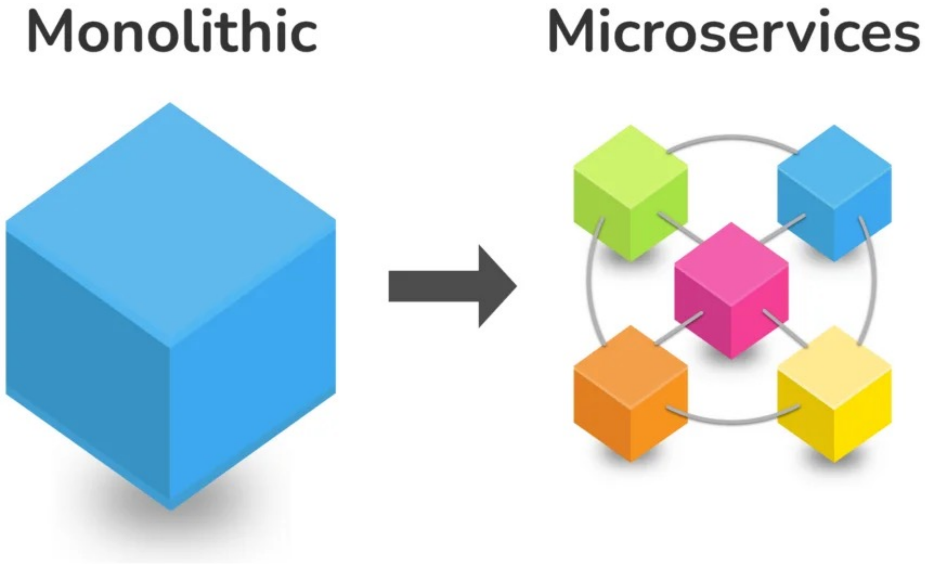
MICROSERVICES ROADMAP

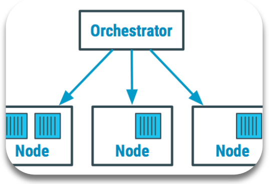
DR. MARYAM LOTFI

SEYED ALIREZA TABATABAEIAN



WHAT WE NEED TO KNOW

Docker



Container Orchestration

Container Management

API gateway

Load Balancing

Service Discovery

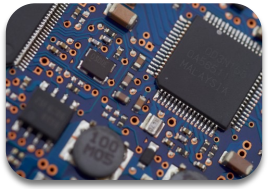
Event Bus

Logging

Monitoring

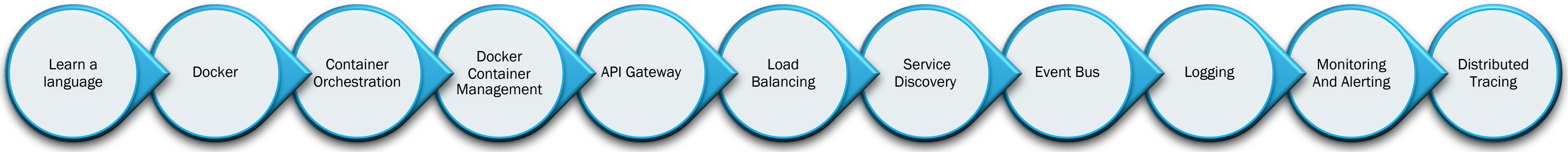
Distributed Tracing

Data Persistence



Caching

ROADMAP



Learn a language

Docker

Container Orchestration

Docker Container Management

API Gateway

Load Balancing

Service Discovery

Event Bus

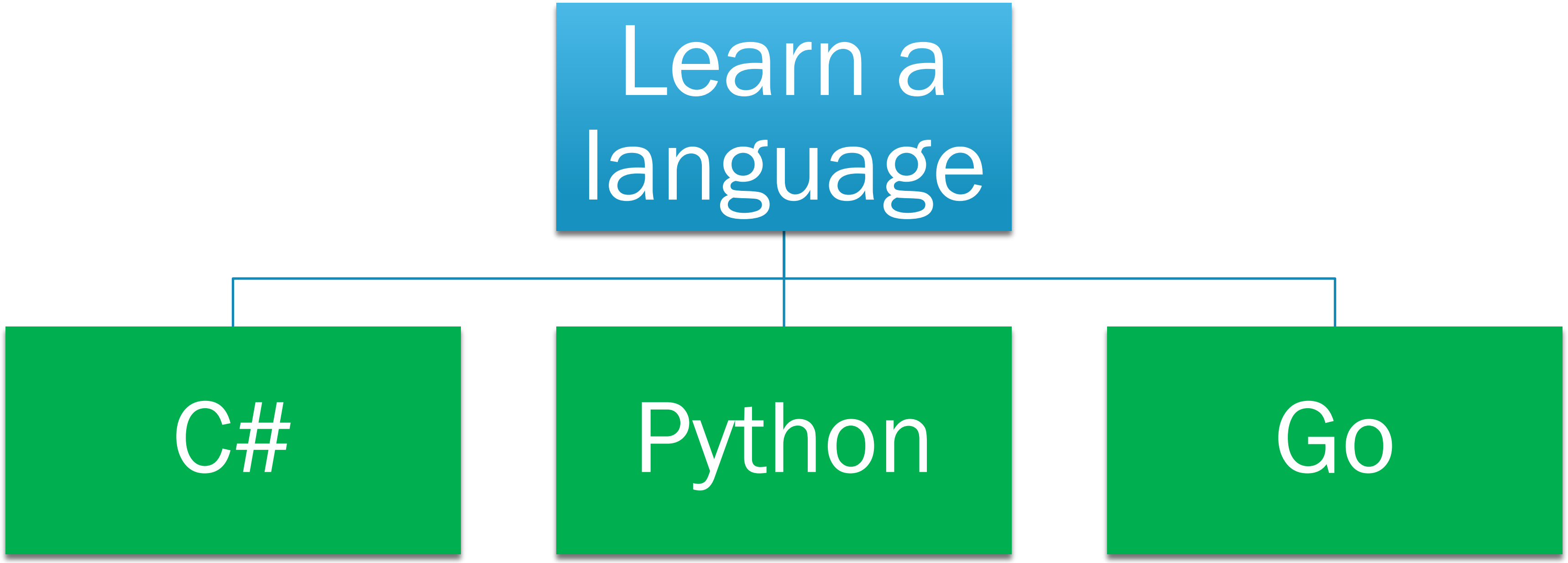
Logging

Monitoring And Alerting

Distributed Tracing

Click on each topic to see its details

LEARN A LANGUAGE



Learn a language

C#

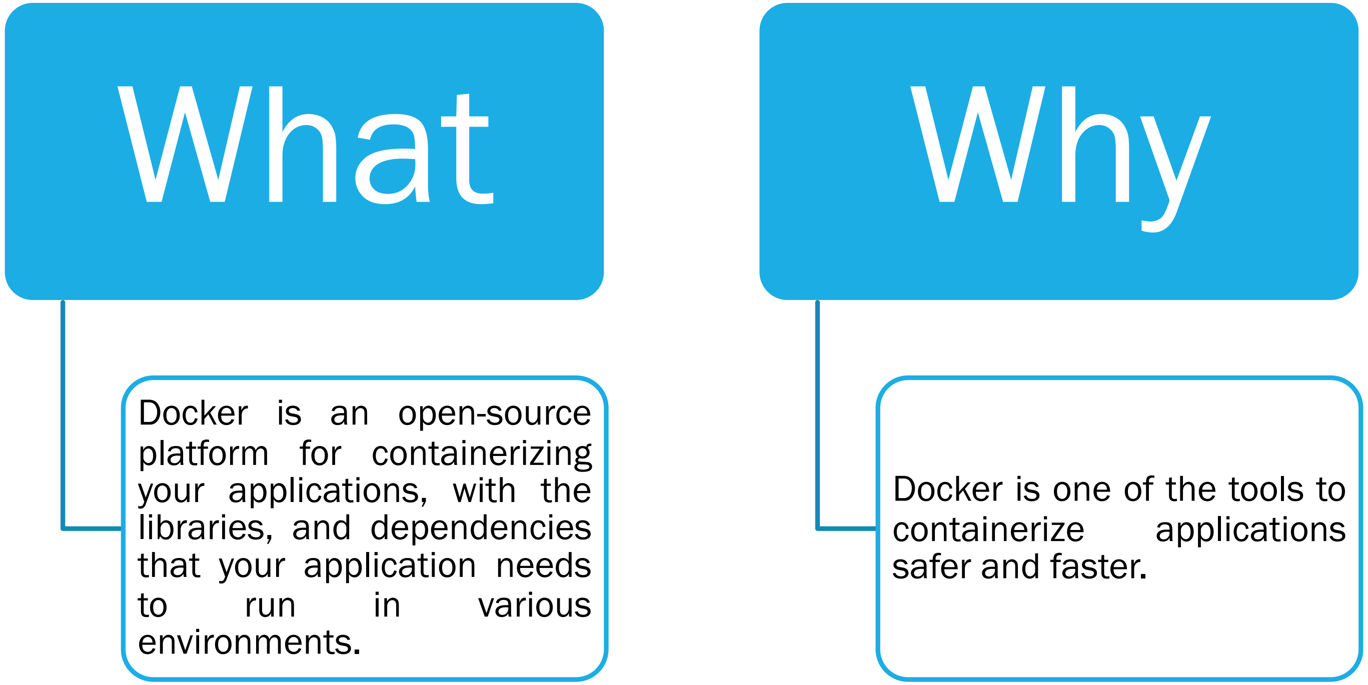
Python

Go

The green icons have its own roadmap
(ctrl-click to view)

