CKNE · CHEAT SHEET · CERTIFYDECK

Certified Kubernetes Network Engineer (CKNE)

Dominios, pesos oficiales y habilidades clave del temario de CKNE, para repasar antes del examen.

Nivel: Intermediate (Associate) Duración: 120 min Fuente: guía de estudio oficial de Microsoft Learn

1. Core Infrastructure and CNI

15–15%

Installing and Configuring CNI Plugins

- El entorno de ejecución de contenedores busca los ejecutables CNI en /opt/cni/bin y la configuración en /etc/cni/net.d; en ADD ejecuta los complementos de la lista en orden y en DEL en orden inverso. Sin complemento de red el nodo queda NotReady y CoreDNS en Pending. Cilium se instala con helm install en kube-system, se comprueba con cilium status y los Pods previos deben reiniciarse.

Managing IPAM and Pod CIDR Allocation

- kube-controller-manager reparte a cada nodo un spec.podCIDR tomado de --cluster-cidr cuando --allocate-node-cidrs está activo, con el tamaño de --node-cidr-mask-size. host-local asigna desde sus rangos y guarda el estado en /var/lib/cni/networks. Cilium cluster-pool usa CiliumNode y se amplía añadiendo CIDR a la lista; las ClusterIP de Service salen de los ServiceCIDR, no del rango de Pods.

Using Linux Tools (iptables, ip, tcpdump) for Packet-level Issues

- ip route get muestra la ruta que usaría el kernel, ip neigh el estado de resolución de vecinos, ip -d link los detalles de veth o VXLAN y nsenter -t PID -n entra en la red de un contenedor. iptables-save revela las cadenas KUBE-SERVICES, KUBE-SVC y KUBE-SEP; en modo nftables, nft list table ip kube-proxy. tcpdump -i any -n con filtros host y port indica si llegan los SYN y si hay respuesta.

Troubleshooting Pod Connectivity (DNS, pod-to-pod)

- Para depurar el DNS desde un Pod, revisa /etc/resolv.conf (nameserver del Service kube-dns, dominios de búsqueda, ndots), comprueba los Pods de CoreDNS (etiqueta k8s-app=kube-dns) y sus EndpointSlices, y recuerda que una política de salida que lo deniega todo bloquea también el DNS. Los Pods se comunican entre nodos sin NAT; kubectl debug añade contenedores efímeros o abre una sesión en el nodo para capturar tráfico.

Configuring Multi-interface Pods

- Multus es un metacomplemento CNI: mantiene la red predeterminada del clúster en eth0 y añade interfaces extra definidas en objetos NetworkAttachmentDefinition (k8s.cni.cncf.io/v1) con una configuración CNI como macvlan, ipvlan o host-device. El Pod las pide con la anotación k8s.v1.cni.cncf.io/networks (namespace/nombre@interfaz o JSON) y el resultado se lee en k8s.v1.cni.cncf.io/network-status, con interfaces net1, net2…

2. Service Networking and DNS

25–25%

Configuring L4 Services

- Un Service expone Pods por selector con port y targetPort (también con nombre): ClusterIP es el tipo predeterminado, NodePort abre el mismo puerto en cada nodo, LoadBalancer requiere una implementación externa, ExternalName devuelve un CNAME y clusterIP: None crea un Service headless. externalTrafficPolicy Local conserva la IP de origen, internalTrafficPolicy Local exige un endpoint local y PreferSameZone prefiere la propia zona.

Understanding kube-proxy and CNI Alternatives

- kube-proxy programa en cada nodo las reglas que llevan el tráfico de la ClusterIP a los endpoints, en modo iptables, nftables o ipvs (obsoleto); el modo se fija con el campo mode de la KubeProxyConfiguration, que en kubeadm vive en el ConfigMap kube-proxy. Cilium puede sustituir a kube-proxy con kubeProxyReplacement=true, k8sServiceHost y k8sServicePort, tras retirar el DaemonSet, el ConfigMap y las reglas KUBE de los nodos.

Customizing coreDNS for Services

- CoreDNS sirve el Service kube-dns y se configura con el Corefile del ConfigMap coredns de kube-system: el plugin kubernetes responde los nombres del clúster, forward envía el resto a servidores externos y un bloque de servidor propio crea un dominio auxiliar. hosts añade entradas estáticas (con fallthrough para no cortar la cadena), rewrite crea alias hacia un Service y reload aplica los cambios.

Troubleshooting Service Network Traffic

- Para depurar un Service se avanza por capas: que exista y se resuelva por DNS, que su selector coincida con las etiquetas de los Pods (kubectl get endpointslices -l kubernetes.io/service-name=web), que el targetPort y el protocolo sean los del contenedor y que kube-proxy haya programado las reglas KUBE-SVC y KUBE-SEP en cada nodo. Las políticas de tráfico Local descartan el tráfico en nodos sin endpoints locales.

Configuring Pod Endpoint Availability

- La readiness probe y las readinessGates deciden si la dirección de un Pod está lista en la EndpointSlice; las sondas de liveness y startup reinician o retrasan, no retiran el tráfico. Al borrar un Pod, su endpoint pasa a terminating con ready a false y un hook preStop en terminationGracePeriodSeconds drena conexiones. Los PodDisruptionBudget limitan los desalojos voluntarios y maxSurge, maxUnavailable y minReadySeconds protegen los despliegues.

Managing Traffic with the Gateway API (Gateway, HTTPRoutes)

- Un Gateway define listeners (protocolo, puerto, hostname) y con allowedRoutes decide qué espacios de nombres pueden adjuntar rutas; la HTTPRoute se vincula con parentRefs y enruta por host, ruta, cabeceras, parámetros y método hacia backendRefs con pesos proporcionales. Los filtros redirigen, reescriben, modifican cabeceras o reflejan tráfico, y las condiciones Accepted y ResolvedRefs explican por qué una ruta no funciona.

