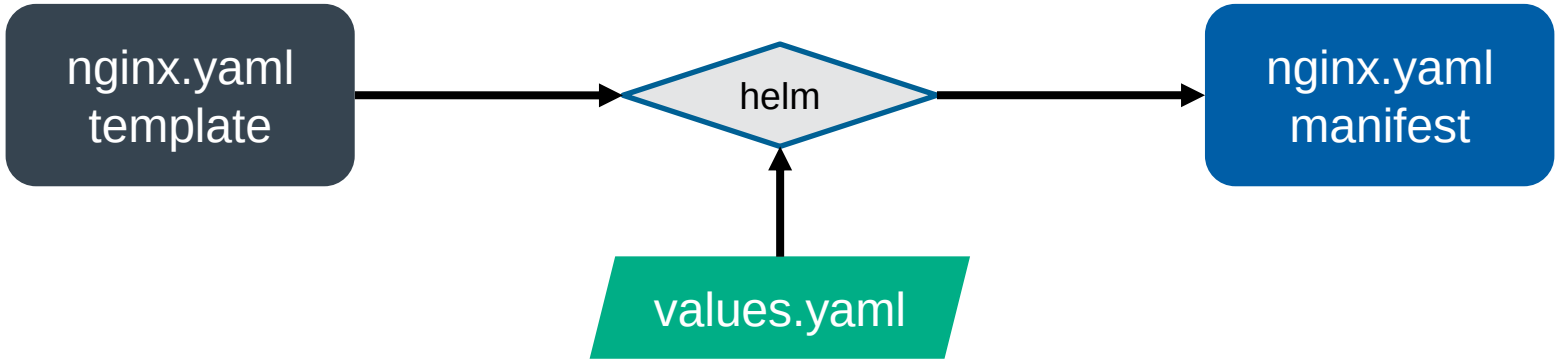


Helm Charts

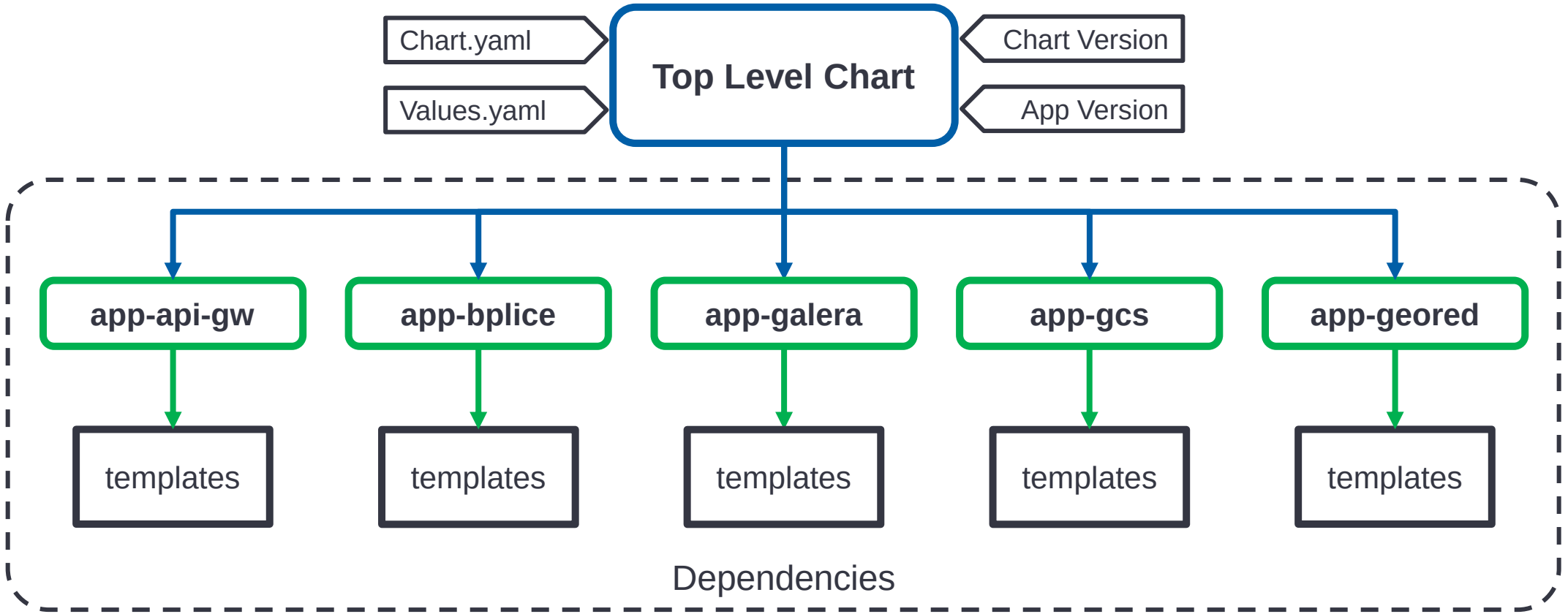
- **Chart.yaml: chart metadata.**
- **values.yaml: key/value pairs used in the template.**
  - Values object stores the key/value pairs.
- **charts: folder for chart dependencies (multi-level nesting).**
- **templates: parametrized YAML manifests to define the application.**
  - **NOTES.txt:** it will be displayed when you **run helm install**.
  - **deployment.yaml:** K8s manifest for deployment.
  - **service.yaml:** K8s manifest for the service endpoint.
  - **_helpers.tpl:** re-usable template helpers.

Chart folder structure

```
nginx/
  Chart.yaml
  values.yaml
  charts/
  templates
    NOTES.txt
    deployment.yaml
    service.yaml
    _helpers.tpl
```



Helm Chart Hierarchy



Summary

Kubernetes Overview

In this section, you learned about basic Kubernetes Architecture and the software deployment process.

- Kubernetes architecture consists of Control Plane and Data Plane functions.
- Kubectl is the default command to interface with K8s API while cURL can be also used for communication with K8s api-server.
- With bphub, customers can download and deploy just what they need for their solution, without creating and transferring large packages.
- Helm is a package manager for K8s. Kubernetes applications have several manifests and with Helm, they can be bundled in a Helm Chart.