Back

DOCKER



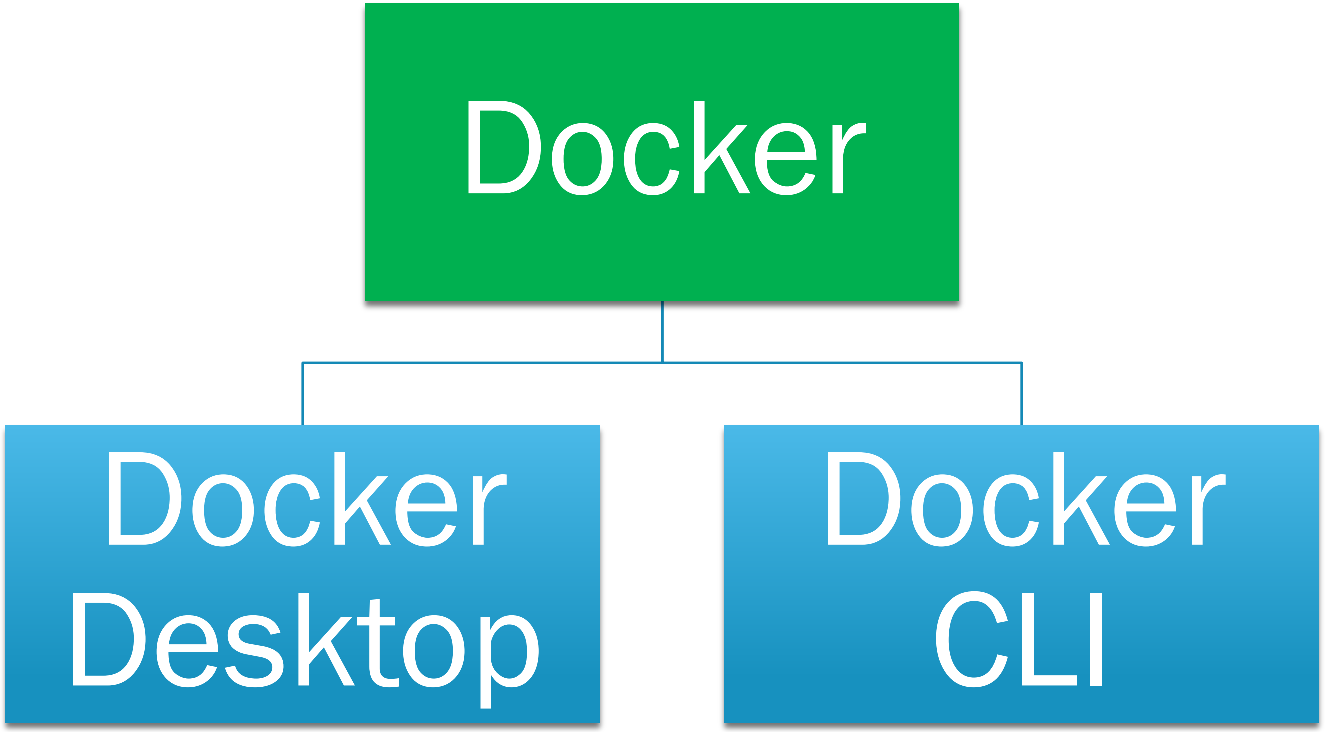
What

Docker is an open-source platform for containerizing your applications, with the libraries, and dependencies that your application needs to run in various environments.

Why

Docker is one of the tools to containerize applications safer and faster.

