White Paper

BlueData Software Architecture

BlueData EPIC™ version 3.5

The BlueData EPIC platform makes it easier, faster, and more cost-effective to deploy large-scale AI and Big Data environments—including TensorFlow, Spark, Kafka, Hadoop, and more—whether on-premises, in multiple public clouds, or in a hybrid model.

Table of Contents

1. BlueData EPIC

The **BlueData Elastic Private Instant Clusters** platform (referred to as **"EPIC"** throughout this white paper) allows users to create distributed environments for machine learning, data science, and analytics in minutes rather than months. With BlueData EPIC, you can offer a self-service experience with the data and tools that your data science teams need while providing enterprise-grade security and reducing costs. The hosts that run the EPIC platform may be located either on-premises, in the public cloud, or in a hybrid environment. This white paper provides an in-depth guide to the main components and architecture for version 3.5 of the BlueData EPIC software platform.

Virtualization and container technologies provide flexibility, agility, and reduced costs to most applications in the enterprise data center. BlueData EPIC uses Docker containers to extend these benefits to Big Data and Artificial Intelligence (AI) applications by allowing enterprises to create, modify, and remove virtual clusters on demand without sacrificing performance.

With the BlueData EPIC software platform, enterprises can simultaneously run hundreds of Big Data and AI workloads with automated policy-based scheduling and self-provisioning. Distributed applications are made efficient and elastic, thanks to EPIC's proprietary application-sensitive caching, data path, network optimization, and policy-based automation and management. IT administrators use a single interface to monitor all clusters, jobs, and infrastructure status. EPIC also automates routine tasks such as provisioning, updates, and monitoring.

BlueData EPIC dramatically reduces the deployment complexity for Big Data and AI applications while improving business agility by providing an elastic self-service infrastructure that reduces the time-to-value from months to days. At the same time, overall costs are reduced by 50%-75% compared to traditional bare-metal deployments using physical clusters. Users can create containerized clusters on demand and execute jobs without ever having to worry about the underlying infrastructure.

Key Features and Benefits

The key features and benefits of BlueData EPIC include:

- **Integrated platform for AI, Big Data analytics, and data science:** EPIC is an infrastructure platform purpose-built for Big Data and/or AI applications—including data science, analytics, machine learning (ML), and deep learning (DL)—using enterprise-grade security, networking, and support for a variety of local and remote storage options.
- **Runs on-premises and/or on public cloud virtual machines (VMs):** EPIC can be deployed on-premises, in the public cloud, or in a hybrid environment that includes both public cloud and on-premises resources.
- **Create virtual clusters:** EPIC uses containers to replicate the functionality of physical clusters while adding flexibility and scalability at reduced cost. You may create, modify, re-prioritize, and remove containerized clusters (referred to as *virtual clusters* throughout this white paper) on demand in response to ever-changing needs within individual business units/departments. EPIC reduces time-to-value from months to hours.
- **Multi-tenancy and enterprise-grade security model:** EPIC integrates with enterprise LDAP and Active Directory authentication systems. Administrators can create groupings of users and resources that restrict access to jobs, data, or clusters based on department membership and/or roles. The result is an integrated, secure, multi-tenant infrastructure.
- **Self-service portal:** EPIC includes a self-service web portal that allows users to create and manage clusters, create and manage nodes, run jobs, and view monitoring statistics. User visibility into resources and ability to take action on the platform vary based on each user's role and tenant membership, in accordance with existing enterprise security policies. For example, department administrators can use the portal to provision nodes/clusters to run their own Big Data and AI applications without impacting nodes/clusters that are assigned to different departments and without having to manage the physical infrastructure.
- **RESTful API:** EPIC supports a RESTful API that surfaces programmable access to the same capabilities available via the self-service portal.
- **Superior performance:** EPIC provides storage I/O optimizations to deliver data to Big Data and AI applications without the penalties commonly associated with virtualization or containerization. The CPU cores and RAM in each host are pooled and then partitioned into virtual resource groups based on tenant requirements.
- **Works with existing infrastructure:** EPIC allows your enterprise to repurpose its existing infrastructure investments for Big Data and AI deployments. EPIC can run on your physical and virtualized infrastructure, including CPUs and GPUs, as well as on all three major public clouds (Amazon Web Services, Google Cloud Platform, and Microsoft

Azure). Existing storage protocols are also supported (HDFS, HDFS with Kerberos, and NFS).

- **Reduced IT overhead:** EPIC streamlines operations and reduces IT costs by automating provisioning, unifying management, and supporting push-button upgrades.
- **Increases utilization while lowering costs:** EPIC delivers hardware and operational cost savings while simultaneously eliminating the complexity of managing multiple physical clusters. EPIC also allows clusters to be paused/unpaused at will, meaning that you only use the resources that you need when you need them.
- **High Availability:** EPIC supports three levels of High Availability to provide redundancy and protection, as described in "High Availability" on page 25.
- **Compute and storage separation:** EPIC decouples analytical processing from data storage, giving you the ability to independently scale compute and storage capacity instantly on an as-needed basis. This enables more effective utilization of infrastructure resource and reduces overall costs.
- **In-place access to on-premises enterprise storage and cloud storage:** EPIC allows you to access and run jobs directly against both existing enterprise-class storage systems and cloud storage systems. The separation of compute and storage provided by EPIC means that you don't need to move or duplicate data before running analytics.

App Store

The BlueData EPIC platform includes an *App Store* with one-click deployment for common Big Data and AI tools, such as:

- Open-source Apache Hadoop distributions from Cloudera, Hortonworks, and MapR.
- Open-source Apache Spark with support for standalone resource manager and YARN.
- Open-source Apache Kafka messaging system.
- Open-source Cassandra NoSQL database from Datastax.
- Integration with JupyterHub, RStudio Server, and Zeppelin data science notebooks.
- Integration with common ML/DL and analytics tools, including Tensor Flow and H2O.

The App Store contains Docker container images of each available application, allowing fully automated self-service deployment. Each image in the App Store provides a particular version, is pre-configured, and ready-to-run on the EPIC platform. EPIC also supports a "bring your own app" model that allows users to quickly add images for any Big Data application or data processing platform to the App Store.

The App Store contains three classes of images:

- **Hadoop, Spark, Kafka, and other Big Data tools provided out-of-the-box by BlueData.** These images contain open-source software that is unmodified and supported by these vendors.
- **ML/DL, analytics, and data science tools supported out-of-the-box by BlueData.** The App Store includes several pre-configured open-source tools as examples. Other tools have also been tested for compatibility with EPIC and can be made available to customers, including both open-source and commercial applications.
- **Custom tools and applications added specifically by individual customers.** BlueData provides an *Application Workbench* that allows customers to create and add their own images to the App Store. Users can then deploy these images and use them in a similar way as any of the out-of-the-box images described above.

Note: App Store images are independent from the BlueData EPIC platform itself. Any tool or application can be added or removed from your EPIC platform to suit your specific needs.

The Platform Administrator may install or uninstall images. Installed images are available for use by Tenant Members when creating jobs and clusters.

BlueData and/or application vendors may provide new images or new versions of existing images. If the EPIC Controller host can access the internet and a new version becomes available for an image that is currently installed, the image will be marked in the App Store with an *Upgrade Available* banner, and its tile will provide a button for upgrading to the new version. Other new images or new versions of currently uninstalled images will display a *New* banner.

See the following for additional information:

- "Hadoop and Spark Support" on page 29 for examples of supported Big Data application services and Hadoop/Spark ecosystem products.
- "Artificial Intelligence and ML/DL Workloads" on page 30 for examples of how BlueData EPIC supports Machine Learning (ML) and Deep Learning (DL) tools for AI applications.

2. BlueData EPIC Architecture

This section describes the BlueData EPIC architecture, user hierarchy, DataTaps, and other important EPIC concepts.

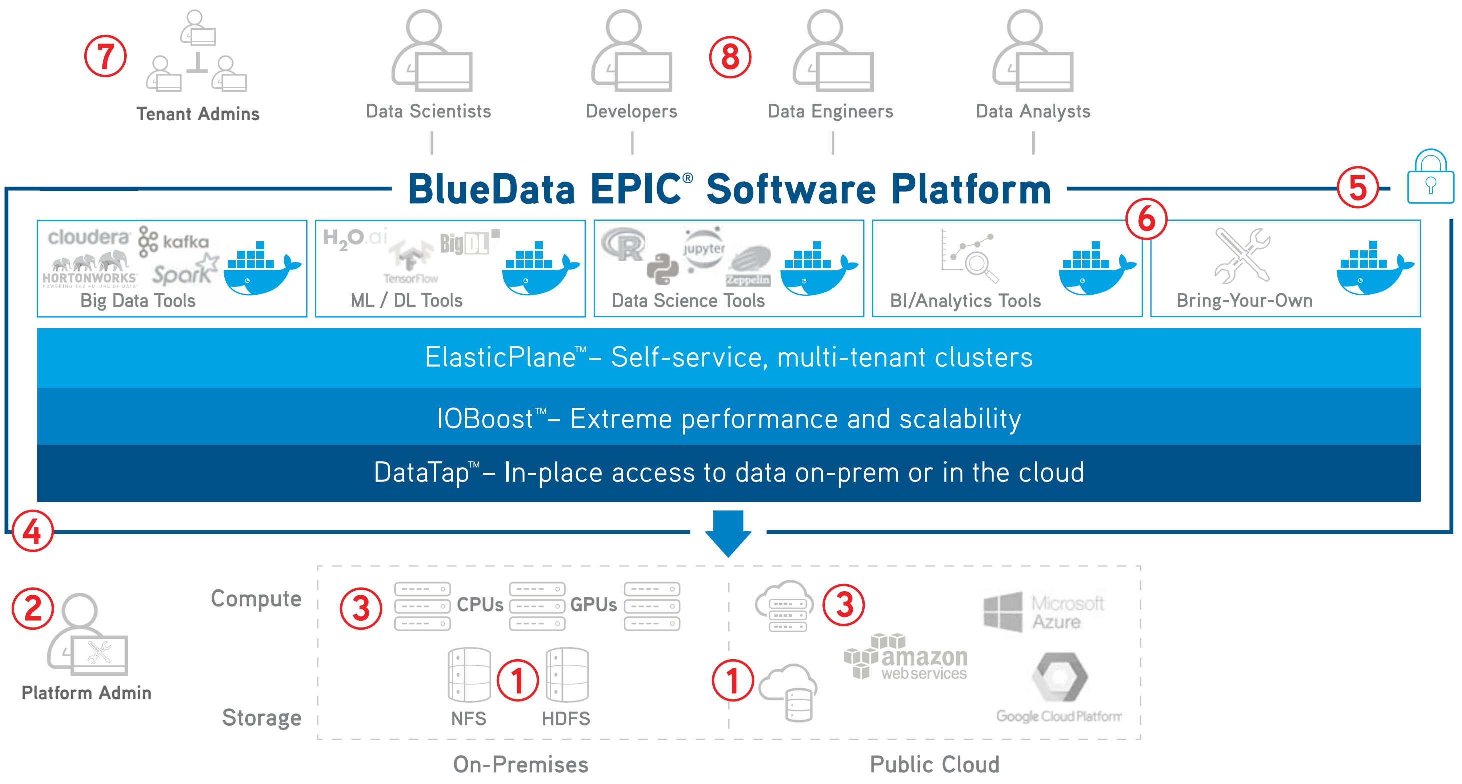
Software Components

BlueData EPIC is an enterprise-grade software platform that forms a layer between the underlying infrastructure and Big Data/ AI applications, transforming that infrastructure into an agile and flexible platform for virtual clusters running on Docker containers.

EPIC consists of three key capabilities:

- **ElasticPlane™** is a self-service web portal interface that spins up virtual clusters on demand in a secure, multi-tenant environment.
- **IOBoost™** provides application-aware data caching and tiering to ensure high performance for virtual clusters running Big Data and AI workloads.
- **DataTap™** accelerates time-to-value by allowing in-place access to any storage environment, thereby eliminating time-consuming data movement.

Figure 1: BlueData EPIC software architecture (below)



The high-level EPIC architecture is as follows (numbers correspond to the callouts in Figure 1, above):

- **Data Storage Resources (1):** On-premises and/or cloud-based storage resources available to the EPIC platform. This comprises the following:
  - **Data Source:** This is where EPIC reads and writes persistent job data required by the tenants and virtual clusters. A data source is typically a DataTap: a shortcut within EPIC that points to existing remote data storage locations on your network. A special DataTap instance, constructed from storage local to the EPIC hosts, is known as Tenant Storage. The use of DataTaps reduces or even eliminates the need to copy large volumes of data to and from the virtual clusters before and after running jobs, thus saving time and reducing network traffic. Please see "About DataTaps" on page 14 for more about how EPIC handles data storage.

- **Cluster file system:** This is the storage where EPIC reads and writes temporary data that is generated while running jobs within an given cluster. The cluster file system is built within the virtual cluster, on storage taken from the node storage space of the underlying EPIC host.
- **Unique file directories for each tenant:** EPIC automatically creates a sandboxed shared-storage area for each tenant within the tenant storage space of the EPIC platform, whether on-premises or on the public cloud. This per-tenant storage can be used to isolate data that should be accessible by only one tenant. Optionally, it can be used to enforce a quota on the tenant's use of that storage capacity.

