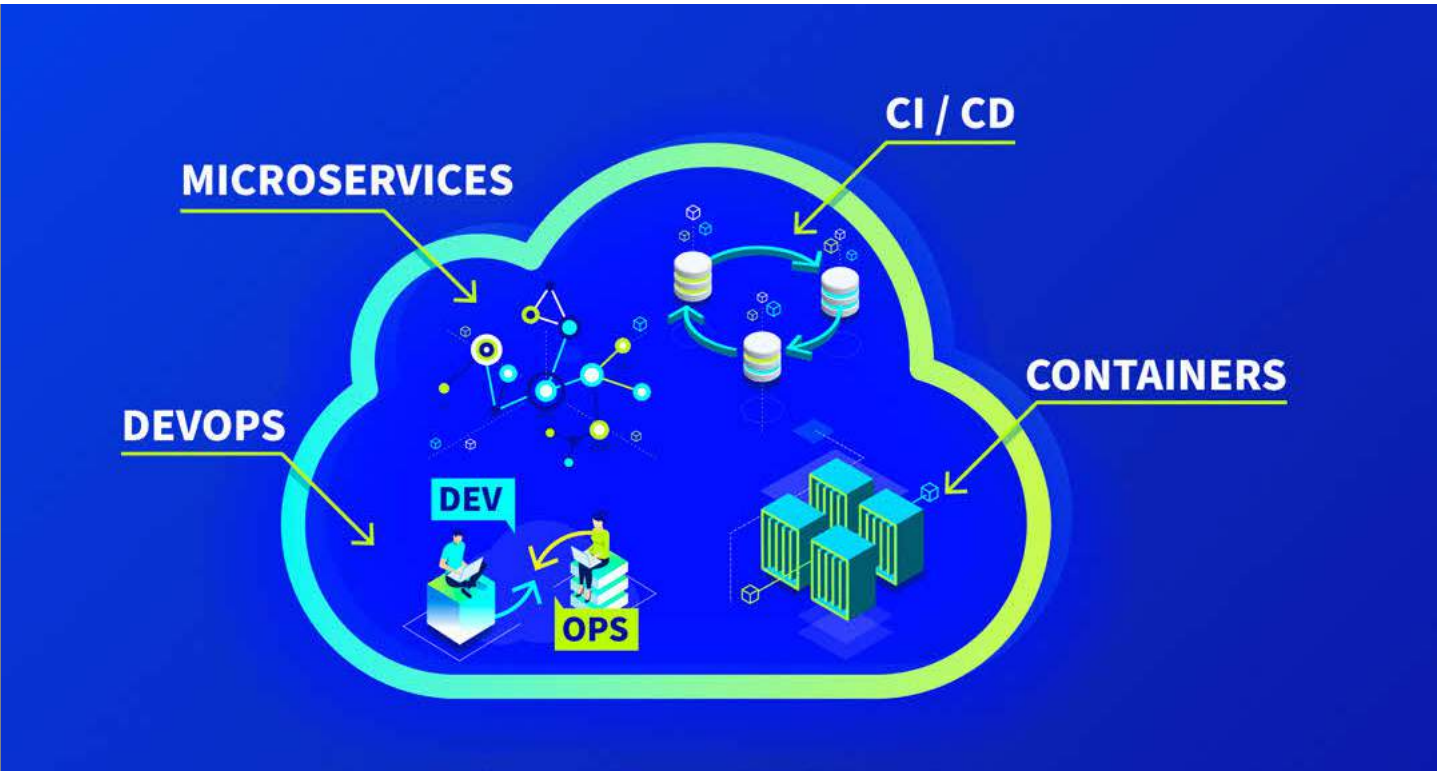
OVHcloud

Migrate towards a cloud-native software architecture



Everything you need to know to successfully migrate towards a cloud-native software architecture

From the historic “monolith” approach to microservices, to the service-oriented architecture (SOA) approach, software architecture has undergone several innovations since the late 1990s. Nowadays, a company will have different generations of architecture co-existing within its infrastructure, and the question of upgrading the oldest infrastructures to take advantage of the “cloud native” approach is regularly raised. This approach takes advantage of modern cloud environments and their managed services, where code and data are deployed without worrying about underlying hardware resources. The responsibility of service availability, load resistance[1], updating and securing the infrastructure’s lower layers is shifted to the cloud provider.

Is it still worth the effort? What benefits should you expect, and which methodology should you adopt to migrate to an OVHcloud native cloud infrastructure step-by-step?