DOCKER



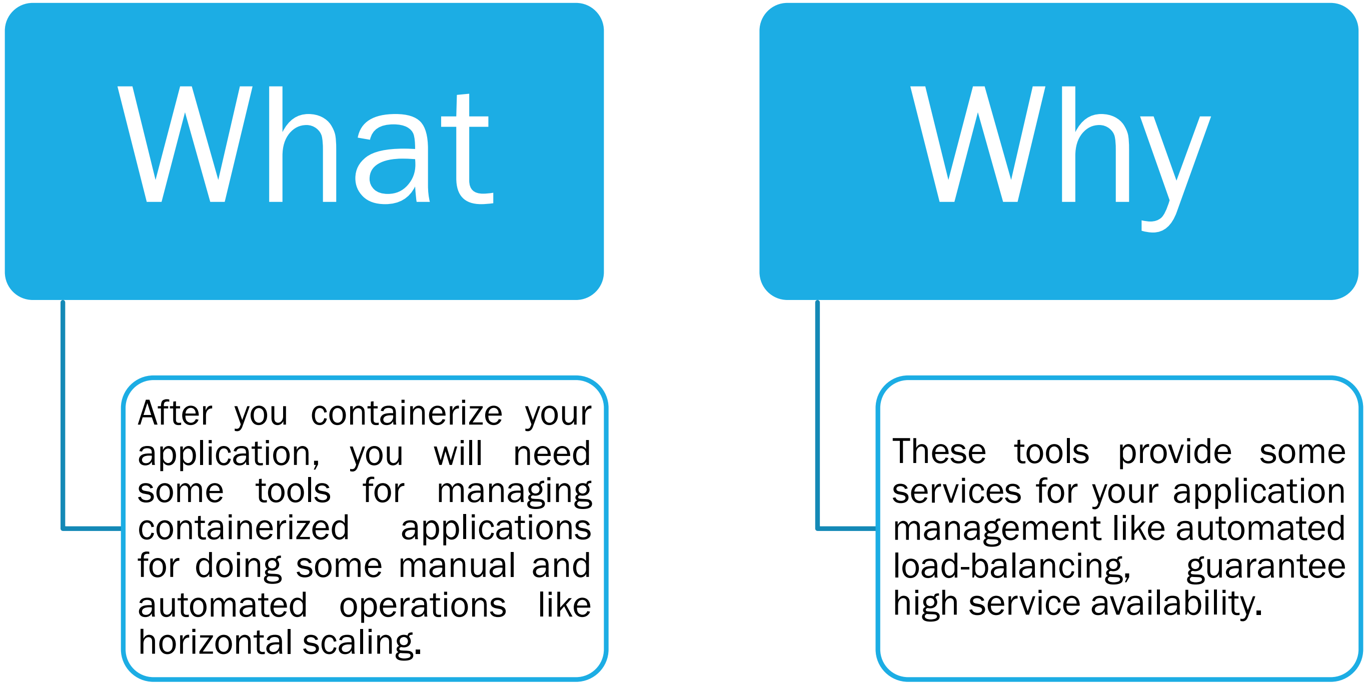
Docker

Docker Desktop

Docker CLI

Back

CONTAINER ORCHESTRATION



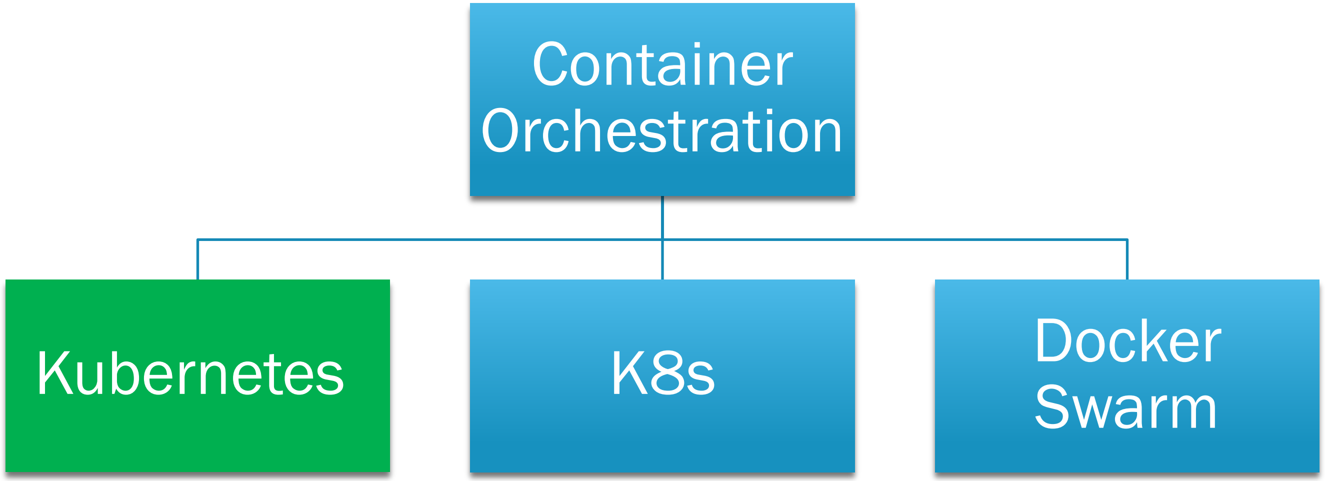
What

After you containerize your application, you will need some tools for managing containerized applications for doing some manual and automated operations like horizontal scaling.

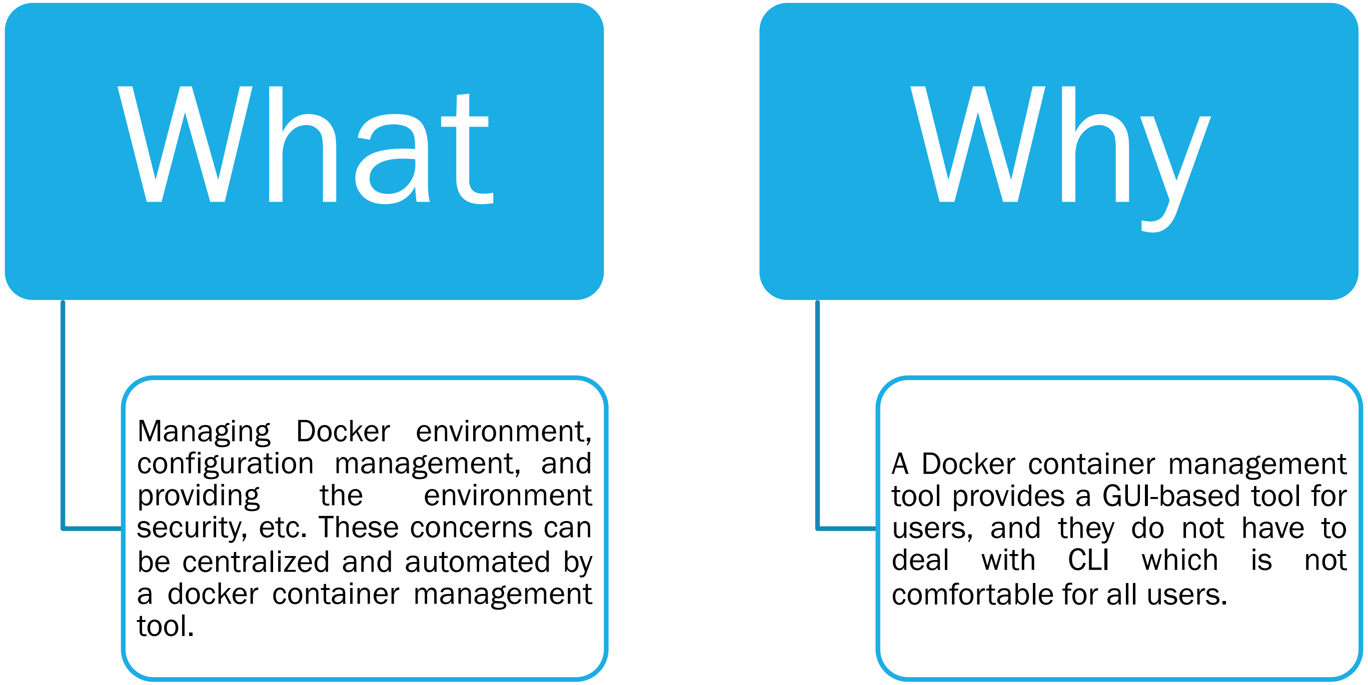
Why

These tools provide some services for your application management like automated load-balancing, guarantee high service availability.

CONTAINER ORCHESTRATION



DOCKER CONTAINER MANAGEMENT



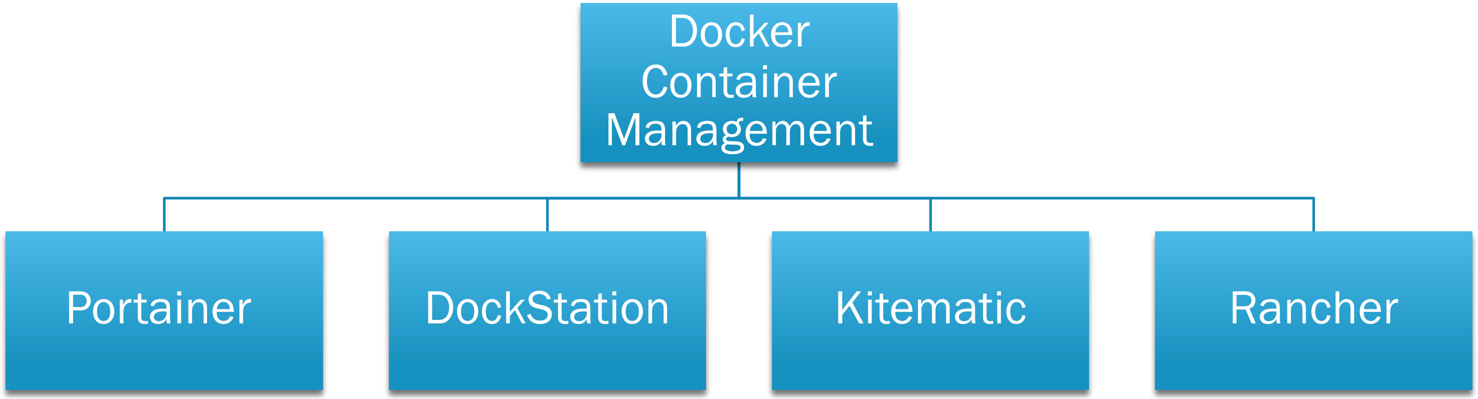
What

Managing Docker environment, configuration management, and providing the environment security, etc. These concerns can be centralized and automated by a docker container management tool.