- **Platform Administrator (2):** One or more Platform Administrator(s) handle overall EPIC administration, including managing hosts and creating tenants.
- **Hosts (3):** Physical and/or virtual machines where the EPIC platform is deployed. Hosts may reside on your premises (such as in your datacenter) and/or on one or more public clouds (Amazon Web Services, Google Cloud Platform, and/or Microsoft Azure). Each EPIC host fulfills one of the following roles:
  - **Controller:** The Controller host is the host where you initially install EPIC. This host controls the rest of the hosts in the EPIC deployment. It is located either on your own infrastructure (on a physical server or virtual machine) or on a public cloud instance. EPIC is supported on the Red Hat Enterprise Linux or CentOS operating system, version 6.8 (minimum), version 7.3, or version 7.4 (recommended). You must have the operating system installed on all of the nodes that you will be using for EPIC before beginning the installation process. If platform High Availability is enabled (see "High Availability" on page 25), then the EPIC platform will include a Shadow Controller host and an Arbiter host.
  - **Worker:** Worker hosts are under the direct control of the Controller host. EPIC dynamically allocates the resources on these hosts to the virtual clusters and jobs within each tenant as needed, based on user settings and resource availability. This dynamic resource allocation means that EPIC achieves a much higher host utilization rate than traditional Hadoop and Spark deployments. If the EPIC platform has High Availability enabled, then two of these Worker hosts will have additional roles, as described in High Availability. The Worker host(s) are located either on your infrastructure (physical server or virtual machine) or on public cloud VMs (for cloud and hybrid deployments). Each Worker host can manage multiple containers/virtual nodes. For example, a virtual cluster that consists of four virtual nodes could potentially be located on a single Worker host.
  - **Gateway:** If EPIC is deployed using a private, non-routable network for virtual nodes/containers, then the EPIC platform will include one or more Gateway host(s) that map both the Controller host and the private service IP endpoints on the virtual nodes/containers inside the EPIC virtual clusters to publicly-accessible IP addresses/ports. See "Gateway Hosts" on page 20.
- **EPIC Platform (4):** The EPIC platform consists of the EPIC services that are installed on each of the hosts. EPIC automatically handles all of the back-end virtual cluster management, thereby eliminating the need for complex, time-consuming IT support. Platform and Tenant Administrator users can quickly perform all of these tasks using the EPIC interface.
- **Role-based security (5):** EPIC includes three user roles (Platform Administrator, Tenant Administrator, and Tenant Member) that allow you to control who can see certain data and perform specific functions. Roles are granted on a per-tenant basis, meaning that you can either restrict users to a single tenant or grant access to multiple tenants. Each user may have at most one role per tenant. You may opt to handle user authentication on either a platform level or on a per-tenant level. When you are using tenant-independent authentication, each tenant may be configured to use a separate authentication method to grant user access. For example, Tenant_1 may use the internal EPIC authentication server while Tenant_2 and Tenant 3 each use different instances of external LDAP/AD servers.
- **Clusters (6):** Clusters can be created to run a wide variety of Big Data and AI/ML/DL applications, services, and jobs.
- **Tenant Administrators (7):** A Tenant Administrator manages the resources assigned to that tenant. Each tenant must have at least one user with the Tenant Administrator role.
- **Tenants (8):** Tenants allow you to restrict EPIC access as needed, such as by department. Each tenant has its own unique sets of authorized users, DataTaps, applications, and virtual clusters that are never shared with other tenants. Users with access to one tenant cannot access or modify any aspect of another tenant unless they have also been assigned a role (Tenant Administrator or Member) on that tenant. If tenant-independent authentication has been enabled, then the user must have a valid account in the authentication system used by each tenant in which that user has a role. In the example from #5, above, a user wanting to access all three

tenants would need three accounts: one in the internal EPIC authentication server, and one each on the two external LDAP/AD servers. Further, if tenant-independent authentication has been enabled, then the same user account cannot be assigned a role in more than one tenant.

On-Premises, Hybrid, and Multi-Cloud Deployments

BlueData EPIC can be deployed standalone on-premises and/or on a public cloud, as well as in a hybrid deployment where some of the EPIC hosts reside on-premises (including in multiple datacenters) while other hosts reside on one or more public cloud(s). See Figures 2 and 3 on the next page. In each case, the network requirements described in "Networks and Subnets" on page 18 must be satisfied. EPIC versions 3.2 and higher support all of the major cloud providers:

- **Amazon Web Services (AWS):** Either a configurable AWS CloudFormation or a purpose-built `grkeevn` command line tool (available with EPIC versions 3.5 and higher) can be used to configure EPIC on EC2 instances.
- **Google Cloud Platform (GCP):** A configurable YAML deployment script in conjunction with the `inenqwf` deployment tool is used to configure EPIC on Google Cloud compute instances (VMs).
- **Microsoft Azure:** A configurable Azure Resource Manager template is used to configure EPIC on Azure instances.

Benefits

EPIC offers the following benefits regardless of the deployment model used:

- The same general approach may be used to install and run EPIC regardless of how it is deployed. See "Single-Cloud Deployments" on page 6 and "Hybrid and Multi-Cloud Deployments" on page 6.
- The code base and management experience are identical for a fully on-premises deployment, fully cloud-only deployment, or hybrid deployment.
- You retain full control over your on-premises and/or cloud infrastructure. For example, you can leverage your existing application machine images (e.g. AWS AMI) for your certified RHEL operating system as well as any cloud specific features. These cloud-specific features include the use of availability zones, spanning hosts across subnets, and multiple instance types.

Note: This white paper uses the term "host" to refer to the physical host(s) and/or virtual machine(s) that run EPIC virtual nodes/ containers.

- EPIC supports vCPU over-provisioning and can place multiple virtual nodes/containers on each host, thereby increasing host utilization and reducing costs. See "Virtual Cores, RAM, Storage, and GPU Devices" on page 11.
- EPIC supports host tags that can be used to control the placement of virtual nodes (containers) among on-premises hosts and/or cloud instances from AWS, Azure or GCP. This enables the placement of virtual nodes based on workload (e.g. Spark, TensorFlow etc), SLA and data gravity considerations. See "Tenant and Host Tags" on page 23.
- EPIC provides common benefits regardless of the deployment model used (see below), including strict tenant isolation without cloud-specific networking constructs.
- Gateway hosts use HAProxy to control access (ingress) to application service endpoints. See "Gateway Hosts" on page 20.

Single-Cloud Deployments

Figure 2 depicts a BlueData EPIC deployment that has been installed either entirely on a single cloud or on-premises on virtual machines in a private cloud.

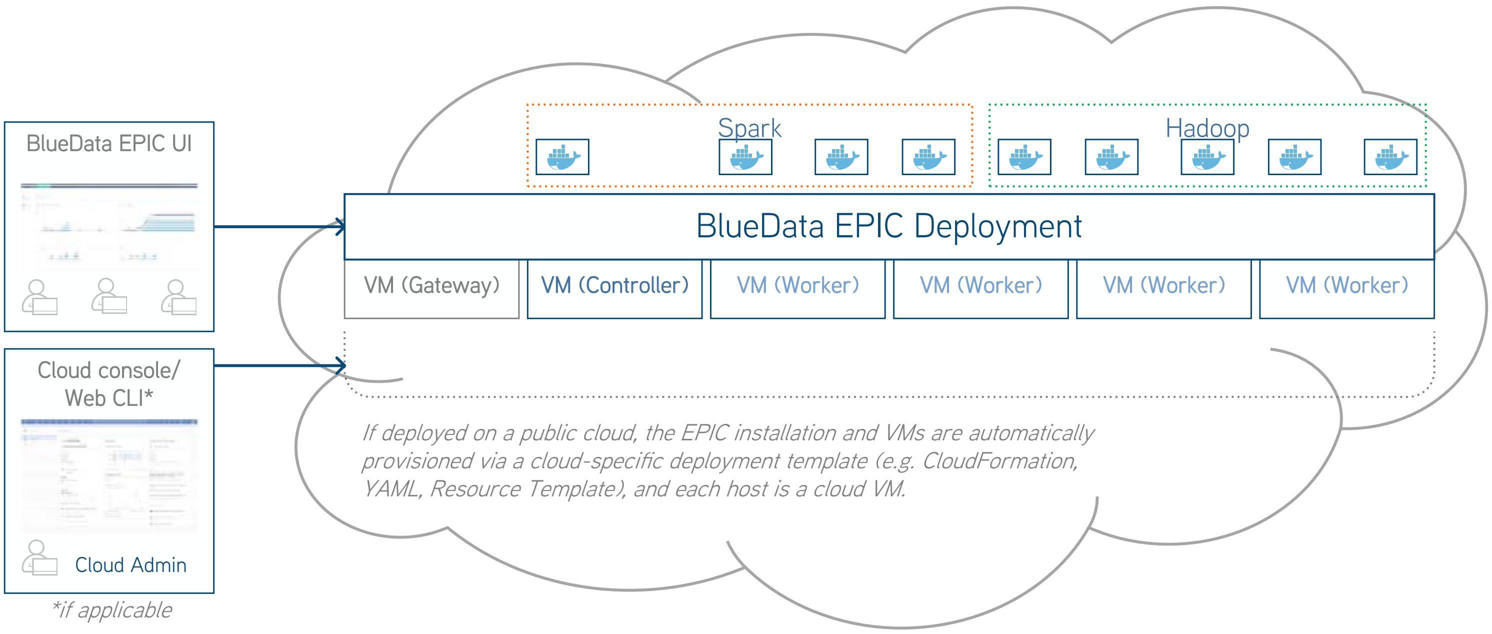

BlueData EPIC UI
Cloud console/ Web CLI*
Cloud Admin
*if applicable
Spark
Hadoop
BlueData EPIC Deployment
VM (Gateway)
VM (Controller)
VM (Worker)
VM (Worker)
VM (Worker)
VM (Worker)
If deployed on a public cloud, the EPIC installation and VMs are automatically provisioned via a cloud-specific deployment template (e.g. CloudFormation, YAML, Resource Template), and each host is a cloud VM.

Figure 2: EPIC deployed on a single cloud.

Hybrid and Multi-Cloud Deployments

Figure 3 depicts a BlueData EPIC deployment that has been installed across one or more public cloud(s) and physical hosts on-premises.

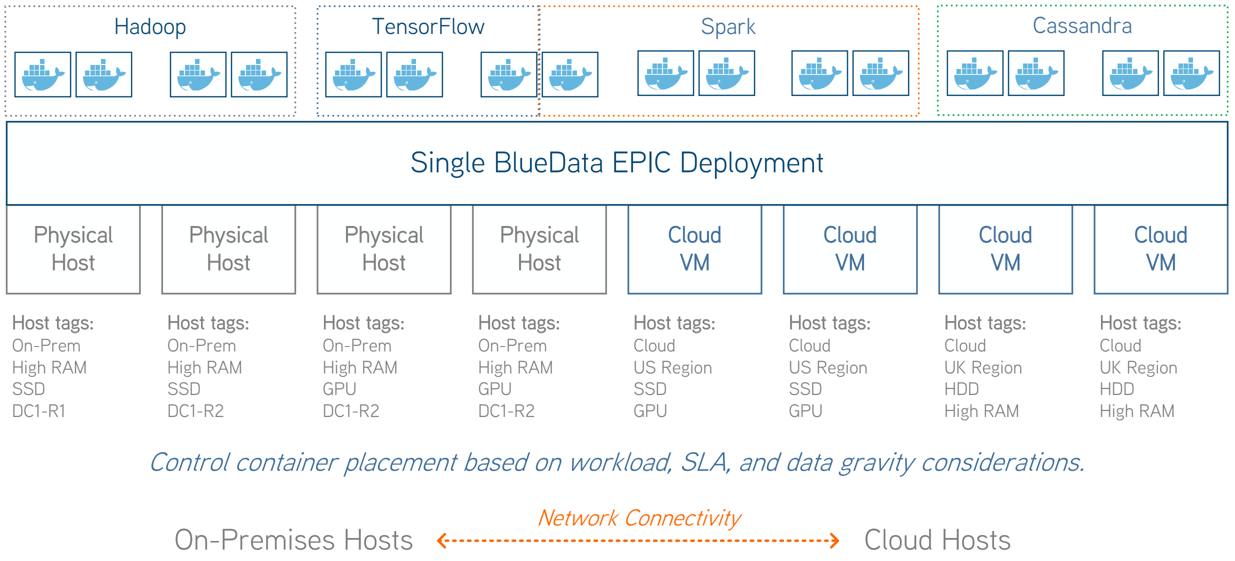

Hadoop
TensorFlow
Spark
Cassandra
Single BlueData EPIC Deployment
Physical Host
Physical Host
Physical Host
Physical Host
Cloud VM
Cloud VM
Cloud VM
Cloud VM
Host tags: On-Prem High RAM SSD DC1-R1
Host tags: On-Prem High RAM SSD DC1-R2
Host tags: On-Prem High RAM GPU DC1-R2
Host tags: On-Prem High RAM GPU DC1-R2
Host tags: Cloud US Region SSD GPU
Host tags: Cloud US Region SSD GPU
Host tags: Cloud UK Region HDD High RAM
Host tags: Cloud UK Region HDD High RAM
Control container placement based on workload, SLA, and data gravity considerations.
Network Connectivity
On-Premises Hosts
Cloud Hosts

Figure 3: EPIC deployed in a hybrid architecture

3. Tenants

Tenants are created by the Platform Administrator after the BlueData EPIC Controller has been installed. The infrastructure resources (e.g. CPU, memory, GPU, storage) available on the BlueData EPIC Worker hosts are split among the tenants on the platform. Each tenant is allocated a set of resources and restricts access to a set of data to only those users authorized to access the tenant. Resources used by one tenant cannot be used by another tenant. All users who are members of a tenant can access the resources and data objects available to that tenant.

You will need to decide how to create tenants to best suit your organizational needs, such as by:

- **Office location:** If your organization has multiple office locations, you could choose to create one or more tenants per location. For example, you could create a tenant for the San Francisco office and one for the New York office. EPIC does not take location into account; this is just an example of how you could use a tenant.
- **Department:** You could choose to create one or more tenants for each department. For example, you could create one tenant each for the Manufacturing, Marketing, Research & Development, and Sales departments.
- **Use cases, application lifecycle, or tools:** Different use cases for Big Data analytics and data science may have different image and resource requirements.
- **Combination:** You could choose to create one tenant by department for each location. For example, you could create a tenant for the Marketing department in San Francisco and another tenant for the Marketing department in New York.

Some of the factors to consider when planning how to create tenants may include:

- **Structure of your organization:** This may include such considerations as the department(s), team(s), and/or function(s) that need to be able to run jobs.
- **Use cases/tool requirements:** Different use cases for Big Data analytics and data science may have different image/ resource requirements.
- **Seasonal needs:** Some parts of your organization may have varying needs depending on the time of year. For example, your Accounting department may need to run jobs between January 1 and April 15 each year but have little to no needs at other times of the year.
- **Amount and location(s) of hosts:** The number and location(s) of the hosts that you will use to deploy an EPIC platform may also be a factor. If your hosts are physically distant from the users who need to run jobs, then network bandwidth may become an important factor as well.
- **Personnel who need EPIC access:** The locations, titles, and job functions of the people who will need to be able to access EPIC at any level (Platform Administrator, Tenant Administrator, or Tenant Member) may influence how you plan and create tenants.
- **IT policies:** Your organization's IT policies may play a role in determining how you create tenants and who may access them.
- **Regulatory needs:** If your organization deals with regulated products or services (such as pharmaceuticals or financial products), then you may need to create additional tenants to safeguard regulated data and keep it separate from non-regulated data.

These are just a few of the possible criteria you must evaluate when planning how to create tenants. EPIC has the power and flexibility to support the tenants you create regardless of the schema you use. You may create, edit, and delete tenants at any time. However, careful planning for how you will use your EPIC platform that includes the specific tenant(s) your organization will need now and in the future will help you better plan your entire EPIC installation from the number and type of hosts to the tenants you create once EPIC is installed on those hosts.

Users and Roles

A user consists of the following components:

- Login credentials (username and password)
- Role

Some of the user- and tenant-specific considerations to be resolved when planning/maintaining the EPIC installation include:

- **Tenants:** The number of tenants and the function(s) each tenant performs will determine how many users with the Tenant Administrator role will be needed and, by extension, the number of users with the Tenant Member role will be needed for each tenant. The reverse is also true, since the number and functions of users needing to run jobs can influence how you define tenants. For example, different levels of confidentiality may mandate separate tenants.
- **Job functions:** The specific work performed by a given user will directly impact the EPIC role they receive. For example, a small organization may designate a single user as the Tenant Administrator for multiple tenants, while a large organization may designate multiple Tenant Administrators per tenant.
- **Security clearances:** You may need to restrict access to information based on a user's security clearance. This can impact both the tenant(s) a user has access to and the role configured for that user within a given tenant.

