EVERY EXAM PREP · CHEAT SHEET

CKA Cheat Sheet 2026

The night-before summary, built like the exam.

—	2h	66%
Questions	Time limit	Passing score
$445	**Cloud Native Computing…**	
Exam fee	Governing body	

The CKA is two hours in a live cluster with no multiple choice — you either produce the object or you don't. Everything below is the working set: the aliases that buy you time, the imperative one-liners that write YAML for you, and the diagnostic sequences for the 30% of the exam that is troubleshooting.

Exam at a glance

Item	Detail
Administered by	CNCF / The Linux Foundation
Format	Online, proctored, performance-based tasks in a live cluster
Time limit	2 hours
Passing score	66%
Cost	$445, includes a second attempt (one free retake)
Scheduling window	12 months from purchase
Certification validity	2 years
Cluster version tested	Kubernetes v1.35
Included simulator	Killer.sh — 2 attempts, 36 hours each

Domain weights

Domain	Weight	What it actually means
Troubleshooting	30%	Broken nodes, pods, DNS, control plane
Cluster Architecture, Installation & Configuration	25%	kubeadm, etcd, RBAC, upgrades
Services & Networking	20%	Services, Ingress, Gateway API, NetworkPolicy
Workloads & Scheduling	15%	Deployments, rollouts, affinity, taints
Storage	10%	PV, PVC, StorageClass

Terminal setup: spend the first 60 seconds here

Type this before you touch a task. It pays for itself by question three.

Source: Cloud Native Computing Foundation (CNCF), in collaboration with The Linux Foundation · Last verified July 18, 2026 · Condensed for print — full s…

Shortcut	Expands to	Use
k	kubectl	Saves ~6 keystrokes per command
$do	--dry-run=client -o yaml	k run nginx --image=nginx $do > pod.yaml
$now	--force --grace-period=0	k delete pod nginx $now — no 30s wait
Namespace pin	k config set-context --current --namespace=<ns>	Stops -n mistakes
Context switch	k config use-context <ctx>	Run the line given in every question
Vim	:set et ts=2 sw=2	YAML indentation survives

Imperative commands that write your YAML

The core technique: never hand-write a manifest. Generate a skeleton with $do, redirect to a file, edit the two fields the question actually asks for, then k apply -f.

Object	Command
Pod	k run nginx --image=nginx --port=80 --labels=app=web $do > pod.yaml
Pod with a command	k run busybox --image=busybox --command $do -- sleep 3600
Throwaway shell	k run tmp -it --rm --restart=Never --image=busybox -- sh
Deployment	k create deployment web --image=nginx --replicas=3 --port=80 $do
Job	k create job pi --image=busybox -- echo done · --from=cronjob/report
CronJob	k create cronjob hello --image=busybox --schedule="*/1 * * * *" -- echo hi
Service	k expose deployment web --port=80 --target-port=8080 --type=NodePort --name=web-svc
Ingress	k create ingress web --class=nginx --rule="foo.com/bar*=web-svc:80,tls=my-cert"
ConfigMap	k create cm app-cfg --from-literal=KEY=val --from-file=./app.conf
Secret	k create secret generic db --from-literal=pass=s3cr3t · tls · docker-registry
Namespace / SA	k create ns dev · k create sa builder
Scale / image	k scale deployment web --replicas=5 · k set image deployment/web nginx=nginx:1.27
Rollout	k rollout status|history|undo deployment/web · --to-revision=2
Edit live	k edit deploy web · k patch svc web -p '{"spec":{"type":"NodePort"}}'

Reading the cluster fast

Need	Command
Field names you forgot	k explain pod.spec.tolerations --recursive
Sort by restarts	k get pods --sort-by='.status.containerStatuses[0].restartCount'
Events in time order	k get events -A --sort-by=.metadata.creationTimestamp

Resource usage	k top nodes · k top pods --sort-by=cpu

Troubleshooting — 30% of the exam

Work the same ladder every time: get for state, describe for events, logs for the application, journalctl for the node.

