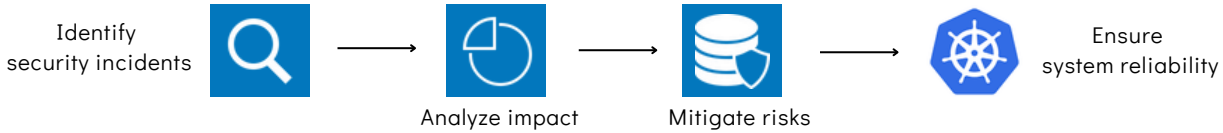
Kubernetes Incident Response (IR) Guide

Incident Response (IR) in Kubernetes refers to the process of identifying, analyzing, and mitigating security incidents within a Kubernetes cluster. It involves coordinated actions to minimize the impact of incidents on applications, data, and infrastructure. Effective IR ensures system reliability, data integrity, and business continuity.



Scope of Kubernetes IR

Incident Identification

- Track security events to identify and report suspected incidents.
- Monitor logs, metrics, and alerts for anomalies

Containment

- Isolate affected components (nodes, pods, services).
- Implement automated rollback mechanisms.
- Leverage self-healing features to minimize downtime.

Eradication

- Investigate root causes.
- Remediate vulnerabilities or misconfigurations.
- Apply security patches promptly.

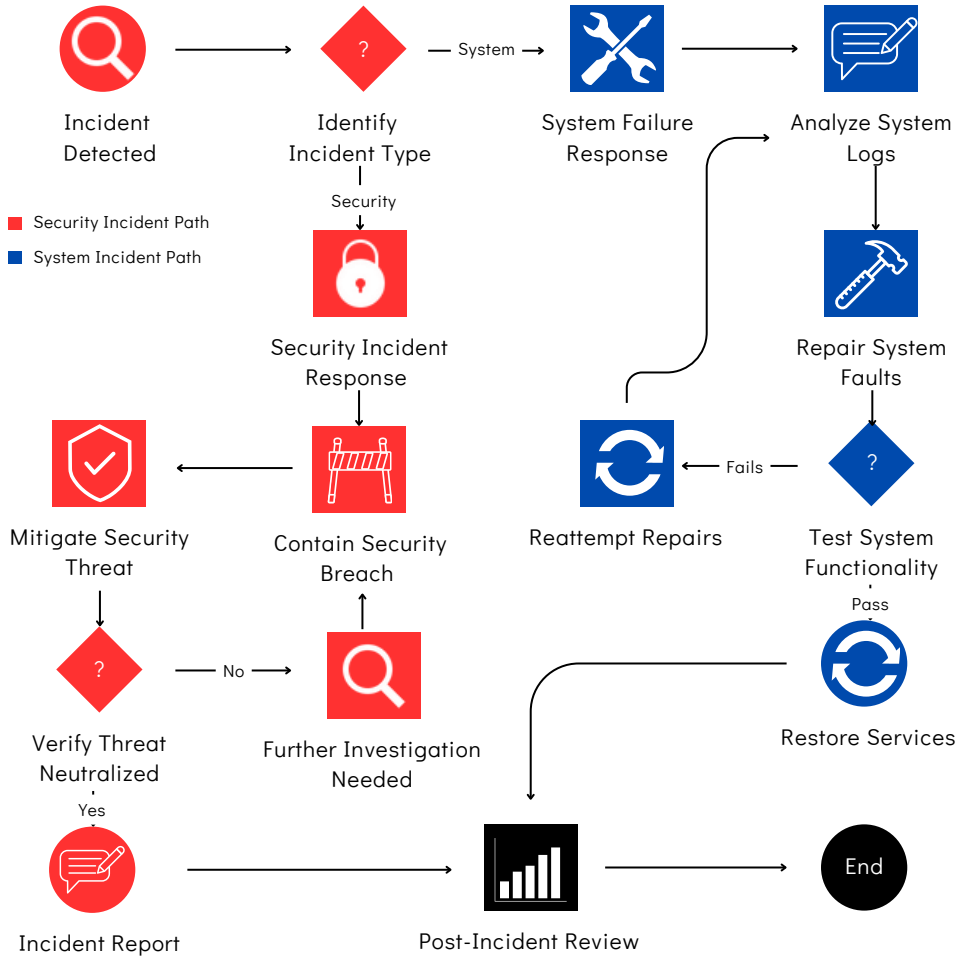
Recovery

- Restore services and data.
- Validate the effectiveness of the response.
- Communicate with stakeholders.

Scope of Kubernetes IR

- Implement robust monitoring tools (e.g., Prometheus, Grafana) for prompt detection.
- Leverage Kubernetes' built-in self-healing capabilities for incident resolution.
- Automatically adjust pod replicas based on resource utilization.
- Safely update or revert Kubernetes deployments without service disruption.
- Define network rules to restrict communication between pods and services.
- Maintain consistent configurations across clusters to prevent misconfigurations.
- Analyze incidents, learn from them, and enhance incident response processes.