Each user may have a maximum of one role per tenant. For example, if your EPIC installation consists of 20 tenants, then each user may have up to 20 separate roles. A user with more than one role may be a Member of some tenants and a Tenant Administrator of other tenants.

Figure 4 lists the specific functions that can be performed within EPIC and the role(s) that can perform each of those actions. In this table:

- **Permission** stands for the right to perform a given action. Users with specific roles receive specific permissions within EPIC.
- **PA** stands for the Platform Administrator role.
- **TA** stands for the Tenant Administrator role.
- **M** stands for the Member (non-administrative) role.
- An **X** in a column means that a user with the indicated role can perform the indicated action.
- A blank entry means that a user with the indicated role cannot perform the indicated action.

Figure 4: EPIC permissions by platform/tenant role (right)

PERMISSION	PA	TA	M
View a tenant	X	X	
Create a tenant	X		
Edit a tenant	X		
Delete a tenant	X		
Configure tenant storage			
Add an EPIC user	X		
Remove an EPIC user	X		
Grant a user platform role	X		
Revoke a user platform role	X		
Grant a user tenant role	X	X	
Revoke a user tenant role	X	X	
View detailed host information	X		
Manage user authentication	X		
View virtual cluster information	X	X	X
Add a virtual cluster (no constraints)		X	X
Add a virtual cluster (w/constraints)		X	X
Access a cluster in Isolated Mode*		X	X
Edit a virtual cluster		X	X
Delete a virtual cluster		X	X
Manage cluster templates		X	X
Run ActionScripts in a cluster		X	X
View detailed DataTap information	X	X	
Add a DataTap		X	
Edit a DataTap		X	
Remove a DataTap		X	
View summary DataTap information			X
View filesystem mounts	X		X
Manage filesystem mounts		X	
View virtual node information	X	X	X
Migrate virtual nodes	X	X	
Manage EPIC platform configuration	X		
Create, edit, delete flavor definitions		X	
Manage App Store images	X		
Add and manage EPIC Worker hosts	X		
Vacate EPIC hosts to migrate nodes	X		
Manage external host access to EPIC	X		
View global usage/health/metrics	X		
View tenant resource usage	X	X	

**The ability to access isolated clusters depends on the tenant Cluster Superuser Privilege setting specified when creating/editing a tenant.*

User Authentication

Each user has a unique username and password that must be provided in order to log in to BlueData EPIC. Authentication is the process by which EPIC matches the user-supplied username and password against the list of authorized users and determines:

- whether to grant access (stored either in the local user database server or in the remote LDAP/Active Directory server), and
- what exact access to allow, in terms of the specific role(s) granted to that user (stored on the EPIC Controller node).

User authentication information is stored on a secure server. EPIC can authenticate users using any of the following methods:

- An EPIC internal user database.
- An existing LDAP or Active Directory server that you can connect to using Direct Bind or Search Bind.

Accessing EPIC (Non-SSO)

The non-SSO user authentication process is identical when using either the internal EPIC user database or an external LDAP/AD server:

1. A user accesses the EPIC **Login** screen using a Web browser.
   - If tenant-independent authentication is not enabled, the URL will be `jvvr<11>kracfftguu@`, where `>kracfftguu@` is the IP address of the Controller host (if platform HA is not enabled) or the cluster IP address (if platform HA has been enabled).
   - If tenant-independent authentication has been enabled, then the URL will be `jvvr<11>kracfftguu@1 dfuygdwk1nqikpAvgpcpvamg{?>mg{@`, where `>mg{@` is the tenant key (approximately 10 characters long) that was assigned by EPIC when the tenant was created or tenant independent authentication was enabled. The user need only do this the first time they access this tenant; EPIC stores a cookie that will automatically redirect the user to that tenant on subsequent login attempts when they navigate to the `jvvr<11>kracfftguu@` address. If the user has access to multiple tenants, then she or he will need to enter the key of the tenant they wish to access.

Note: If the EPIC platform requires a secure connection, then replace `jvvr` with `jvvru`, as appropriate.

2. The user enters her or his username and password in the appropriate fields and attempts to login.
3. EPIC securely passes the user-supplied username and password to the authentication server.
4. The authentication server returns a response that indicates either a valid (allow user to login) or invalid (prevent user from logging in) login attempt.
5. If the login attempt is valid, EPIC will match the user with the role(s) granted to that user and allow the proper access.

Using the internal user database included with EPIC is fast and convenient from an IT perspective. However, it may complicate user administration for various reasons, such as:

- The user may be required to change their password on the rest of the network but this change will not be reflected in EPIC.
- A user who is removed from the network (such as when they leave the organization) must be independently removed from the EPIC user database.

Connecting EPIC to your existing user authentication server requires you to supply some information about that server when installing EPIC. Contact your user administrator for the following information:

- **LDAP:** LDAP Host, User Attribute, User Subtree DN
- **Active Directory:** AD Host, User Attribute, User Subtree DN

Accessing EPIC (SSO)

Single Sign On (SSO) allows users to supply login credentials once, and then gain access to all authorized resources and applications without having to log in to each application separately. When BlueData EPIC is configured for SSO, authorized users will proceed directly to the **Dashboard** screen without having to log in, by navigating to either of the following, as appropriate:

- `jvvr<11>kracfftguu@`, if tenant independent authentication is not enabled.
- `jvvr<11>kracfftguu@1dfuygdwk1 nqikpAvgpcpvamg{?>mg{@`, if tenant independent authentication has been enabled.

*Note: If tenant-independent authentication has been enabled and the user has access to a single tenant, then navigating to that tenant as described Step 1 of "Accessing EPIC (Non-SSO)" on page 9 will take the user to the **Dashboard** screen for that tenant. If the user has SSO access to more than one tenant, then she or he will need to provide the key of the desired tenant to access the **Dashboard** screen for that tenant. If the user has access to one or more tenant(s) that do not have SSO enabled, then the user will need to supply credentials when accessing the non-SSO tenant(s)*

Configuring EPIC for SSO requires both of the following:

- A metadata XML file that is provided by the Identity Provider (IdP)
- Configuring the XPATH parameter to refer to a location in the SAML response that contains the LDAP/AD username of the authenticated user, such as `11ucon<CvvtkdwvgUvcvgogpv1 ucon<Cvvtkdwvg]BPcog?$RgtuqpKoowvcdngKF$_1 ucon<CvvtkdwvgXcnwg1vgzv*+`

You can then use LDAP/AD groups to assign roles to EPIC users.

For EPIC deployments where platform HA is not enabled, the hostname of the EPIC Controller host must be mapped to the IP address of the Controller host via a DNS server that can be accessed by the user. This allows a user-initiated browser GET request to correctly resolve to the Controller host.

For EPIC deployments where platform HA is enabled, this will be a hostname that resolves to the cluster IP address. Figure 5 shows the DNS name resolution process.

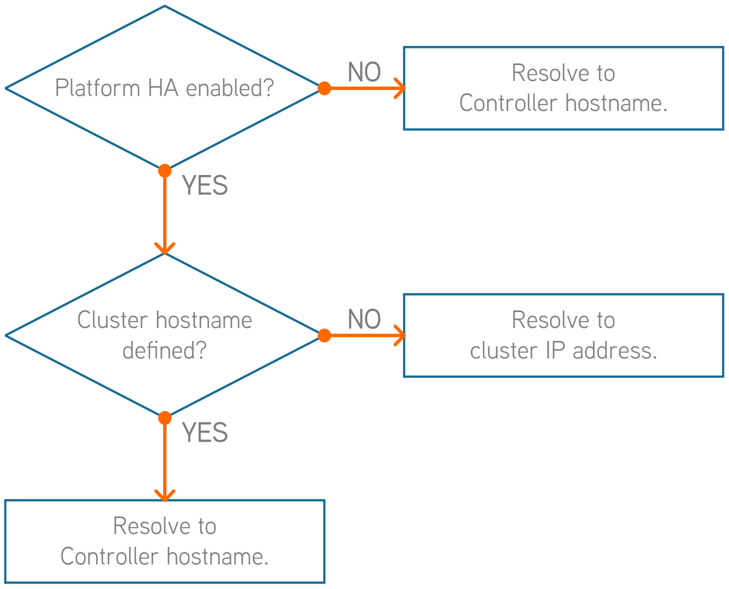

Platform HA enabled?
NO
Resolve to Controller hostname.
YES
Cluster hostname defined?
NO
Resolve to cluster IP address.
YES
Resolve to Controller hostname.

Figure 5: DNS name resolution process

The IdP must be configured with the following information:

- **Audience:** This field is not required; however, providing the base URL of the SAML server is more secure than a blank entry. If you do enter a URL, then this URL must exactly match the **SAML Application Name** that you will specify in EPIC.
- **Recipient:** `]>jquvpcog@~>kracfftguu@_1dfuygdwk1 nqikp`, where `>jquvpcog@` is the name of either the Controller host or HA cluster, as appropriate, and `>kracfftguu@` is either the Controller or cluster IP address. Use either a hostname or an IP address, but not both. For example, `eqpvtqnngtpcog1dfuygdwk1nqikp` or `32054030321dfuygdwk1nqikp`.
- **Consumer URL Validator:** Enter `>grkeakphq@1 dfuygdwk1nqikp1`, where `>grkeakphq@` is either:
  - `0,` - This is a valid generic entry, but is less secure. For example, `0,1dfuygdwk1nqikp1`.
  - `>"pcog"qt"kr@`, which will be either the FQDN or IP address of either the Controller host or HA cluster, as described above. This entry is more secure than the generic entry. For example, `32054020971dfuygdwk1 nqikp1` or `grke/230qticpk|cvkqp0eqo1 dguygdwk1nqikp1`.
- **Consumer URL:** `>grkeakphq@1dfuygdwk1 uconanqikp@`, where `>grkeakphq@` is either a generic or specific entry, as described above.

The IdP will provide a SAML IdP XML metadata file that you will use when configuring EPIC for SSO.

4. Virtual Compute and Memory

The BlueData EPIC software platform allows you to deploy multiple virtual clusters using fully managed and embedded Docker containers. A virtual cluster is a collection of virtual nodes (Docker containers) that are available to a specific tenant. Each virtual node is allocated resources, as described below.

Virtual Cores, RAM, Storage, and GPU Devices

EPIC models virtual CPU (vCPU) cores as follows:

- The EPIC platform license specifies the maximum number of CPU cores that can be utilized by all of the on-premises and/or public cloud hosts in a given EPIC deployment. The use or non-use of CPU hyperthreads does impact the EPIC license and vCPU count.
- The number of available vCPU cores is the number of physical CPU cores multiplied by the CPU allocation ratio specified by the Platform Administrator. For example, if the hosts have 40 physical CPU cores and the Platform Administrator specifies a CPU allocation ratio of 3, then EPIC will display a total of 120 available cores. EPIC allows an unlimited number of vCPU cores to be allocated to each tenant. The collective core usage for all nodes within a tenant will be constrained by either the tenant's assigned quota or the available cores in the system, whichever limit is reached first. The tenant quotas and the CPU allocation ratio act together to prevent tenant members from overloading the system's CPU resources.
- When two or more nodes are assigned to the same host, they contend for the same physical CPU resources of that host. EPIC allocates access to CPU resources to such nodes in a ratio determined by their vCPU core count. For example, a node with 8 cores will receive twice as much CPU time as a node with 4 cores.
- The Platform Administrator can also specify a Quality of Service (QOS) multiplier for each tenant. In the case of CPU resource contention, the node vCPU count is multiplied by the tenant QOS multiplier when determining the physical CPU time that will be granted. For example, a node with 8 vCPU cores in a tenant with a QOS multiplier of 1 will receive the same physical CPU time as a node with 4 vCPU cores in a tenant with a QOS multiplier of 2. The QOS multiplier is used to describe relative tenant priorities when CPU resource contention occurs; it does not affect the overall cap on CPU load established by the CPU allocation ratio and tenant quotas.

EPIC models RAM as follows:

- The total amount of available RAM is equal to the amount of unreserved RAM in the EPIC platform. Unreserved RAM is the amount of RAM remaining after reserving some memory in each host for EPIC services. For example, if your EPIC platform consists of four hosts that each have 128GB of physical RAM with 110GB of unreserved RAM, the total amount of RAM available to share among EPIC tenants will be 440GB.
- EPIC allows an unlimited amount of RAM to be allocated to each tenant. The collective RAM usage for all nodes within a tenant will be constrained by either the tenant's assigned quota or the available RAM in the system, whichever limit is reached first.

EPIC models storage as follows:

- Root disk storage space is allocated from the disk(s) on each Worker host that are assigned as Node Storage disks when adding the Worker to the platform. Each virtual node consumes node storage space equivalent to its root disk size on the Worker host where that virtual node is placed.
- If desired, you may specify 20GB or more of persistent storage space, which EPIC uses to facilitate container migration, as described in "Stateful Container Migration" on page 15.
    - The persistent storage resource is configured using the EPIC web interface.
    - Persistent storage capacity is specified as another resource when defining a flavor. See "Virtual Node Flavors" on page 12.
    - When creating containers using a flavor that includes a persistent storage amount, the persistent storage resource must have an amount of free space equal to the **Persistent Storage Size (GB)** value times the number of container(s) that are being created using that flavor.

If the EPIC platform includes compatible GPU devices, then EPIC models those GPU devices as follows:

- The EPIC platform must be running version 7.x of the RHEL/CentOS operating system; GPU devices are not supported when running any 6.x versions of the RHEL/CentOS OS.
- Deploying an EPIC platform does not automatically install NVIDIA drivers on the hosts. You must install the NVIDIA drivers on the hosts before deploying EPIC.

- The total number of available GPU resources is equal to the number of physical GPU devices in the EPIC platform. For example, if your EPIC platform consists of four hosts that each have 8 physical GPU devices, then the EPIC platform will have a total of 32 GPU devices available to share among EPIC tenants.
- EPIC allows an unlimited number of GPU resources to be allocated to each tenant. The collective GPU resource usage for all virtual nodes within a tenant will be constrained by either the tenant's assigned quota or the available GPU devices in the system, whichever limit is reached first.
- GPU devices are expensive resources. EPIC therefore strives to maximize their use by handling virtual node/container placement as follows:
    - If a virtual node does not require GPU devices, then EPIC attempts to place that node on a host that does not have any GPU devices installed.
    - If a virtual node does require GPU resources, then EPIC attempts to place that container in such a way as to maximize GPU resource utilization on a given host and to reduce/eliminate wasted resources.
    - EPIC does not have the concept of a virtual GPU, nor does it allow sharing the same GPU core between multiple containers simultaneously.

During tenant creation, EPIC will suggest default values in the various quota fields. These default values will be 25% of the total system resources for most fields. The exception to this rule is the quota for GPU devices where the default value is 0. When configuring each resource quota, the EPIC UI displays the total available amount of that resource for comparison. You may edit these quota values or delete a value and leave the field blank to indicate that the tenant does not have a quota defined for that resource.