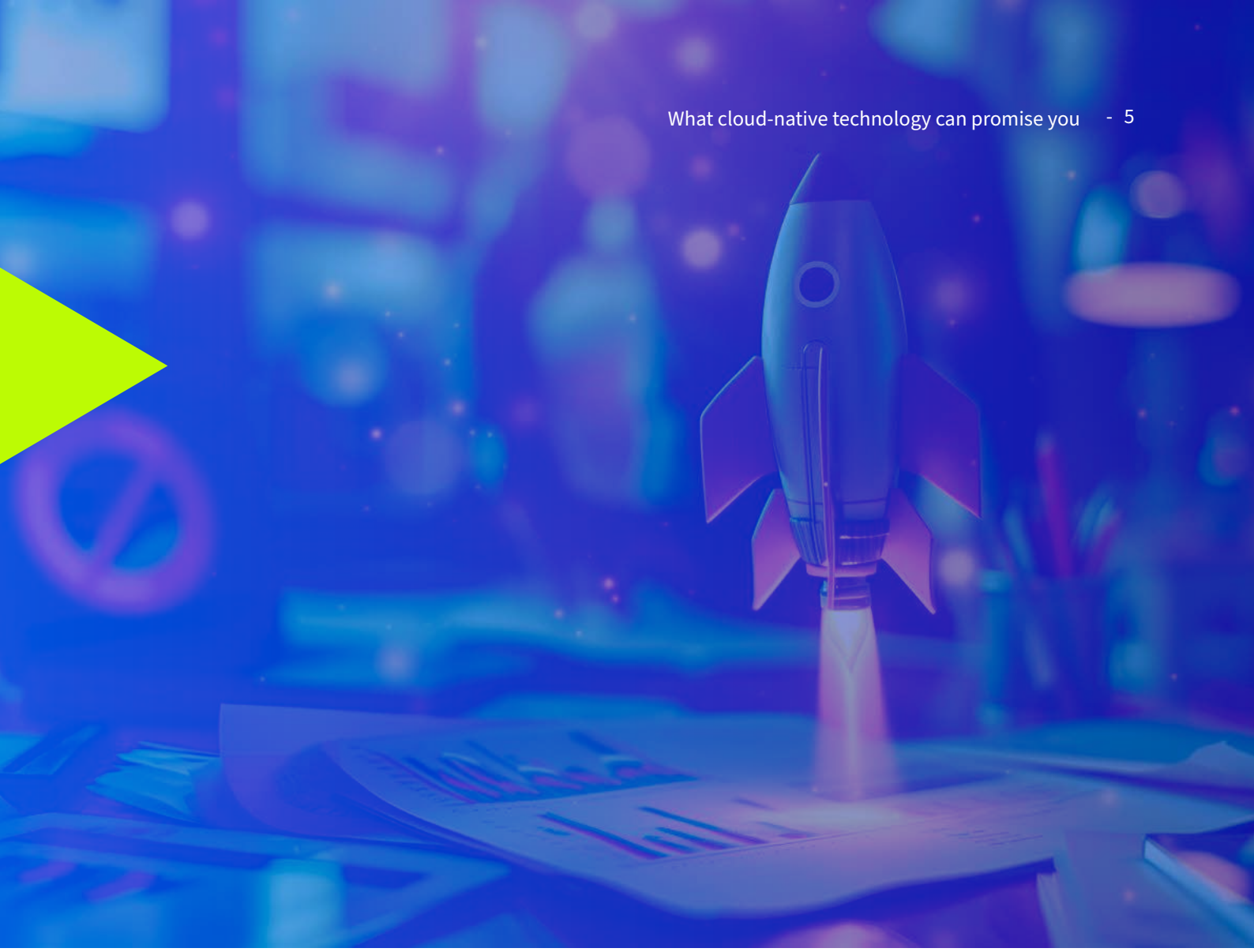
[v] Limited to the maximum number of resources you can allocate to your platform, set by you, and provided that the core of your application supports scaling (see Part 2 of the document).

Content

OVHcloud

PART 1

In which use cases is a cloud-native approach beneficial?



WHAT CLOUD-NATIVE TECHNOLOGY CAN PROMISE YOU

While the monolithic approach is still a good option for early development, particularly when it comes to creating an MVP (minimum viable product), it is mainly associated with older “legacy” applications.

Its limitations are well-known: close coupling between different parts of the system, difficulties scaling or evolving the code, and a strong reliance on a single technology stack.

By contrast, a cloud-native approach offers compelling advantages: scalability and high availability, pay-as-you-go billing, and being able to delegate the administration of the lower layers. It enables businesses to gain agility and competitiveness by accelerating both the deployment and maintenance of new features. With this approach, your development teams can work on several aspects of the application simultaneously, and utilise software building blocks that can be reused from one project to another – all while accessing on-demand resources, billed solely according to usage, with no long-term contract.

These resources include a diverse range of managed services, such as databases, compute instances, object-mode storage, as well as pre-trained AI functions (OCR, voice recognition, computer vision, etc.), ready to be integrated into your applications.

In short, a cloud-native approach helps create applications that are more scalable, efficient, resilient, and easy to manage.

. It also simplifies the portability of applications from one cloud provider to another, perfectly aligning with OVHcloud's commitments on reversibility[2].

[2] Thanks to the conformance programme established by the Cloud Native Computing Foundation (CNCF), a number of cloud licensors and providers guarantee total data reversibility. Kubernetes has in fact become the standard for multi-cloud (different cloud providers or datacentres) and hybrid (on-premises and cloud distribution) scenarios. The same configuration can be instantly transferred from one provider to another.

THE CHALLENGES OF MOVING TO A CLOUD-NATIVE APPROACH

This evolution of software architecture is radically changing the way applications are developed, which now involves skillfully assembling and interfacing technological building blocks. However, migrating from a monolithic architecture doesn't just mean transposing code; it often requires re-adapting or even re-thinking entire parts of the application to make it scalable and effective.

It is therefore crucial to ask the right questions before considering migrating. For example, what is the durability of the application? With an average software lifespan of 6-10 years (2-3 years for a mobile application), it's essential to evaluate how critical it is to your business, as well as how variable your workload is.

In other words, migrating an older software, such as one that generates payslips every month, would not bring a benefit unless the number of users changes dramatically, or unless the calculations are complex.

However, if your application has usage uncertainties and workload variations, the cloud-native approach could be very beneficial to you. Still, we recommend taking a phased approach, starting with the features where cloud-native computing will bring the greatest benefits.

We'll come back to this point later.

Another scenario that justifies migration to a microservices architecture is the need for a software publisher to deploy a flexible, scalable architecture that is adapted to each customer's specific needs. This flexibility allows you to deliver the same SaaS platform for both an SME and a multinational company, without any particular complexity. It also helps you to adopt a more agile business model, with a transition from traditional licensing to per-user pricing. This makes it possible to adjust the resources allocated to the platform in a granular manner according to your peak loads, offering optimal scalability.

Migrating to a microservices architecture is often the first step towards adopting a cloud-native approach.

There are significant technical and financial challenges, but there are also human resource implications. You will need to adapt your teams' skills to meet the new requirements, by accelerating the transition to DevOps practices and site reliability engineering (SRE).

This transition will also require a redistribution of roles and responsibilities within your teams, with a refocusing of efforts from system administration (outsourced to OVHcloud, for example) to operational management and application support.

CLOUD-NATIVE APPROACH: WHAT YOU NEED TO KNOW ABOUT MICROSERVICES ARCHITECTURE AND CONTAINERISATION

The cloud-native approach, which leverages managed cloud services to develop applications, is based on a modular architecture, broken down into independent microservices. Each microservice interacts with others to perform complex functions, while allowing updates, changes, or upgrades to be made without affecting the other components.

These microservices are often combined with containerisation: a technology that provides them with an isolated execution environment. Each container groups the microservice code with its dependencies (necessary files and libraries), making it portable and adaptable to different environments. Unlike standard virtualisation, containers don't include the full operating environment, making them lightweight and quick to deploy. They facilitate horizontal (container multiplication) and vertical (increase in resources such as CPU and RAM) scalability, thus contributing to the independence from hardware that virtualisation has brought about.

