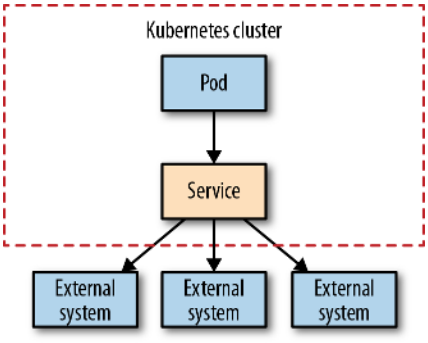

Figure 12-4. Manual service discovery

While connecting to an external resource is this mechanism's most common use, it is not the only one. Endpoints can hold IP addresses of Pods, but not virtual IP addresses of other Services. One good thing about the Service is that it allows adding and removing selectors and pointing to external or internal providers without deleting the resource definition that would lead to a Service IP address change. So service consumers can continue using the same Service IP address they first pointed to, while the actual service provider implementation is migrated from on-premises to Kubernetes without affecting the client.

In this category of manual destination configuration, there is one more type of Service, as shown in Example 12-5.

Example 12-5. Service with an external destination

```
apiVersion: v1
kind: Service
metadata:
  name: database-service
spec:
  type: ExternalName
  externalName: my.database.example.com
  ports:
  - port: 80
```

This Service definition does not have a `selector` either, but its type is `ExternalName`. That is an important difference from an implementation point of view. This Service definition maps to the content pointed by `externalName` using DNS only. It is a way of creating an alias for an external endpoint using DNS CNAME rather than going through the proxy with an IP address. But fundamentally, it is another way of providing a Kubernetes abstraction for endpoints located outside of the cluster.