Assigning a quota of resources to a tenant does not reserve those resources for that tenant when that tenant is idle (not running one or more clusters). This means that a tenant may not actually be able to acquire system resources up to the limit of its configured quota.

- You may assign a quota for any amount of resources to any tenant(s) regardless of the actual number of available system resources. An EPIC deployment where the total amount of configured tenant resource quotas exceeds the current amount of system resources is called over-provisioning. Over-provisioning occurs when one or more of the following conditions are met:
- You have a tenant which has resource quotas that either exceed the system resources or are undefined quotas. This tenant will only be able to obtain the amount of resources that are actually available to the EPIC platform; this arrangement is typically a convenience to make sure that the tenant is always able to fully utilize the platform, even if you add more hosts in the future.
- You have multiple tenants where none have overly large or undefined quotas, but where the sum of their quotas exceeds the resources available to the EPIC platform. In this case, you are not expecting all tenants to attempt to use all their allocated resources simultaneously. Still, you have given each tenant the ability to claim more than its "fair share" of the EPIC platform's resources when these extra resources are available. In this case, you must balance the need for occasional bursts of usage against the need to restrict how much a "greedy" tenant can consume. A larger quota gives more freedom for burst consumption of unused resources while also expanding the potential for one tenant to prevent other tenants from fully utilizing their quotas.

Note: Over-provisioning is useful in certain situations; however, avoiding over-provisioning prevents potential resource conflicts by ensuring that all tenants are guaranteed to be able to obtain their configured quota of virtual CPU cores, RAM, and GPU devices.

Virtual Node Flavors

BlueData EPIC defines a virtual node flavor as the number of vCPU cores, RAM capacity, root disk storage capacity, persistent storage capacity (if any), and number of GPUs (if any) required by a virtual node. For example, if the flavor **small** specifies a single vCPU core and 3GB of RAM, then all virtual nodes created with the **small** flavor will have those specifications, subject to the rules described in "Virtual Cores, RAM, Storage, and GPU Devices" on page 11. EPIC creates a default set of flavors (such as **Small**, **Medium**, and **Large**) during installation.

Note: Flavors are tenant-specific. A flavor that is created or edited in one tenant will not affect a flavor with the same name in another tenant.

The Platform Administrator should create flavors with virtual hardware specifications appropriate to the clusters that tenant members will create. Application characteristics will guide these choices, particularly the minimum virtual hardware requirements per node. Using nodes with excessively large specifications will waste resources (and count toward a tenant's quota). It is therefore important to define a range of flavor choices that closely match user requirements.

The Platform Administrator may freely edit or delete these flavors. When editing or deleting a flavor:

- If you edit or delete an existing flavor, then all virtual nodes using that flavor will continue using the flavor as specified before the change or deletion.
- You may delete all of the flavors defined within your EPIC installation; however, if you do this, then you will be unable to create any new clusters until you create at least one flavor.
- You may specify a root disk size when creating or editing a flavor. This size overrides the default root disk size specified in each App Store image. Specifying a root disk size that is smaller than the minimum size indicated by a given image will prevent you from being able to instantiate that image on a cluster using that flavor. Creating a larger root disk size may slow down cluster creation.

Note: Consider using DataTaps (see "About DataTaps" on page 14*) where possible for optimal performance.*

5. Storage

This section describes how BlueData EPIC uses DataTaps, tenant storage, and node storage.

About DataTaps

DataTaps expand access to shared data by specifying a named path to a specified storage resource. Big Data jobs within EPIC virtual clusters can then access paths within that resource using that name. This allows you to run jobs using your existing data systems without the need to make time-consuming copies or transfers of your data. Tenant Administrator users can quickly and easily build, edit, and remove DataTaps using the DataTaps screen. Tenant Member users can use DataTaps by name.

Each DataTap requires the following properties to be configured, depending on type (NFS, HDFS, or HDFS with Kerberos):

- **Name:** A unique name for each DataTap. This name may contain letters (A-Z or a-z), digits (0-9), and hyphens (-), but may not contain spaces. You can use the name of a valid DataTap to compose DataTap URIs that you give to applications as arguments. Each such URI maps to some path on the storage system that the DataTap points to. The path indicated by a URI might or might not exist at the time you start a job, depending on what the application wants to do with that path. Sometimes the path must indicate a directory or file that already exists, because the application intends to use it as input. Sometimes, the path must not currently exist, because the application expects to create it. The semantics of these paths are entirely application- dependent, and are identical to their behavior when running the application on a physical Hadoop or Spark platform.
- **Description:** Brief description of the DataTap, such as the type of data or the purpose of the DataTap.
- **Type:** Type of file system used by the shared storage resource associated with the DataTap (HDFS, or NFS). This is completely transparent to the end job or other process using the DataTap.
- **Host:** DNS name or address of the service providing access to the storage resource. For example, this could be the NameNode of an HDFS cluster.
- **Share:** For NFS DataTaps, this is the exported share on the selected host.
- **Port:** For HDFS DataTaps, this is the port for the NameNode server on the host used to access the HDFS filesystem.
- **Path:** Complete path to the directory containing the data within the specified NFS share or HDFS file system. You can leave this field blank if you intend the DataTap to point at the root of the specified share/volume/file system.
- **Standby NameNode:** DNS name or IP address of a standby NameNode that an HDFS DataTap will try to reach if it cannot contact the primary host. This field is optional; when used, it provides high-availability access to the specified HFDS DataTap.
- **Kerberos parameters:** If the HDFS DataTap has Kerberos enabled, then you will need to specify additional parameters. EPIC supports two modes of user access/authentication.
    - Passthrough mode passes the individual user's (client principal) Kerberos credentials from the compute Hadoop cluster to the remote HDFS cluster for authentication.
    - Proxy mode permits a “proxy user” to be configured to have access to the remote HDFS cluster. Individual users are granted access to the remote HDFS cluster by the proxy user configuration. Mixing and matching distributions is permitted between the compute Hadoop cluster and the remote HDFS.
    - HDFS file systems configured with TDE encryption as well as cross-realm Kerberos authentication are supported.

The storage pointed to by a BlueData DataTap can be accessed by a Map/Reduce job (or by any other Hadoop-or Spark-based activity in an EPIC virtual node) via a URI that includes the name of the DataTap.

A DataTap points to the top of the “path” configured for the given DataTap. The URI has the following form:

```
fvcr<11fcvcvcrapcog
```

In this example, `fcvcvcrapcog` is the name of the DataTap that you wish to use. You can access files and directories further in the hierarchy by appending path components to the URI:

```
fvcr<11fcvcvcrapcog1uqogauwdfktgevqt{1
cpqvjgtauwdfktgevqt{1uqogahkng
```

For example, the URI `fvcr<11o{fcvcvcr1jqog1 o{fktgevqt{` means that the data is located within the `1jqog1 o{fktgevqt{` directory in the storage that the DataTap named `o{fcvcvcr` points to.

DataTaps exist on a per-tenant basis. This means that a DataTap created for Tenant A cannot be used by Tenant B. You may, however, create a DataTap for Tenant B with the exact same properties as its counterpart for Tenant A, thus allowing both tenants to access the same storage resource. Further, multiple jobs within a tenant may use a given DataTap simultaneously. While such sharing can be useful, be aware that the same cautions and restrictions apply to these use cases as for other types of shared storage: multiple jobs modifying files at the same location may lead to file access errors and/or unexpected job results.

Tenant Storage

Tenant storage is a storage location that is shared by all nodes within a given tenant. The Platform Administrator configures tenant storage while installing EPIC and can change it at any time thereafter. Tenant storage can be configured to use either a local HDFS installation (configured by BlueData EPIC on the host storage) or a remote HDFS or NFS system. Alternatively, you can create a tenant without dedicated storage.

Note: If all tenants are created using the same tenant storage service settings, then no tenant can access the storage space of any other tenant.

When a new tenant is created, that tenant automatically receives a DataTap called "TenantStorage" that points at a unique directory within the Tenant Storage space. This DataTap can be used in the same manner as other DataTaps, but it cannot be edited or deleted. This does not apply if no tenant storage has been defined (meaning that you selected None for tenant storage when installing EPIC).

The **TenantStorage** DataTap points at the top-level directory that a tenant can access within the tenant storage service. The Tenant Administrator can create or edit additional DataTaps that point at or below that directory.

If the tenant storage is based on a local HDFS, then the Platform Administrator can specify a storage quota for each tenant. EPIC uses the HDFS back-end to enforce this quota, meaning that the quota applies to storage operations that originate from both the EPIC DataTap browser or the nodes within that tenant.

EPIC creates root Tenant Storage folders under the deployment global Tenant Storage root. For example, given a global Tenant Storage root of `1c1d.` EPIC will create tenant-specific Tenant Storage root directories such as `1c1d13` for Tenant 1 and `1c1d1 4` for Tenant 2. EPIC versions 3.4 and previous restricted a tenant's ability to create DataTaps to the global Tenant Storage to subdirectories within that tenant's Tenant Storage root directory.

- EPIC versions 3.5 and later allow you to create DataTaps that point to any subdirectory within the global Tenant Storage root, so long as that location cannot access another tenant's Tenant Storage root directory, nor the global Tenant Storage root. For example:
- You could create a DataTap in Tenant 1 that points to `1c1d1UjctgfUvqtcig`, because that directory is not part of any existing tenant's Tenant Storage root.
- You will also be able, as Tenant 2, to create another DataTap that points to `1c1d1UjctgfUvqtcig`, thereby allowing data sharing between Tenant 1 and Tenant 2.

Note: Tenant 2 cannot create a DataTap to the `1c1d131UjctgfUvqtcig` *directory, because the* `1c1d13"` *directory is the root Tenant Storage directory for Tenant 1.*

Users who have a Tenant Administrator role may view and modify detailed DataTap information. Members may only view general DataTap information and are unable to create, edit, or remove a DataTap.

Note: Data conflicts may occur if more than one DataTap points to a location being used by multiple jobs at once.

Node Storage

Node storage is built from the local storage in each host in the EPIC deployment and is used for the disk volumes that back the local storage for each virtual node. Using SEDs (Self-Encrypting Drives) will ensure that any data written to node storage is encrypted on write and decrypted on read by the OS. A tenant can optionally be assigned a quota for how much storage the nodes in that tenant can consume.

BlueData instances running on public cloud VMs (such as AWS EC2) utilize storage within the instance (such as AWS Elastic Block Storage, or EBS) as node storage.

Stateful Container Migration

BlueData EPIC allows you to specify an external persistent storage pool. Persistent storage exists on a remote storage resource that is pointed to but not managed by EPIC. You can create, expand, and shrink storage capacity just as you would any other resource. This feature allows you to migrate containers between hosts by (default) preserving the following critical container folders for ongoing use:

- /usr
- /opt

- /var
- /etc
- /home

The contents of other folders can be preserved during container migration by specifying their names in the metadata JSON file when creating a new application image generated using the App Workbench.

Use Cases

Big Data applications such as Hadoop and Spark offer robust high availability capabilities; however, some enterprise customers have operational requirements that require the ability to move virtual nodes/containers from one host to another. These operational requirements include:

- An EPIC host crashing and the containers that were running on that host must be redeployed on other working EPIC hosts with minimal downtime and no additional configuration required.
- One or more EPIC host(s) needs to be replaced for maintenance and/or as part of a server refresh cycle. In this scenario, all of the containers running on those hosts must be seamlessly moved to other EPIC hosts that are not being replaced, with minimal downtime to the applications running in the containers.
- Resolving a condition where there is an inability to meet the SLA for an application running in a containerized clusters (e.g. Spark or Hadoop) due to poor performance (CPU, network, or storage). This is a resource contention (bottleneck) condition that requires a re-balancing of virtual nodes onto EPIC hosts with more available resources.

Enabling Container Migration

Once you have enabled persistent storage, the next step is to create one or more flavor(s) that include at least 20GB of persistent storage. Containers created using a flavor with persistent storage enabled will be preserved as described above. You may also assign a persistent storage quota to EPIC tenants. You may create a flavor that specifies persistent storage even if no persistent storage has been defined; however, EPIC will return an error if you attempt to use this flavor before enabling persistent storage. Further, containers created using a flavor that does not have persistent storage enabled will not benefit from this feature, even if you later edit the flavor to enable persistent storage.

Note: The persistent storage resource must have enough free capacity to accommodate the sum of all tenant persistent storage quotas or to accommodate the amount of persistent storage specified in all applicable flavor(s) times the number of containers that use the flavor(s), whichever is greater. Further, if you specify a per-tenant persistent storage quota, then that quota must be large enough to accommodate the flavor-defined persistent storage times the number of containers using the applicable flavor(s).

There are two ways to use persistent storage to migrate a container:

- **Worker Vacate:** If a Worker host goes down and some or all of the containers on that host use persistent storage, then you can perform a Worker vacate function. All jobs running on the affected container(s) will end, but the containers themselves will be recovered as follows:
  - EPIC will not place any new containers on the affected host.
  - The protected containers are removed from the affected host.
  - EPIC automatically migrates those containers to one or more different EPIC host(s), provided that those EPIC hosts have sufficient available resources. All applicable virtual node placement constraints, as described in "Tenant and Host Tags" on page 23 are enforced.
- **Node Migration:** This use case applies to a scenario where the hosts are functioning properly but are overburdened. In this case, the Tenant Administrator or Platform Administrator can add new hosts to the EPIC platform. Containers can then be migrated to the new host(s) on a container-by-container basis. Placement constraints apply to this type of container migration as well.

The following storage systems are supported for EPIC persistent storage:

- CEPH RBD
- NFS
- ScaleIO

Migrating a container (virtual node) has the following effects:

- The cluster to which the container belongs will be stopped and unavailable until the migration completes.
- Any jobs or ActionScripts running on a cluster with one or more migrating container(s) will be lost and must be run again after completing the migration.
- Any data residing in non-persistent storage directories of a container being migrated will be lost.

- Any external host(s) that have access to the EPIC platform will be unable to access the affected container(s) until the migration process completes.
- Migrated containers maintain their configuration and IP addresses.

Application Path Inputs

Applications running within EPIC virtual clusters will typically accept input arguments that specify one or more storage paths. These paths identify files or directories that either currently exist or are intended to be created by the application. Each path can be specified in one of three ways:

- You can use a UNIX-style file path, such as `1jqog1 fktgevqt{`. This path refers to a location within a file system that is local to either the container itself or to the HDFS file system running within the containerized cluster, if one exists. Remember that data stored in any local file systems or cluster local HDFS file system will be deleted when the cluster is deleted.
- You can use a DataTap URI, such as `fvcr<11o{fcvcvcr1 jqog1o{fktgevqt{`. This path refers to a location within the storage system pointed to by the named DataTap within the same tenant. A Hadoop or Spark application will “mount” a given DataTap before using any paths within that DataTap. If the DataTap changes or is deleted, a running application will be affected until it attempts to mount that DataTap again.
- You can use a `u5c<11>Dwemgv"Pcog@1>hqnfgt"pcog@` path, such as `u5c<11nqifcvc142391`, to access Amazon S3 storage. Storage from other public cloud providers can be accessed in a similar fashion.