Blue Planet Installer Overview

Objectives

- Discover Blue Planet Installer (bpi)
- Examine the Blue Planet Software installation steps
- Explain bpi options
- Demonstrate a bpi example

blueplanet
a division of Ciena

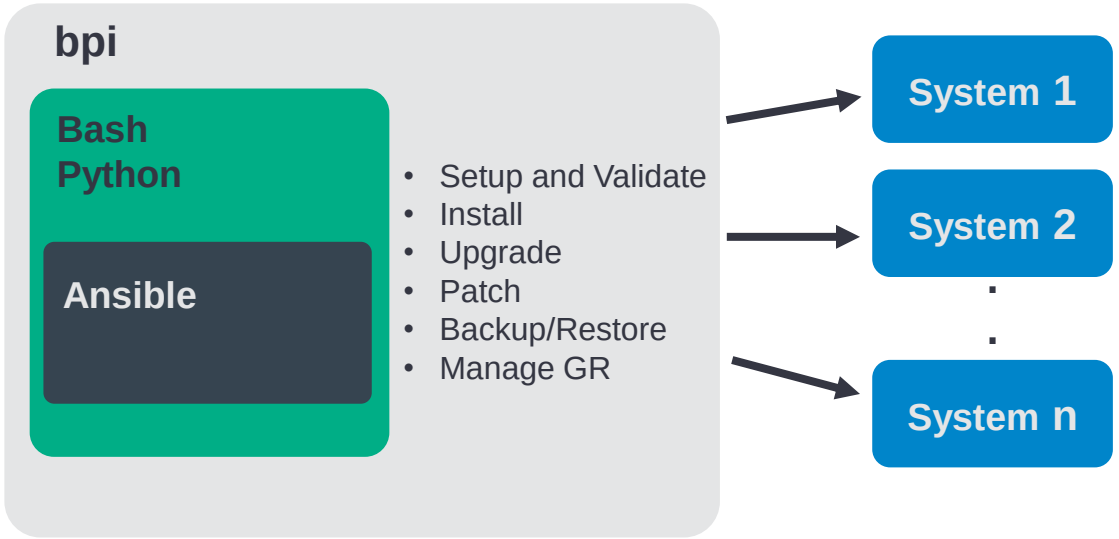
Blue Planet Installer Overview

Agenda

1. **Blue Planet Installer Overview**
2. Blue Planet Software Installation Steps

Blue Planet Installer Overview

- **The Blue Planet Installer (bpi) is an Ansible-based tool, used to:**
  - Evaluate if the target system fulfills the requirements and is capable of running Blue Planet software.
  - Download, install, upgrade, and patch Blue Planet software on one or more target systems.
  - Perform site and solution backup and restore.
  - Manage Geo-redundancy between target systems.
- **bpi is Python plus Bash script that calls Ansible.**
  - Executed as a shell script.
  - Supports custom playbooks.



Initializing the Blue Planet Installer

- **bpi is obtained as part of the Blue Planet software download process, or from the Blue Planet artifact repository.**
- **bpi.sh is a single binary file, which is executed and self-extracted.**
  `bpi-<release number>-<version number>.sh`
- **./bpi directory is created, which contains:**
  - bpi Python script and other shell scripts.
  - Ansible hosts file, roles, playbooks, and the group_vars file.
  - A Python virtual environment.
  - bpi log files.
- **Different versions of the tool are available for bp2, K8s, on-prem, or cloud deployments.**
  - The example shown is bpi for AWS EKS deployments.
  - k8s-installer for on prem and eks-installer for AWS EKS deployments available.

```
./bpi
├── ansible.cfg
├── bp-aws-install.sh
├── bpi
├── bp-uninstall.sh
├── cleanup.sh
├── hosts
├── LICENSE.md
├── logs
│   ├── history.log
│   └── setup-2022-05-31.log
├── playbooks
│   ├── all_yml_files.txt
│   ├── eks-reset.yml
│   ├── eks-setup.yml
│   ├── group_vars
│   ├── roles
│   └── terraform
├── setup.sh
├── venv
│   ├── bin
│   ├── lib
│   ├── lib64
│   └── pyvenv.cfg
└── workspace
```

Agenda

1 Blue Planet Installer Overview

2 **Blue Planet Software Installation Steps**

Blue Planet Software Installation Steps - High Level Overview

- **Update the hosts file.**
  - ./bpi/hosts
  - Standard Ansible hosts file.
  - Provide IP addresses of target systems.
  - Enable HA.
  - For K8s, configure single or multiple master and worker nodes.

```
# Single Node
[master_node_primary]
k8s-master-0 ansible_host=10.78.4.78
```

```
# Single Master
[master_node_primary]
k8s-master-0 ansible_host=10.78.4.78

[master_nodes:children]
master_node_primary

[worker_nodes]
k8s-worker-0 ansible_host=10.78.4.46
k8s-worker-1 ansible_host=10.78.4.101
k8s-worker-2 ansible_host=10.78.4.98
```

```
# Multi Master
[master_node_primary]
k8s-master-0 ansible_host=10.78.4.78

[master_node_secondary]
k8s-master-1 ansible_host=10.78.4.113
k8s-master-2 ansible_host=10.78.4.58

[master_nodes:children]
master_node_primary
master_node_secondary

[worker_nodes]
k8s-worker-0 ansible_host=10.78.4.46
k8s-worker-1 ansible_host=10.78.4.101
k8s-worker-2 ansible_host=10.78.4.9
```

NOTE: Refer to the Blue Planet Installation Guides for details about this and other steps.

Blue Planet Software Installation Steps - High Level Overview

- **Configure a License Server IP.**
  - Ansible group_vars for bp2 platform
  - staticnbi (site-config) for Kubernetes platform
- **Validate the hosts.**
  - ./bpi --validate
  - Hardware (CPU, RAM, SWAP)
  - Operating System (type, architecture, version, system-level packages for bpi execution, SELinux mode.)
  - Disk (mountpoints, size, Docker volume group if required.)
  - Network (ports and unique hostnames.)
- **Set up system users and keys.**
  - ./bpi --setup-users
  - Setup SSH keys and passwordless SSH users (bpadmin, bpuser, bpmaint).

NOTE: Refer to the Blue Planet Installation Guides for details about these and other steps.

Blue Planet Software Installation Steps - High Level Overview

- **Setup the Kubernetes infrastructure.**
  - Options for single/multiple nodes and with or without Persistent Volume.
- **Verify the Kubernetes cluster.**
  - kubectl get nodes
- **Install a Lineup.**
  - A lineup is a configuration file used by the bpi tool to indicate which solutions should be deployed, in what order, and where to access them. Default lineups are delivered with each release; however, these should be customized for each customer at the time of installation.
  - For the Kubernetes platform, use Helm to access the appropriate Helm Charts from the bphub registry.

- **Access the Nagios web interface to verify and monitor deployment progress and status.**

NOTE: Refer to the Blue Planet Installation Guides for details about these and other steps.

Some Blue Planet Installer Options - General

Option	Description
bpi --validate	Validate the hardware and software against a given validation profile (lab or production environment).
bpi --setup-users	Setup users and their sudo permissions, create user keys, and copy them to remote hosts for passwordless authentication.
bpi --history	List of bpi commands that have been run.
bpi --get-logs	Retrieve log files from the cluster. The logs are in ./bpi/target/logs.
bpi --help	Display the help message containing all the options.
bpi --version	Display the installer version.
--playbook-args ='optional_args_for_ansible-playbook'	Pass optional arguments to ansible-playbook. Examples: --playbook-args='-vvv' for verbose debug --playbook-args='--skip-tags pv' for deploying without Persistent Volume

Different options call different scripts or Ansible playbooks. For example, when bpi is run with --get-logs, the getlog.yml and getmessages.yml playbooks are executed.