Strategic Approach

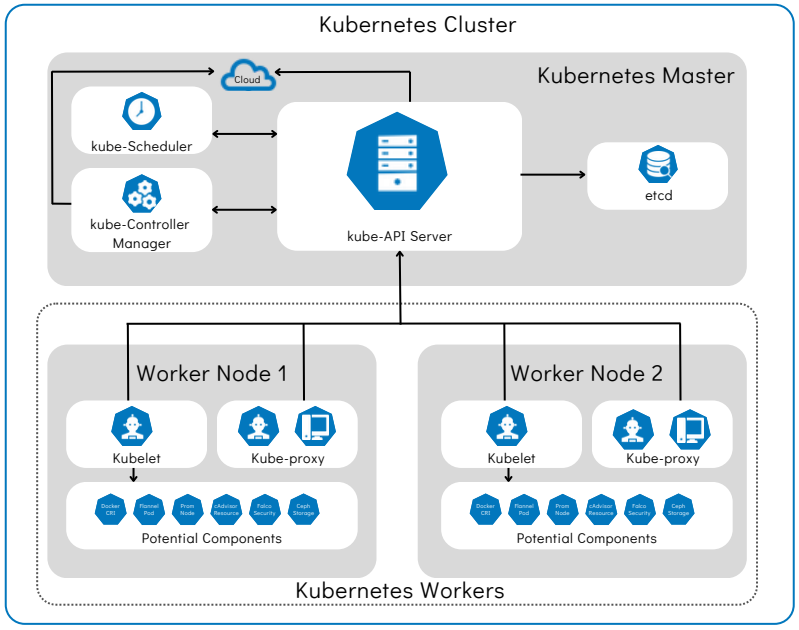


Kubernetes

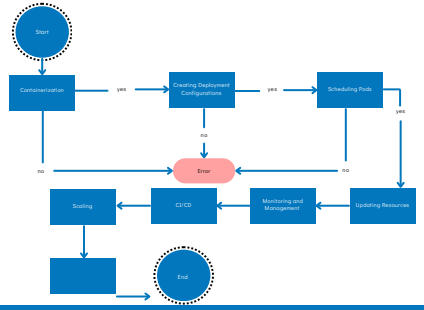
Kubernetes is an open-source container orchestration platform used for automating deployment, scaling, and management of containerized applications.

It simplifies the process of deploying and managing applications in a dynamic and scalable environment.

Kubernetes Architecture



Deployment Process



Key Components

Pods: Containers grouped together for deployment.
Services: Abstraction to access Pod functionality.
Deployments: Manages lifecycle of replicated Pods.
Control Plane: Oversees cluster management operations.

kubectl command List

Pod Management

Commands	Example	Commands	Example
kubectl create pod	`kubectl create -f pod.yaml`	kubectl delete pod	`kubectl delete pod my-pod`
kubectl get pods	`kubectl get pods`	kubectl logs	`kubectl logs my-pod`
kubectl describe pod	`kubectl describe pod my-pod`		

Deployment Management

Commands	Example	Commands	Example
kubectl rollout history	`kubectl rollout history deployment/nginx`	kubectl scale deployment	`kubectl scale deployment nginx --replicas=3`
kubectl rollout status	`kubectl rollout status deployment/nginx`	kubectl create deployment	`kubectl create deployment nginx --image=nginx`
kubectl get deployments	`kubectl get deployments`		

Service Management

Commands	Example	Commands	Example
kubectl delete service	`kubectl delete service my-service`	kubectl describe service	`kubectl describe service my-service`
		kubectl get services	`kubectl get services`
kubectl expose	`kubectl expose deployment my-deployment --port=80 --target-port=8080`	kubectl create service	`kubectl create service nodeport my-service --tcp=80:8080`

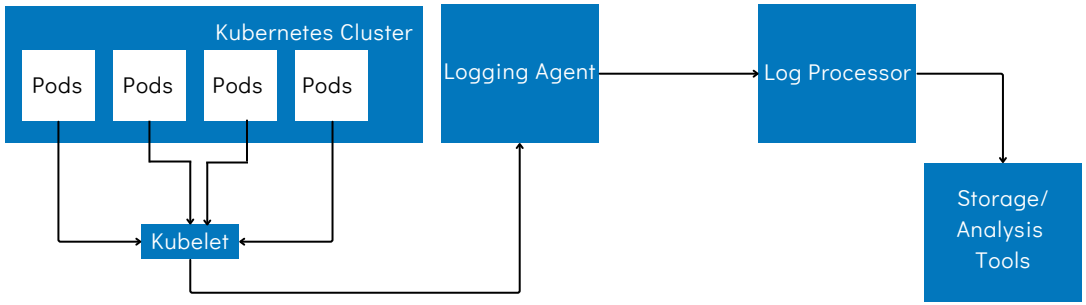
Namespace Management

Commands	Example	Commands	Example
kubectl apply -f	`kubectl apply -f pod.yaml --namespace=my-namespace`	kubectl create namespace	`kubectl create namespace my-namespace`
kubectl describe namespace	`kubectl describe namespace my-namespace`	kubectl delete namespace	`kubectl delete namespace my-namespace`
kubectl get namespaces	`kubectl get namespaces`		

Node Management

Commands	Example	Commands	Example
kubectl get nodes	`kubectl get nodes`	kubectl describe node	`kubectl describe node my-node`
kubectl cordon	`kubectl cordon my-node`	kubectl drain	`kubectl drain my-node`
kubectl uncordon	`kubectl uncordon my-node`		

Kubernetes Logging



Kubernetes Logging - Security Logging Analysis

Pod Security Events: Log events related to pod security violations, such as unauthorized access attempts or privilege escalation.