Please see the technical white paper BlueData EPIC Storage Options for additional details about how EPIC makes storage datasets available to containerized clusters.

6. Networking

This section describes how BlueData EPIC manages networking, including public and private container networks, subnets, and Gateway hosts.

Networks and Subnets

EPIC operates on two networks, as shown in Figure 6:

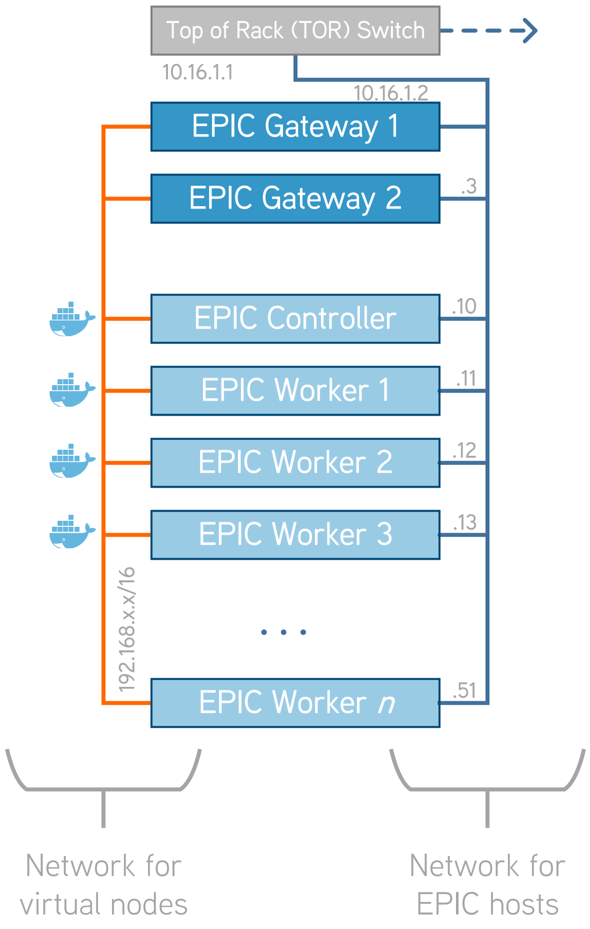


Figure 6: Dual networks on the BlueData EPIC platform

The two networks are laid out as follows:

- **Network for the physical EPIC Controller, Worker, and Gateway hosts:** This network must be both routable and part of the organization's network that is managed by that organization's IT department. All of the physical hosts must be routable. If the EPIC Platform HA feature is enabled (see "High Availability" on page 25), then both the Primary Controller and Shadow Controller must be in the same subnet. If different subnets are used for the Worker hosts, then all subnets used must share the same path MTU setting; Gateway hosts can use subnets with different MTU settings. See "Multiple Subnets" on page 20.
- **Network for virtual nodes (Docker containers):** EPIC creates and manages this network, which can be either public (routable) or private (non-routable). See "Private (Non-Routable) Virtual Node Network" on page 18 and "Public (Routable) Virtual Node Network" on page 18

Private (Non-Routable) Virtual Node Network

Private, non-routable virtual node networks keep the virtual node IP addresses private and hidden within the private network. EPIC replaces the IP addresses of outgoing and incoming packets, as shown in Figure 7 on the following page.

BlueData EPIC software is deployed on a set of hosts. Each host has an IP address and FQDN such as Host IP1, Host IP2, etc. Hosts are typically deployed as one or more rack(s) of servers that are connected to an external switch for access to other networks in the organization (e.g. end user network etc).

BlueData EPIC provisions clusters of embedded, fully-managed Docker containers. Each cluster spins up within a tenant and receives distinct assigned IP addresses and FQDNs from a private IP range (or optionally from a routable IP range provided by the network teams), which appear in the diagram above as BD IP1, BD IP2, etc. Containers in one tenant do not have network access to containers in a different tenant despite potentially being placed on the same host.

End user access to services in the containers (such as SSH or web applications) is routed through a Gateway host that runs the HAProxy service. This access is purely for control traffic. All other traffic, including access to remote HDFS or other enterprise systems such as Active Directory (AD), MIT KDC (Kerberos provider), SSO (Identity providers), and Certificate Authority (CA), is performed via the host network interface, as opposed to the Gateway host.

Public (Routable) Virtual Node Network

Network traffic in a public EPIC virtual node network flows as shown in Figure 8 on the following page, using a routable IP range for containers. In this case, the BlueData EPIC platform does not require Gateway hosts. All the key functionality is identical to the recommended approach for a non-routable private IP range (see "Private (Non-Routable) Virtual Node Network" on page 18). This approach allows the Docker containers to directly access the external network via the host network interface connector (NIC).

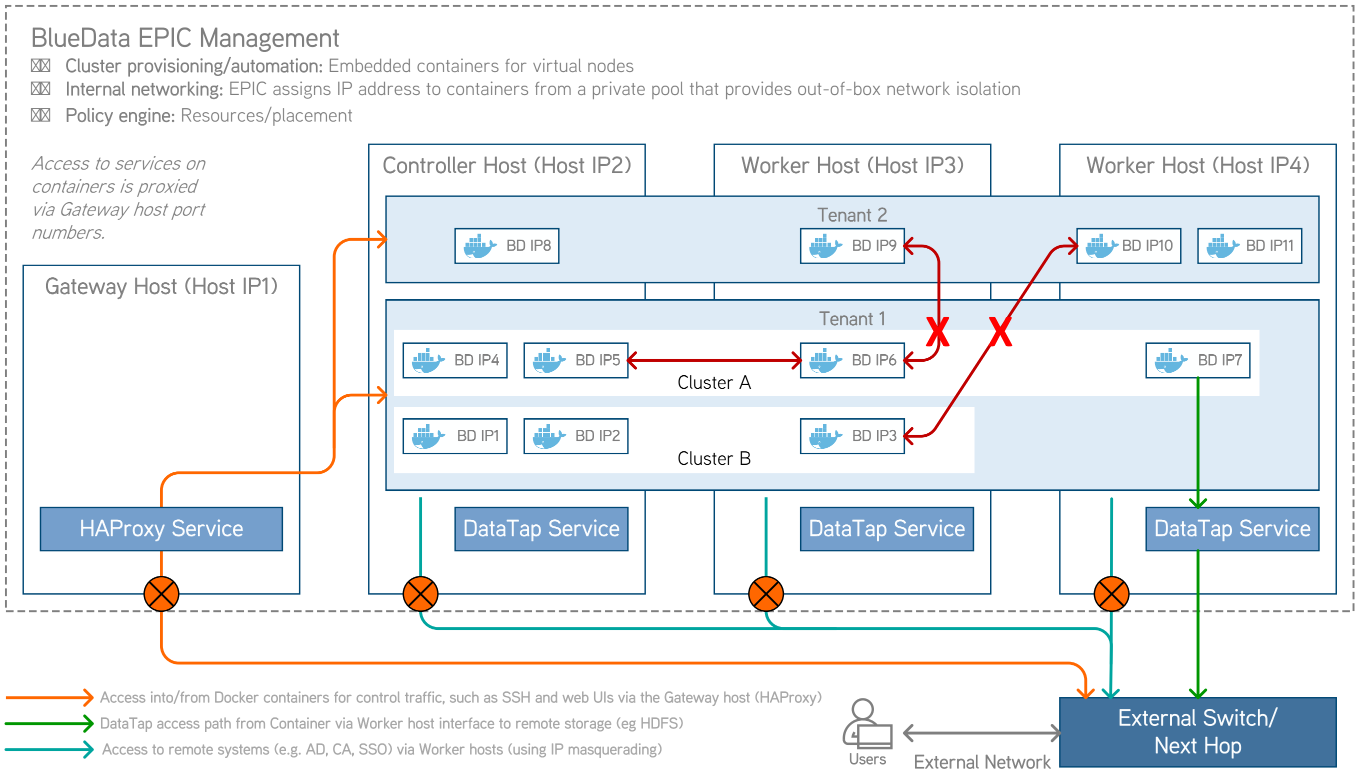

BlueData EPIC Management
Cluster provisioning/automation: Embedded containers for virtual nodes
Internal networking: EPIC assigns IP address to containers from a private pool that provides out-of-box network isolation
Policy engine: Resources/placement
Access to services on containers is proxied via Gateway host port numbers.
Gateway Host (Host IP1)
HAProxy Service
Controller Host (Host IP2)
Worker Host (Host IP3)
Worker Host (Host IP4)
Tenant 2
BD IP8
BD IP9
BD IP10
BD IP11
Tenant 1
BD IP4
BD IP5
BD IP6
BD IP7
Cluster A
BD IP1
BD IP2
BD IP3
Cluster B
DataTap Service
DataTap Service
DataTap Service
Access into/from Docker containers for control traffic, such as SSH and web UIs via the Gateway host (HAProxy)
DataTap access path from Container via Worker host interface to remote storage (eg HDFS)
Access to remote systems (e.g. AD, CA, SSO) via Worker hosts (using IP masquerading)
Users
External Network
External Switch/ Next Hop

Figure 7: BlueData EPIC platform configured to use a private, non-routable virtual node network.

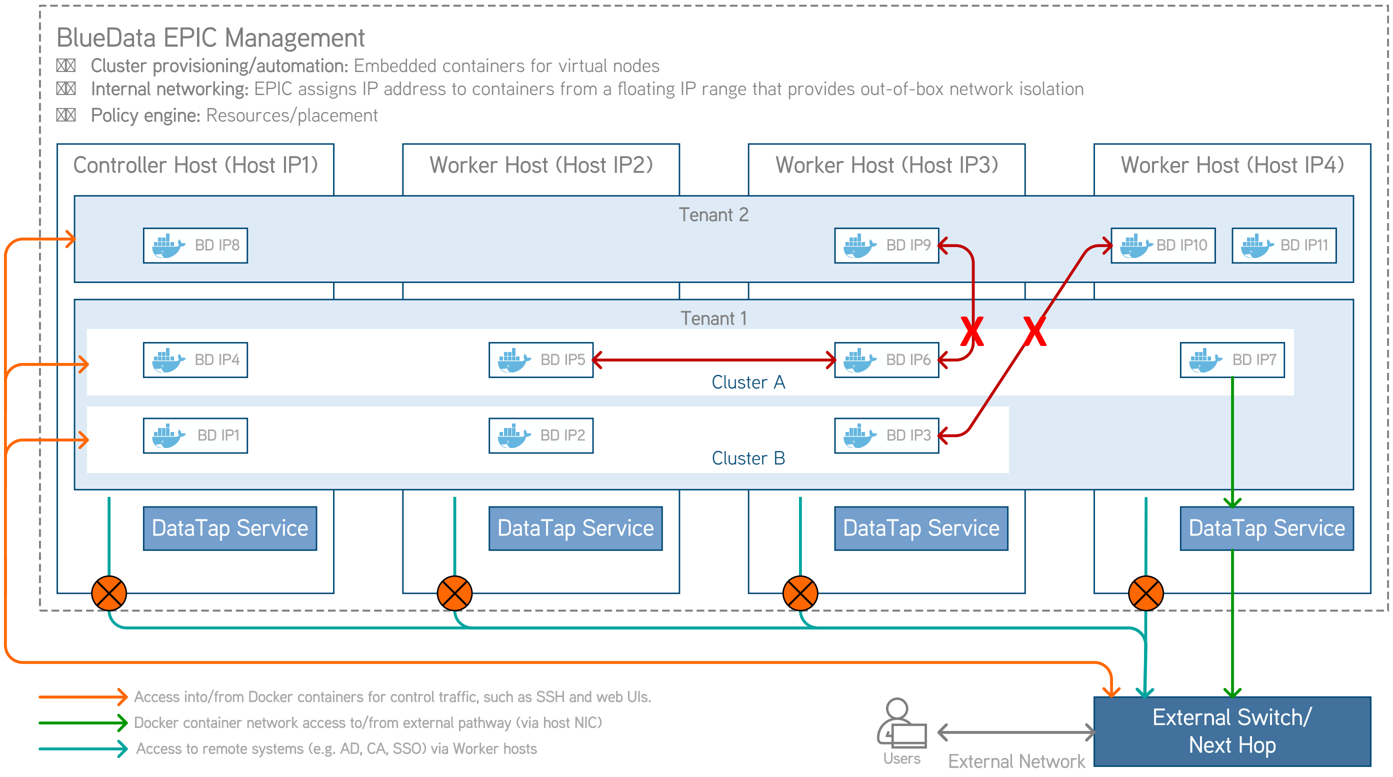

BlueData EPIC Management
Cluster provisioning/automation: Embedded containers for virtual nodes
Internal networking: EPIC assigns IP address to containers from a floating IP range that provides out-of-box network isolation
Policy engine: Resources/placement
Controller Host (Host IP1)
Worker Host (Host IP2)
Worker Host (Host IP3)
Worker Host (Host IP4)
Tenant 2
BD IP8
BD IP9
BD IP10
BD IP11
Tenant 1
BD IP4
BD IP5
BD IP6
BD IP7
Cluster A
BD IP1
BD IP2
BD IP3
Cluster B
DataTap Service
DataTap Service
DataTap Service
DataTap Service
Access into/from Docker containers for control traffic, such as SSH and web UIs.
Docker container network access to/from external pathway (via host NIC)
Access to remote systems (e.g. AD, CA, SSO) via Worker hosts
Users
External Network
External Switch/ Next Hop

Figure 8: BlueData EPIC platform configured to use a public, routable virtual node network

Multiple Subnets

Figure 9, below, illustrates a sample EPIC deployment using multiple subnets.