3. Advanced Traffic Management

20–20%

Optimizing LLM Traffic

- Con Gateway API Inference Extension, una InferencePool agrupa los Pods de servidores de modelos (selector y targetPorts) y sustituye al Service como backendRef de la HTTPRoute. Su endpointPickerRef apunta a un Endpoint Picker que elige el Pod según métricas como la cola de peticiones y el uso de la caché KV; failureMode decide si, cuando falla, se reenvía igualmente (FailOpen) o se descarta la petición (FailClose).

Implementing Routing to Expose Networks

- Con Cilium, el BGP Control Plane (CiliumBGPClusterConfig, CiliumBGPPeerConfig y CiliumBGPAdvertisement) anuncia a los routers el CIDR de Pods de cada nodo y las VIP de los Service, cuyas IP de LoadBalancer asigna LB IPAM desde un CiliumLoadBalancerIPPool. La encapsulación VXLAN o Geneve no exige que la red conozca los CIDR de Pods; el enrutamiento nativo sí. externalIPs está obsoleto y hostPort ata los Pods a los nodos.

Configuring Egress Gateways for Cluster Exit Traffic

- Con Cilium, una CiliumEgressGatewayPolicy (de ámbito de clúster; requiere bpf.masquerade y kubeProxyReplacement) envía el tráfico de los Pods seleccionados hacia destinationCIDRs, menos excludedCIDRs, por un nodo de puerta de enlace que aplica SNAT con egressIP o la IP de una interfaz. En Istio, istio-egressgateway con ServiceEntry, Gateway, VirtualService y DestinationRule, y REGISTRY_ONLY bloquea hosts no registrados.

Implementing Cross Cluster Service Discovery and Load Balancing

- La Multi-Cluster Services API exporta un Service con un ServiceExport del mismo nombre; el controlador crea el ServiceImport y los EndpointSlices, y el nombre <svc>.<ns>.svc.clusterset.local llega a los backends de todo el ClusterSet. En Cilium Cluster Mesh, cada clúster necesita nombre e ID únicos y CIDR de Pods sin solapes; los Global Services usan service.cilium.io/global, shared y affinity.

4. Network Security and Policy

25–25%

Securing Traffic with Network Policies

- Una NetworkPolicy solo funciona si el complemento de red la aplica; aísla los Pods seleccionados por sentido y sus reglas de permiso se suman. podSelector: {} sin reglas crea la denegación por defecto, y un único elemento from con namespaceSelector y podSelector es un AND. Para reglas HTTP, toFQDNs o ámbito de clúster, Cilium ofrece CiliumNetworkPolicy y CiliumClusterwideNetworkPolicy.

Implementing Node and Pod Level Encryption

- Cilium cifra de forma transparente el tráfico entre Pods de nodos distintos con WireGuard (interfaz cilium_wg0, claves por nodo publicadas en CiliumNode) o con IPsec (Secret cilium-ipsec-keys en el espacio de nombres de Cilium y rotación incrementando el ID de clave). Istio añade TLS mutuo entre cargas de trabajo, con sidecars o con ztunnel y HBONE en modo ambient, que además autentica identidades.

Managing TLS Certificates for Gateway API

- En Gateway API, un listener HTTPS o TLS en modo Terminate referencia un Secret kubernetes.io/tls en tls.certificateRefs; si el Secret está en otro espacio de nombres hace falta una ReferenceGrant allí, y Passthrough se enruta con TLSRoute. BackendTLSPolicy valida el TLS hacia el Service. cert-manager, con su soporte de Gateway API activado, crea Certificates a partir de las anotaciones del Gateway.

Implementing Pod-level Authentication and Authorization

- En Istio, PeerAuthentication fija el modo mTLS (STRICT, PERMISSIVE, DISABLE) para la malla, un espacio de nombres o una carga de trabajo, y AuthorizationPolicy permite o deniega por identidad (cluster.local/ns/<ns>/sa/<sa>), método o ruta; DENY prevalece sobre ALLOW. RequestAuthentication valida JWT y necesita una política con requestPrincipals. En modo ambient, ztunnel aplica L4 y las reglas L7 exigen un waypoint.

5. Observability

15–15%

Analyzing Network Health Using Metrics

- Los contadores como istio_requests_total se leen con rate() o increase(), aplicando rate() antes de sum, y la latencia con histogram_quantile() sobre las series _bucket conservando le. Las etiquetas reporter, response_code y source_workload localizan el fallo. En Cilium, las métricas de Hubble (drop, flow, dns, httpV2) se habilitan aparte con hubble.metrics.enabled; CoreDNS expone coredns_* con su complemento prometheus.

Troubleshooting End-to-End Network Performance with Tracing

- En Istio los proxies envían intervalos por sí mismos, pero la aplicación debe reenviar traceparent, tracestate y x-request-id para que todos los saltos formen una sola traza. El proveedor se define en meshConfig.extensionProviders (por ejemplo opentelemetry hacia Jaeger o un OpenTelemetry Collector) y se activa con un recurso Telemetry, que también fija randomSamplingPercentage por malla, espacio de nombres o carga de trabajo.

Auditing Traffic with Logs

- Con Cilium, hubble observe filtra registros de flujo por espacio de nombres, Pod, veredicto (DROPPED) o protocolo; el exportador de Hubble los escribe en ficheros con listas de filtros y máscaras de campos. En Istio los registros de acceso de Envoy se activan con Telemetry o meshConfig.accessLogFile y se leen con kubectl logs en el contenedor istio-proxy; CoreDNS anota consultas con el plugin log; la auditoría solo recoge llamadas a la API.