The trend in IT is to break free of the constraints of provisioning, deploying and administering physical resources. As the technological research firm Gartner explains:

> **Platform engineering enhances developer productivity by providing self-service capabilities with automated infrastructure operations. Platform engineering is trending because of its promise to optimise the developer experience and accelerate product teams' delivery of customer value.**

This trend is combined with Infrastructure-as-Code (IaC), which allows infrastructure needs to be described in code, whether they are microservices or containers. The cloud provider (such as OVHcloud) takes over the management of the physical and software layers, guaranteeing performance and availability.

In short, a cloud-native approach can deliver previously unattainable service levels, while freeing your teams from infrastructure tasks, so they can focus on their main job: software development. By pooling resources and industrialising facilities management, the cloud provider will enable you to benefit from an optimised infrastructure without management constraints.

Find out more about the microservices architecture
https://www.ovhcloud.com/asia/learn/what-is-microservices-architecture/

IS A CLOUD-NATIVE APPROACH NECESSARILY COST-EFFECTIVE?

Balancing the price/performance ratio is often complex, but the basic principle is simple. Hardware generates fixed costs, which are largely separate from usage. The challenge is to optimise resource allocation, so that we can use the machines as close as possible to their maximum capacity. Virtualisation was born with this in mind, allowing resources to be shared at different scales.

Containerisation follows this logic, further reducing the size of the base unit, namely the container inside the virtual machine (VM). This is done in a way that maximises the occupancy rate of a physical machine. The smaller the blocks, the better the fill rate – especially since a container can use 100% of the CPU/RAM resources allocated to it. The idea is therefore to avoid underusing machines and/or overprovisioning infrastructure by sharing the physical resources that a provider like OVHcloud uses on an industrial scale.

However, your savings will be limited if your software is already running on a server that is perfectly suited to its workload, without significant variations in demand. In this case, the infrastructure is already optimised, so the potential gains will be minimal.

Furthermore, virtualisation itself consumes around 15% of CPU/RAM resources, which can further reduce the cost savings.

On the other hand, in a high availability (HA) infrastructure running on a cluster of machines with a utilisation rate of less than 60%, the cost savings can be substantial. You will benefit from better performance and a higher level of service in this scenario, notably thanks to reduced hardware management costs and delegating the responsibility of service availability to a provider.

One final point to keep in mind:

with the cloud-native approach, you would switch from stable, predictable billing to a variable billing model. This means your monthly costs may fluctuate up or down depending on the workload your app receives. It is therefore essential to set upper and lower limits in order to anticipate these variations.

To manage this variability, we recommend adopting the FinOps practice. This approach involves implementing tools and indicators that offer a view into your cloud resource usage. This not only allows you to regain a certain predictability in your spending, but also to quickly detect any anomalies linked to an improper configuration of your services.

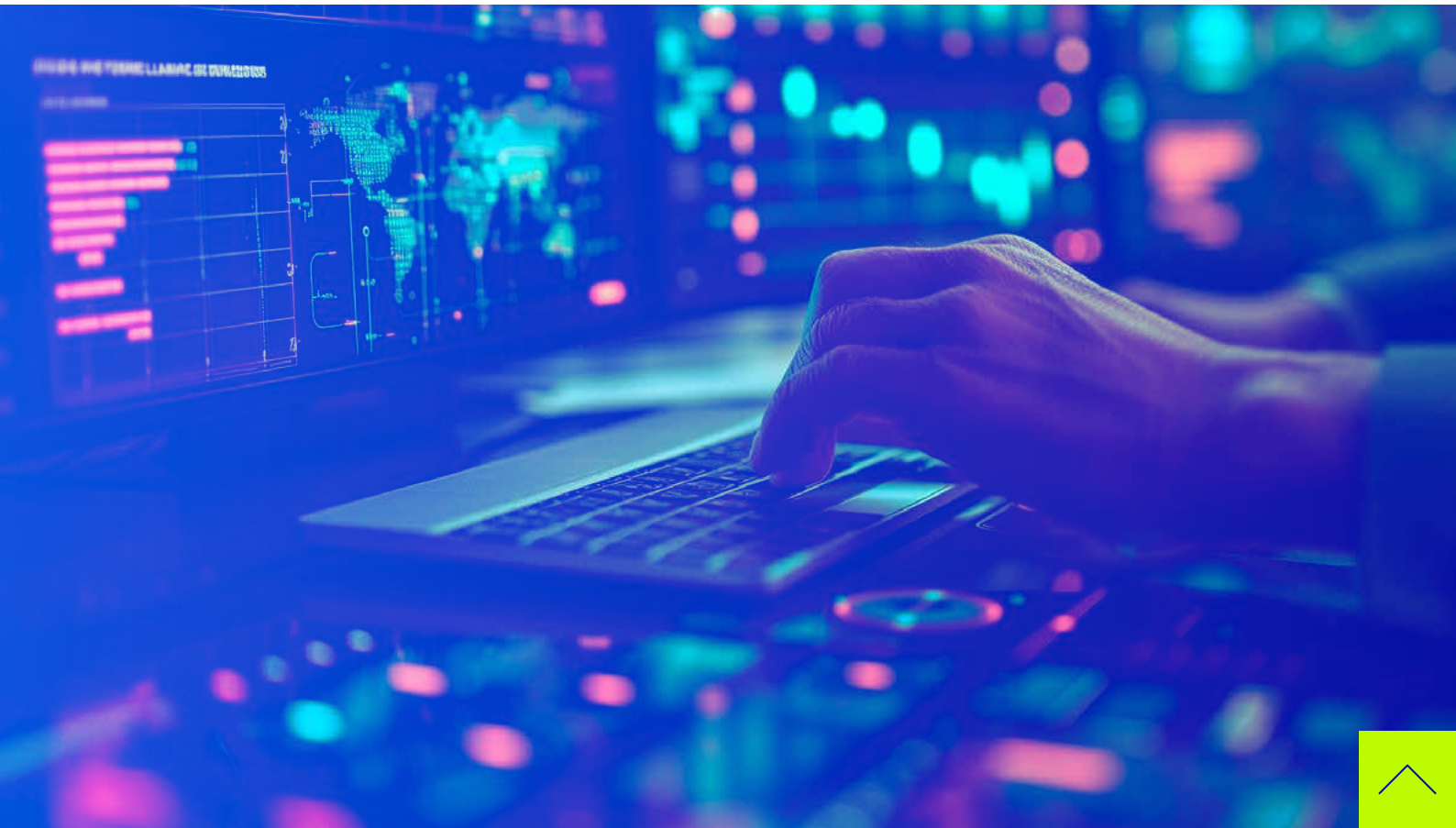
OVHcloud

PART 2

Adopting the cloud-native approach

In principle, there are no applications that are incompatible with the cloud-native approach;