Some Blue Planet Installer Options - EKS

Option	Description
bpi --eks-setup	Setup EKS.
bpi --eks-reset	Reset EKS.
bpi --k8s-bp-aws-deploy	Deploy Blue Planet apps (xp, core, inventory) and logging (fluentbit) on the EKS cluster using Helm.
bpi --k8s-bp-undeploy	Undeploy Blue Planet apps (xp, core, inventory) and logging (fluentbit) from the K8s cluster using Helm.

Usage:

```
./bpi --eks-setup
```


```
./bpi --k8s-bp-aws-deploy <k8s_lineup> <lineup_version> <k8s_namespace> <helm_repo> <eks_cluster>
<aws_profile> <bphub_user> <bphub_pass>
```

Some Blue Planet Installer Options - Kubernetes On-Prem

Option	Description
bpi --k8s-setup	Setup Kubernetes.
bpi --k8s-setup-single-node	Setup Kubernetes on a single node.
bpi --k8s-reset	Reset Kubernetes.
bpi --k8s-reset-single-node	Reset Kubernetes from a single node.
bpi --k8s-bp-onprem-deploy	Deploy bp apps (xp, core, inventory) and logging (fluentbit) on the on-premise K8s cluster using Helm.
bpi --k8s-bp-onprem-deploy-single-node	Deploy bp apps (xp, core, inventory) and logging (fluentbit) on one node on-premise K8s cluster using Helm.
bpi --k8s-bp-undeploy	Undeploy bp apps (xp, core, inventory) and logging (fluentbit) from the K8s cluster using Helm.

Usage：

```
./bpi --k8s-setup

./bpi --k8s-bp-onprem-deploy <k8s_lineup> <lineup_version> <k8s_namespace> <helm_repository>
```

Summary

Blue Planet Installer Overview

In this section, you learned about the Blue Planet Installer (bpi) tool.

- The Blue Planet Installer (bpi) is an Ansible-based tool used to evaluate if the target system fulfills the installation requirements and it is capable of Blue Planet software downloads, installation, upgrading, and patching.
- bpi is obtained as a part of the Blue Planet software download process, or from the Blue Planet artifact repository.
- A lineup is a configuration file used by the bpi tool to indicate which solutions should be deployed, in what order, and where to access them.
- Different bpi options call different scripts or Ansible playbooks.

Blue Planet Platform Preferences

Objectives

- Describe the Blue Planet platform preferences
- Explain preference hierarchies
- Explore preference examples
- Describe Blue Planet UI themes and how to manage them
- Observe how to modify themes

blueplanet
a division of Ciena

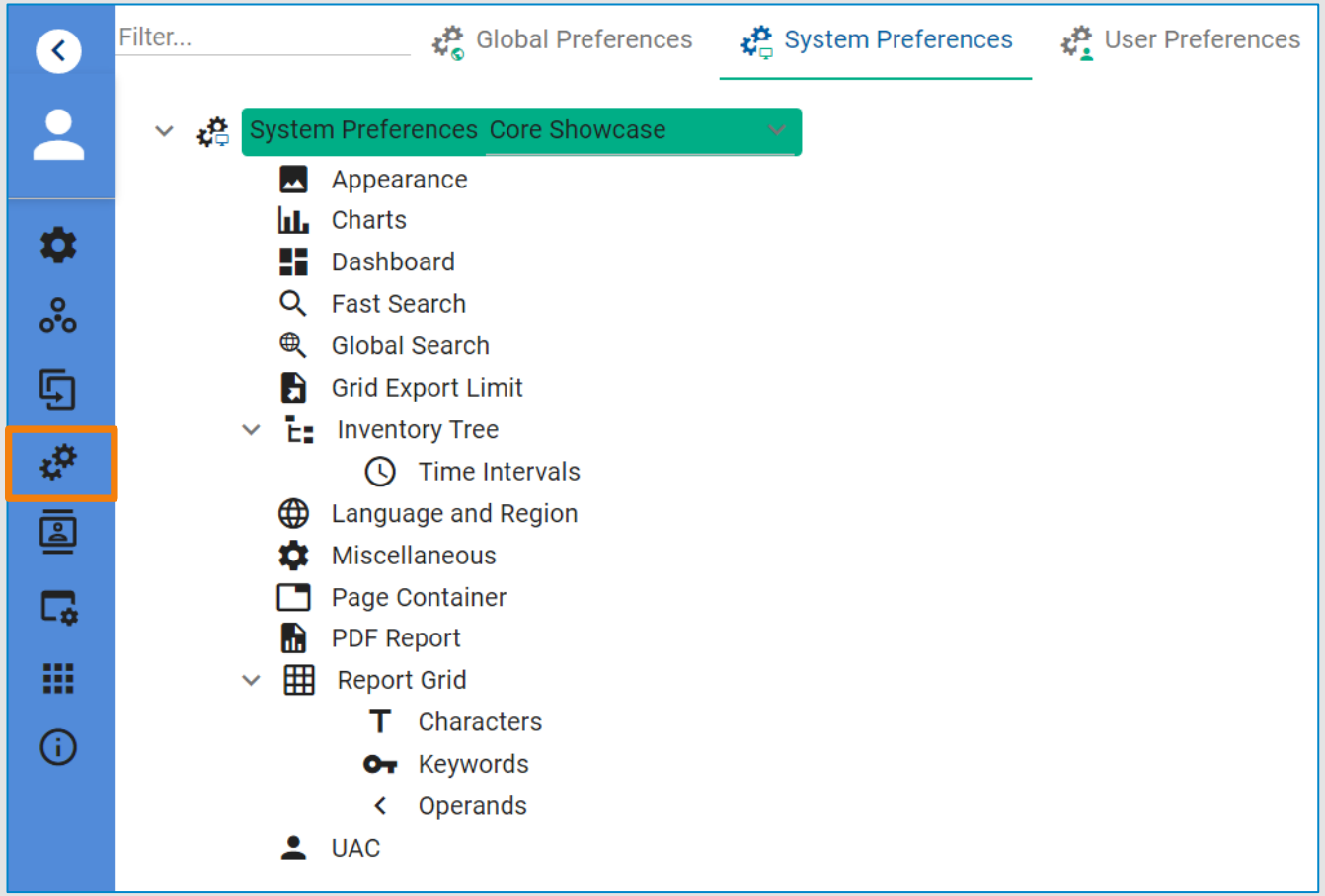
Blue Planet Platform Preferences

Agenda

1. **Blue Planet Platform Preferences**
2. Themes

Blue Planet Platform Preferences

- **Customize the behavior and appearance of common platform features.**
- **Global and System preferences are managed by administrators and define the default behavior.**
- **Users can override and customize the settings under User Preferences.**
- **Some additional features are product-specific.**
- **Accessible from the application bar.**



Preference Hierarchies

Global preferences

- Define the default values for all UIs (Systems) on the Blue Planet Platform instance.

