



Best Practices for Securing Kubernetes

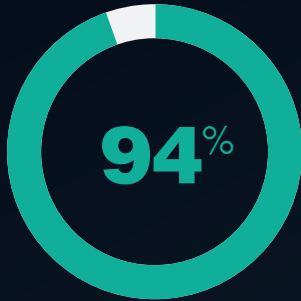
**How to Apply Zero-Trust
Principles to Secure
Access to Kubernetes**

WHITE PAPER

CONTENTS

INTRODUCTION	03
WHAT IS ZERO-TRUST SECURITY	04
ZERO-TRUST SECURE ACCESS FOR KUBERNETES	05
AUTHENTICATION	06
AUTHORIZATION	06
ADMISSION CONTROL	07
LOGGING AND AUDITING	07
BEST PRACTICES WHEN APPLYING ZERO-TRUST FOR KUBERNETES	08
ZERO-TRUST SECURE ACCESS AT RAFAY	09
ESTABLISHING ZERO-TRUST ACCESS	10
USER AND ADMINISTRATOR EXPERIENCE	11
BREAK GLASS	12
AUDITING AND COMPLIANCE	12
THE RAFAY KUBERNETES OPERATIONS PLATFORM	13

INTRODUCTION



**Over 94 percent
experienced security
incidents in the last year**

RECENT SURVEY OF
OVER 540 RESPONDENTS*

The footprint of Kubernetes in the enterprise is expanding rapidly as IT teams take advantage of its power, efficiency, and flexibility to develop modern applications quickly and deliver them at scale. Your organization may already have a large number of development and production Kubernetes clusters spread across on-premises data centers, multiple public cloud providers, and edge locations.

Ensuring security in any dynamic IT environment is growing more challenging by the day. Given Kubernetes' relative newness, rapid growth, and dynamic operating model, the security challenges are significant. Most teams have multiple clusters in multiple locations—often running different distributions with different management interfaces—and teams of developers, operators, contractors, and partners who need access.

Many security incidents are occurring, and many of the companies breached—like Microsoft, Docker, and Jenkins—aren't new to Kubernetes. In fact, in a recent survey, 94 percent had experienced security incidents in the last year! Teams deploying Kubernetes have to do everything possible to ensure their K8s infrastructure is secure, at scale. This white paper explores zero-trust principles applied to Kubernetes.

It also explains how Rafay's Zero-Trust Access Service makes it easier to create and maintain an end-to-end zero-trust Kubernetes environment spanning data centers, public clouds, and edge locations.

*Source: <https://techgenix.com/5-kubernetes-security-incidents/>



WHAT IS ZERO-TRUST SECURITY?

Traditional IT network security is based on a castle-and-moat model where users inside the castle are trusted by default. The fundamental problem with this approach is that once an attacker gains access to the network, they have access to everything inside.

Zero-trust assumes that all actors, systems, and services cannot be trusted. It draws on technologies such as authentication, authorization, and encryption and is continuously validated for security configuration and posture to ensure compliance.

- **Authentication:** The process of verifying the identity of a user or process. Multi-factor authentication (MFA) uses two or more methods, providing protection even if one method is compromised.
- **Authorization:** An authenticated user or service also has to be authorized to perform the requested function. Role-based access control (RBAC) is widely used to limit the tasks that a given user or service is able to perform.
- **Encryption:** Cryptography is at the core of zero-trust, enabling data to be encrypted and signed. For example, public-key cryptography can enable identification, authentication, and secure data transport.

KUBERNETES ACCESS: BEFORE RAFAY

A fast-growing customer has applications that stream video and audio, requiring Kubernetes clusters that are geographically close to its global customer base. The company operates roughly 90 clusters ranging from pre-production to QA to staging to Prod.

This company faced three main problems with regard to secure user access to Kubernetes:

Managing user accounts:

With 300-400 developers and 20 ops staff, manually adding and removing users from each cluster was an operational and security nightmare.

Slow access:

Time to remediation is critical when a “severity 1” incident occurs. With *traditional* access methods it took 5 minutes just to log into an affected cluster.

Complex compliance reporting:

With log data spread across 90 clusters, centralized auditing and compliance reporting was nearly impossible.