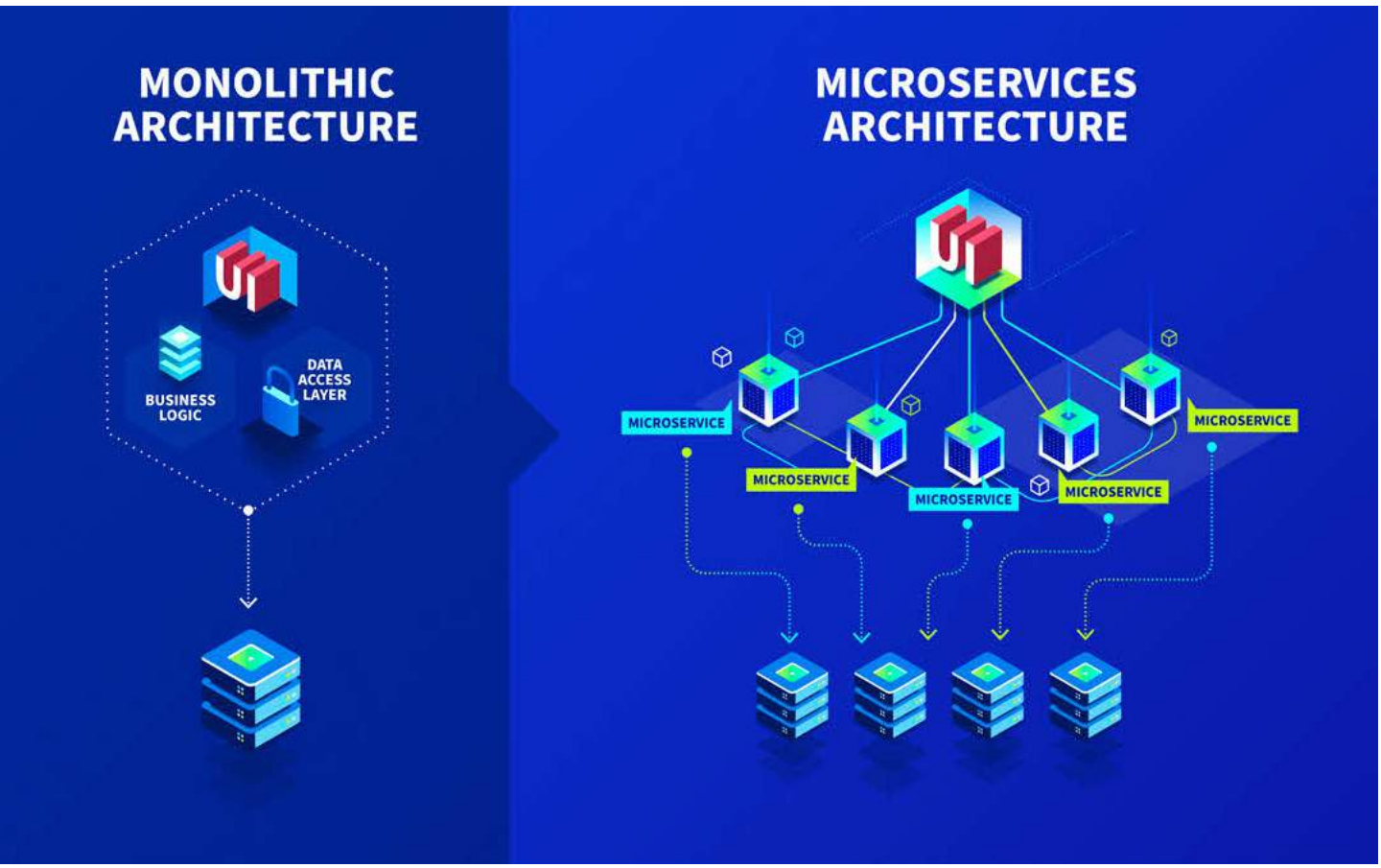
the question is more about the cost-effectiveness of the transition. Modernising your application, sometimes referred to as “platforming” when adopting a microservices architecture, often involves significant work, and requires your teams to develop their skills. Here are the two main factors to consider when moving to a cloud-native environment.



DECOUPLING YOUR APPLICATION COMPONENTS

Migrating to a cloud-native architecture involves identifying the different software building blocks that make up your application, and isolating each component in a separate container. Each container therefore hosts a microservice, running autonomously and independently. This decoupling allows the resources allocated to each service to be precisely adjusted, avoiding the typical bottlenecks of monolithic architectures.

Furthermore, these small, self-contained software building blocks are interoperable, increasing agility in application development. They can be easily reused in other projects or shared, improving the efficiency and flexibility of your future developments.