System Preferences

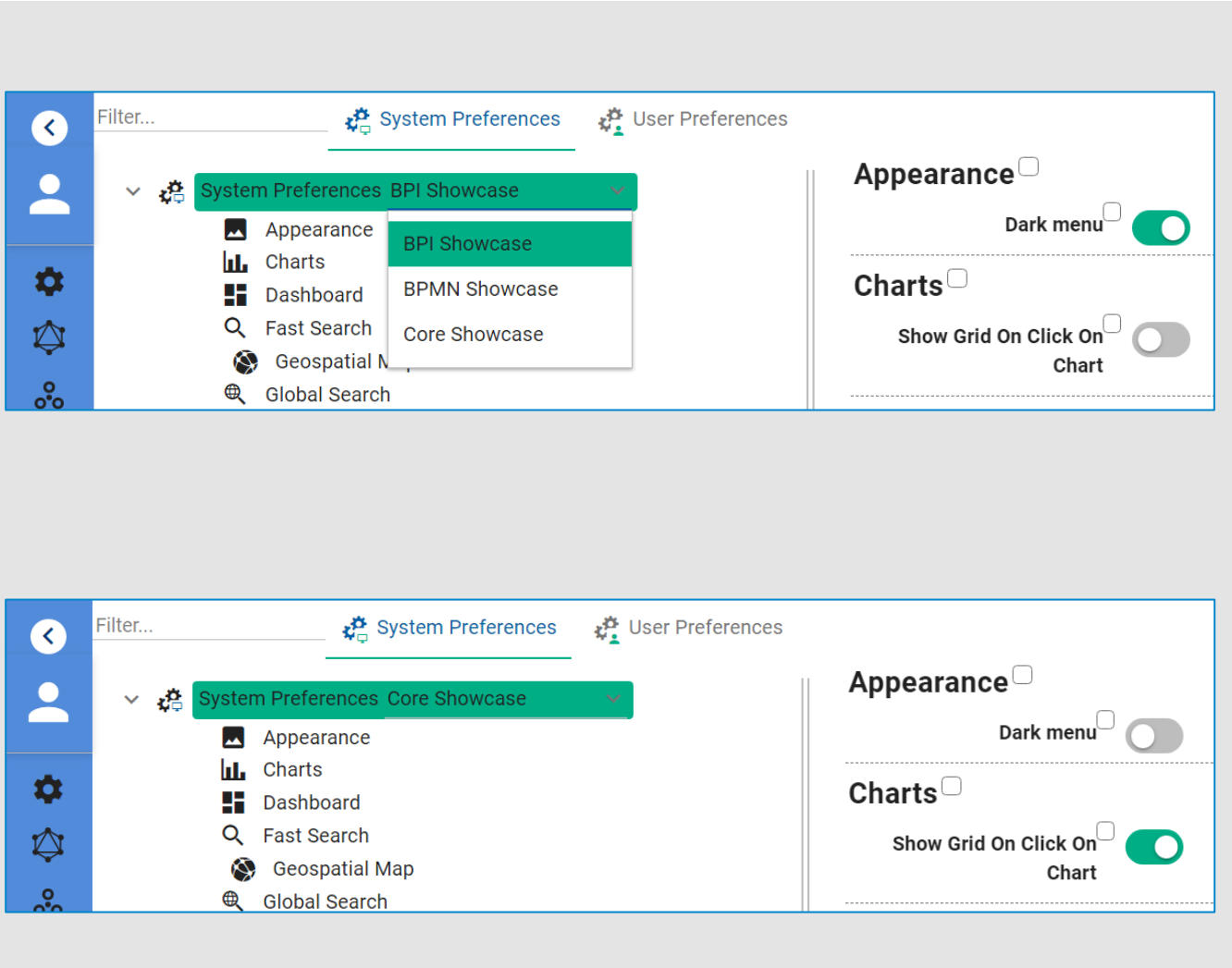
- Define the default values for the specific UI (specific System).
- Override the Global Preferences.

User Preferences

- Each user can define their own set of preferences.
- Override the System Preferences.

The option to change different preference levels is defined by user permissions.

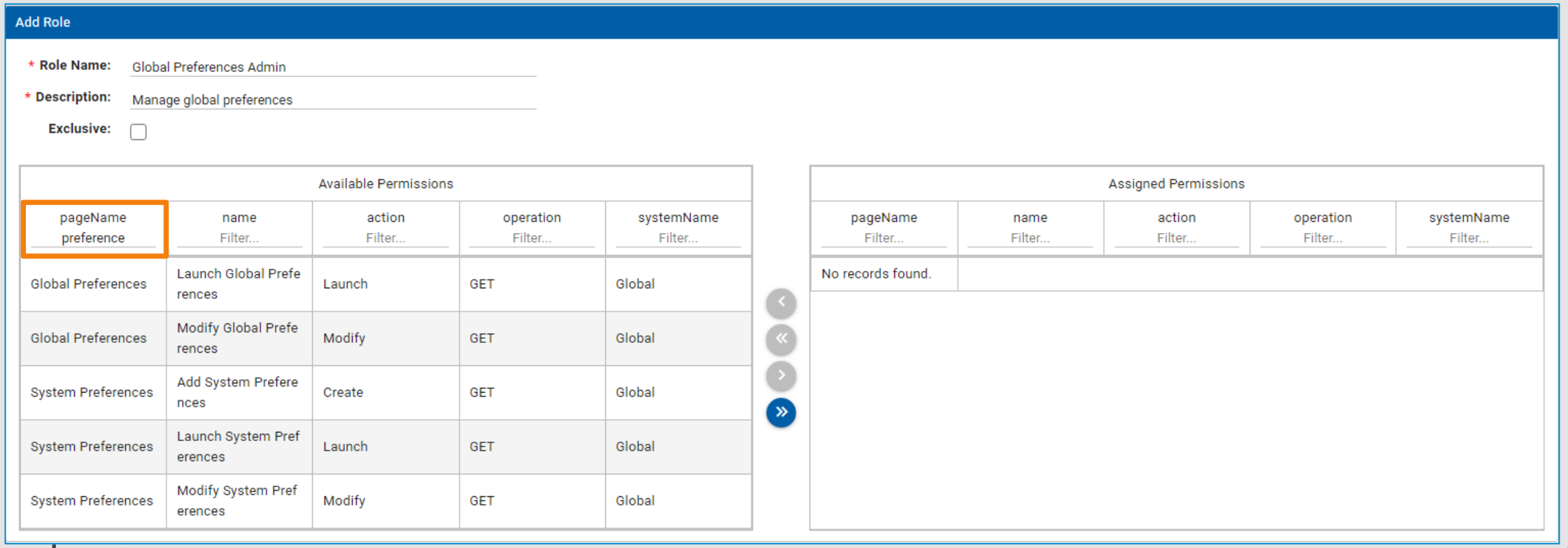
- By default, administrators or any other users do not have permission to change Global and System preferences.