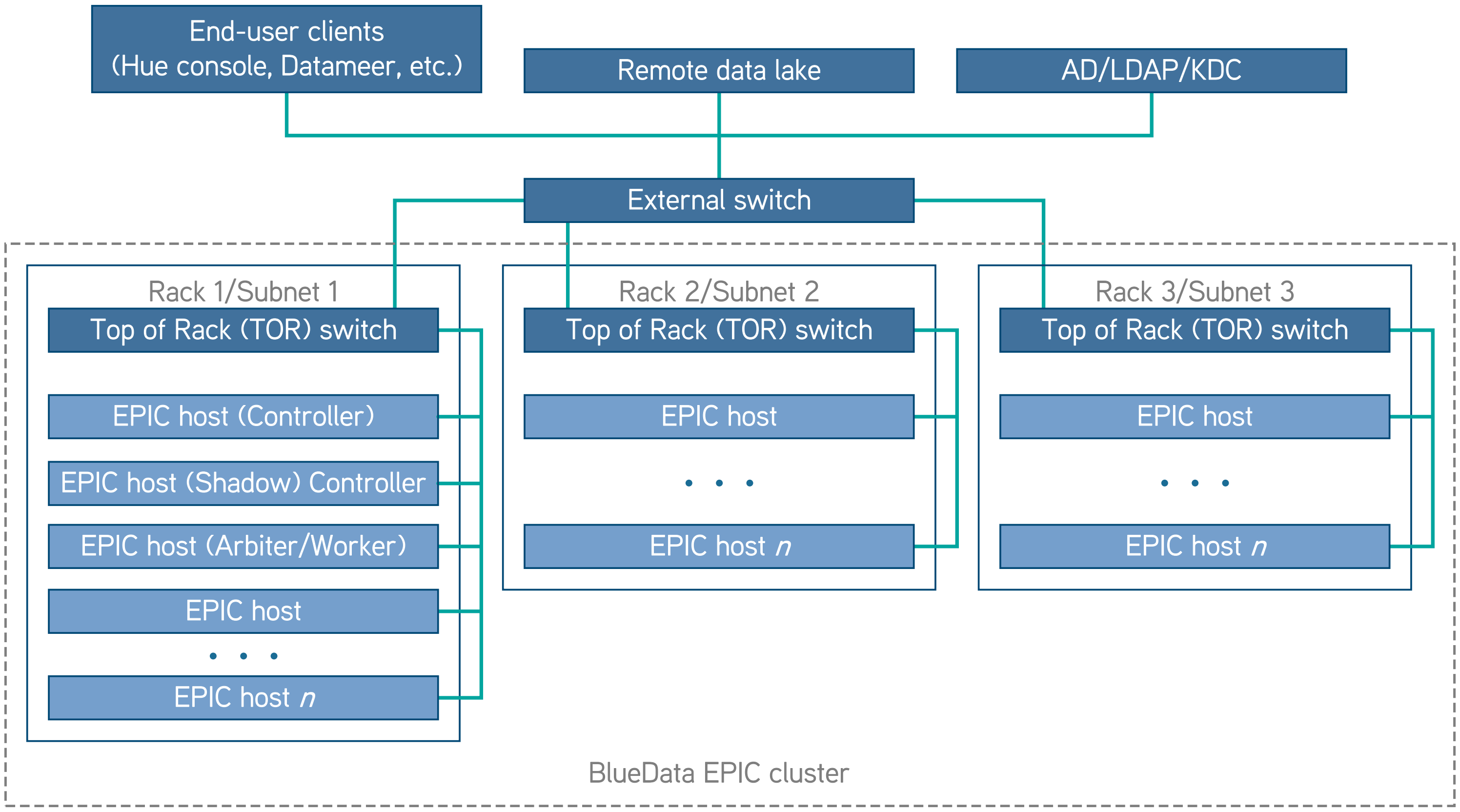


Figure 9: BlueData EPIC platform that spans multiple subnets

When multiple subnets are used:

- The EPIC hosts may be located on-premises and/or in a public cloud. For example, EPIC hosts can reside on multiple racks and/or can be virtual machines residing on cloud-based services (such as AWS, Azure, or GCP).
- If the EPIC platform includes cloud-based hosts, then the container network must be private and non-routable.
- All of the subnets used by EPIC Worker hosts must share the same path MTU setting.
- The subnet(s) used by EPIC Gateway hosts may have different path MTU settings.

Gateway Hosts

A Gateway host is an optional first-class role that is managed by EPIC in a manner similar to the Controller and Worker hosts. One or more Gateway host(s) are required when the IP addresses used by the virtual nodes in the EPIC platform are private and non-routable, meaning that the virtual nodes cannot be accessed via the corporate network. Gateway hosts must conform to all applicable requirements listed in the EPIC documentation. Unlike Worker hosts, Gateway hosts do not run containers/virtual nodes. Instead, they enable access to user-facing services such as Hue console, Cloudera Manager, Ambari, and/or SSH running on containers via High Availability Proxy (HAProxy). You can configure multiple Gateway hosts with a common Fully Qualified Domain Name (FQDN) for round-robin load balancing and High Availability. You may also use a hardware load balancer in front of the multiple Gateway hosts, and can also configure one or more custom port range(s) between 10000 and 50000 for use as proxies.

The ability to run EPIC virtual nodes in a private, non-routable network can drastically reduce the routable IP address pool requirement. For example, a /16 private network can support thousands of containers. The corporate network need only manage the physical host addresses (for example, 10.16.1.2-51), while EPIC Gateway hosts provide access to the virtual nodes running on those hosts. All control traffic to the virtual nodes/ Docker containers from end-user devices (browsers and command line), such as https, SSH, and/or AD/KDC, goes through the Gateway host(s), while all traffic from the virtual nodes/ Docker containers is routed through the EPIC hosts on which those nodes/containers reside. Every container spawned within the EPIC platform is assigned a private IP address that is managed by the EPIC private network.

Support for multiple subnets increases Gateway host flexibility. For example, you can use "small" virtual machines that meet all Gateway host requirements located on different racks or in different areas of your network, instead of having to place these hosts on the same rack as Compute hosts. This can help optimize resource usage.

Figure 10 (right) displays the physical architecture of an EPIC platform with Gateway hosts. Figure 11 (below) displays the logical architecture of an EPIC platform with Gateway hosts.

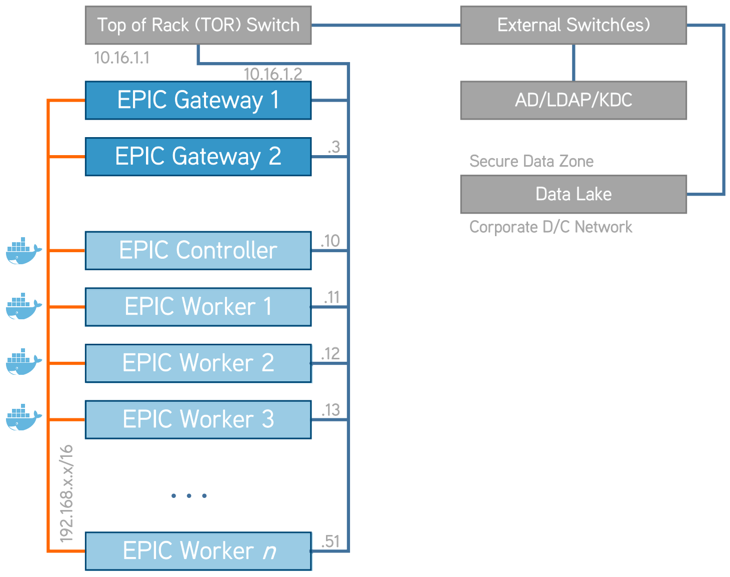


Figure 10: Physical EPIC platform architecture with Gateway hosts

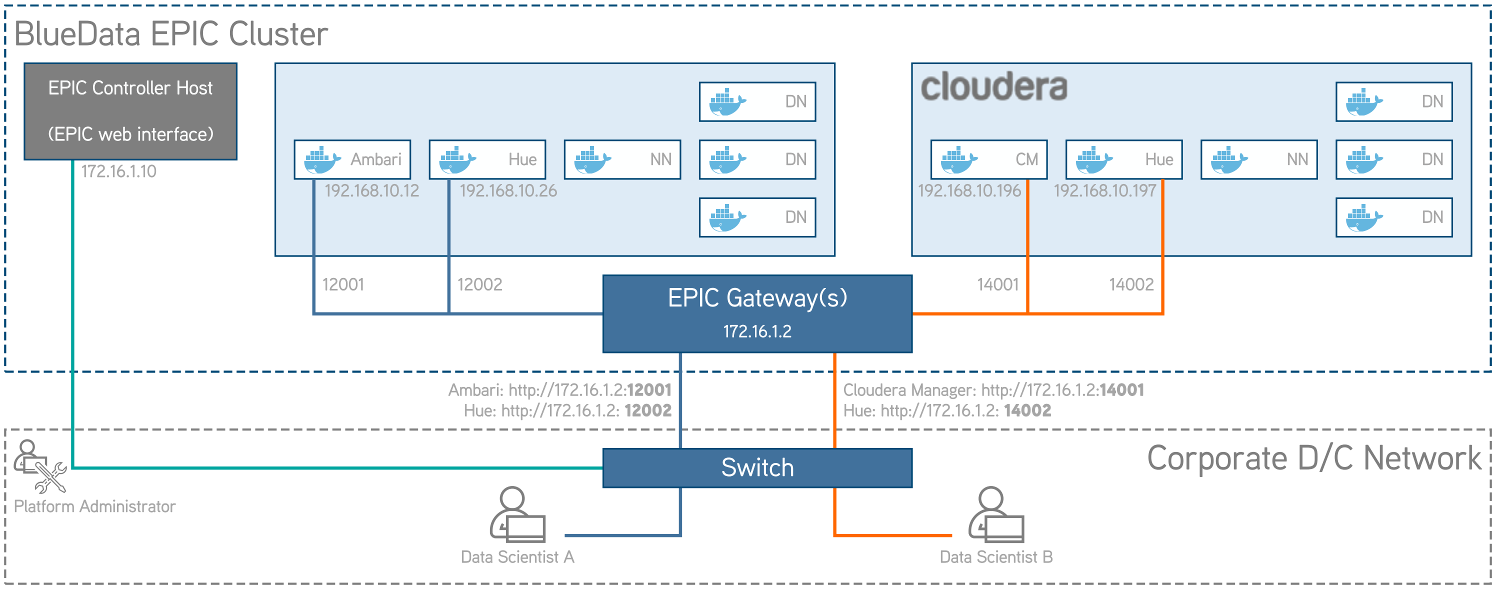


Figure 11: Logical EPIC platform architecture with Gateway hosts

Please see the technical white paper BlueData EPIC Network Design for additional details about the networking design and network configurations for the BlueData EPIC software platform.

7. Cluster Management

This section describes how BlueData EPIC creates clusters, as well as the cluster Isolated Mode, and host tag concepts.

Cluster Creation

EPIC performs the following tasks when you create a cluster:

1. The EPIC Controller checks whether cluster creation would violate any tenant quotas or licensing limits.
2. The Controller checks whether enough total free system resources are available for cluster creation.
3. The Controller chooses the host(s) in the EPIC platform on which to deploy virtual nodes. The hosts are selected based on node affinity, scheduling constraints, matching host tag(s) (see "Tenant and Host Tags" on page 23), and the per-host free resources.
4. The Controller copies the appropriate application image onto each chosen host if it is not already available there.
5. The EPIC Host Agent service runs on each host and manages the lifecycle of the virtual nodes (containers) deployed on that host. This service creates a Docker container for each virtual node and then configures the networking and storage for the node(s).
6. The Controller copies the Node Agent package into each virtual node, and then logs into those containers via SSH to install and start that agent.
7. The Node Agent in each virtual node registers the node with its host's DataTap service and requests the application setup package and cluster metadata from either the Controller or the specified Docker-compatible registry, as appropriate.
8. The Node Agent unpacks the application setup package, and then runs the extracted startscript.
9. The Node Agent begins reporting cluster configuration status and services status to the Controller.
10. The startscript in each virtual node queries `dfaxenk` namespaces to obtain information from the cluster metadata, which includes the IP addresses and DNS names of nodes in the cluster, node roles, service definitions, etc.
11. The startscript performs final runtime configuration appropriate for the application.
12. The startscript registers application services with the Node Agent to identify services that must be started every time the node is (re)started.
13. After the startscript has succeeded on all nodes, the Node Agent reports a "ready" cluster status to the Controller, and the startscript output from each node is uploaded to the Controller to be displayed as the cluster setup logs in the EPIC interface and API.
14. If the EPIC platform is configured to use LDAP/AD user authentication, then an additional post-config setup takes place to allow users to access the containers.

Figure 12 on the following page illustrates how this works.

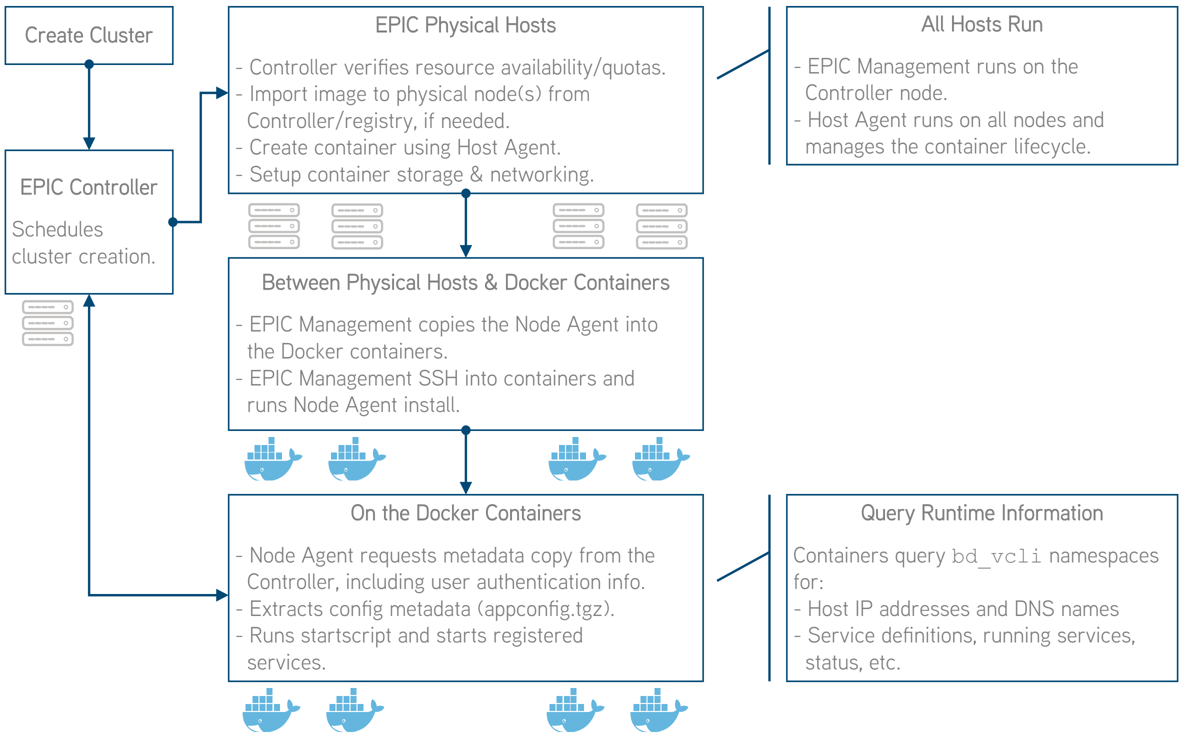
Isolated Mode

There may be times when you need to limit access to a cluster in order to perform configuration or maintenance. Clusters running RHEL/CentOS 7.x can be placed into Isolated Mode. Running a cluster in Isolated Mode allows authorized users to access clusters for maintenance or updates using either automated scripts or manual commands, while preventing other users from accessing the services running in that cluster. Clusters can be placed into Isolated Mode at any time. For example, you can create a new cluster in Isolated Mode and then configure Kerberos protection before making that cluster available for tenant-wide access, thereby avoiding any security issues that may arise from running an unprotected cluster. You may also specify a "bootstrap" ActionScript that will automatically run as soon as a new cluster comes up.

Placing a cluster into Isolated Mode limits access to only those users who have access to the keypair based on the Cluster Superuser Privilege setting for the tenant. If LDAP/AD is used for authenticating EPIC users, then these users may also log in with their LDAP/AD credentials instead of using the tenant keypair.