MOVING TO A STATELESS ARCHITECTURE

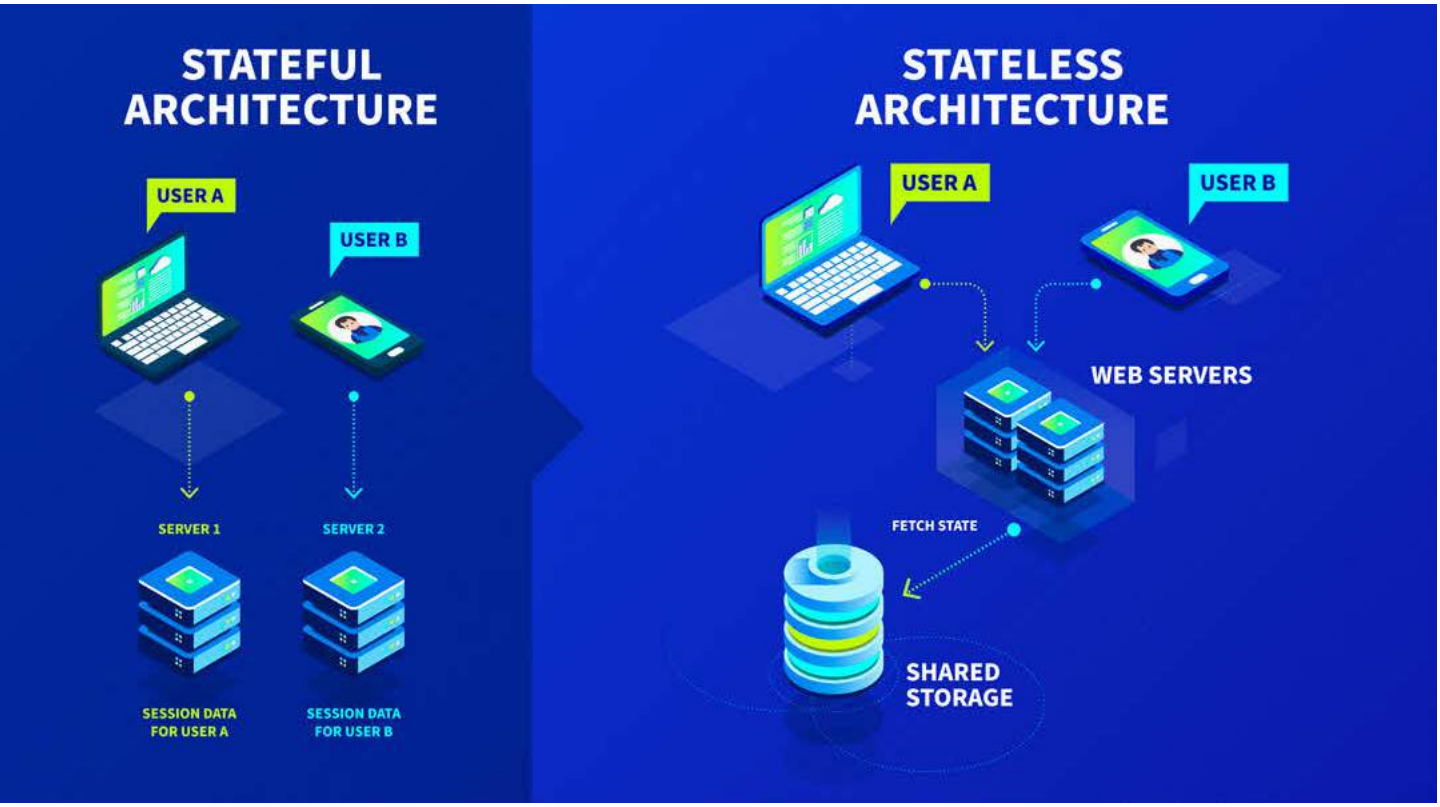
Once your application has been divided into independent components, you just need to make it stateless. Although this decoupling allows each component to operate independently in a container, it does not bring any significant added value compared to a virtual machine (VM) if the application remains stateful (state-dependent).

This is exactly what Kubernetes, the most popular container management technology on the market, does to help you scale your architecture in a simple way: either vertically, dynamically allocating more resources to a container, or horizontally, by multiplying container instances.

This also guarantees high availability (if a node becomes inaccessible, the container it was hosting is recreated instantly elsewhere), and it can be used to set up a rolling update, which is used to carry out continuous updates without any service interruptions (containers update in succession)[3].

To support this “stop & play” model and allow a peak load from 1 to 2 containers or more, several conditions must be met. First, the independent components of your architecture must be able to communicate with each other, whether synchronously, asynchronously, or via event broadcasting. These three modes of communication must be compatible in order to avoid a domino effect in the event of a service failure. In other words, a microservice must be able to run its processing operations without depending on the availability of another microservice. Secondly, to increase the application’s scalability, your processes need to be parallelised, so that several processes can be run simultaneously. Finally, container volatility involves minimising the use of local storage in the container’s memory, by favouring external and centralised storage solutions.

[3] This mechanism also allows an easy rollback in the event of a version upgrade of a microservice that ends up causing incompatibility or side effects in other microservices in the architecture.



As you'll see, this upgrade involves a lot of API work for your microservices. Beyond a certain level of complexity, the use of APIs becomes essential to facilitating the development of your application and ensuring communication between your microservices, as well as their interaction with third-party services.

The APIs expose the features of your microservices and must be properly documented and updated regularly to guarantee their smooth operation and durability.

OVHcloud

PART 3

Managed services from the OVHcloud Public Cloud ecosystem that facilitate the implementation of a cloud-native architecture

OUTSOURCE DATABASES AND STORAGE

As mentioned above, refactoring your application's code as part of the migration to a cloud-native architecture is a substantial process. However, there is one aspect of your infrastructure that can quickly be simplified: database administration. Database clustering, often one of the most complex tasks in IT, can now be easily outsourced with Database as a Service (DBaaS) solutions, offering fully managed administration.

The migration can be done in minutes by importing your database into this new environment, performing tests, and decommissioning the old database once everything is up and running.

In order of increasing complexity, you can then deal with content storage management by opting for Object Storage. This not only allows you to delegate the management of content availability, but also reduces the load on your web servers.

You will no longer be directly responsible for requests relating to high volumes of content (images, videos, audio).

Once you have completed these two steps, the rest will depend on the specific features of your application. You will need to divide your services into containerisable microservices, taking into account the specific characteristics of your project.

CONTAINERISATION WITH MANAGED KUBERNETES SERVICE

Kubernetes is the leading orchestrator for easy deployment and management of your Docker container groups. It supports essential features such as self-healing (automatic component failure remediation) and **autoscaling** (automatic scaling and load balancing).

Although it's an essential tool, Kubernetes is difficult to administer. Of course, while you can deploy and manage your own Kubernetes cluster, this is usually more complex than administering a cluster of virtual machines.

What's more, ensuring the cluster's high availability is crucial, as without it, Kubernetes can become a

a Single Point of Failure (SPOF),

transforming a tool meant to simplify your infrastructure into a point of vulnerability.

Fortunately, the administration of your Kubernetes cluster can be delegated to a cloud provider like OVHcloud, via its Managed Kubernetes Service. Based on OVHcloud Public Cloud instances, this service includes additional load balancers and disks, allowing you to host any type of workload with full reversibility.