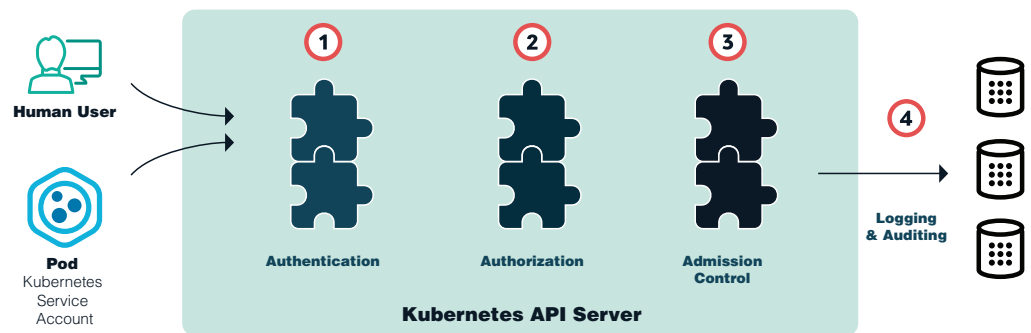


ZERO-TRUST SECURE ACCESS FOR KUBERNETES

The Kubernetes API Server is the core of the Kubernetes control plane for each cluster. API calls are used to query and manipulate the state of all Kubernetes objects (namespaces, pods, config maps, etc.). Therefore, controlling access to API use is key to controlling Kubernetes access and achieving zero-trust for Kubernetes. Securing the Kube API server should be the first line of defense for any organization.

The first step is to protect network traffic to and from the API server with Transport Layer Security (TLS), which securely encrypts all packets.

Once transport is secure, Kubernetes includes all the necessary hooks to implement Kubernetes zero-trust and control API Server access for each Kubernetes cluster. These hooks fall into four areas:



The above list of requirements is critical to any production-grade K8s environment operated and supported by IT operations and, as a result, also serves as the minimum requirements list for any company evaluating K8s management and operations offerings. Each of these features - and its specific applicability for K8s management - is discussed below.

1: Authentication

2: Authorization

3: Admission Control

4: Logging and Auditing

Integrate with Your Corporate Identity System

For teams that are new to Kubernetes, a common mistake is to keep authorization independent from the established single sign-on (SSO) or identity system. This lack of integration may be just a minor inconvenience at the outset, but as your deployment grows it adds administrative overhead, creates new points of vulnerability, and increases the time it takes to gain access to Kubernetes

1 Authentication

With zero-trust for Kubernetes, all user and service accounts must be authenticated before they can execute an API call. There are a number of Kubernetes security modules and plugins to ensure that the platform will work with your preferred authentication system:

- Client certificates
- HTTP Basic auth
- Bearer tokens
- OpenID Connect tokens
- Authentication Proxy (to support LDAP, SAML, Kerberos, etc.)
- Webhook token authorization

We recommend that you implement multi-factor authentication or MFA with at least two authentication methods. The most commonly used two-factor authentication methods combine something the user knows (like a password) with something the user has (like a token). Authentication methods can also include biometrics (for example fingerprint or facial recognition) or be location-based using network address or GPS.

2 Authorization

User and service accounts accessing a Kubernetes cluster must not be able to carry out every possible action. With zero-trust for Kubernetes, a request is only authorized if an authenticated user has permission to complete the requested action. Each request specifies the username of the requester, the action requested, and the objects affected.

As with authentication, Kubernetes supports multiple ways to handle authorization, including:

- **RBAC mode:** Role-based access control or RBAC authorizes access based on the user's role in the organization such as admin, developer, security, etc.
- **ABAC mode:** Attribute-based access control or ABAC authorizes access dynamically based on a combination of user, environment, and resource attributes.

Most organizations that implement access control choose RBAC as the more practical solution because it's easier to manage and provides the necessary granularity for most use cases. ABAC can be time consuming to define and configure properly and troubleshooting can be challenging if issues arise.

3 Admission Control

Kubernetes Admission Control allows you to exercise additional controls over user actions and implement business logic to refine your Kubernetes zero-trust approach. Admission controllers provide methods to act on requests that create, modify, delete, or connect to Kubernetes objects. You can have multiple admission controllers, and if any admission controller rejects a request, then the request is immediately rejected.

There are a variety of built-in admission controllers you can enable to enforce policies and implement various actions. Dynamic controllers enable you to modify requests on the fly to adhere to your established rules. For example, the ResourceQuota admission controller observes incoming requests and ensures that they don't violate the constraints you've enumerated in the ResourceQuota object for a Namespace. For more details, see [Using Admission Controllers](#).

