**100 Kubernetes Diagnostics Commands with Kubectl**

Here is a list of 100 kubectl commands that can be useful for diagnosing issues in a Kubernetes cluster.

**Cluster Information:**

**1.** Show the Kubernetes version**: kubectl version**

2. Display cluster information: **kubectl cluster-info**

**3.** List all nodes in the cluster: **kubectl get nodes**

**4.** Describe a specific node: **kubectl describe node <node-name>**

**5.** List all namespaces**: kubectl get namespaces**

**6.** List all pods in all namespaces: **kubectl get pods --all-namespaces**

**Pod Diagnostics:**

**1.** List pods in a specific namespace**: kubectl get pods -n <namespace>**

**2.** Describe a pod: **kubectl describe pod <pod-name> -n <namespace>**

**3.** View pod logs**: kubectl logs <pod-name> -n <namespace>**

**4.** Tail pod logs: **kubectl logs -f <pod-name> -n <namespace>**

**5.** Execute a command in a pod: **kubectl exec -it <pod-name> -n <namespace> -- <command>**

**Pod Health Checks:**

1. Check pod readiness: **kubectl get pods <pod-name> -n <namespace> -o** jsonpath='{.status.conditions[?(@.type=="Ready")].status}'

**2.** Check pod events**: kubectl get events -n <namespace> --field-selector involvedObject.name=<pod-name>**

**Service Diagnostics:**

**1.** List all services in a namespace: **kubectl get svc -n <namespace>**

**2.** Describe a service**: kubectl describe svc <service-name> -n <namespace>**

**Deployment Diagnostics:**

**1.** List all deployments in a namespace**: kubectl get deployments -n <namespace>**

**2.** Describe a deployment: **kubectl describe deployment <deployment-name> -n <namespace>**

**3.** View rollout status: **kubectl rollout status deployment/<deployment-name> -n <namespace>**

**4.** View rollout history: **kubectl rollout history deployment/<deployment-name> -n <namespace>**

**StatefulSet Diagnostics:**

**1.** List all StatefulSets in a namespace: **kubectl get statefulsets -n <namespace>**

**2.** Describe a StatefulSet**: kubectl describe statefulset <statefulset-name> -n <namespace>**

**ConfigMap and Secret Diagnostics:**

**1.** List ConfigMaps in a namespace**: kubectl get configmaps -n <namespace>**

**2.** Describe a ConfigMap**: kubectl describe configmap <configmap-name> -n <namespace>**

**3.** List Secrets in a namespace: **kubectl get secrets -n <namespace>**

**4.** Describe a Secret: **kubectl describe secret <secret-name> -n <namespace>**

**Namespace Diagnostics:**

**1.** Describe a namespace: **kubectl describe namespace <namespace-name>**

**Resource Usage:**

**1.** Check resource usage for a pod: **kubectl top pod <pod-name> -n <namespace>**

**2.** Check resource usage for nodes**: kubectl top nodes**

**Networking Diagnostics:**

1. Show the IP addresses of pods in a namespace: **kubectl get pods -n <namespace -o custom-**columns=POD:metadata.name,IP:status.podIP --no-headers

**2.** List all network policies in a namespace: **kubectl get networkpolicies -n <namespace>**

**3.** Describe a network policy: **kubectl describe networkpolicy <network-policy-name> -n <namespace>**

**Persistent Volume (PV) and Persistent Volume Claim (PVC) Diagnostics:**

1. List PVs: **kubectl get pv**

**2.** Describe a PV**: kubectl describe pv <pv-name>**

**3.** List PVCs in a namespace: **kubectl get pvc -n <namespace>**

**4.** Describe a PVC**: kubectl describe pvc <pvc-name> -n <namespace>**

**Node Diagnostics:**

**1.** Get the list of pods running on a specific node: **kubectl get pods --field-selector spec.nodeName=<node-name> -n <namespace>**

**Resource Quotas and Limits:**

**1.** List resource quotas in a namespace: **kubectl get resourcequotas -n <namespace>**

**2.** Describe a resource quota: **kubectl describe resourcequota <resource-quota-name> -n <namespace>**

**Custom Resource Definitions (CRD) Diagnostics:**

**1.** List custom resources in a namespace: **kubectl get <custom-resource-name> -n <namespace>**

**2.** Describe a custom resource**: kubectl describe <custom-resource-name> <custom-resource-instance-name> -n <namespace>**

Remember to replace <namespace>, <pod-name>, <service-name>, <deployment-name>, <statefulset-name>, <configmap-name>, <secret-name>, <namespace-name>, <pv-name>, <pvc-name>, <node-name>, <network-policy-name>, <resource-quota-name>, <custom-resource-name>, and <custom-resource-instance-name> with your specific values when using these commands. These commands should help you diagnose various aspects of your Kubernetes cluster and applications running within it.