What's more, the service is completely free – you only pay for the on-demand instances and storage used in your Kubernetes cluster.

Speaking of auto-scaling with OVHcloud Managed Kubernetes Service: here's a little feature that makes all the difference...

The autoscaling offered by Kubernetes (K8s) is actually a form of pod autoscaling. This means that Kubernetes can independently adjust the number of containers in real time, based on workload, within the limits you set. For example, for a specific microservice, K8s will be able to create up to 10 containers, while guaranteeing a minimum of 3. But what happens if K8s doesn't find enough resources (computing instances) in your cluster to create the necessary containers? In this case, it will only create 7 containers instead of the 10 authorised.

This is where OVHcloud's solution comes in with its node autoscaling solution. In addition to the native K8s mechanism, you can specify a minimum and maximum number of deployable instances in your cluster. If additional resources are required, OVHcloud automatically adds and removes them once the peak load has passed. This way, you only pay for the resources you have actually consumed.

Be careful not to ignore a potential resource overconsumption caused by a bug in the code, which could go unnoticed thanks to these two mechanisms.

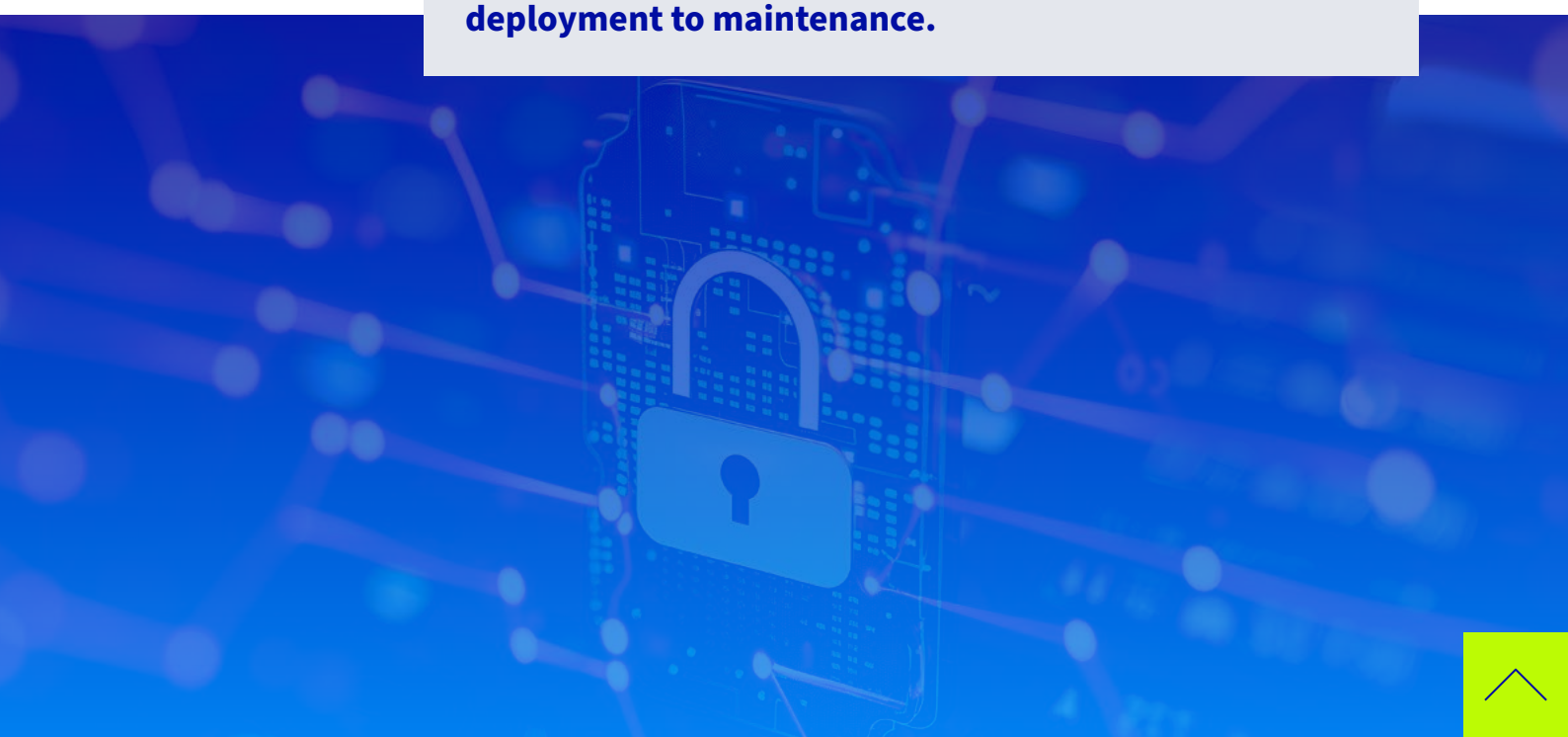
This is why monitoring is so important, which we'll discuss in more detail later on.

OPT FOR CENTRALISED, SIMPLIFIED MANAGEMENT OF YOUR KUBERNETES CLUSTERS WITH THE OVHCLOUD MANAGED RANCHER SERVICE

Once you have converted to Kubernetes, you may want to coordinate multiple clusters to take advantage of on-premises (hybrid cloud) resources, or to distribute your infrastructure across different geographical areas, either with a single cloud provider, or with multiple clusters (in a multi-cloud context). In this case, using Rancher can bring you a huge benefit, by simplifying the deployment and management of multiple clusters.

With the Rancher service managed by OVHcloud, you get all the features of Rancher to unify and simplify the management of your Kubernetes clusters, no matter which deployment you choose. This applies to a managed Kubernetes cluster (managed by OVHcloud or another provider), self-managed via Rancher Kubernetes Engine (RKE), or K3s clusters (the lightweight distribution of Kubernetes, ideal for IoT and Edge Computing), deployed on the infrastructure of your choice.

This centralised management includes user authentication, implementing security policies, and managing the entire lifecycle of your clusters, from deployment to maintenance.



STREAMLINE AND SECURE YOUR IMAGE MANAGEMENT WITH MANAGED PRIVATE REGISTRY (HARBOR)

To make your container-based projects more reliable, OVHcloud offers a private registry service, hosted and managed by its teams. This service allows you to easily store, manage and access images of your containers as well as Helm charts (via an API). It is based on Harbor, a project of the Cloud Native Computing Foundation, and guarantees secure access through RBAC (Role-Based Access Control) and a Content Trust mechanism to ensure the integrity of your image sources.