Placing a cluster into Isolated Mode has the following effects:

- None of the cluster ports except SSH can be accessed from outside the cluster.
- Only authorized users (see above) can access the cluster via SSH or make changes via the EPIC interface or API, such as configuration, power, ActionScripts, and/or cluster deletion.



Create Cluster
EPIC Controller
Schedules cluster creation.
EPIC Physical Hosts
- Controller verifies resource availability/quotas.
- Import image to physical node(s) from Controller/registry, if needed.
- Create container using Host Agent.
- Setup container storage & networking.
All Hosts Run
- EPIC Management runs on the Controller node.
- Host Agent runs on all nodes and manages the container lifecycle.
Between Physical Hosts & Docker Containers
- EPIC Management copies the Node Agent into the Docker containers.
- EPIC Management SSH into containers and runs Node Agent install.
On the Docker Containers
- Node Agent requests metadata copy from the Controller, including user authentication info.
- Extracts config metadata (appconfig.tgz).
- Runs startscript and starts registered services.
Query Runtime Information
Containers query bd_vcli namespaces for:
- Host IP addresses and DNS names
- Service definitions, running services, status, etc.

Figure 12: On-premises container orchestration

- Non-authorized users can place a cluster into Isolated Mode, but only authorized users can cause the cluster to exit this state.
- Any SSH connections that are in progress when the cluster is placed into Isolated Mode will be terminated.
- Users cannot access cluster services via the Node(s) Info tab of the **Cluster Details** screen.
- A cluster can be configured to require two-step deletion. If this option is selected, then an attempt to delete a cluster will place that cluster into Isolated Mode. An authorized user can then delete that cluster once they determine that it is proper to do so.
- Any jobs being run by the cluster when placed into Isolated Mode will continue; however, no new jobs can be run in this cluster until it has exited Isolated Mode.
- Isolated Mode events (enabling and disabling this mode) are recorded in the **Cluster History** tab of the **Cluster Details** screen.

Tenant and Host Tags

Host Tags are a method of permitting you to control which EPIC Worker host(s) a given container will be deployed to. Tags are a node role-specific placement constraint. For example, you could use tags to:

- Deploy specific application clusters only to those EPIC hosts containing SSD storage. Tags such as `uuf?vtwg` or `uvqtcig?uuf/qpn{` would be used to do this.
- Isolate virtual clusters of a specific tenant onto specific EPIC hosts for physical isolation in addition to network isolation, using tags such as `vgpcpv?octmgvkpi` or `vgpcpv?hkpcpeg`.
- Differentiate between cloud based EPIC Worker hosts and on-premises based Worker hosts in a hybrid EPIC deployment, using tags such as `Qprtgo?vtwg` or `nqecvkqp?qprtgo`.

The general process of creating and using tags is:

1. Create one or more tag(s). Tag names are arbitrary and may include special characters and/or spaces; however, best practice is to assign consistent tag names.
2. Associate the tag(s) with one or more Worker host(s), and then specify the tag values for each host. Values are arbitrary; however, best practice is to assign consistent values to each tag.

You may then:

- Create or edit a placement constraint for a node role where you can force EPIC to place containers with that role on a host where the tag equals or does not equal a given value. For example:
    - You could add a placement constraint that the node role must be placed on a Worker host with the tag `erwurggf` set to `4049IJ\` by specifying `crwurggf"Gswcnu" 4049IJ\`.
    - You could add a similar constraint that the node role must be placed on a host with the tag `erwurggf` set to a value that is not `4049IJ\` by specifying `erwurggf"Pqv" Gswcn"4049IJ\`.
    - You could specify that the node role must be placed on unique hosts with the `tcempwodgt` tag by specifying `tcempwodgt"Wpkswg`. In this example, the first virtual node in the cluster with the specified role may be placed on any host with the `tcempwodgt` tag set to any value (such as `3`), the second virtual node with the same role will be placed on any host with the `tcempwodgt` tag to set to any value that is not `3`, and so forth. If the cluster has more virtual nodes with the specified role than there are hosts with unique values, then attempting to create a cluster will fail.
    - You could specify that the node role must be placed on any host(s) that have the `erwurggf` tag set to any value by specifying `erwurggf"Ku"Ugv`.
    - You could specify that the node role must be placed on any host(s) that do not have the `erwurggf` tag set by specifying `erwurggf"Ku"Pqv"Ugv`.
- Add tags to a tenant. This allows you to control the tag(s) and value(s) that the user of a given tenant sees when creating or editing clusters within that tenant. For example, you could select the `erwurggf` tag and the `4049IJ\` value. A user who adds a placement constraint to a cluster in this tenant using the `erwurggf` tag will only see the `4049IJ\` value, regardless of how many values the tag has.
- Add placement constraints to a tenant. This functions identically to creating a placement constraint for a cluster; however, all of the clusters created within the tenant must conform to the tenant-level placement constraints in addition to any cluster-level placement constraints. The EPIC Worker host(s) must have sufficient resources that meet all placement criteria in order for you to be able to create clusters within the tenant.

8. High Availability

BlueData EPIC supports three levels of High Availability protection:

- **Platform High Availability:** This level of protection only applies to the EPIC platform services.
- **Virtual Cluster High Availability:** This level of protection applies to the Big Data and AI/ML/DL clusters running on the EPIC platform
- **Gateway Host High Availability:** This level of protection applies to the Gateway host(s).

Platform High Availability

EPIC supports platform-level High Availability functionality that protects your EPIC platform against the failure of the Controller host. Your EPIC platform must include at least three hosts that conform to all of the requirements listed in the EPIC documentation in order to support this feature, as shown in Figure 13.

Platform-level High Availability requires designating two Worker hosts as the Shadow Controller and Arbiter, respectively. If the Controller host fails, then EPIC will failover to the Shadow Controller host within approximately two or three minutes, and a warning will appear at the top of the EPIC screens. You may need to log back in to the EPIC interface if this occurs.

Each host in an EPIC deployment has its own IP address. If the Controller host fails, then attempting to access the Shadow Controller host using the same IP address will fail. Similarly, accessing the Shadow Controller host using that host's IP address will fail once the Controller host recovers. To avoid this problem, you must specify a cluster IP address that is bonded to the node acting as the Controller host, and then log into EPIC using that cluster IP address. EPIC will automatically connect you to the Controller host (if the EPIC platform is running normally) or to the Shadow Controller host with a warning message (if the Controller host has failed and triggered the High Availability protection).

Platform-level EPIC High Availability protects against the failure of any one of the three hosts being used to provide this protection. The warning message will therefore appear if either the Shadow Controller or Arbiter host fails, even if the Controller host is functioning properly.

When platform High Availability is enabled:

- The Controller host manages the EPIC platform, and the Shadow Controller and Arbiter hosts function as Worker hosts.
- If the Controller host fails, then the Arbiter host switches management of the EPIC Platform to the Standby Controller host, and the EPIC platform is no longer protected against any failure of the Shadow Controller host(s).
- If the Arbiter host fails, then the Controller host continues working, and the EPIC platform is not protected against any further failure of the Controller or Shadow Controller host(s).

When a failure of a High Availability host occurs, EPIC takes the following actions:

- If the Controller host has failed, then EPIC fails over to the Standby Controller host and begins running in a degraded state. This process usually takes 2-3 minutes, during which you will not be able to log in to EPIC.
- If the Shadow Controller or Arbiter host fails, then EPIC keeps running on the Controller host in a degraded state.
- A message appears in the upper right corner of the EPIC interface warning you that the system is running in a degraded state. Use the Service Status tab of the Platform

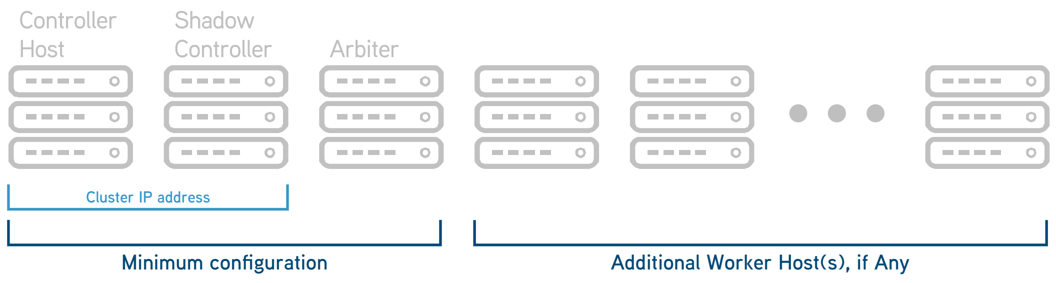


Figure 13: BlueData EPIC platform configured for High Availability

- EPIC analyzes the root cause of the host failure and attempts to recover the failed host automatically. If recovery is possible, then the failed host will come back up and EPIC will resume normal operation.
- If EPIC cannot resolve the problem, then the affected host will be left in an error state and you will need to manually diagnose and repair the problem (if possible) and then reboot that host. If rebooting solves the problem, then the failed host will come back up and EPIC will resume normal operation. If this does not solve the problem, then you will need to contact BlueData Technical Support for assistance.

Enabling platform High Availability protection in EPIC is a two-stage process: First, you must designate one Worker host as the Shadow Controller host in the Installation tab of the EPIC Installation screen. This is a one-time assignment; you cannot transfer the Shadow Controller role to another Worker host. Second, you must enable High Availability protection and then assign the Arbiter role to a third Worker host in the HA Settings tab of the Settings screen.

EPIC does not support enabling platform High Availability if any virtual clusters already exist. If you create one or more virtual clusters before deciding to enable High Availability, then you should delete those clusters before proceeding with the High Availability. In general, you should implement High Availability just after first installing EPIC.

Virtual Cluster High Availability

Some Big Data applications allow you to create cluster with High Availability protection. This is separate and distinct from the EPIC platform High Availability described above, as follows:

- A cluster with High Availability enabled for that virtual cluster is still dependent on the EPIC platform. If the Controller host of an EPIC deployment without High Availability fails, then the virtual cluster will also fail.
- If the EPIC deployment has High Availability enabled and the Master node of a virtual cluster fails, then that virtual cluster will fail if High Availability for that virtual cluster was not enabled when that virtual cluster was created.

Figure 14 displays the relationship between virtual cluster and EPIC platform High Availability protection under a variety of scenarios.

Gateway Host High Availability

You can add redundancy for Gateway hosts by mapping multiple Gateway host IP addresses to a single hostname. When this is done, then either the DNS server or an external load balancer will load-balance requests to the hostname among all of the Gateway hosts on a round-robin basis. This ensures that there is no single point of failure for the Gateway host.

FAILURE TYPE	NO HA	CLUSTER HA ONLY	EPIC HA ONLY	BOTH HA
EPIC Controller host down; virtual cluster Master node up.	X	X	protected	protected
EPIC Controller host up; virtual cluster Master down.	X	protected	X	protected
EPIC Controller host up; Other virtual cluster node down.	protected	protected	protected	protected

Figure 14: BlueData EPIC High Availability protection under various scenarios

Appendix

This Appendix defines the key terms used in this white paper. It also describes how BlueData EPIC supports Big Data applications in the Hadoop and Spark ecosystem, as well as ML/DL tools for AI applications.

Definitions

This white paper uses the following terms:

- **Host:** A *host* is a physical or virtual server that is available to the EPIC platform. Nodes and clusters reside on hosts.
- **Node:** A *node* (also called a *virtual node* or *instance*) is a Docker container that is created when creating a persistent cluster. This white paper may also use the terms *container* and/or *Docker container*.
- **Microservice:** A *microservice* is a method of developing software applications as a suite of small, modular, and independently deployable services in which each service runs a unique process and communicates through a well-defined, lightweight mechanism to serve a business goal.
- **Big Data/AI application:** A *Big Data/AI application* generally refers to a distributed, multi-node, inter-related service that can process large amounts of data computing on several nodes. Some examples of Big Data and AI applications include Hadoop, Spark, Kafka, TensorFlow, H2O, and others. Big Data/AI applications should not be confused with microservices.
- **Docker container:** A *Docker container* is a lightweight, standalone, executable software package that runs specific services. This software package includes code, runtime, system libraries, configurations, etc. that run as isolated process in user space. A Docker container is typically used to deploy scalable and repeatable microservices.
- **EPIC platform:** An *EPIC platform* consists of the hosts that comprise the overall infrastructure available to a virtual cluster.
- **Controller:** A *Controller* is a host that manages the other nodes while also serving as a Worker host in the EPIC platform.
- **Worker host:** A *Worker host* is a host that is managed by a Controller.
- **Compute host (or Compute Worker):** A *Compute host* or *Compute Worker* is a Worker host that runs the virtual nodes (Docker containers) used by persistent clusters.
- **Gateway host (or Gateway Worker):** A *Gateway host* or *Gateway Worker* is a Worker host that maps services running on virtual nodes to ports in order to allow users to access those services when the virtual node IP addresses are non-routable and thus cannot be accessed directly.
- **Master node:** A *Master node* is a container that manages a cluster.
- **Worker node:** A *Worker node* is a container that is managed by a Master node in a cluster.
- **Shadow Controller:** A *Shadow Controller* is a designated Worker host that assumes the Controller host role if the primary Controller host fails. Your EPIC platform must meet all applicable requirements and have High Availability enabled for this protection to be active.
- **Arbiter:** An *Arbiter* is a designated Worker host that triggers the Shadow Controller host to assume the Controller role if the primary Controller host fails.
- **Hadoop job tracker:** Within a virtual cluster running MapReduce version 1, a *Hadoop job tracker* is the service that is responsible for coordinating the MapReduce application efforts of the remaining virtual nodes within that virtual cluster.
- **Hadoop task tracker:** Within a virtual cluster running MapReduce version 1, a *Hadoop task tracker* is a service that accepts tasks (Map, Reduce, and Shuffle operations) from the Hadoop job tracker. Each task tracker configuration specifies the number of “slots” or tasks that it can accept. Task trackers notify the job tracker when tasks complete. They also send out periodic “heartbeat” messages with the number of available slots, to keep the job tracker updated.
- **Resource manager:** Within a virtual cluster running YARN, a *resource manager* is the service that is responsible for coordinating the MapReduce application efforts of the remaining virtual nodes within that virtual cluster.
- **Node manager:** Within a virtual cluster running YARN, a *node manager* is a service that accepts tasks (Map, Reduce, and Shuffle operations) from the resource manager. Each node manager configuration specifies the number of “slots” or tasks that it can accept. Node managers notify the resource manager when tasks complete. They also send out periodic “heartbeat” messages with the number of available slots, to keep the resource manager updated.

