**Práctica Service Discovery & Service Mesh**

**Por Prof. Oscar H. Mondragón**

## Objetivos

* Comprender el funcionamiento de las mallas de servicios (service mesh)
* Construir un service mesh usando Consul y explorar sus servicios de descubrimiento

## Herramientas a utilizar

* Vagrant
* VirtualBox
* Consul

## Introducción

En esta práctica implementaremos un service mesh con la ayuda de Consul (<https://www.consul.io/docs/connect>) y exploraremos sus servicios de descubrimiento. Implementaremos una aplicación sencilla en JavaScript, la cual registrará varias replicas en Consul.

## Desarrollo de la práctica

## Configuración de Vagrant

Esta práctica la desarrollaremos usando boxes de Ubuntu 22.04 en Vagrant. El Vagrantfile que usaremos es el siguiente (el mismo con el que venimos trabajando):

# -\*- mode: ruby -\*-

# vi: set ft=ruby :

Vagrant.configure("2") do |config|

if Vagrant.has\_plugin? "vagrant-vbguest"

config.vbguest.no\_install = true

config.vbguest.auto\_update = false

config.vbguest.no\_remote = true

end

config.vm.define :clienteUbuntu do |clienteUbuntu|

clienteUbuntu.vm.box = "bento/ubuntu-22.04"

clienteUbuntu.vm.network :private\_network, ip: "192.168.100.2"

clienteUbuntu.vm.hostname = "clienteUbuntu"

end

config.vm.define :servidorUbuntu do |servidorUbuntu|

servidorUbuntu.vm.box = "bento/ubuntu-22.04"

servidorUbuntu.vm.network :private\_network, ip: "192.168.100.3"

servidorUbuntu.vm.hostname = "servidorUbuntu"

end

end

## Instalación de Consul

Abra una sesión SSH con servidorUbuntu

vagrant ssh servidorUbuntu

Dirigirse a <https://www.consul.io/downloads> y seguir los pasos de instalación de Consul en Linux:

$ wget -O- https://apt.releases.hashicorp.com/gpg | gpg --dearmor | sudo tee /usr/share/keyrings/hashicorp-archive-keyring.gpg

$ echo "deb [signed-by=/usr/share/keyrings/hashicorp-archive-keyring.gpg] https://apt.releases.hashicorp.com $(lsb\_release -cs) main" | sudo tee /etc/apt/sources.list.d/hashicorp.list

$ sudo apt update && sudo apt install consul

Verifique la versión de Consul instalada:

$ consul -v

Consul v1.13.2

Revision 0e046bbb

Build Date 2022-09-20T20:30:07Z

Protocol 2 spoken by default, understands 2 to 3 (agent will automatically use protocol >2 when speaking to compatible agents)

## Arrancar un agente de cónsul

Para iniciar un agente de cónsul, puede usar el siguiente comando:

consul agent -ui -dev -bind=192.168.100.3 -client=0.0.0.0 -data-dir=.

La opción -ui se agrega para poder acceder al dashboard de cónsul

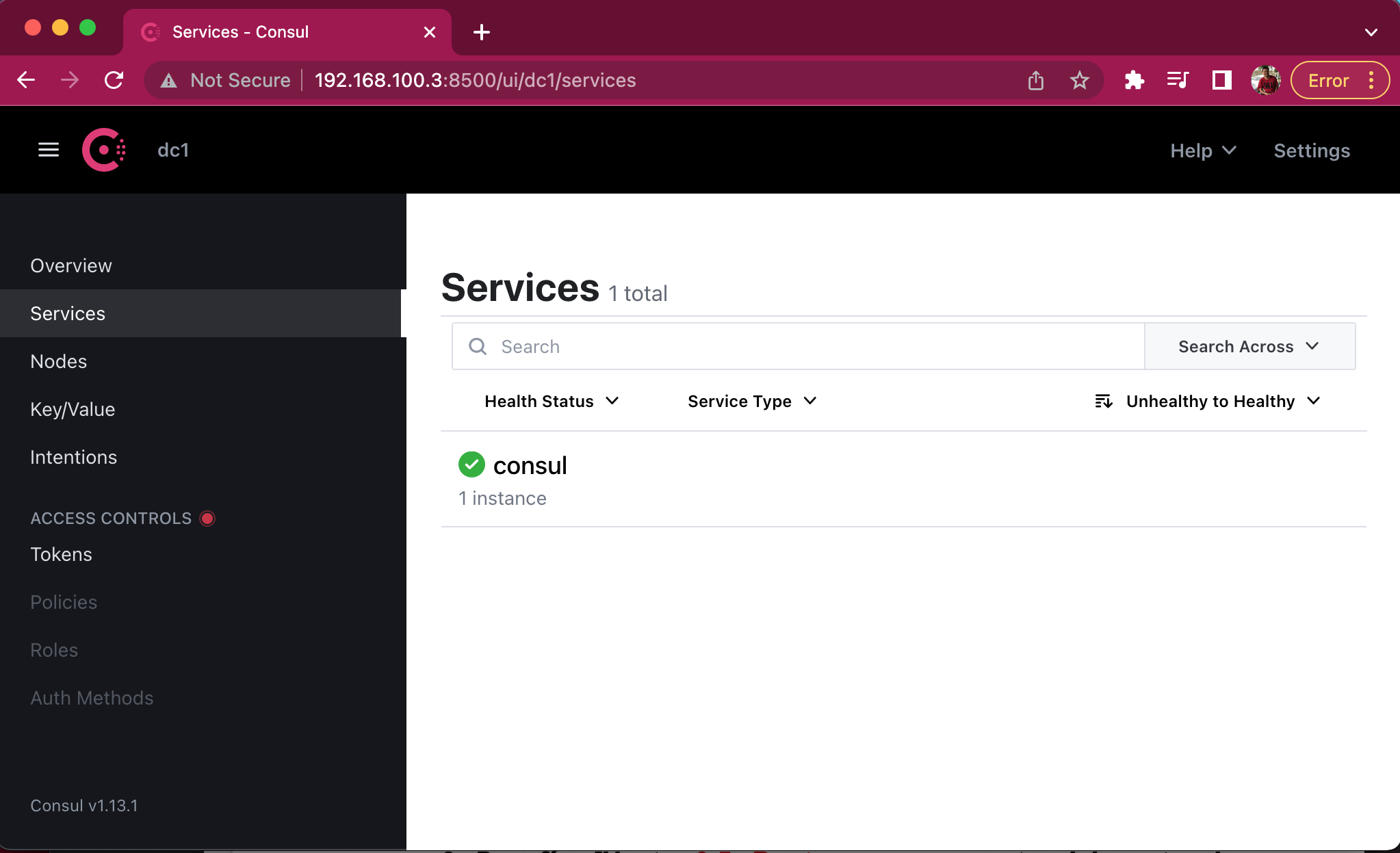
Ya que no desplegaremos por ahora un cluster cónsul, debemos usar la opción -dev para hacer una prueba local