**Resource Scaling and Autoscaling:**

**1.** Scale a deployment: **kubectl scale deployment <deployment-name> --replicas=<replica-count> -n <namespace>**

**2.** Set autoscaling for a deployment**: kubectl autoscale deployment <deployment-name> --min=<min-pods> --max=<max-pods> --cpu-percent=<cpu-percent> -n <namespace>**

**3.** Check horizontal pod autoscaler status**: kubectl get hpa -n <namespace>**

**Job and CronJob Diagnostics:**

**1.** List all jobs in a namespace**: kubectl get jobs -n <namespace>**

**2.** Describe a job**: kubectl describe job <job-name> -n <namespace>**

**3.** List all cron jobs in a namespace: **kubectl get cronjobs -n <namespace>**

**4.** Describe a cron job**: kubectl describe cronjob <cronjob-name> -n <namespace>**

**Volume Diagnostics:**

**1.** List persistent volumes (PVs) sorted by capacity**: kubectl get pv --sort-by=.spec.capacity.storage**

2. Check PV reclaim policy: **kubectl get pv <pv-name> -o=jsonpath='{.spec.persistentVolumeReclaimPolicy}'**

**3.** List all storage classes**: kubectl get storageclasses**

**Ingress and Service Mesh Diagnostics:**

**1.** List all ingresses in a namespace: **kubectl get ingress -n <namespace>**

**2.** Describe an ingress**: kubectl describe ingress <ingress-name> -n <namespace>**

3. List all VirtualServices (Istio) in a namespace: kubectl get virtualservices -n <namespace>

**4.** Describe a VirtualService (Istio**): kubectl describe virtualservice <virtualservice-name> -n <namespace>**

**Pod Network Troubleshooting:**

**1.** Run a network diagnostic pod (e.g., busybox) for debugging: **kubectl run -it --rm --restart=Never --image=busybox net-debug-pod -- /bin/sh**

**2.** Test connectivity from a pod to a specific endpoint**: kubectl exec -it <pod-name> -n <namespace> -- curl <endpoint-url>**

**3.** Trace network path from one pod to another: **kubectl exec -it <source-pod-name> -n <namespace> -- traceroute <destination-pod-ip>**

**4.** Check DNS resolution from a pod: **kubectl exec -it <pod-name> -n <namespace> -- nslookup <domain-name>**

**Config and Resource Validation:**

**1.** Validate a Kubernetes YAML file without applying it: **kubectl apply --dry-run=client -f <yaml-file>**

2. Validate a pod’s security context and capabilities: kubectl auth can-i list pods --as=system:serviceaccount:<namespace>:<serviceaccount-name>

**RBAC and Security:**

**1.** List roles and role bindings in a namespace: **kubectl get roles,rolebindings -n <namespace>**

**2.** Describe a role or role binding: **kubectl describe role <role-name> -n <namespace>**

**Service Account Diagnostics:**

**1.** List service accounts in a namespace: **kubectl get serviceaccounts -n <namespace>**

**2.** Describe a service account: **kubectl describe serviceaccount <serviceaccount-name> -n <namespace>**

**Node Drain and Uncordon:**

**1.** Drain a node for maintenance**: kubectl drain <node-name> --ignore-daemonsets**

**2.** Uncordon a previously drained node: **kubectl uncordon <node-name>**

**Resource Cleanup:**

**1.** Delete a pod forcefully (not recommended): **kubectl delete pod <pod-name> -n <namespace> --grace-period=0 --force**

**Pod Affinity and Anti-Affinity:**

**1.** List pod affinity rules for a pod: **kubectl get pod <pod-name> -n <namespace> -o=jsonpath='{.spec.affinity}'**

2. List pod anti-affinity rules for a pod: **kubectl get pod <pod-name> -n <namespace> -**o=jsonpath='{.spec.affinity.podAntiAffinity}'