4 Logging and Auditing

Auditing is an essential part of Kubernetes security. Kubernetes auditing capabilities enable you to track the actions performed within a cluster, including actions by users, applications, and the control plane itself. Available audit levels include:

- **None:** Don't log this event.
- **Metadata:** Log request metadata.
- **Request:** Log event metadata and the request.
- **Request Response:** Log event metadata, the request, and the response.

You can also control where audited events are logged. The log backend writes events to the cluster's local filesystem, while the webhook backend sends audit events to an external logging system.



BEST PRACTICES WHEN APPLYING ZERO-TRUST FOR KUBERNETES

As a summary, here are the best practices for implementing zero-trust for Kubernetes access:

Kubernetes API Server

- Enable TLS everywhere
- Use a private endpoint for the API Server
- Use third-party authentication for the API Server
- Close firewall inbound rules to the API Server, ensuring it is cloaked and not directly accessible from the Internet

Authentication

- Use at least two authentication methods
- Rotate client certificates regularly

Authorization

- Enable RBAC with least privilege

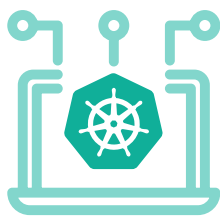
Admission Control

- Enable appropriate admission controllers for your needs

Logging and Auditing

- Enable audit logs with central log aggregation

While applying the best practices in each area gives you the ability to apply zero-trust principles to your Kubernetes environment, keeping all the individual elements correctly configured and aligned becomes a real challenge once you have more than a few clusters, especially when multiple workloads and Kubernetes distributions are involved.



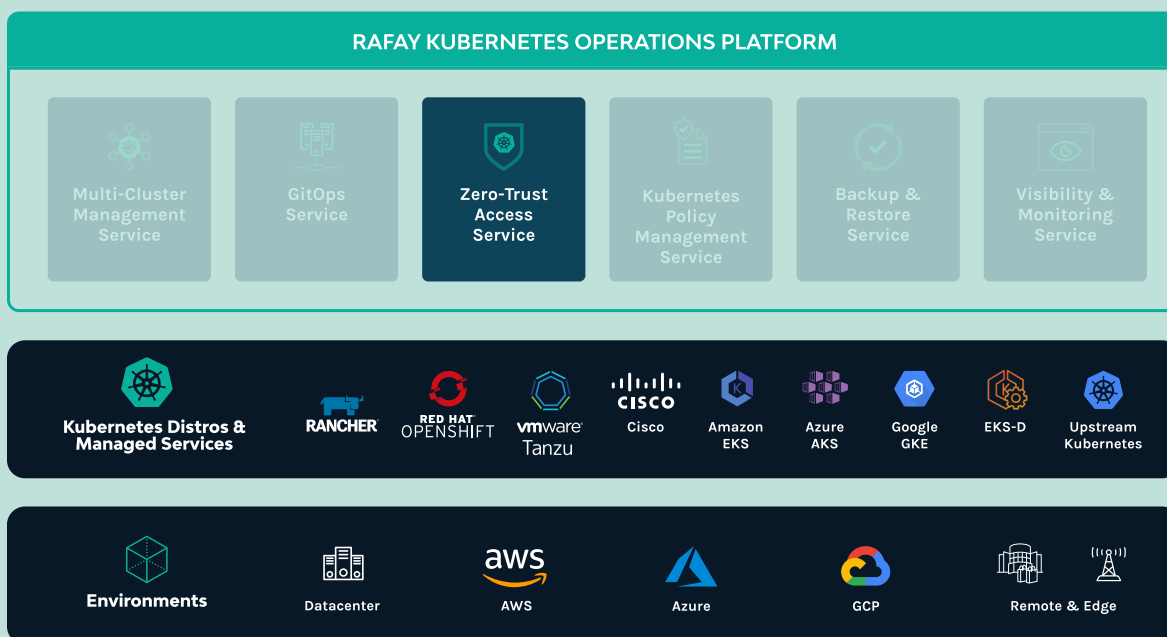
ZERO-TRUST SECURE ACCESS AT RAFAY

Companies with multiple Kubernetes clusters and rapidly growing Kubernetes deployments can leverage **Rafay's Zero-Trust Access Service—a key component of the Rafay Kubernetes Operations Platform—to secure and govern Kubernetes access everywhere.**

The [Rafay Zero-Trust Access Service](#) ensures that Kubernetes best practices are applied and enforced to secure your entire Kubernetes fleet, eliminating the need to manually apply best practices to each cluster so you can operate Kubernetes at scale with far less risk and wasted time.