This service is compatible with all containerisation and orchestration platforms, including Kubernetes, and greatly simplifies the setup of your **CI/CD** processes (continuous integration and deployment), ensuring smooth and secure deployments.

Did you know? OVHcloud is a Certified Kubernetes Service Provider

Our Professional Services teams are certified by the Cloud Native Computing Foundation (CNCF), and can help you deploy, secure and scale your containerised workloads on Kubernetes.

Find out more:

https://www.ovhcloud.com/asia/professional-services/kcsp/

Why does OVHcloud's choice of AMD processors offer better performance for your Kubernetes clusters?

OVHcloud has opted for AMD processors for the physical machines that host its managed Kubernetes cluster service. These processors offer a higher number of cores and threads at a similar price. The ability to parallelise computations is a core tenet of cloud native computing, which translates into greater efficiency in use cases involving virtualisation and containerisation.

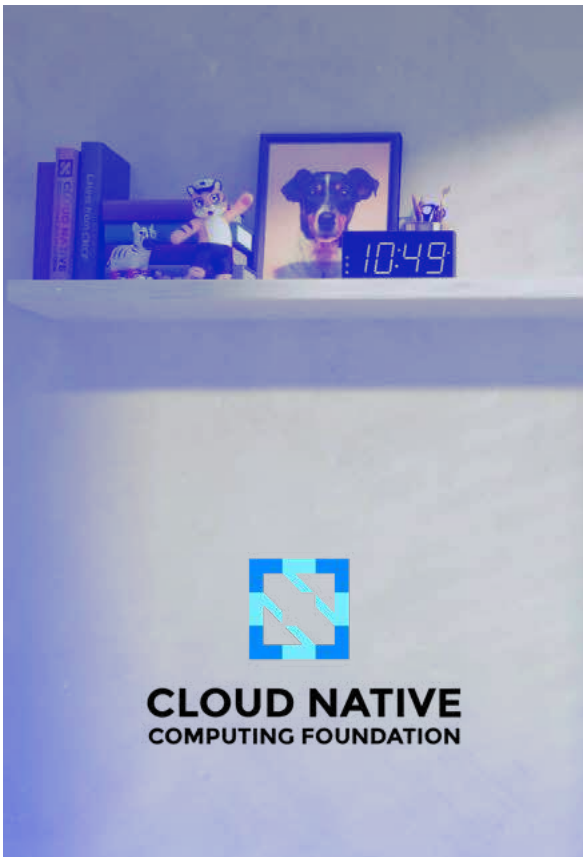
> **Cloud native** workloads require a fast, flexible, resilient, automated, highly efficient and quickly scalable infrastructure. They typically operate in multi-tenant environments and are designed to make use of microservices, with support for ongoing integration and deployment. These workloads can greatly benefit from a cloud-native hardware architecture with a high number of cores and threads. Benefits are measured by performance per **vCPU, server** or **rack**, and per watt consumed. The cloud native hardware architecture continues to evolve, with future designs likely to include more integrated and interchangeable accelerator units, as new developments and standards emerge to support modular architectures based on silicon chiplets, tiles and intellectual properties

WANT MORE? LET'S MOVE ONTO AUTOMATION!

The final step is to automate your infrastructure, also known as Infrastructure as Code (IaC). The idea is to treat all infrastructure resources (VM, storage, private networks, etc.) as commodities, and code-describe the infrastructure you want to deploy. This is an investment of time and skills development, but automation using tools like Terraform can be extremely cost-effective if you need to deploy multiple identical environments, version your environments, facilitate audits and certifications, implement a business continuity plan (BCP) or disaster recovery plan (DRP), or scale your infrastructure.

Develop skills through Cloud Native Computing Foundation courses

As you can see, by offloading the burden of managing physical machines, you increase the complexity of your software architecture. Without complete control, you risk adding unnecessary complexity to your infrastructure, especially if your team is unsure about their scope of responsibility: what is within their remit, which tasks fall to Kubernetes, and what OVHcloud take care of on the lower layers.



Note that the learning curve is longer with containerisation than with virtualisation.

However, the time invested in training your teams will quickly pay off. Migrating a legacy application is a long process, but you can already begin secondary projects directly with a cloud-native approach, which will boost efficiency.

Your teams’ technological growth is also at stake. Learning a new technology is challenging, but a key advantage of a cloud-native approach is that you can assemble independent development teams that can work on different functional areas, delivering them as and when – meaning no more production stress! With continuous deployment and integration, the innovation cycle is shortened, delivering real-world satisfaction to your developers.

Train with the Cloud Native Computing Foundation https://www.cncf.io/training/courses/