Node NotReady

Step	Command	Looking for
1	k get nodes -o wide · k describe node <node>	Conditions: MemoryPressure, DiskPressure, PIDPressure; taints
2	ssh <node>; systemctl status kubelet	Service dead or crash-looping
3	journalctl -u kubelet -f · journalctl -xeu kubelet	Bad config path, bad cert, runtime down
4	systemctl status containerd · crictl ps	Runtime socket unavailable
5	systemctl daemon-reload && systemctl restart kubelet	After fixing /var/lib/kubelet/config.yaml

Pod stuck in Pending

- k describe pod <pod> — read the Events block first; the scheduler writes the reason there.
- insufficient cpu -> lower resources.requests or free capacity.
- had untolerated taint -> add a toleration, or k taint nodes <node> key=value:NoSchedule-.
- didn't match Pod's node affinity/selector -> fix nodeSelector, or k label node <node> disk=ssd.
- unbound immediate PersistentVolumeClaims -> no matching PV; check k get pvc,pv,sc.
- No events at all -> scheduler is down: k get pods -n kube-system.

CrashLoopBackOff

Command	Purpose
k logs <pod> --previous	Logs from the container that just died — the most useful flag on the exam
k logs <pod> -c <container>	Multi-container pods and initContainers
k describe pod <pod>	Last State: Terminated, Exit Code, OOMKilled
k get pod <pod> -o yaml	Bad command / args, missing env, failing probes
k debug <pod> -it --image=busybox --target=<c>	Ephemeral container in a pod with no shell

ImagePullBackOff / ErrImagePull

- k describe pod <pod> -> Events name the exact failure: typo, tag not found, or 401 Unauthorized.
- Auth failure -> create a docker-registry secret, then add spec.imagePullSecrets.
- Air-gapped node -> verify with crictl pull <image> and crictl images.

Service not resolving or not routing

Step	Command	Failure it isolates
1	k get svc <svc> -o yaml	Wrong port / targetPort, wrong type

2	k get endpointslices -l kubernetes.io/service-name=<svc>	Empty = selector matches no ready pod
3	k get pods -l <selector> --show-labels	Label mismatch between Service and Pods
4	k run tmp -it --rm --restart=Never --image=busybox -- nslookup <svc>.<ns>.svc.cluster.local	DNS resolution
5	k get pods -n kube-system -l k8s-app=kube-dns · k logs -n kube-system <coredns-pod>	CoreDNS down or misconfigured
6	k get daemonset -n kube-system kube-proxy	kube-proxy not running on the node
7	wget -qO- <clusterIP>:<port> from a pod	IP works, name doesn't -> DNS; neither -> NetworkPolicy or CNI

Control plane down

Control-plane components are static pods. If kubectl itself fails, go to the node.

- crictl ps -a then crictl logs <container-id> — the only way to read a kube-apiserver that won't start.
- Manifests: /etc/kubernetes/manifests/ — kube-apiserver.yaml, kube-controller-manager.yaml, kube-scheduler.yaml, etcd.yaml.
- Edit the file and the kubelet restarts the pod — no apply. Never keep backups in that directory.
- Mirror pod name is <pod>-<node>; path set by staticPodPath in /var/lib/kubelet/config.yaml.
- Certs: kubeadm certs check-expiration · kubeadm certs renew all (/etc/kubernetes/pki).

kubeadm cluster lifecycle

Task	Command
Init control plane	kubeadm init --pod-network-cidr=<cidr> --apiserver-advertise-address=<ip>
Set up kubeconfig	mkdir -p ~/.kube && cp -i /etc/kubernetes/admin.conf ~/.kube/config
Recover a join command	kubeadm token create --print-join-command · kubeadm token list (--ttl 24h0m0s)
Join a worker	kubeadm join <endpoint> --token <token> --discovery-token-ca-cert-hash sha256:<hash>
Recompute the CA hash	openssl x509 -pubkey -in /etc/kubernetes/pki/ca.crt | openssl rsa -pubin -outform der 2>/dev/null | openssl dgst -sha256 -hex | sed 's/^.* //'
Join a control plane	add --control-plane --certificate-key <key>
Reset a node	kubeadm reset

Upgrade order — control plane first, one node at a time

#	Control plane node	Worker node
1	apt-mark unhold kubeadm && apt-get install -y kubeadm='1.35.x-*' && apt-mark hold kubeadm	same
2	kubeadm upgrade plan	—
3	kubeadm upgrade apply v1.35.x (first CP only; others use kubeadm upgrade node)	kubeadm upgrade node