Permissions for Managing Preferences

For managing System and Global Preferences, permissions need to be specifically added to the User Role.

Search for "preference" in the pageName column.



Limiting Preference Modification

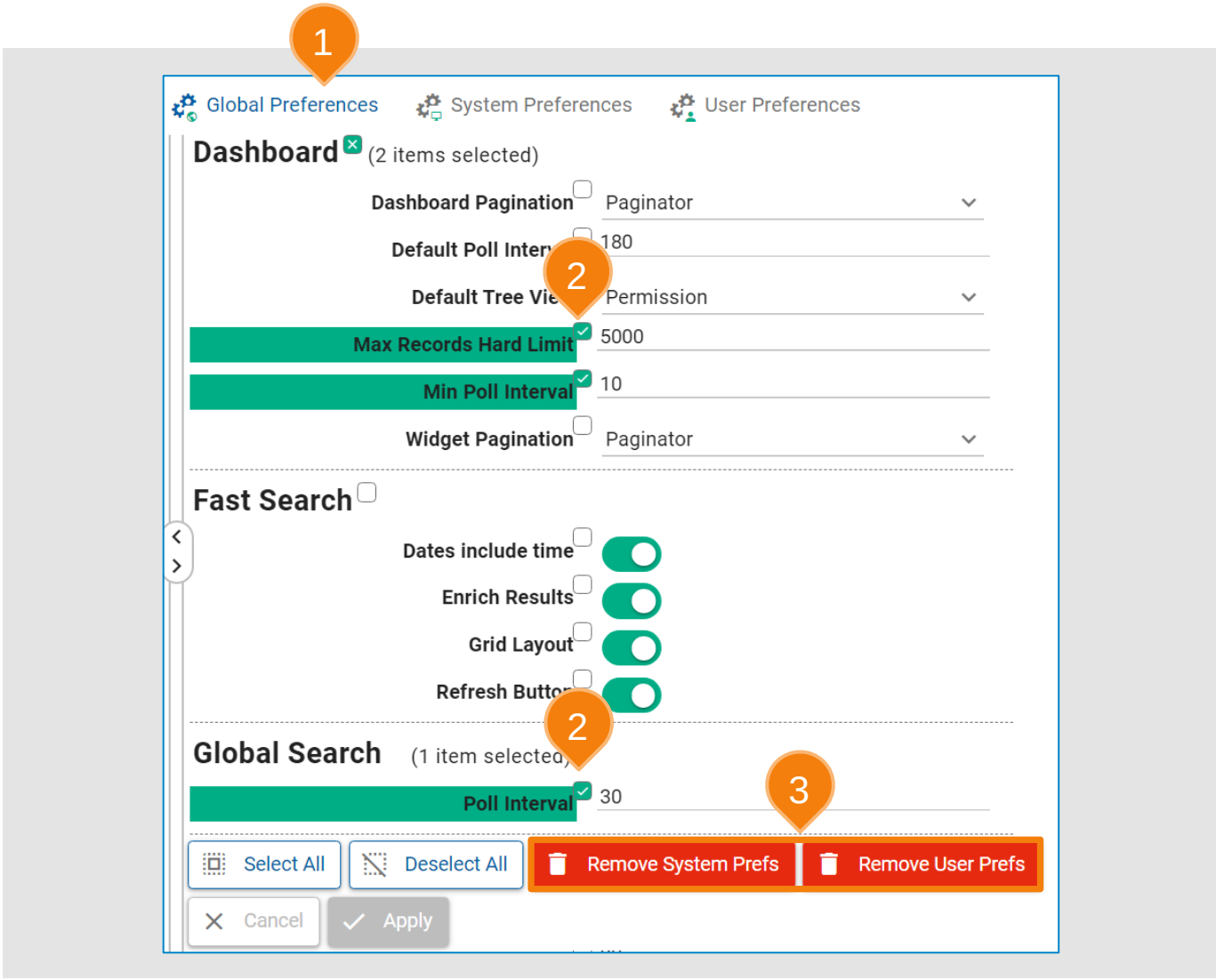
Privileged users can choose which preferences can be modified by less privileged users.

- Users with Global privileges can disable modifications of specific System preferences and/or User preferences.
- Users with System privileges can disable modifications of specific User preferences.

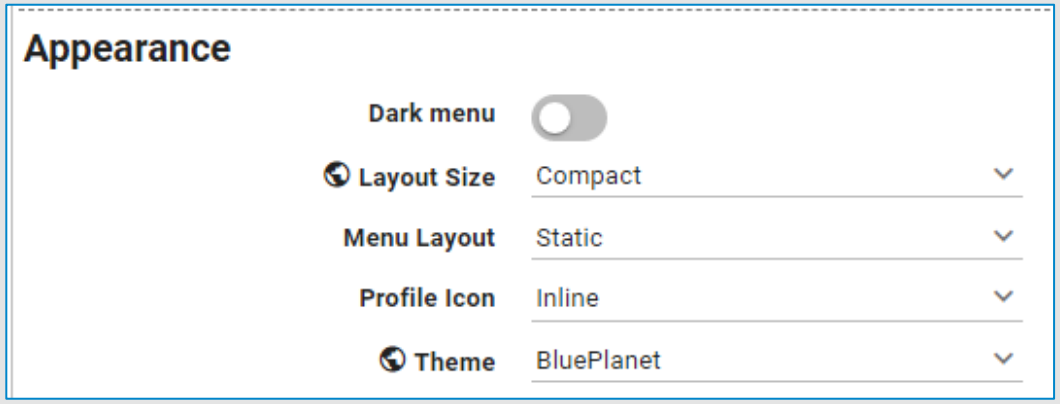
This ensures that some specific settings are only managed by administrators.

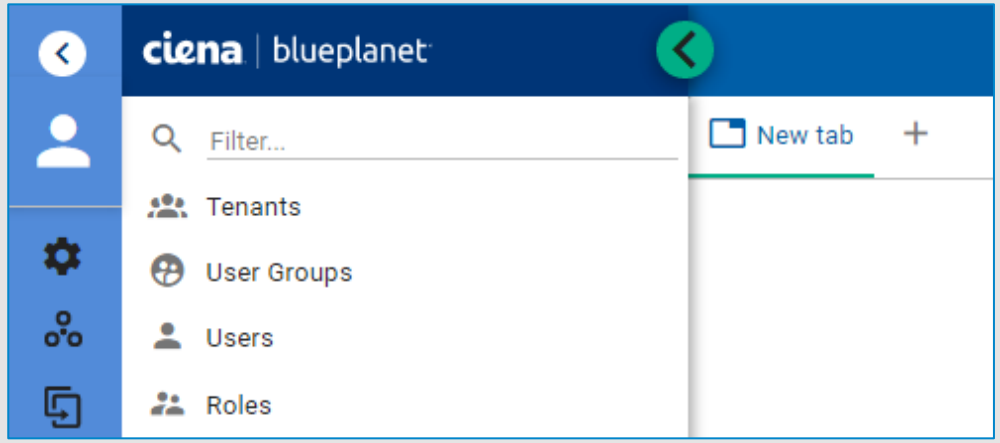
For example, the user has Global privileges:

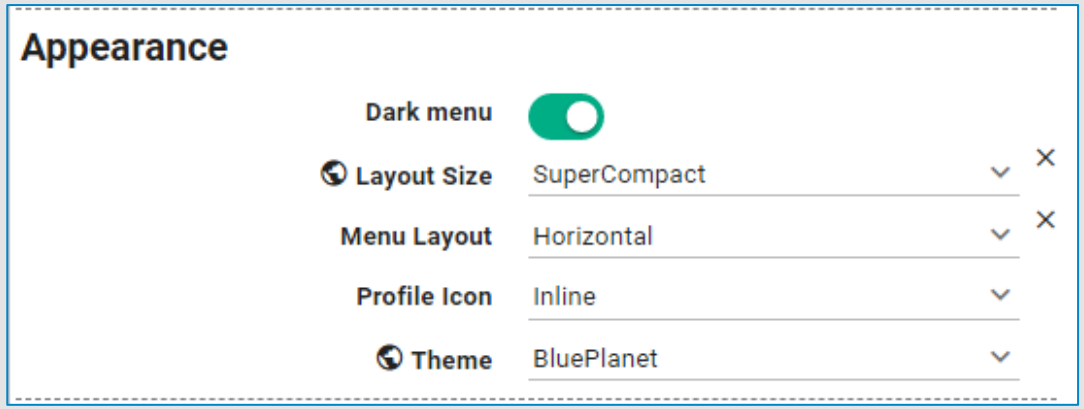
1. Global Preferences tab is selected.
2. Some preferences are checked.
3. Selected preferences can be removed from the System and the User preferences.

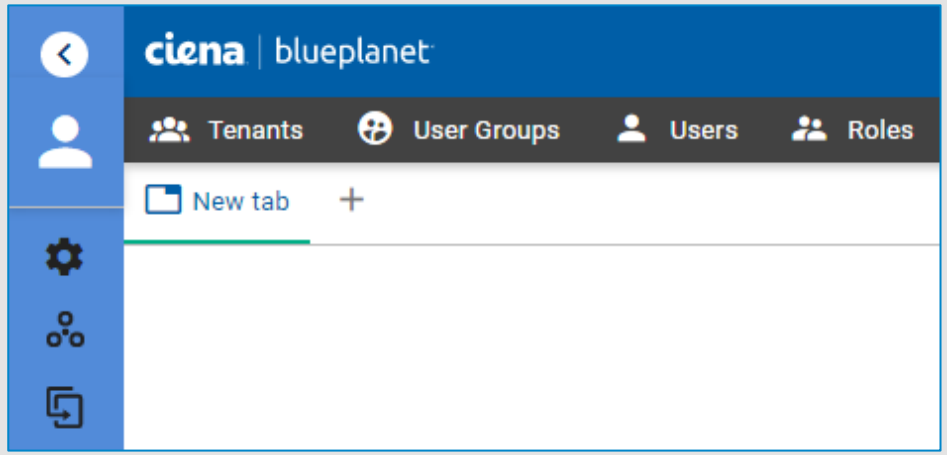


Preference Examples: Appearance



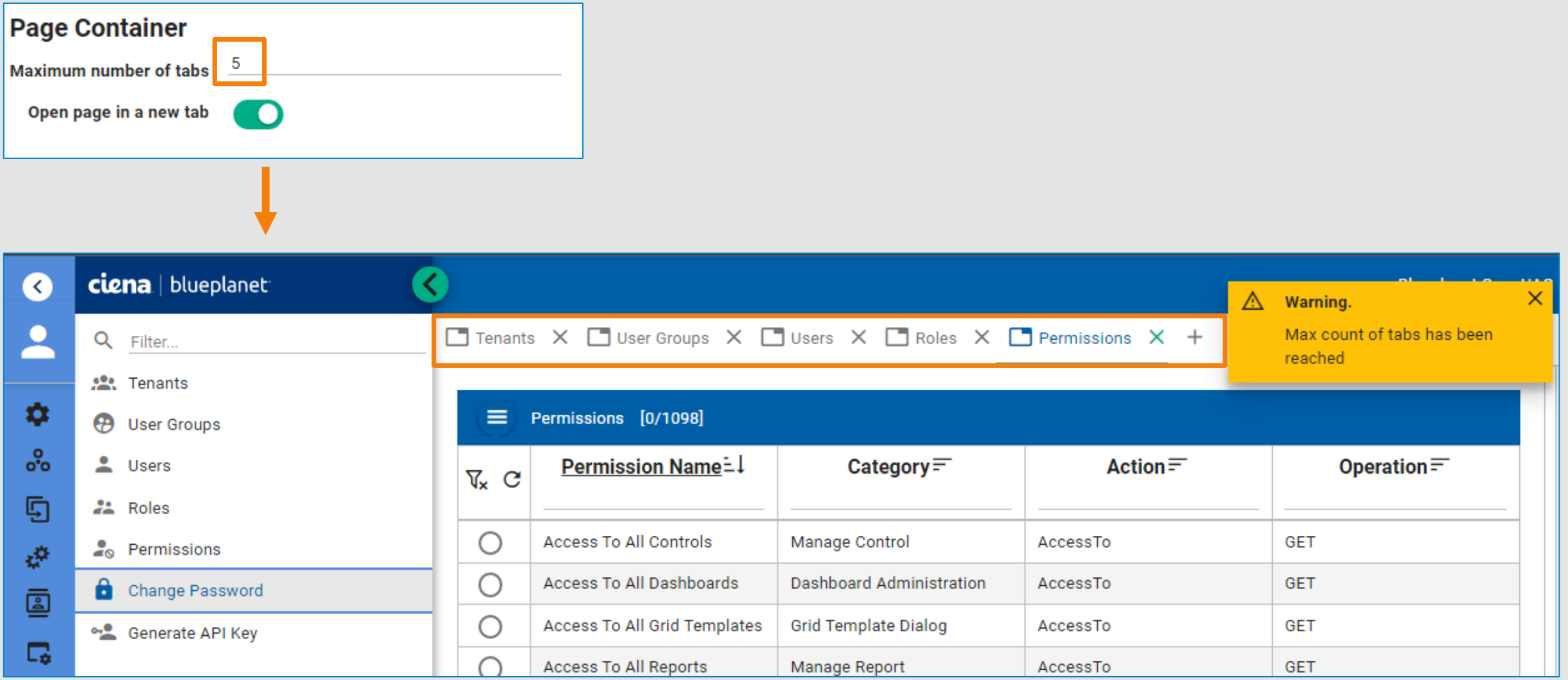






For new settings to take effect, the user needs to refresh the web page or logout/login.