Why

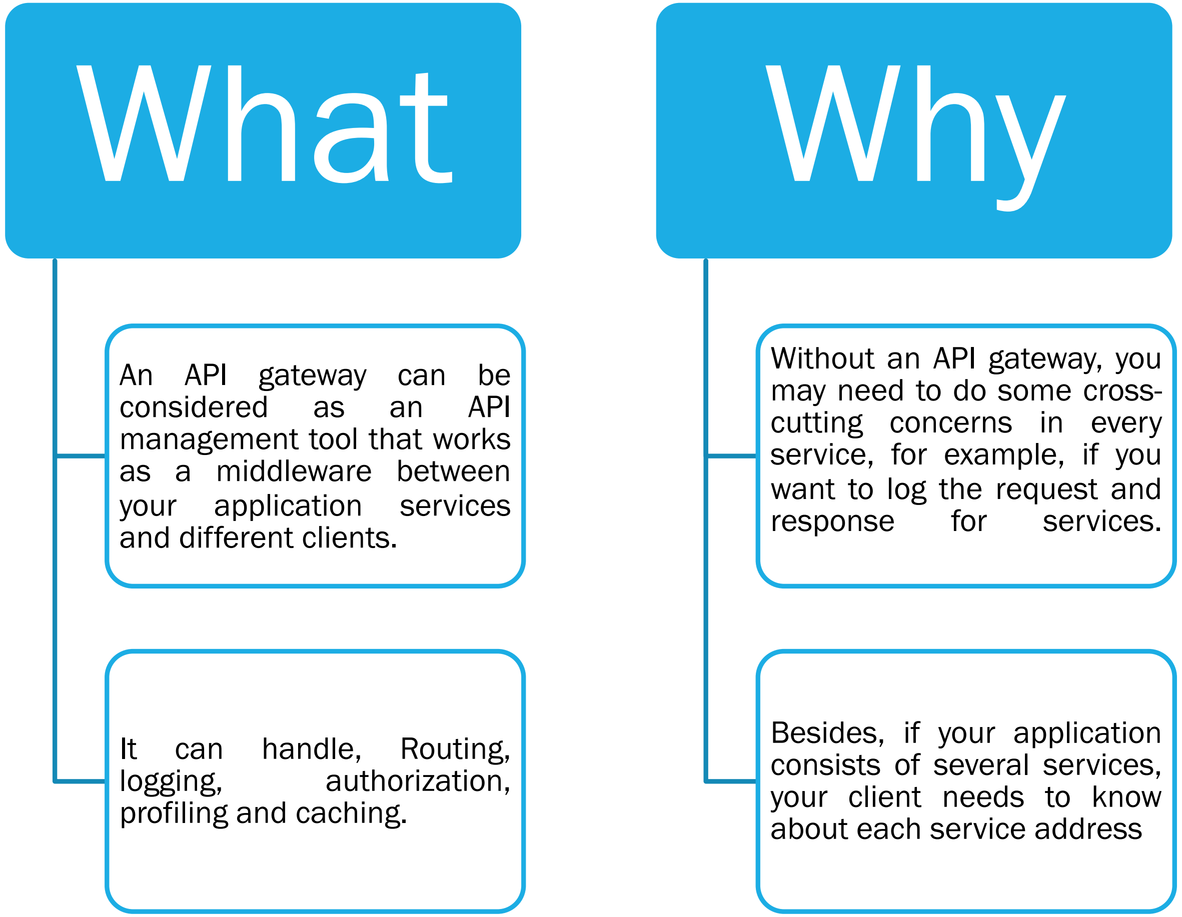
A Docker container management tool provides a GUI-based tool for users, and they do not have to deal with CLI which is not comfortable for all users.

DOCKER CONTAINER MANAGEMENT



Back

API GATEWAY



What

An API gateway can be considered as an API management tool that works as a middleware between your application services and different clients.

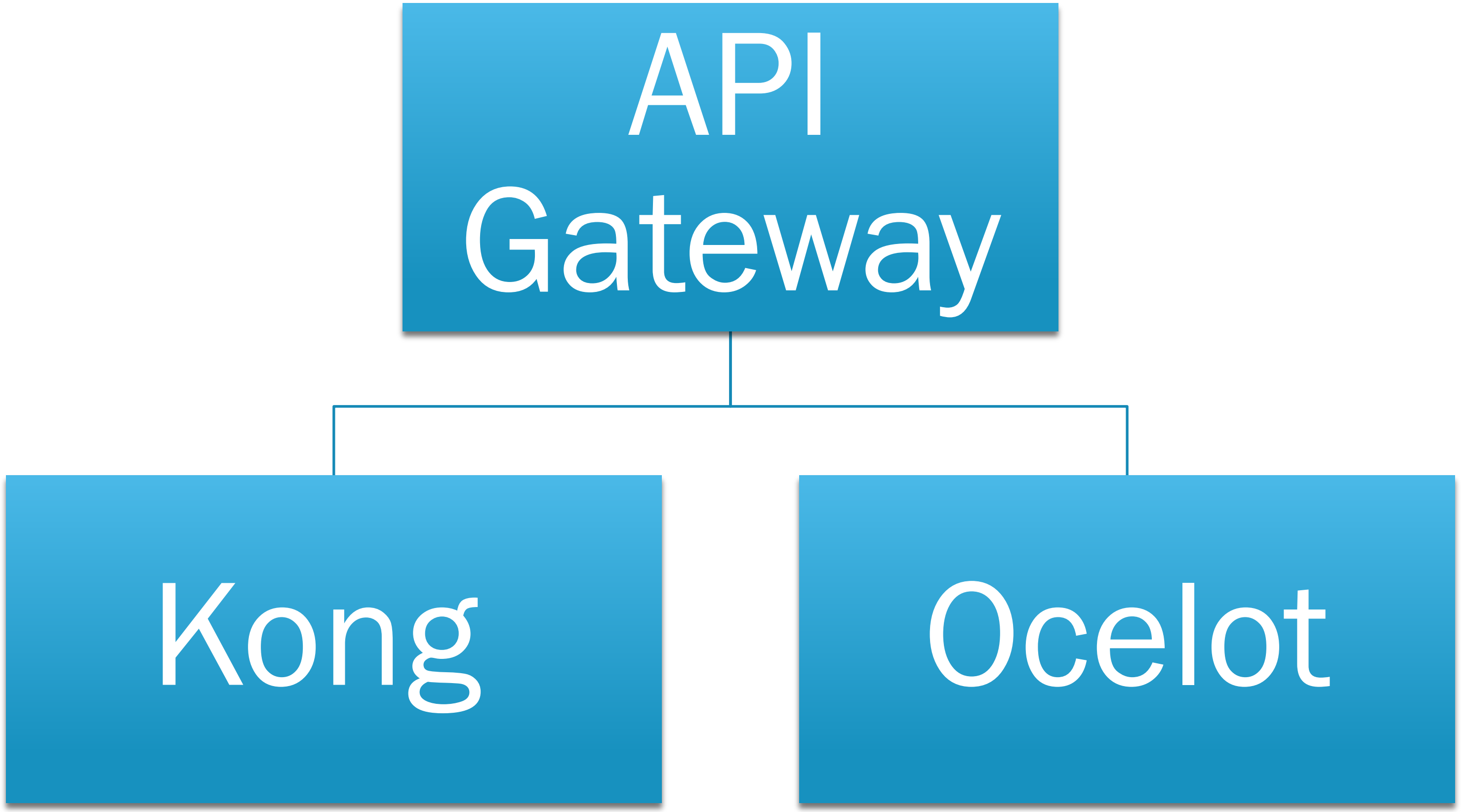
It can handle, Routing, logging, authorization, profiling and caching.

Why

Without an API gateway, you may need to do some cross-cutting concerns in every service, for example, if you want to log the request and response for services.

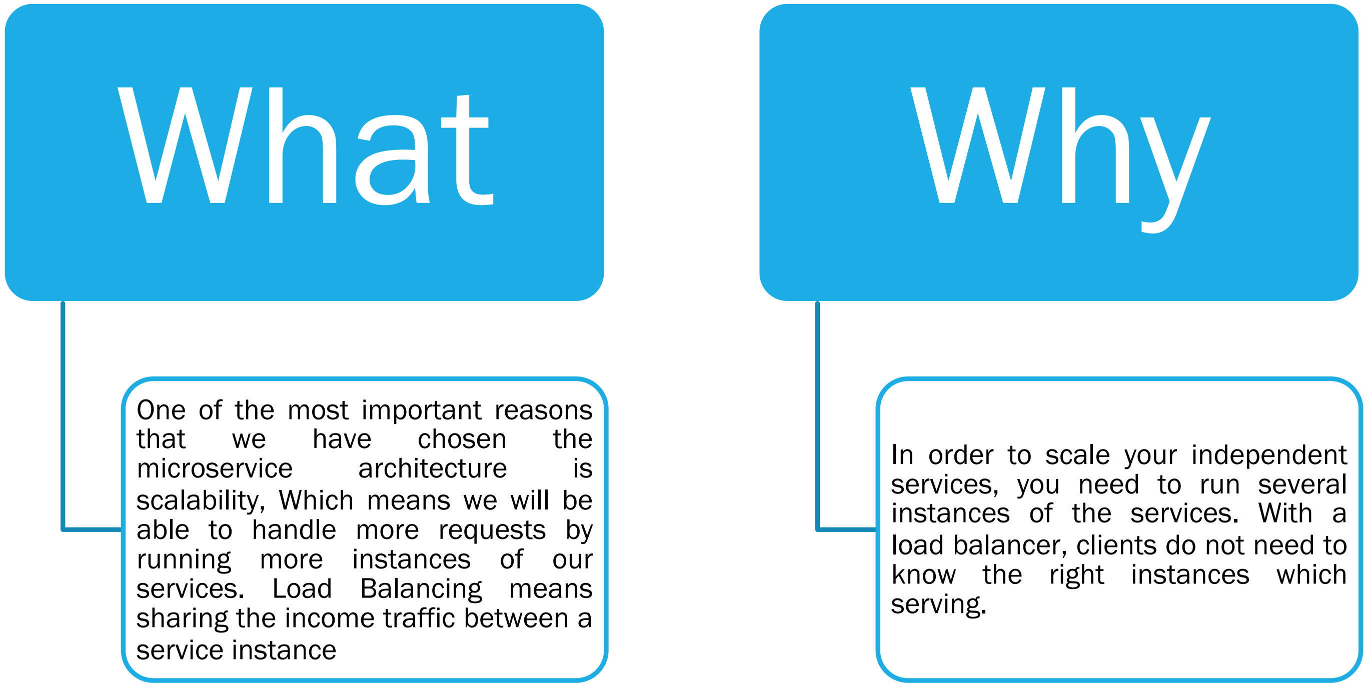
Besides, if your application consists of several services, your client needs to know about each service address