Rafay utilizes the latest zero-trust principles and security best practices throughout our Kubernetes Operations Platform to guarantee security, and—because our platform works with all major cloud providers and certified Kubernetes distributions—you can increase security everywhere, including enabling zero-trust on-premises, in public clouds, and at the edge. With Rafay, you can:

- Centralize and secure your entire fleet with auditing of all actions
- Instantly enable/revoke user access to your entire fleet of clusters
- Integrate and unify zero-trust, role-based access policies with third-party identity management systems (e.g. corp IdPs and cloud identities)
- Remove the need to leverage a VPN solution/bastion server access model and/or open firewall ports
- Reduce complexity and eliminate the risk of security breaches and productivity losses



JUST-IN-TIME ACCOUNTS

Old and unused service accounts make clusters vulnerable. With the Zero-Trust Access Service, service accounts for authorized users are created on remote clusters when access is requested and automatically flushed from the cluster after a specified period of time.

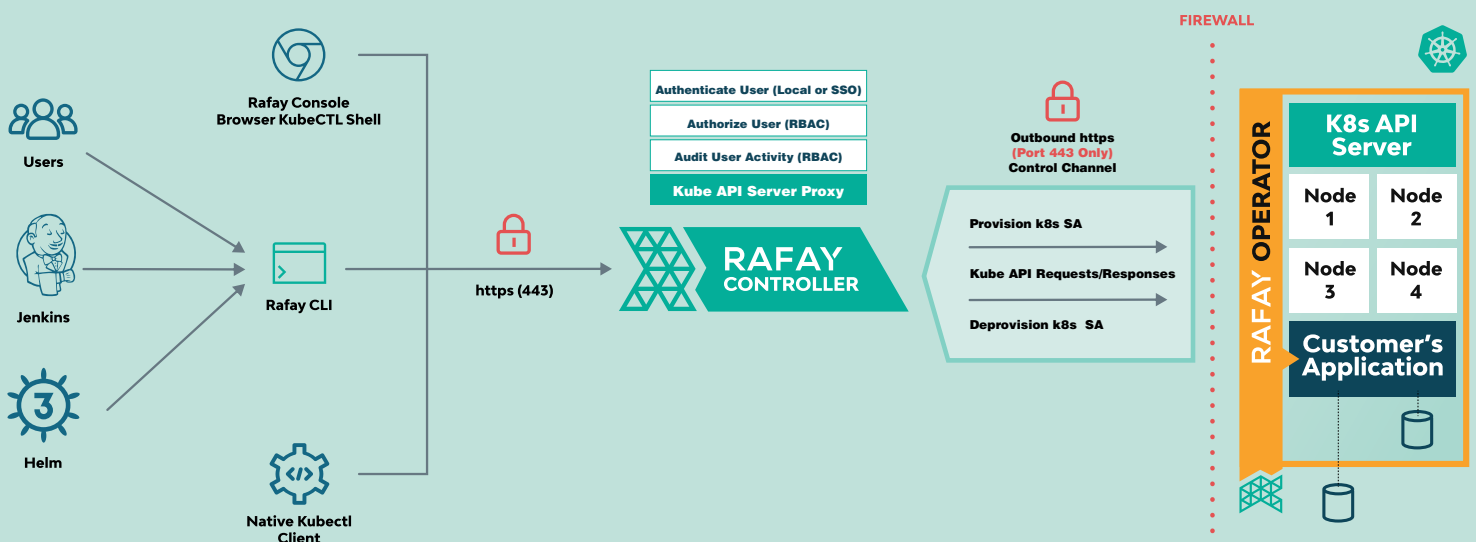
Establishing Zero-Trust Access

Rafay's Zero-Trust Access Service streamlines and consolidates access control for your fleet of clusters spanning different operating environments, different clouds and firewall-protected, on-premises data centers.

Rafay enables controlled, audited access to your Kubernetes infrastructure for users (such as developers and SREs) as well as automation systems—with just-in-time service account creation and user-level credentials management integrated with your enterprise's single sign-on (SSO) solution, eliminating the burden of configuring and managing Role Based Access Control on a cluster-by-cluster basis.

