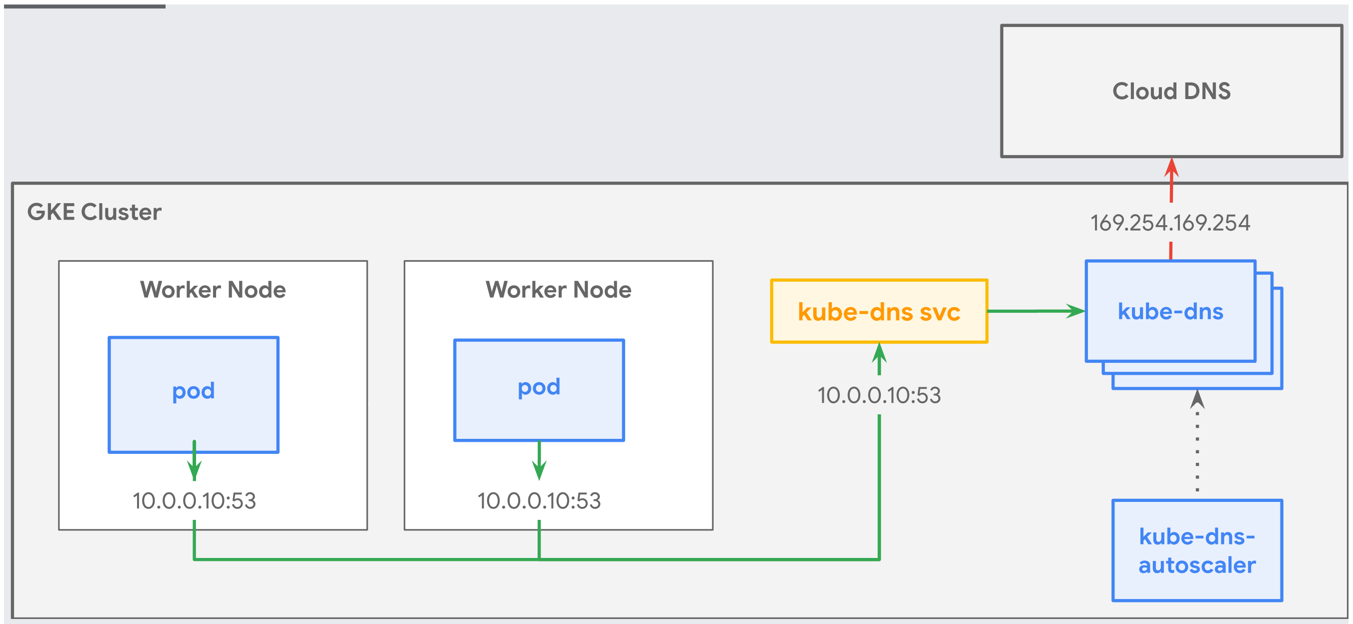
This page describes how Google Kubernetes Engine (GKE) implements service discovery and managed DNS. For a general overview of how DNS is used in Kubernetes clusters, see DNS for Services and Pods (https://kubernetes.io/docs/concepts/services-networking/dns-pod-service/).

In Kubernetes, service discovery is implemented with autogenerated service names that map to the service's IP address. Service names follow a standard specification (https://github.com/kubernetes/dns/blob/master/docs/specification.md): `my-svc.my-namespace.svc.my-zone`. Pods can also access external services, like example.com, through their names. See DNS for Services and Pods (https://kubernetes.io/docs/concepts/services-networking/dns-pod-service/) for more information on the behaviour of DNS in Kubernetes.

GKE provides managed DNS for resolving service names and for resolving external names. This is implemented by kube-dns, a cluster add-on that is deployed by default in all GKE clusters. kube-dns runs as a Deployment (/kubernetes-engine/docs/concepts/deployment) that schedules redundant `kube-dns` Pods to nodes in the cluster. The `kube-dns` Pods are in the `kube-system` namespace. The kube-dns deployment is accessed through a corresponding Service (/kubernetes-engine/docs/concepts/service) that groups the `kube-dns` Pods and gives them a single IP address. By default, all Pods in a cluster use this service to resolve DNS queries.



kube-dns scales to serve the DNS demands of the cluster. This scaling is controlled by the `kube-dns-autoscaler` which is deployed by default in all GKE clusters. `kube-dns-autoscaler` adjusts the number of replicas in the `kube-dns` deployment based on the number of nodes and cores in the cluster.

The kubelet agent running on each Pod configures the Pod's `etc/resolv.conf` to use the `kube-dns` service's ClusterIP. An example of this configuration is shown below, in this example the IP address of the `kube-dns` service is `10.0.0.10` (this IP address will be different in other clusters):

kube-dns is the authoritative name server for the cluster domain (`cluster.local`) and it recursively resolves external names. Short names that are not fully qualified, like `myservice`, are completed first with local search paths. For example, `myservice.default.svc.cluster.local`, `myservice.svc.cluster.local`, `myservice.cluster.local`, `myservice.c.my-project-id.internal`, and `myservice.google.internal`.

- Learn how to provide scalable DNS resolution using NodeLocal DNSCache (/kubernetes-engine/docs/how-to/nodelocal-dns-cache) for clusters requiring high volumes of DNS queries.