API GATEWAY



Back

LOAD BALANCING



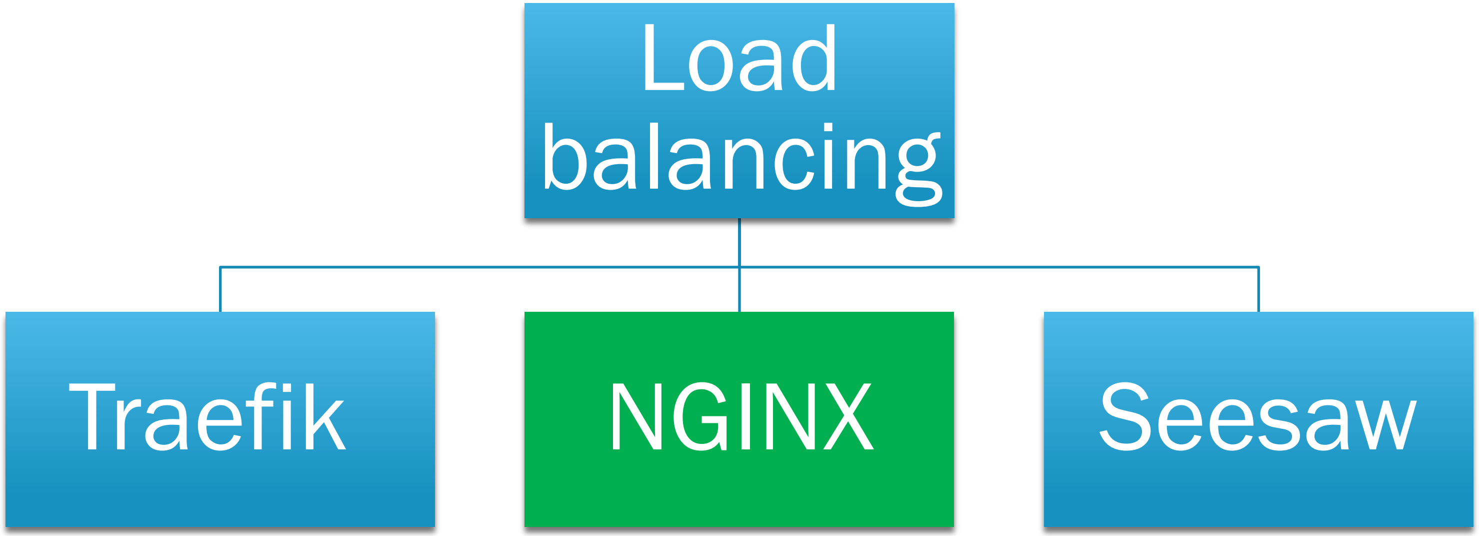
What

One of the most important reasons that we have chosen the microservice architecture is scalability, Which means we will be able to handle more requests by running more instances of our services. Load Balancing means sharing the income traffic between a service instance

Why

In order to scale your independent services, you need to run several instances of the services. With a load balancer, clients do not need to know the right instances which serving.

LOAD BALANCING



Load balancing

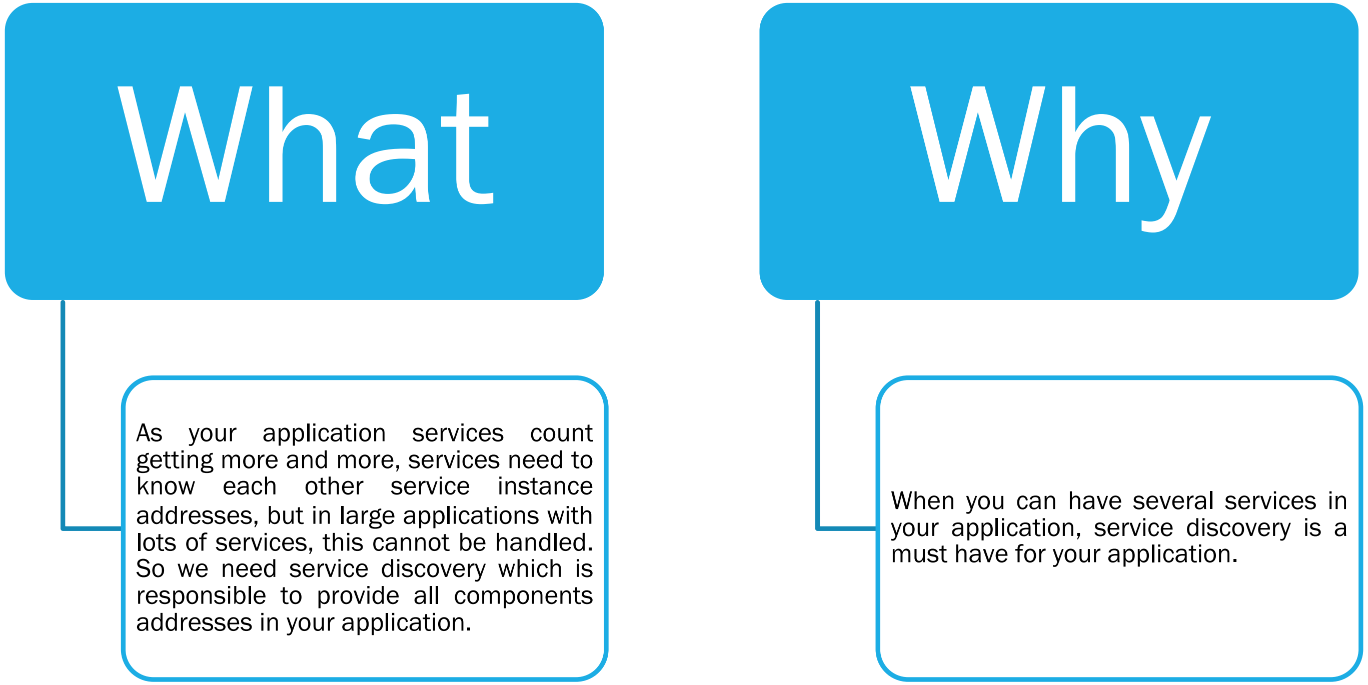
Traefik

NGINX

Seesaw

Back

SERVICE DISCOVERY



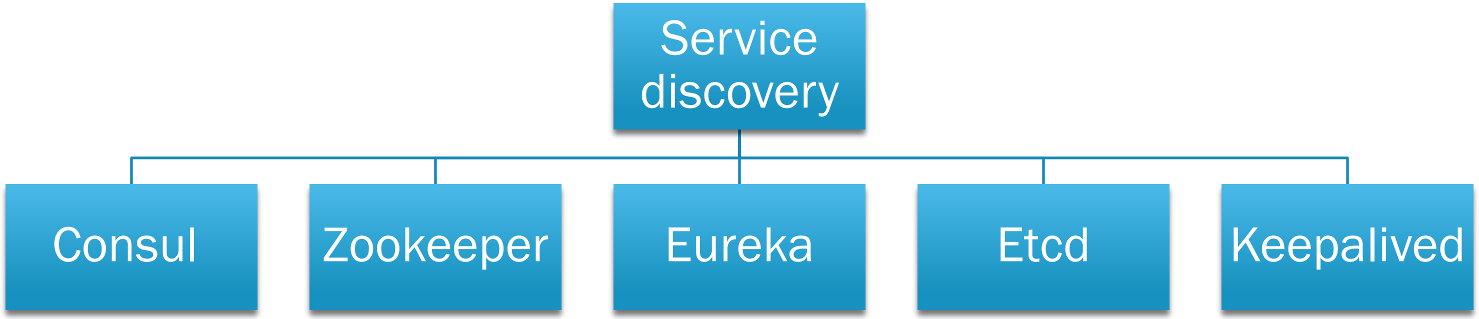
What

As your application services count getting more and more, services need to know each other service instance addresses, but in large applications with lots of services, this cannot be handled. So we need service discovery which is responsible to provide all components addresses in your application.