Zero-Trust Kubernetes access is achieved through the use of a Kube API Server proxy that is part of the Rafay Controller. All incoming requests—whether they originate from Rafay's browser-based kubectl shell, kubectl running in a terminal, or from an application executing kubectl commands or Kube API calls—are authenticated, authorized, and audited before execution.

Rafay's Kubernetes Operation Platform includes extensive abilities to automate routine workflows and integrate with other management applications and services. In all cases, the Zero-Trust Access Service provides access control as shown below.



User and Administrator Experience

A single login gives authorized users seamless and secure access to all clusters with a native and familiar kubectl experience and significant operational enhancements, including:

- Centralize security and governance for all your K8s infrastructure without inbound access to clusters
- Configure and consolidate access for your fleet of remote clusters
- Integrate with your corporate directory to simplify identity management
- Automatically implement and enforce read-only access for non-privileged users
- Limit access to kubectl commands such as “get,” “describe,” and “logs”
- Provide contractors temporary access, with access terminating automatically after a specified time window
- Ephemeral access with Just in Time (JIT) creation of service accounts on target clusters
- Centralize audit logs across all users and clusters
- Prove compliance with internal policies, regulations, and audits
- Flexible cluster access via integrated terminal or CLI

KUBERNETES ACCESS: AFTER RAFAY

The same company described earlier (in the sidebar on page 4) turned to Rafay's Zero-Trust Access Service to enable its globally distributed team to manage user access for its entire fleet of clusters from a central location, with centralized audit logging for compliance. For this customer, there were three clear benefits:



Flexible role-based access control:

If a user changes roles, access permissions are updated automatically so they never have too much or too little access.



Fast and streamlined access:

No more 5-minute delays. An authorized user can get access to a cluster instantly.



Just-in-time credentials:

Service accounts for authorized users are created on remote clusters “just in time”, eliminating the risk of out-of-date credentials.

Break Glass

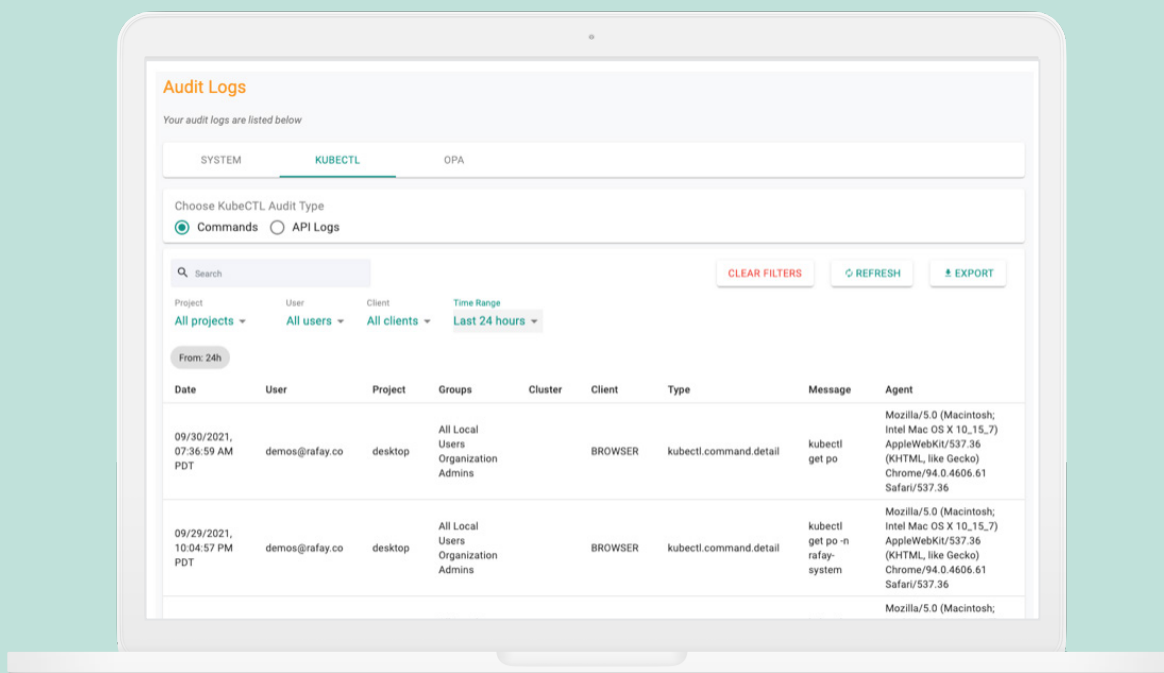
For mission-critical environments, Rafay allows you to easily implement a centralized and audited “break glass” process in case of emergencies. For example, if a security breach is suspected all kubectl access to clusters can be disabled until a privileged administrator reapproves access.