Mediante la opción -bind especificamos la interfaz de red que usaremos en el servidor para cónsul

-client=0.0.0.0 nos permitera acceder al dashboard desde maquinas remotas

En el browser del anfitrión abrir:

<http://192.168.100.3:8500/ui/dc1/services>



## Aplicación JavaScript & Consul

Deje corriendo en agente de Consul lanzado en el punto anterior y abra otra consola del servidor

vagrant ssh servidorUbuntu

Desarrollaremos un servicio en JavaScript y lo registraremos en cónsul.

Instalar nodejs

$ sudo apt install nodejs

$ node -v

$ sudo apt install npm

$ npm -v

Si se desea una versión superior de node se puede instalar de esta forma

curl -fsSL https://deb.nodesource.com/setup\_20.x | sudo -E bash - &&\

sudo apt-get install -y nodejs

Clonar el repositorio con la aplicación JavaScript

git clone <https://github.com/omondragon/consulService>

cd consulService/app

Visualice y analice el archivo index.js

const Consul = require('consul');

const express = require('express');

const SERVICE\_NAME='mymicroservice';

const SERVICE\_ID='m'+process.argv[2];

const SCHEME='http';

const HOST='192.168.100.3';

const PORT=process.argv[2]\*1;

const PID = process.pid;

/\* Inicializacion del server \*/

const app = express();

const consul = new Consul();

app.get('/health', function (req, res) {

console.log('Health check!');

res.end( "Ok." );

});

app.get('/', (req, res) => {

console.log('GET /', Date.now());

res.json({

data: Math.floor(Math.random() \* 89999999 + 10000000),

data\_pid: PID,

data\_service: SERVICE\_ID,

data\_host: HOST

});

});

app.listen(PORT, function () {

console.log('Servicio iniciado en:'+SCHEME+'://'+HOST+':'+PORT+'!');

});

/\* Registro del servicio \*/

var check = {

id: SERVICE\_ID,

name: SERVICE\_NAME,

address: HOST,

port: PORT,

check: {

http: SCHEME+'://'+HOST+':'+PORT+'/health',

ttl: '5s',

interval: '5s',

timeout: '5s',

deregistercriticalserviceafter: '1m'

}

};

consul.agent.service.register(check, function(err) {

if (err) throw err;

});

Desde el directorio de trabajo ~/consulService/app instalar dependencias de consul, express

~/consulService/app$ npm install consul

~/consulService/app$ npm install express

Note que con el comando npm install estas dependencias se instalan localmente en el directorio de trabajo ~/consulService/app$ , si requiere una instalación global debe usar npm install -g.

## Pruebas

En diferentes consolas, desde el directorio de trabajo ~/consulService/app corra varias instancias del servicio:

~/consulService/app$ node index.js 3000

~/consulService/app$ node index.js 3001

~/consulService/app$ node index.js 3002

Visualice de nuevo el dashboard de cónsul. En el browser del anfitrión abrir:

<http://192.168.100.3:8500/ui/dc1/services>

Consulte la lista de servicios a traves del API Rest de Consul (en una nueva consola

)

curl http://192.168.100.3:8500/v1/agent/services

## Ejercicio

Ejecute los siguientes comandos, y analice sus resultados.

1. Haga un Query a la interfaz DNS que en cónsul corre en el puerto 8600 por defecto. Explique el resultado.

dig @127.0.0.1 -p 8600 mymicroservice.service.consul

1. Agregue SRV al comando anterior y explique el resultado.

dig @127.0.0.1 -p 8600 mymicroservice.service.consul SRV

1. HTTP API. Ejecute y explique los resultados de los siguientes comandos.

curl http://localhost:8500/v1/catalog/service/mymicroservice

curl 'http://localhost:8500/v1/health/service/mymicroservice?passing'

## Entregables y Evaluación

* Sustentación de la práctica

## Bibliografía

* Hashicorp Consul. <https://www.consul.io/docs/connect>
* Microservicios | Service Discovery con Hashicorp Consul y Javascript. <https://www.albertcoronado.com/2020/06/03/microservicios-service-discovery-con-hashicorp-consul-y-javascript/>
* Documentacion Consul JavaScript. <https://www.npmjs.com/package/consul>