Why

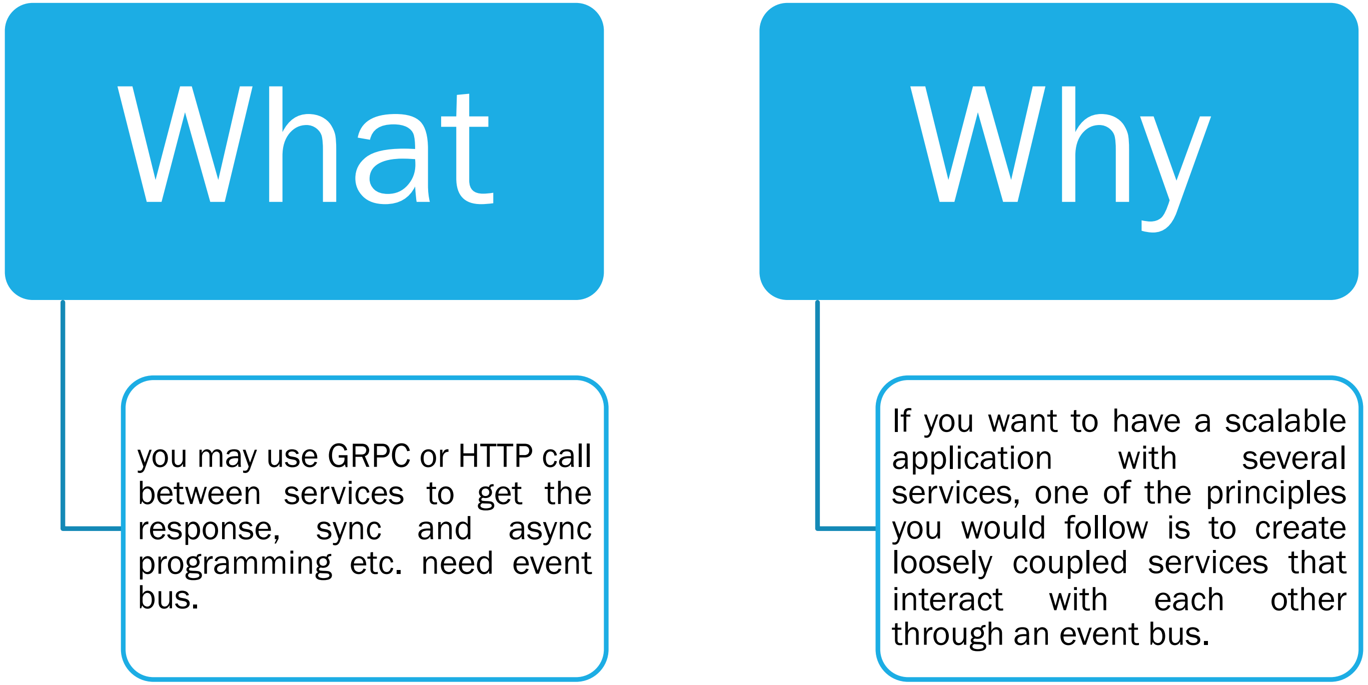
When you can have several services in your application, service discovery is a must have for your application.

SERVICE DISCOVERY



Back

EVET BUS



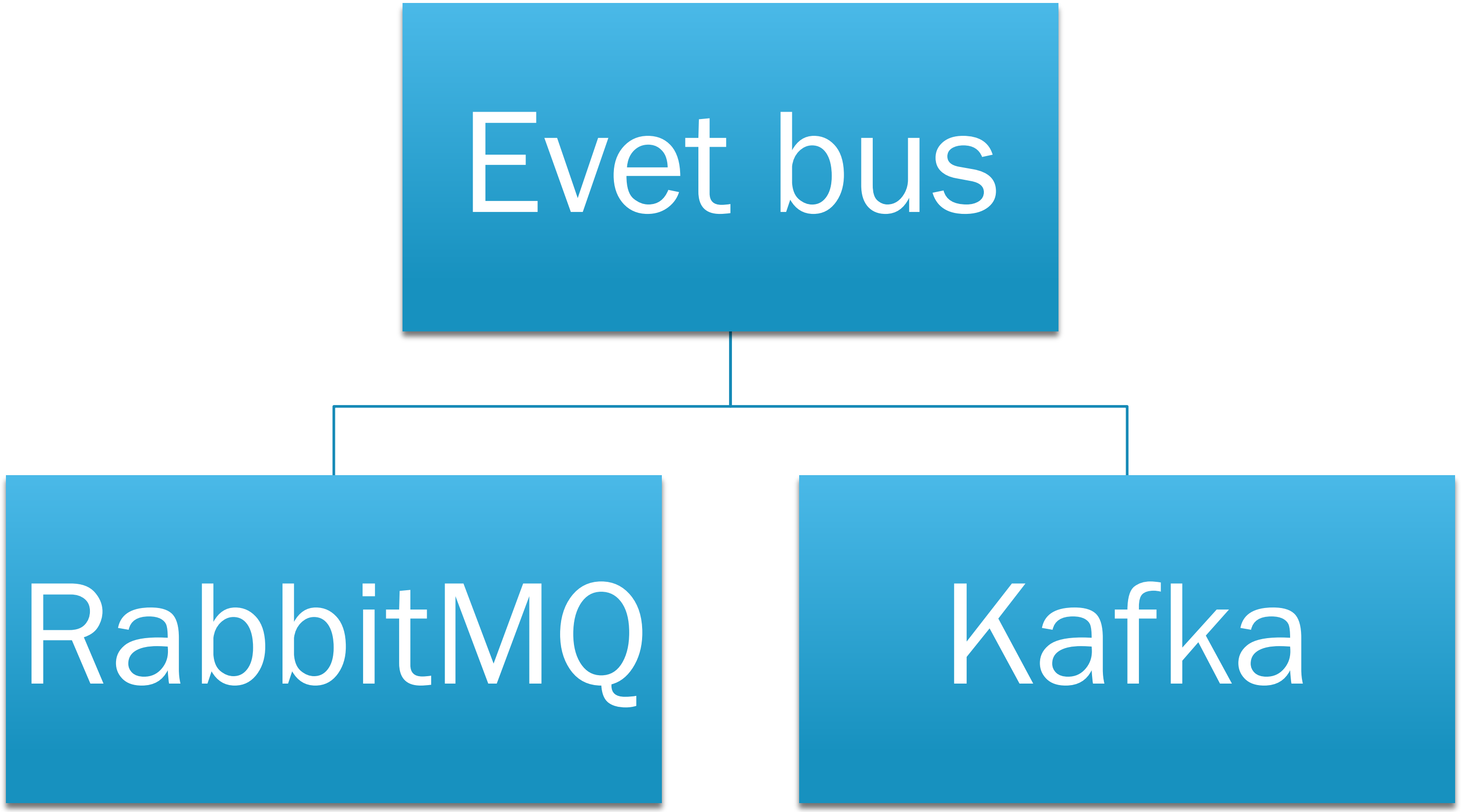
What

you may use GRPC or HTTP call between services to get the response, sync and async programming etc. need event bus.