**Pod Security Policies (PSP):**

**1.** List all pod security policies (if enabled): **kubectl get psp**

**Kubernetes Events:**

**1.** View recent cluster events: **kubectl get events --sort-by=.metadata.creationTimestamp**

**2.** Filter events by a specific namespace: **kubectl get events -n <namespace>**

**Node Troubleshooting:**

**1.** Check node conditions**: kubectl describe node <node-name> | grep Conditions -A5**

**2.** List node capacity and allocatable resources: **kubectl describe node <node-name> | grep -E "Capacity|Allocatable"**

**Ephemeral Containers (Kubernetes 1.18+):**

**1.** Run an ephemeral debugging container: **kubectl debug -it <pod-name> -n <namespace> --image=<debug-image> -- /bin/sh**

**Resource Metrics (Metrics Server required):**

**1.** Get CPU and Memory usage for pods: **kubectl top pod -n <namespace>**

**Kubelet Diagnostics:**

**1.** View kubelet logs on a node: **kubectl logs -n kube-system kubelet-<node-name>**

**Advanced Debugging with Telepresence:**

**1.** Debug a pod with Telepresence: **telepresence --namespace <namespace> --swap-deployment <pod-name>**

**Kubeconfig and Contexts:**

**1.** List available contexts: **kubectl config get-contexts**

**2.** Switch to a different context: **kubectl config use-context <context-name>**

**Pod Security Standards (PodSecurity admission controller):**

**1.** List PodSecurityPolicy (PSP) violations: **kubectl get psp -A | grep -vE 'NAME|REVIEWED'**

**Pod Disruption Budget (PDB) Diagnostics:**

**1.** List all PDBs in a namespace: **kubectl get pdb -n <namespace>**

**2.** Describe a PDB: **kubectl describe pdb <pdb-name> -n <namespace>**

**Resource Lock Diagnostics (if using resource locks):**

**1.** List resource locks in a namespace**: kubectl get resourcelocks -n <namespace>**

**Service Endpoints and DNS:**

**1.** List service endpoints for a service**: kubectl get endpoints <service-name> -n <namespace>**

**2.** Check DNS configuration in a pod: **kubectl exec -it <pod-name> -n <namespace> -- cat /etc/resolv.conf**

**Custom Metrics (Prometheus, Grafana):**

1. Query Prometheus metrics: Use **kubectl port-forward** to access Prometheus and Grafana services to query custom metrics.

**Pod Priority and Preemption:**

**1.** List priority classes: **kubectl get priorityclasses**

**Pod Overhead (Kubernetes 1.18+):**

**1.** List overhead in a pod: **kubectl get pod <pod-name> -n <namespace> -o=jsonpath='{.spec.overhead}'**

**Volume Snapshot Diagnostics (if using volume snapshots):**

**1.** List volume snapshots**: kubectl get volumesnapshot -n <namespace>**

**2.** Describe a volume snapshot: **kubectl describe volumesnapshot <snapshot-name> -n <namespace>**

**Resource Deserialization Diagnostics:**

**1.** Deserialize and print a Kubernetes resource**: kubectl get <resource-type> <resource-name> -n <namespace> -o=json**

**Node Taints:**

**1.** List node taints: **kubectl describe node <node-name> | grep Taints**

**Mutating and Validating Webhook Configurations:**

**1.** List mutating webhook configurations**: kubectl get mutatingwebhookconfigurations**

**2.** List validating webhook configurations: **kubectl get validatingwebhookconfigurations**

**Pod Network Policies:**

**1.** List pod network policies in a namespace: **kubectl get networkpolicies -n <namespace>**

**Node Conditions (Kubernetes 1.17+):**

List node conditions: kubectl get nodes -o custom-columns=NODE:.metadata.name,READY:.status.conditions[?(@.type==”Ready”)].status -l ‘node-role.kubernetes.io/worker=’**Audit Logs:**

1. Retrieve audit logs (if enabled): Check your Kubernetes audit log configuration for the location of audit logs.

**Node Operating System Details:**

1. Get the node’s OS information: kubectl get node <node-name> -o jsonpath='{.status.nodeInfo.osImage}'

**List All Running Pods in All Namespaces (Short Command):**

1. List all running pods in all namespaces in a short format: kubectl get pods --all-namespaces

These commands should cover a wide range of diagnostics scenarios in Kubernetes. Make sure to replace placeholders like <namespace>, <pod-name>, <deployment-name>, etc., with actual values specific to your cluster and use case.