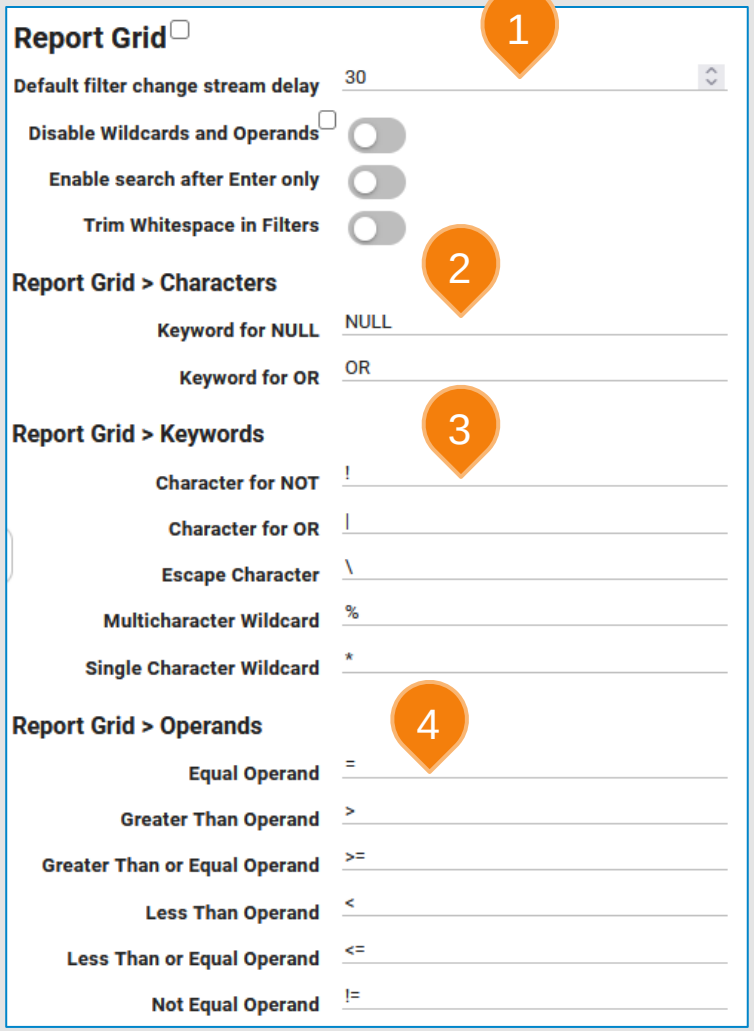
Preference Examples: Page Container



Preference Examples: Report Grid

Report Grid preferences define the behavior of filters on the data grid pages. Filters can include logical statements.

1. Define the responsiveness of the data grid filters in milliseconds.
2. Define characters for the different logical operations.
3. Define keywords for the different logical operations.
4. Define the operands.