Why

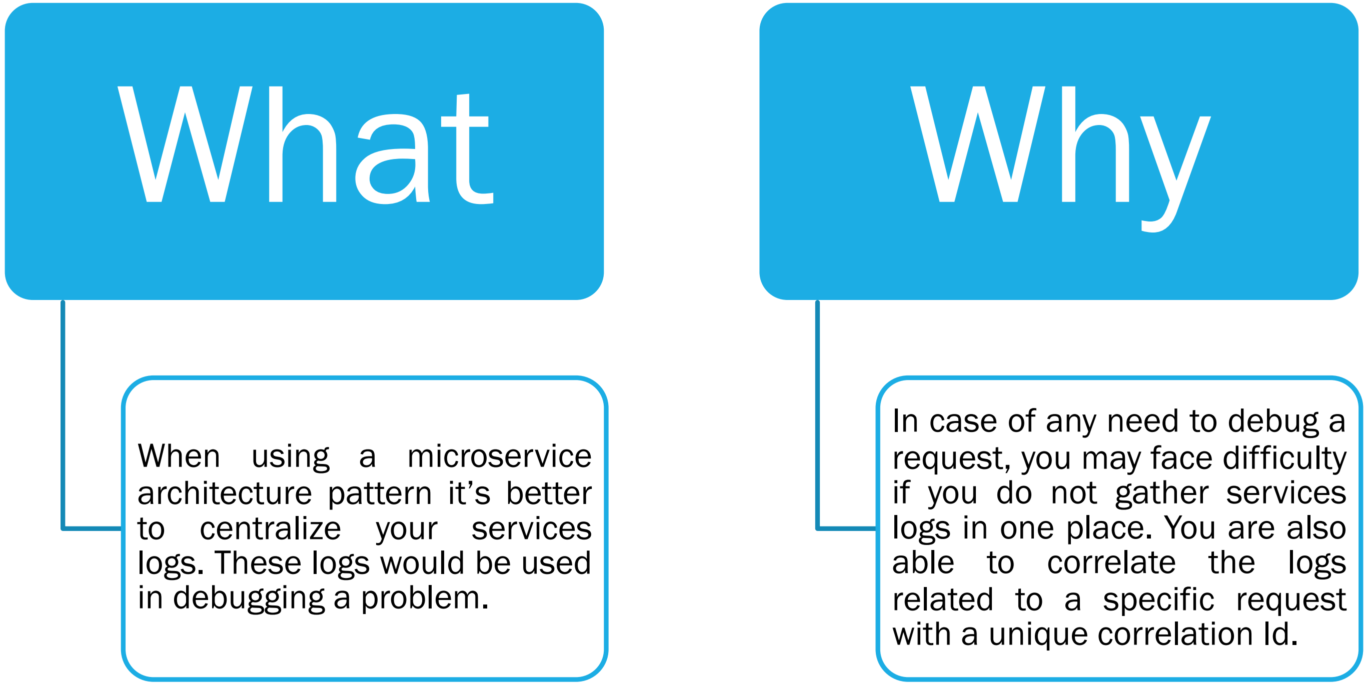
If you want to have a scalable application with several services, one of the principles you would follow is to create loosely coupled services that interact with each other through an event bus.

EVET BUS



Back

LOGGING



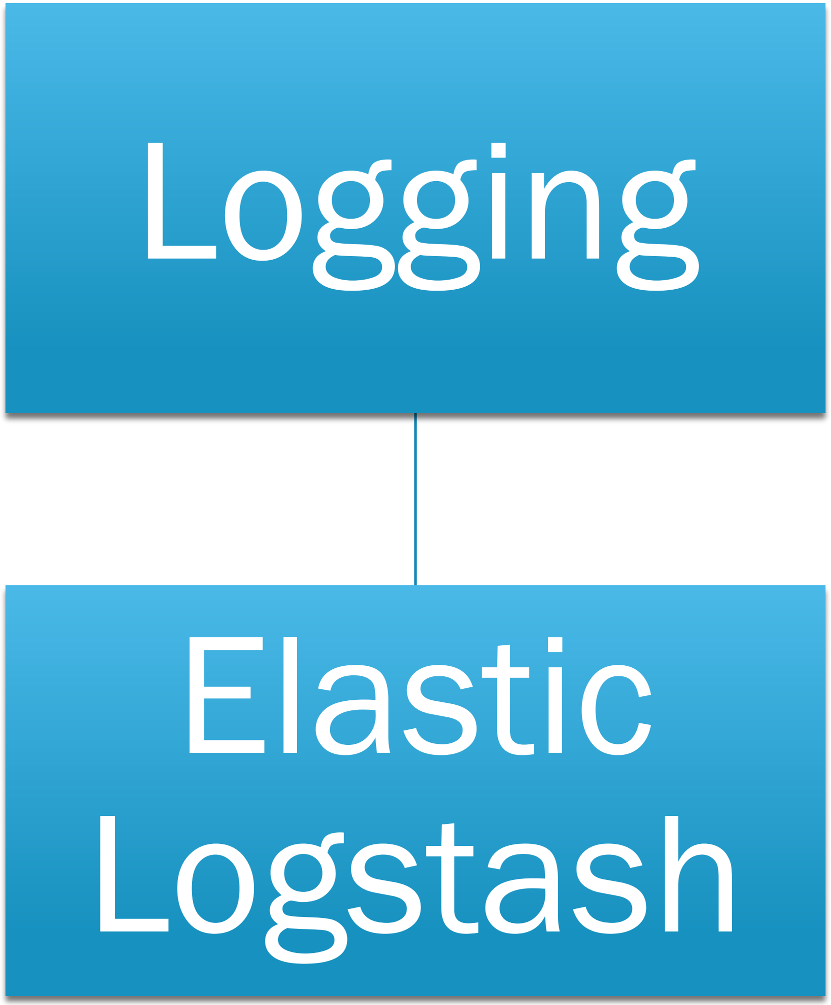
What

When using a microservice architecture pattern it’s better to centralize your services logs. These logs would be used in debugging a problem.

Why

In case of any need to debug a request, you may face difficulty if you do not gather services logs in one place. You are also able to correlate the logs related to a specific request with a unique correlation Id.