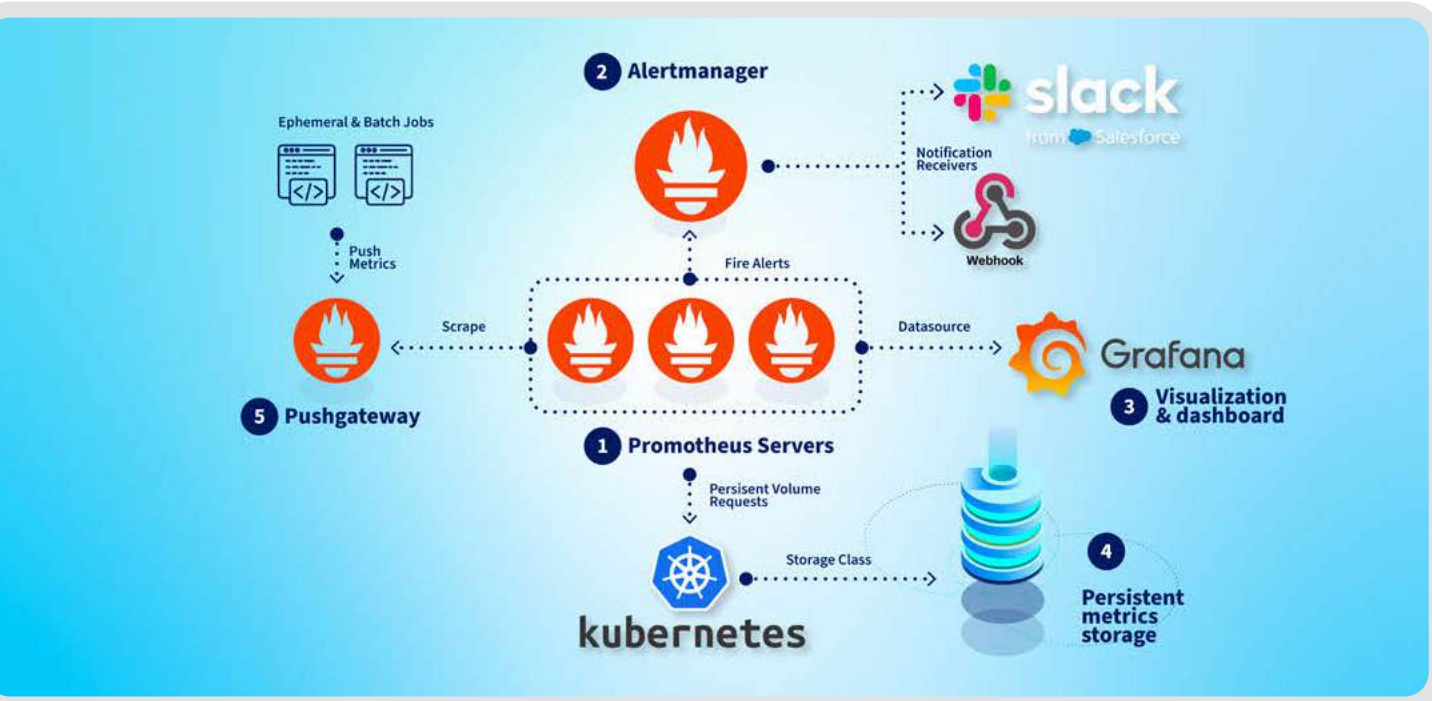
OVHcloud

PART 4

How a cloud-native approach will transform the way you work

ADAPT YOUR PLATFORM MONITORING

With a cloud-native approach and the use of managed services through a cloud provider like OVHcloud, it's essential to reconsider how you monitor your architecture. Monitoring the lower layers is no longer necessary, as this responsibility is delegated to OVHcloud's teams. However, you will need to implement tools to quickly identify problems, such as a poorly-sized container or a weakness in your architecture. Monitoring therefore evolves from a metrology logic (hardware monitoring) to application and business monitoring.



Logging, or traceability of actions via logs, also becomes crucial as your platform reboots on its own in the event of a failure.

For security, you could configure your containers in "no limit" mode during migrations or tests, but this configuration can make the diagnosis more complex and hide bugs that cause resource overconsumption.

For effective monitoring in a cloud-native environment, you will need a combination of tools: a tool for collecting and archiving data, such as Prometheus or InfluxDB, and another for visualisation through dashboards, such as Grafana, both available as a service.

EASILY CLONE YOUR PRODUCTION FOR A DEVELOPMENT (AND DEBUGGING) ENVIRONMENT

With the CI/CD approach, which is a natural by-product of cloud-native principles, a pre-production environment is often no longer necessary. Having a development environment, on the other hand, is essential for diagnosis and debugging, as you can easily duplicate your production environment. This creates a strictly identical development environment for testing before patching the production.

For the pre-production environment, there are several approaches.

One approach is to use a single Kubernetes cluster, where your development and production environments are logically isolated, but share the same resources.

You can even turn off the development environment on Friday nights and restart it on Monday mornings, freeing up all resources for production. However, this approach is risky

if you need to test a patch in an emergency following an incident, and it can be difficult to accurately estimate the cost of the development environment, especially for chargebacks between companies.

Another approach is to have a Kubernetes cluster with two separate “namespaces”: one for production and the other for development. This solution is especially well-suited when you manage several productions for each customer, while maintaining a single development environment.

REDUCED BACKUP REQUIREMENTS

The final question that arises when moving to a cloud-native architecture is the need for backups.

Your database is managed by OVHcloud, with the ability to revert to a previous state. Your data is stored in solution Y, replicated three times, and your code is probably on GitHub – so you may question whether you even need a backup.

The answer is yes, but mainly in hybrid architectures where the data is not fully outsourced. In this case, it is crucial to be able to reassemble the containers identically rather than reboot them from scratch, as this can lead to data loss. Solutions such as Velero (open source) or Kasten (via Veeam) can help you manage these backups. These tools can also be used to migrate your applications from one cloud provider to another, or set up BCPs and DRPs with an optimised RTO (Recovery Time Objective).

GET YOUR KUBERNETES CLUSTER UP AND RUNNING, AT AN ULTRA-COMPETITIVE

Example of a simplified cloud-native architecture on a Kubernetes cluster, consisting of three Discovery D2-4 instances

The benefits of this type of architecture:

- discover the power of automation on a full Kubernetes cluster, with three nodes;
- rely on the robustness of Discovery instances for your first steps with Kubernetes;
- upgrade to more powerful instances at any time, in just a few clicks;
- focus on your software development with services from the Public Cloud ecosystem, such as Managed Private Registry, Load Balancer or Object Storage;
- adopt the Infrastructure as Code approach to automate your deployments with Terraform.

Find out more :
https://www.ovhcloud.com/asia/public-cloud/architecture-cloud-native/

Migrate towards a cloud-native software architecture

OVHcloud

> **“Data is yours. The choice is yours too, and it matters. AI is what you make of it. Take control and innovate in a trusted environment!”**

About OVHcloud

OVHcloud is a global player in the cloud and the European leader in the sector. The company operates more than 450,000 servers in 40 datacenters on 4 continents, serving 1.6 million customers in over 140 countries. Spearheading a trusted cloud and pioneering a sustainable cloud with the best price-performance ratio, the Group has for over 20 years relied on an integrated model that gives it complete control over its value chain. From the design of its servers to the construction and management of its data centers, via the orchestration of its fiber optic network. This unique approach enables OVHcloud to independently cover all its customers’ needs, while allowing them to benefit from the virtues of an environmentally sound model, with frugal use of resources and an industry-leading carbon footprint. Today, OVHcloud offers state-of-the-art solutions that combine performance, predictable pricing and total data sovereignty to support its customers’ growth in complete freedom.