Network Policy Violations: Capture events related to network policy violations, such as unauthorized network access between pods.

Cluster Authentication Failures: Log authentication failures within the Kubernetes cluster, indicating potential unauthorized access attempts.

Container Runtime Anomalies: Monitor container runtime activities for anomalies, such as suspicious process execution or file system modifications.

API Server Authorization Events: Log events related to API server authorization, such as denied requests or policy enforcement.

Kubernetes Logging - Identity Logging Analysis

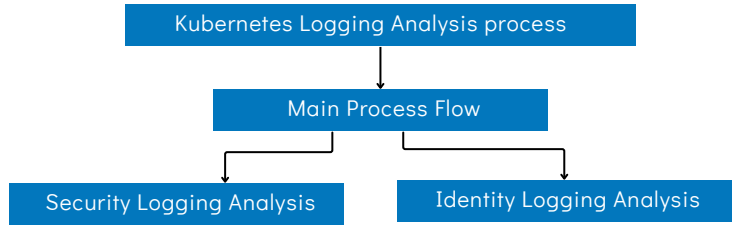
User Activity Logs: Track user activity within the Kubernetes cluster, including authentication events and resource access.

Service Account Activity: Monitor service account usage and activity within the cluster, including creation, deletion, and resource access.

Role-Based Access Control (RBAC) Changes: Log changes to role-based access control (RBAC) configurations, including role assignments and policy updates.

Audit Trail of Kubernetes API Calls: Maintain an audit trail of Kubernetes API calls made by users and service accounts, including requests and responses.

Service Identity Management: Manage and log service identities used by applications and services within the Kubernetes environment.



Kubernetes security controls to the MITRE ATT&CK framework

Initial Access	Discovery	Credential Access	Privilege Escalation	Defense Evasion	Persistence	Lateral Movement	Execution	Impact
Role-Based Access Control (RBAC)	Kubernetes Audit Logs	Kubernetes Secrets Management	Role-Based Access Control (RBAC)	Pod Security Context	Pod Security Policies (PSP)	Network Segmentation	Container Runtime Security	Disaster Recovery Planning
Kubernetes API Server Configuration	Kubernetes API Server Configuration	Secure Kubernetes API Access	Service Account Usage	Image Signing and Verification	Network Policies	Service Mesh	Kubernetes Resource Quotas	Pod Security Policies (PSP)
Secure Kubernetes API Access	Network Policies	Pod Security Context	Pod Security Policies (PSP)	Kubernetes Audit Logs	Kubernetes Secrets Management	Kubernetes Audit Logs	Kubernetes Audit Logs	Testing and Validating Disaster Recovery Procedures Regularly
Secure Kubernetes Dashboard Configuration	Kubernetes RBAC Misconfigurations	Rotation and Lifecycle Management for Kubernetes Secrets	Least Privilege Principle for Role Bindings	Continuous Monitoring for Anomalies in Pod Behavior	Automated Backup and Restore for Kubernetes Secrets	Kubernetes Security Best Practices Documentation	Secure Configuration of Container Runtimes	Automated Failover and Redundancy for Critical Cluster Components
Multi-Factor Authentication (MFA) for Kubernetes API Access	Pod Security Context Misuse	Encryption at Rest for Kubernetes Secrets	Regular Review and Audit of Service Account Permissions	Automated Image Scanning and Verification in CI/CD Pipelines	Role-Based Access Control for etcd Data	Secure Communication between Microservices in Service Mesh	Implementing Resource Quotas for Namespace Isolation	High Availability Cluster Configuration
Restrictive Network Access Policies	Container Image Vulnerability Scanning	Secure Configuration for Service Account Tokens	Automated Remediation for PSP Violations	Automated Threat Detection Mechanisms	Kubernetes Security Best Practices Documentation	Implementing Network Policies for Inter-Pod Traffic	Real-time Alerting and Monitoring for Unauthorized Pod Creation	Minimizing downtime risks.
Limiting ingress/egress traffic flow.	Centralized Kubernetes Cluster Logging	Secure Storage Encryption Mechanisms	Role-Based Access Control Auditing	Early detection of suspicious activities.	Immutable Infrastructure Deployment Strategy	Secure Configuration of Service Mesh	Container Image Signing Enforcement	
	Unified logs for better visibility.	Protecting data at rest.	Monitoring RBAC policy changes.		Ensuring consistent and reliable environments.	Ensuring encrypted and authenticated communication.	Verifying container image authenticity.	