LOGGING

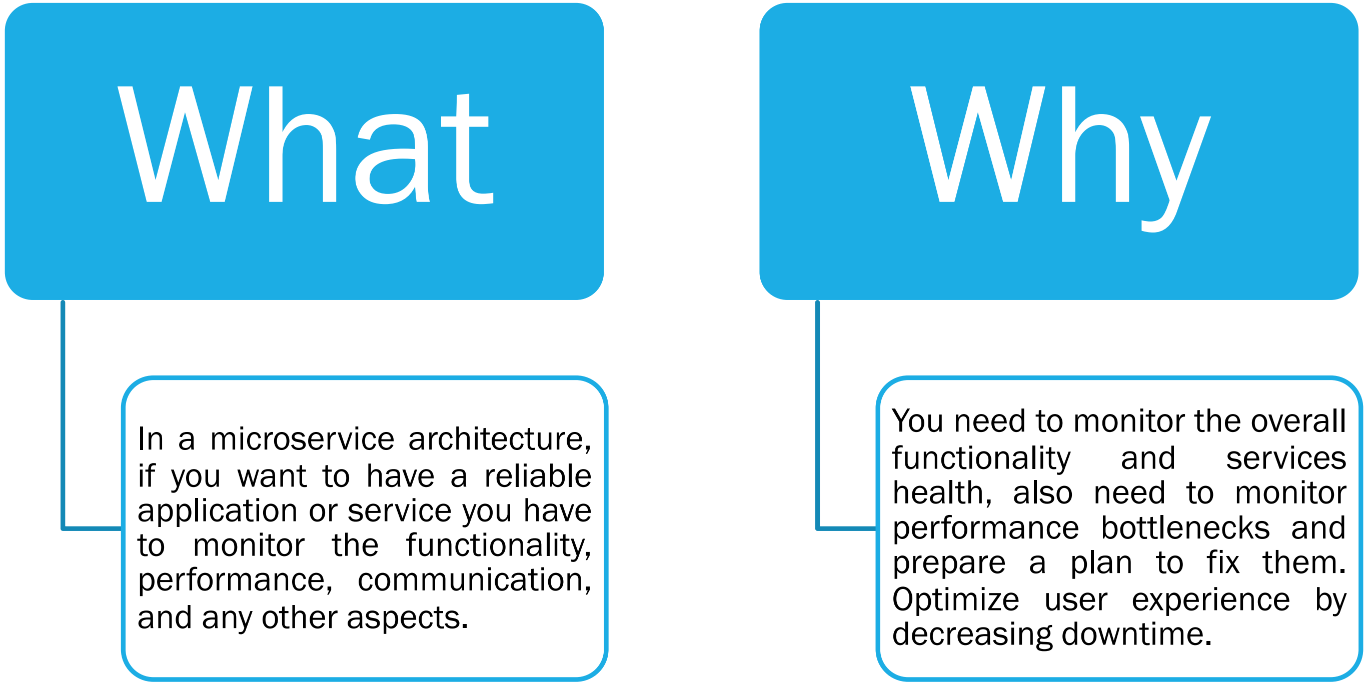


Logging

Elastic Logstash

Back

MONITORING AND ALERTING



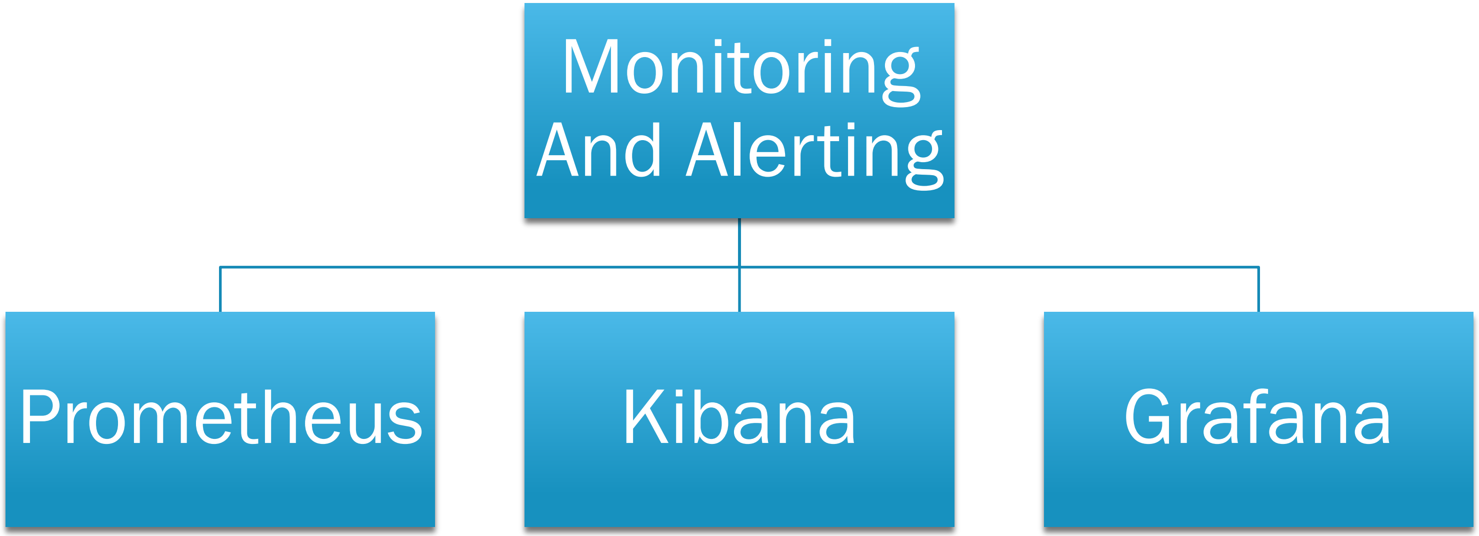
What

In a microservice architecture, if you want to have a reliable application or service you have to monitor the functionality, performance, communication, and any other aspects.

Why

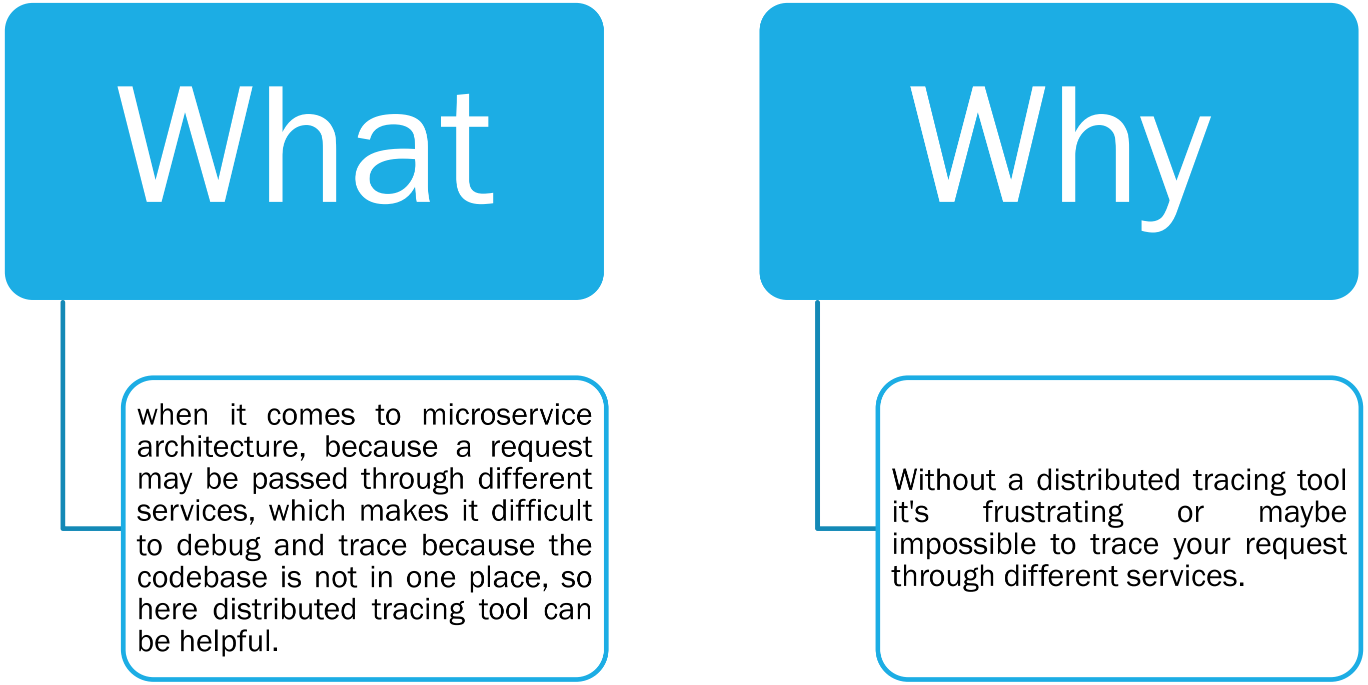
You need to monitor the overall functionality and services health, also need to monitor performance bottlenecks and prepare a plan to fix them. Optimize user experience by decreasing downtime.

MONITORING AND ALERTING



Back

DISTRIBUTED TRACING



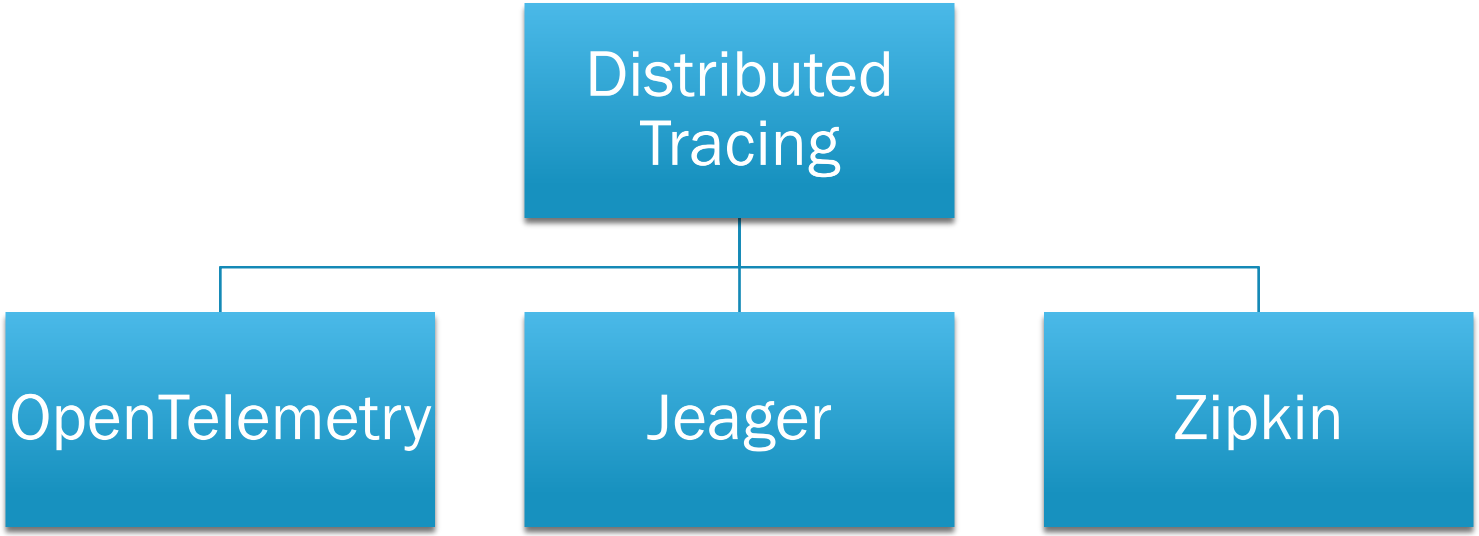
What

when it comes to microservice architecture, because a request may be passed through different services, which makes it difficult to debug and trace because the codebase is not in one place, so here distributed tracing tool can be helpful.

Why

Without a distributed tracing tool it's frustrating or maybe impossible to trace your request through different services.

DISTRIBUTED TRACING



Back

SOURCES

name	link
Medium - Matt Ghafouri	Ctrl-Click
Roadmap	Ctrl-Click