- **Platform:** A *platform* includes all of the tenants, nodes, virtual clusters, and users that exist on a given EPIC deployment.
- **Tenant:** A *tenant* is a unit of resource partitioning and data/ user access control in a given deployment. The resources of an EPIC platform are shared among the tenants on that platform. Resources used by one tenant cannot be used by another tenant. All users who are a member of a tenant can access the resources and data objects available to that tenant.
- **Edge node:** An *edge node* is a container running a business intelligence tool that interacts with the cluster's Hadoop or Spark framework.
- **Virtual cluster:** A *virtual cluster* is a collection of virtual nodes that is available to a specific tenant.
- **cnode:** *cnode* is the BlueData caching node service. The transfer of storage I/O requests from the BlueData implementation of the HDFS Java client to this caching node service is being optimized as latency is reduced.
- **Node flavor:** A *node flavor* describes the amounts and sizes of resources (virtual CPU cores, memory, and hard disk size) allotted to a virtual node.
- **Platform Administrator:** The *Platform Administrator* (or *Platform Admin*) is a role granted to an EPIC user. A user with this role has the ability to create/delete tenants. This user will typically also be responsible for managing the hosts in the EPIC platform.
- **Tenant Administrator:** A *Tenant Administrator* (or *Tenant Admin*) is a role granted to an EPIC user. A user with this role has the ability to manage the specific tenant(s) for which they have been granted this role, including creating DataTaps for that tenant.
- **Tenant Member:** A *Tenant Member* (or *Member*) is a role granted to an EPIC user. A user with this role has non-administrative access to the specific tenant(s) for which they have been granted this role. Members may use existing DataTaps for reading and writing data.
- **User:** A *user* is the set of information associated with each person accessing the EPIC platform, including the authentication and site roles.
- **DataTap:** A *DataTap* is a shortcut that points to a storage resource on the network. A Tenant Administrator creates a DataTap within a tenant and defines the storage namespace that the DataTap represents (such as a directory tree in a file system). A Tenant Member may then access paths within that resource for data input and/or output. Creating and editing DataTaps allows Tenant Administrators to control which storage areas are available to the members of each tenant, including any specific sharing or isolation of data between tenants.
- **Node storage:** *Node storage* is storage space available for backing the root filesystems of containers. Each host in the EPIC platform contributes node storage space that is used by the virtual nodes (Docker containers) assigned to that host. The Platform Administrator may optionally specify a quota limiting how much node storage a tenant's virtual nodes may consume.
- **Tenant storage:** *Tenant storage* is a shared storage space that may be provided by either a local HDFS installation on the EPIC platform or a remote storage service. Every tenant is assigned a sandbox area within this space that is accessible by a special, non-editable **TenantStorage** DataTap. All virtual nodes within the tenant can access this DataTap and use it for persisting data that is not tied to the life cycle of a given cluster. Tenant storage differs from other DataTap-accessible storage as follows:
    - A tenant may not access tenant storage outside of its sandbox.
    - The Platform Administrator can choose to impose a space quota on the sandbox.
- **Cluster file system:** Many types of EPIC clusters set up cluster-specific shared storage that consists of services and storage resources inside the cluster nodes. For example, an EPIC cluster running Hadoop will set up an in-cluster HDFS instance. This shared storage is referred to as the *cluster file system*. The cluster file system is often used for logs and temporary files. It can also be used for job input and output; however, the cluster file system and all data therein will be deleted when the cluster is deleted.
- **Filesystem Mount (or FS Mount):** A *filesystem mount* allows EPIC to automatically add NFS volumes or mounts to virtual nodes. This allows virtual nodes to directly access NFS shares as if they were local directories. You can use this feature to provide common files across all of the virtual node in a given tenant, such as a common configuration file that will be used by all of the virtual nodes in a tenant.
- **Active Directory (or AD):** This is a Microsoft directory service for Windows domain networks.
- **Lightweight Directory Access Protocol (LDAP):** This is a client-server directory service protocol that runs on a layer above the TCP/IP stack and provides a mechanism for connecting to, searching, and modifying networked directories.

- **Security Assertion Markup Language (SAML):** This is an open standard for exchanging authentication and authorization data between parties, such as between an identity provider (IdP) and a service provider.

Hadoop and Spark Support

The BlueData EPIC platform includes pre-configured, ready-to-run versions of major Hadoop distributions, such as Cloudera (CDH), Hortonworks (HDP), and MapR (CDP). It also includes recent versions of Spark standalone as well as Kafka and Cassandra. Other Big Data distributions, services, commercial applications, and custom applications can be easily added to a BlueData EPIC deployment, as described in "App Store" on page 2.

BlueData EPIC supports a wide range of Big Data application services and Hadoop/Spark ecosystem products out-of-the-box, such as:

- **Ambari (for HDP):** Ambari is an open framework that allows system administrators to provision, manage, and monitor Apache Hadoop clusters and integrate Hadoop with the existing network infrastructure.
- **Cassandra (from Datastax):** Cassandra is a free, open-source, distributed NoSQL database management system for processing large datasets on commodity servers with no single points of failure.
- **Cloudera Manager (for CDH):** Cloudera Manager provides a real-time view of the entire cluster, including a real-time view of the nodes and services running, in a single console. It also includes a full range of reporting and diagnostic tools to help optimize performance and utilization.
- **Cloudera Navigator (for CDH):** Cloudera Navigator is a fully integrated Hadoop data management tool that provides critical data governance capabilities for regulated enterprises or enterprises with strict compliance requirements. These capabilities include verifying access privileges and auditing all Hadoop data access.
- **Flume:** Flume-NG is a distributed, reliable, and available service for efficiently collecting, aggregating, and moving large amounts of server log data. It is robust and fault tolerant with many failover and recovery mechanisms. It uses a simple extensible data model that allows one to build online analytic applications.
- **GraphX (for Spark):** GraphX works seamlessly with graphs and collections by combining Extract/Transform/Load (ETL), exploratory analysis, and iterative graph computation within a single system. The Pregel API allows you to write custom iterative graph algorithms.
- **Hadoop Streaming:** Hadoop Streaming is a utility that allows you to create and run MapReduce jobs with any script as the mapper and/or the reducer taking its input from `uvfkp` and writing its output to `uvfqwv`.
- **HAWQ:** Apache HAWQ is a native Hadoop application that combines high-performance Massively Parallel Processing (MPP)-based analytics performance with robust ANSI SQL compliance, integration, and manageability within the Hadoop ecosystem, as well as support for a variety of datastore formats with no connectors required.
- **HBase:** HBase is a distributed, column-oriented data store that provides random, real-time read/write access to very large data tables (billions of rows and millions of columns) on a Hadoop cluster. It is modeled after Google's BigTable system.
- **Hive:** Hive facilitates querying and managing large amounts of data stored on distributed storage. This application provides a means for applying structure to this data and then running queries using the HiveQL language. HiveQL is similar to SQL and supports using custom mappers when needed.
- **Hue:** Hue is an open-source Web interface that integrates the most common Hadoop components into a single interface that simplifies accessing and using a Hadoop cluster.
- **Impala:** Impala by Cloudera is an open-source MPP query engine that allows users to query data without moving or transforming that data, thus enabling real-time analytics without the need to migrate data sets. EPIC supports Impala on CDH.
- **JupyterHub:** JupyterHub is a multi-user server that provides a dedicated single-user Jupyter Notebook server for each user in a group.
- **Kafka:** Kafka allows a single cluster to act as a centralized data repository that can be expanded with zero down time. It partitions and spreads data streams across a cluster of machines to deliver data streams beyond the capability of any single machine.
- **MapReduce:** MapReduce assigns segments of an overall job to each Worker, and then reduces the results from each back into a single unified set.
- **MLlib:** MLlib is Spark's scalable machine learning library that contains common learning algorithms, utilities, and underlying optimization primitives.
- **Oozie:** Oozie is a workflow scheduler system for managing Hadoop jobs that specializes in running workflow jobs with actions that run Hadoop MapReduce and Pig jobs.

- **Pig:** Pig is a language developed by Yahoo that allows for data flow and transformation operations on a Hadoop cluster.
- **Ranger:** Apache Ranger is a central security framework that allows granular access control to Hadoop data access components, such as Hive and HBase. Security administrators manage policies that govern access to a wide array of resources in an individual and/or group basis. Additional capabilities include auditing, policy analytics, and encryption.
- **RStudio®:** RStudio is a free, open-source integrated development environment (IDE) for the R programming language that is used for statistical computing and graphics.
- **Spark SQL:** Spark SQL is a Spark module designed for processing structured data. It includes the DataFrames programming abstraction and can also act as a distributed SQL query engine. This module can also read data from an existing Hive installation.
- **SparkR:** This R package provides a lightweight front end for using Spark from R.
- **Spark Streaming:** Spark Streaming is an extension of the core Spark API that enables fast, scalable, and fault-tolerant processing of live data streams.
- **Sqoop:** Sqoop is a tool designed for efficiently transferring bulk data between Hadoop and structured datastores, such as relational databases. It facilitates importing data from a relational database, such as MySQL or Oracle DB, into a distributed filesystem like HDFS, transforming the data with Hadoop MapReduce, and then exporting the result back into an RDBMS.
- **Sqoop 2:** Sqoop 2 is a server-based tool designed to transfer data between Hadoop and relational databases. You can use Sqoop 2 to import data from a relational database management system (RDBMS) such as MySQL or Oracle into the Hadoop Distributed File System (HDFS), transform the data with Hadoop MapReduce, and then export it back into an RDBMS.
- **Zeppelin:** Web-based Zeppelin notebooks allow interactive data analytics by bringing data ingestion, exploration, visualization, sharing, and collaboration features to Spark as well as Hadoop.

Artificial Intelligence and ML/DL Workloads

Enterprises are increasingly turning to AI to solve complex problems, conduct research, and maintain or boost their competitive advantages in the marketplace. AI and machine learning (ML)/deep learning (DL) technologies have moved into the mainstream with a broad range of data-driven enterprise applications: credit card fraud detection, stock market prediction for financial trading, credit risk modeling for insurance, genomics and precision medicine, disease detection and diagnosis, natural language processing (NLP) for customer service, autonomous driving and connected car IoT use cases, and more.

A typical distributed ML/DL workflow may look something like this:

1. The model is conceptualized.
2. The model is built in one or more sandbox/custom environment(s) that require access to data and model storage.
3. Further iterations are created that may add libraries and/or features and that require rerunning the model.
4. The model is saved and deployed, and any API endpoints are published.

Measurements to determine model efficacy occur both in real time and in batch feedback loops, and this feedback is used to continue conceptualizing the model.

Needs

Enterprises wanting to deploy distributed ML/DL infrastructures typically have some or all of the following needs:

- Role-based access control to some or all of the following:
  - Preferred ML/DL tools, such as TensorFlow, H2O, MXNet, BigDL for Spark, Caffe, and SparkMLib.
  - Common "big" and "small" data frameworks, such as Kafka, HDFS, HBase, Spark, model storage, and workflow management,
  - Data science notebooks, such as Jupyter, RStudio, and Zeppelin.
  - Various related analytics, business intelligence (BI), and ETL tools.
- Choice of modeling techniques.
- Ability to build, share, and iterate.
- Reproducibility.
- Easy scaling for testing on actual data sets.
- Support for varying roles and actions.

Challenges

Some of the key challenges enterprises face when looking to build, deploy, and operationalize their ML/DL pipelines to meet the needs described above include:

- Traditional analytics tools were built to process structured data in databases. AI use cases that require ML/DL tools require a large and continuous flow of data that is typically unstructured.

- Data scientists and developers may have built and designed their initial ML/DL algorithms to operate in a single-node environment (e.g. on a laptop, virtual machine, or cloud instance) but need to parallelize the execution in a multi-node distributed environment.
- Enterprises cannot meet their AI use case requirements using the data processing capabilities and algorithms of a single ML/DL tool. They need to use data preparation techniques and models from multiple open source and/or commercial tools.
- Data science teams are increasingly working in more collaborative environments where the workflow for building distributed ML/DL pipelines spans multiple different domain experts.
- Many ML / DL deployments use hardware acceleration such as GPUs to improve processing capabilities. These are expensive resources, and this technology can add to the complexity of the overall stack.
- ML/DL technologies and frameworks are different from existing enterprise systems and traditional data processing frameworks.
- ML/DL stacks are complex because they require both multiple software and infrastructure components and version compatibility and integration across those components.
- Assembling all of the required systems and software is time consuming, and most organizations lack the skills to deploy and wire together all of these components.

For example, TensorFlow is one of the most popular ML/DL tools. It is often used together with technologies like Python and GPUs to create an end-to-end pipeline from data preparation to modeling, scoring, and inference. However, different use cases may call for different tools. Data scientists and developers want to evaluate and work with a variety of ML/DL tools and require rapid prototyping to compare different libraries and techniques. Most large organizations require them to comply with enterprise security, network, storage, user authentication, and access policies. These users often start with a single-node environment; but these technologies are difficult to implement in complex, multi-node distributed environments for large-scale enterprise use cases. Most enterprises lack the skills to deploy and configure these tools with their existing data infrastructure and systems, whether on-premises, in the public cloud, using CPUs and/or GPUs, with a data lake or with cloud storage.

The BlueData Solution EPIC for ML/DL

BlueData EPIC goes beyond Hadoop and Spark support by leveraging the inherent infrastructure portability and flexibility of Docker containers to support distributed AI for both ML and DL use cases. The separation of compute and storage for Big Data and ML/DL workloads is one of the key concepts behind this flexibility, because organizations can deploy multiple containerized compute clusters for different workflows (e.g. Spark, Kafka, or TensorFlow) while sharing access to a common data lake. This also enables hybrid and multi-cloud EPIC deployments, with the ability to mix and match on- and/or off-premises compute and storage resources to suit each workload. Further, compute resources can be quickly and easily scaled and optimized independent of data storage, thereby increasing flexibility and improving resource flexibility while eliminating data duplication and reducing cost by reusing existing storage investments.

Some of the key ML/DL features and benefits that EPIC provides include:

- **Container-based automation:** EPIC creates virtual clusters that each contain one or more Docker container(s). Containers are now widely recognized as a fundamental building block to simplify and automate deployments of complex application environments, with portability across on-premises infrastructure and public cloud services.
- **Deployment of ML/DL workloads:** BlueData EPIC can be used to deploy distributed ML/DL environments such as TensorFlow, Caffe2, H2O, BigDL, and SparkMLlib. This allows organizations embarking on AI initiatives to quickly spin up multi-node ML/DL sandbox environments for their data science teams. If available, they can also easily and securely tap into an existing data lake to build and deploy their ML/DL pipelines.
- **Rapid and reproducible provisioning:** Users can spin up new, fully-provisioned distributed ML/DL applications in multi-node containerized environments on any infrastructure, whether on-premises or in the cloud, using either CPUs and/or GPUs. These fully-configured environments can be created in minutes via self-service, via RESTful APIs, or via a few mouse clicks in the EPIC interface. IT teams can ensure enterprise-grade security, data protection, and performance with elasticity, flexibility, and scalability in a multi-tenant architecture. The template feature allows organizations to preserve specific cluster configurations for reuse at any time with just a few mouse clicks. EPIC publishes service endpoint lists for each virtual node/cluster.

- **Decoupling of compute and storage resources:** As described above, the separation of compute from storage allows organizations to reduce costs by scaling these infrastructure resources independently while leveraging their existing storage investments in file, block, and object storage to extend beyond their petabyte-scale HDFS clusters. EPIC allows secure integrations with distributed file systems including HDFS, NFS, and S3 for storing data and ML / DL models, including pass-through security from the compute clusters.