



# Service Discovery

In this lesson, we'll study service discovery with Consul.

## We'll cover the following



- Distinguishing features
  - HTTP REST API & DNS support
  - Configuration file generation
  - Health checks
  - Replication
  - Multiple data centers
  - Service configuration
- Consul dashboard
- Reading data with DNS
- Consul Docker image

Consul (<http://www.consul.io>) is a service discovery technology that ensures microservices can communicate with each other.

## Distinguishing features #

Consul has some features that set it apart from other service discovery solutions.

## HTTP REST API & DNS support #

Consul has an *HTTP REST API* and supports **DNS**



- DNS (<http://www.zytrax.com/books/dns/>) (Domain Name System) is the system that maps host names such as [www.innoq.com](http://www.innoq.com) (<http://www.innoq.com>) to IP addresses on the Internet.
- In addition to returning IP addresses, it can return ports at which a service is available.
- This is a feature of the SRV DNS records.

## Configuration file generation #

With Consul Template (<https://github.com/hashicorp/consul-template>), Consul can generate configuration files.

- The files may contain IP addresses and ports of services registered in Consul.
- *Consul Template* also provides Consul's service discovery to systems that cannot access Consul via the API.
- The systems have to use some kind of configuration file, which they often already do anyway.

## Health checks #

Consul can perform **health checks** and exclude services from service discovery when the health check fails.

- For example, a health check can be a request to a specific HTTP resource to determine whether the service can still process requests.
- A service may still be able to accept HTTP requests, but it may not be able to process them properly due to a database failure.
- The service can signal this through the health check.

## Replication #

Consul supports **replication** and can ensure high availability.



If a Consul server fails, other servers with replicated data take over and compensate for the failed server.

## Multiple data centers #

Consul also supports **multiple data centers**.

- Data can be replicated between data centers to further increase availability and protect Consul against the failure of a data center.
- The search for services can be limited to the same data center.
- Services in the same data center usually deliver higher performance.

## Service configuration #

Finally, Consul can be used not only for service discovery, but for the **configuration of services**.

- Configuration has different requirements.
- Availability is important for service discovery.
- Faulty information can be tolerated.
- If the service is accessed and is not available, it doesn't matter. You can simply use another instance of the service.
- However, if services are configured incorrectly, this can lead to errors.
- **Correct information is more important** for configuration than for service discovery.

## Consul dashboard #

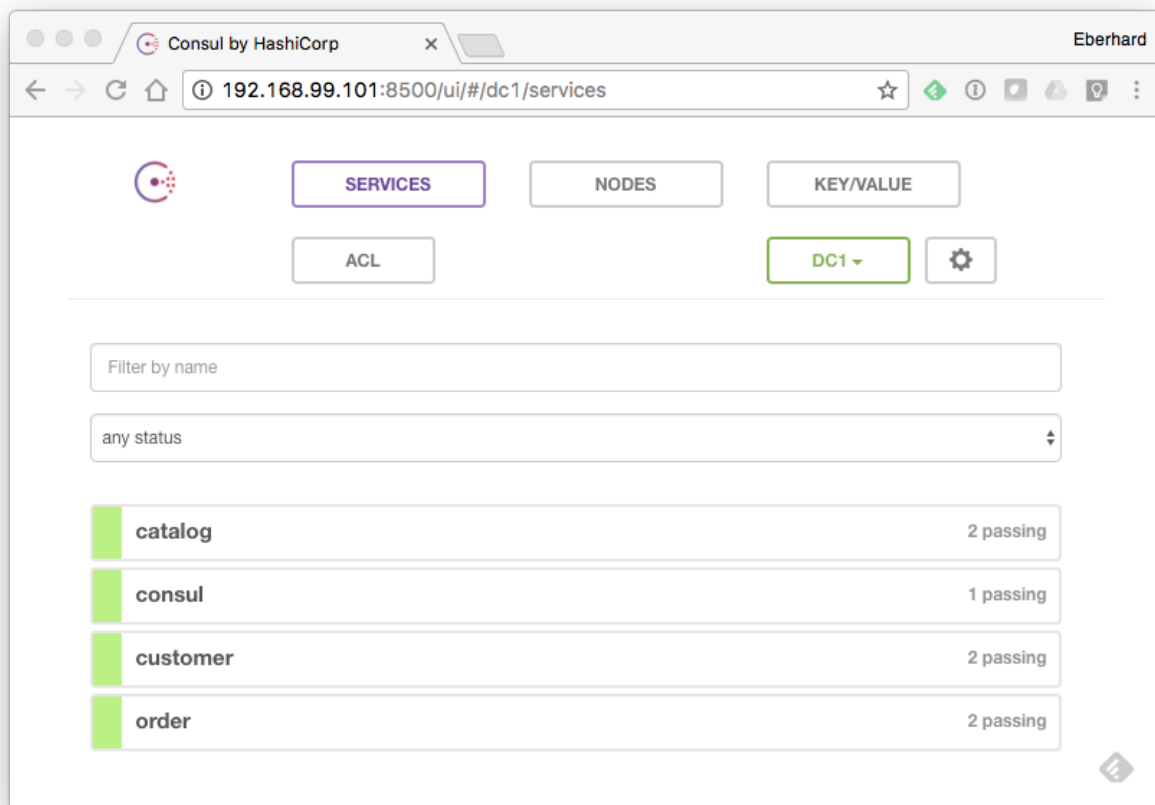


Access to the information about registered microservices is provided by Consul in its **dashboard**, see the screenshot below.

It shows **all services** that have registered at the Consul server and the result of their **health checks**.

In the example, all microservices are healthy and can accept requests.

In addition to the registered services, it displays the **servers on which Consul is running** and the **contents of the configuration database**. Access to the dashboard is possible at port 8500 on the Docker host.



Consul Dashboard

## Reading data with DNS #



It is also possible to access the data from the Consul server via DNS. This can be done, for example, with the tool `dig`. `dig @localhost -p 8600 order.service.consul.` returns the IP address of the order microservice if the Docker containers run on the local computer `localhost`.

Communication to the Consul DNS server takes place via the UDP port 8600. `dig @localhost -p 8600 order.service.consul. SRV` returns the IP address and the port at which the service is available. DNS SRV records are used for this purpose; they are part of the DNS standard and allow you to specify a port for services in addition to the IP address.

## Consul Docker image #

The example uses a Consul Docker image directly from the manufacturer Hashicorp. It is configured so that Consul only stores the data in the main memory and runs on a single node. This simplifies the setup of the system and reduces resource consumption.

This configuration is **unsuitable for a production environment** because data can be lost, and a failure of the Consul Docker container would bring the entire system to a standstill. In production, a cluster of Consul servers should be used, and the data should be stored persistently.

# QUI

# Z

microservice that lists customers?



- ☐ A) It can ask to list all customers and see if the result is returned appropriately.
- ☐ B) It can ask to list all customers and see if the request is accepted.
- ☐ C) It can try to check for its DNS record with `dig`.

Submit Answer



Question 1 of 2  
0 attempted



Reset Quiz

In the next lesson, we'll discuss Apache httpd.

[← Back](#)

Example

[Next →](#)

Routing: Apache httpd



Mark as Completed



Report an Issue