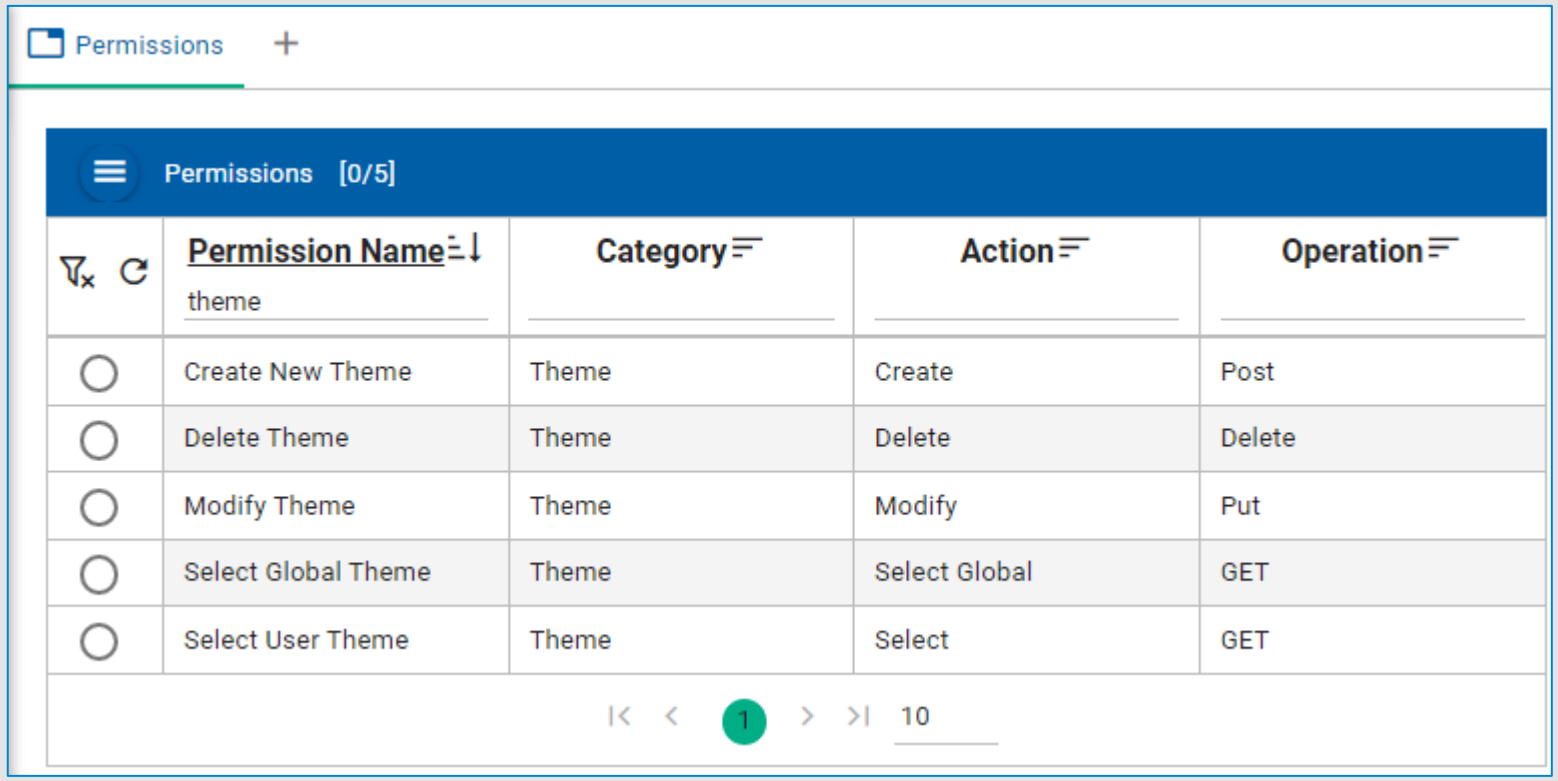
Blue Planet Platform Preferences

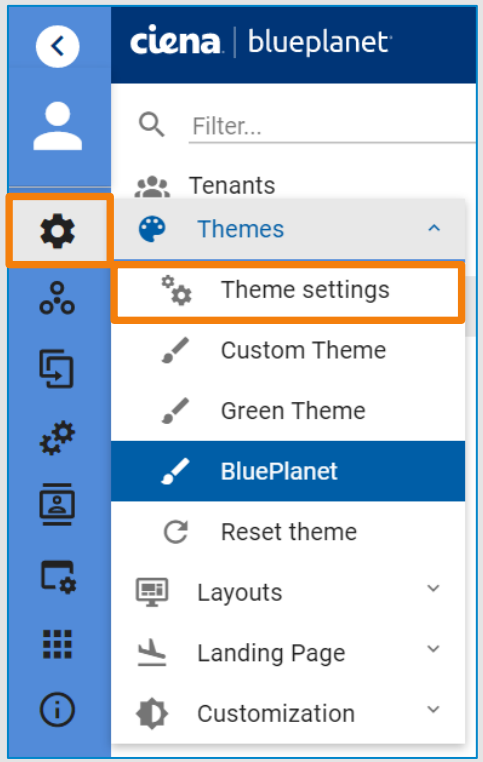
Agenda

1. Blue Planet Platform Preferences
2. **Themes**

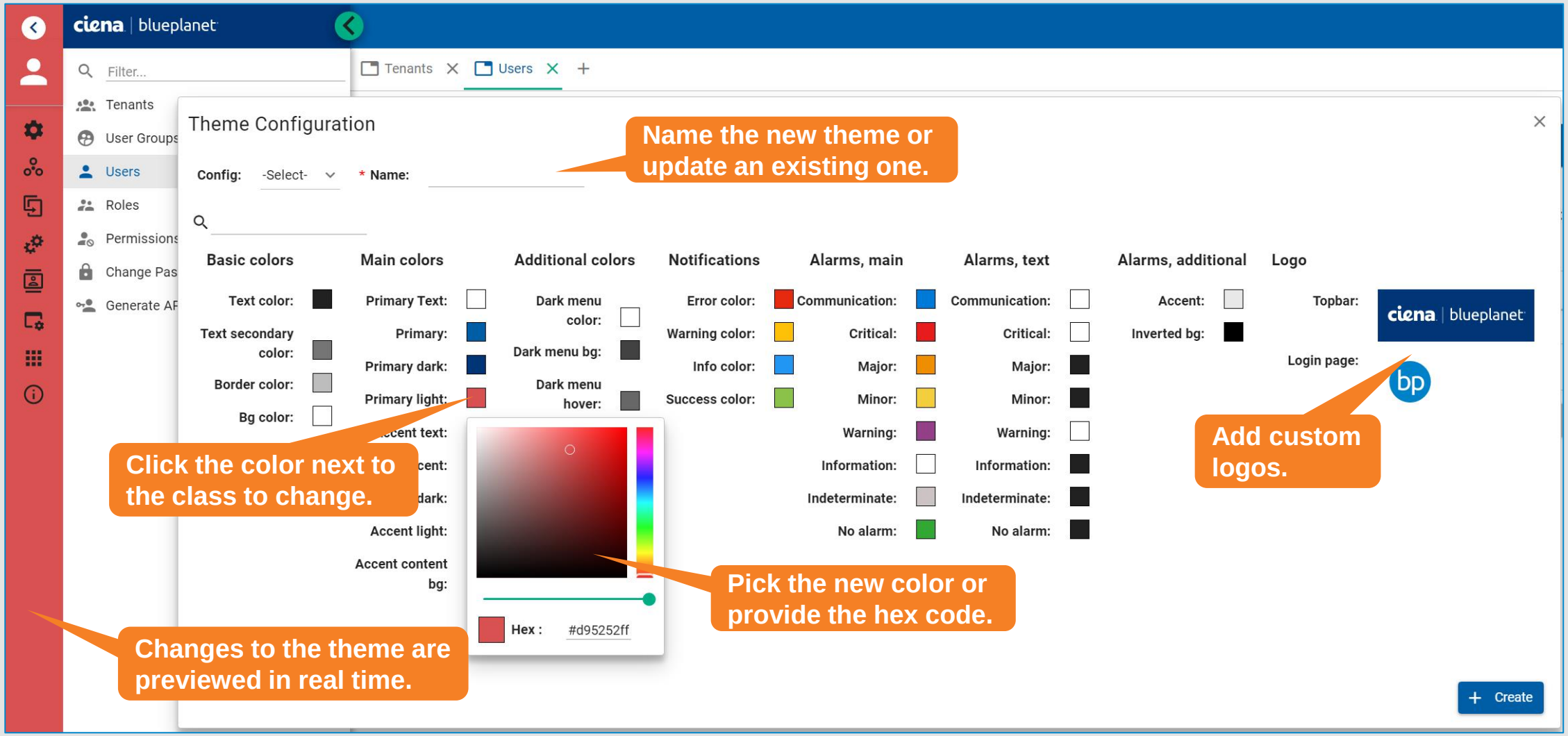
Themes

- **Custom themes can be created and made available for users.**
- **Themes are managed from the Settings menu in the application bar.**
- **Users can choose a theme from the Settings menu or through User Preferences.**
- **"Theme" user permissions are required to access Theme Settings.**





Configuring Themes



Summary

Blue Planet Platform Preferences

In this section, you learned about Blue Planet UI preferences and themes.

- Blue Planet platform preferences customize the behavior and appearance of common platform features.
- Global and System preferences are managed by administrators and define the default behavior.
- Users can override and customize the settings under User Preferences.
- For managing System and Global Preferences, permissions need to be specifically added to the User Role.
- Themes are managed from the Settings menu in the application bar.
- Users can choose a theme from the Settings menu or through User Preferences.

Monitoring Tools

Objectives

- Discover monitoring tools available in Blue Planet
- Explore Nagios for system monitoring
- Examine performance visualization with Grafana
- Demonstrate logs operations in Kibana

Monitoring Tools

Agenda

1. Monitoring Tools Overview
2. Working with Nagios
3. Working with Grafana
4. Working with Kibana

Monitoring Tools

The content for this section is covered in the BPS103 Blue Planet Monitoring Tools online course. Attendees should complete this course before attending the BPPA training.

https://learning.ciena.com/course/view.php?id=1874

Lab 2: Platform Operations and Maintenance

In this Lab you will learn how to:

- Get familiar with the Blue Planet installer and basic Helm commands.
- Use Kubernetes commands to manage the cluster, nodes, services, and other objects.
- Manage Platform backup and restore operations using snapshots.
- Manage the health of your deployment by using Nagios.
- Display built-in graphs and create custom graphs in Grafana.
- Monitor BP solution logs using Kibana.

blueplanet
a division of ciena

Thank You