Alternatively, many companies allow only a few privileged admins to have access to production clusters, but developers may need access if there’s an issue. An audited break glass procedure can give developers web-based kubectl access to an affected cluster. Once the emergency is over, access is disabled, closing the door. The entire process is monitored, controlled, and lowers risk.

Auditing and Compliance

The Zero-Trust Access Service records all actions performed by all users via the UI, CLI, and API so your company can provide true cluster and application safety. You can easily search audit logs by cluster, namespace, user, and method. Detailed auditing and logging make it relatively straightforward to prove compliance with internal policies, industry regulations, and audit processes.

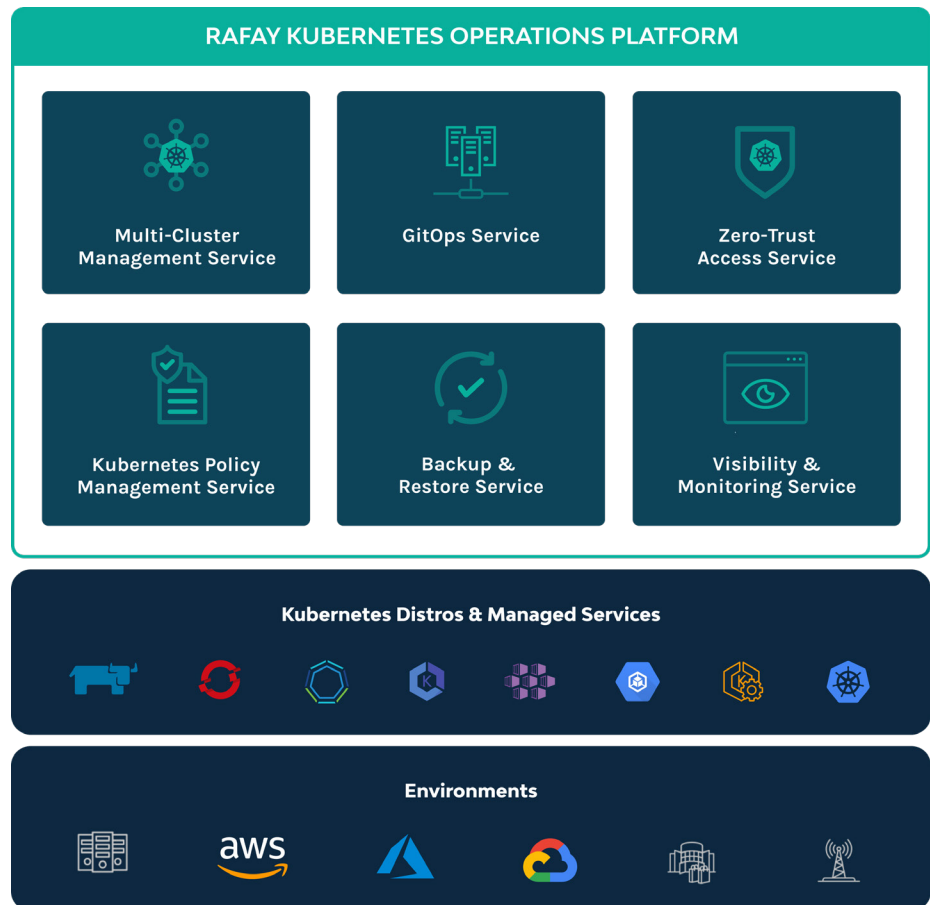
The kubectl CLI takes a high-level user command and breaks it down into a sequence of API calls. A full audit trail of the sequence of the API calls is available for administrators. Useful filters can be used to quickly sort and zero in on the details you require.





THE RAFAY KUBERNETES OPERATIONS PLATFORM

The Rafay Kubernetes Operations Platform is a cloud-based platform purpose-built to streamline Kubernetes operations and enable companies to deploy and manage modern applications across data centers, public cloud, and edge environments – at any scale.



Modernize security with integrated zero-trust and role-based access control for your entire fleet with Rafay Systems. **Sign up for a free trial today.**

Learn More About Rafay Systems

