
## Victor Hang

Site Reliability Engineer
vhvictorhang@gmail.com
https://www.linkedin.com/in/victor-hang
https://github.com/Banh-Canh
Bordeaux, Aquitaine, FR


### SKILLS

Cloud & Infrastructure, Development & Automation, Systems & Databases

### EDUCATION

in DevOps Consultant | ib - groupe Cegos
, 2021 - 2021

in Network and System Administration | EPSI
, 2020 - 2020

### CERTIFICATIONS

GCP Architect ($certification.Date)
CKA Kubernetes Administrator ($certification.Date)
Gitlab Associate ($certification.Date)

### LANGUAGES

French (Native)
English (Professional)

### HOBBIES

Coding , Gym , Concerts , Travels

### EXPERIENCES

Site Reliability Engineer | S3NS
| Since Jun, 2023
As a Site Reliability Engineer on the Compute & Virtualization team, I am responsible for engineering the core of a sovereign cloud platform from a Thales-Google Cloud partnership. My primary mission is to design, automate, and operate the internal Infrastructure-as-a-Service (IaaS) layer, ensuring high availability, robust security, and strict compliance with the French government's SecNumCloud standards.

- Design, deploy, and maintain the lifecycle of highly available Kubernetes clusters that serve as the foundation for the sovereign cloud.
- Engineer a VM-as-a-Service experience using KubeVirt to run and manage virtual machines directly within Kubernetes via standard APIs.
- Architect and manage the cloud's virtual network infrastructure using KubeOVN and Multus, configuring complex multi-homed networking topologies and security policies.
- Implement and manage NixOS to create a fully declarative, reproducible, and immutable operating system environment, enhancing security and reducing configuration drift.
- Develop custom Kubernetes operators and controllers in Golang using the Kubebuilder framework to automate complex operational tasks and manage custom resources.
- Operate within a hybrid environment leveraging GCP primitives while ensuring all operations adhere to strict sovereign cloud principles and SecNumCloud controls.

DevOps Consultant at Orange Business Service | SFEIR
| Nov, 2022 - Jun, 2023
As a Level 3 DevOps Consultant for SFEIR, I was embedded with the client, Orange Business Services, to serve as a senior technical authority. My role involved architecting resilient cloud-native solutions, resolving complex systemic issues, and mentoring client teams on DevOps best practices for automation, security, and large-scale observability.

- Architected and implemented scalable and secure cloud infrastructure on AWS using Terraform, creating reusable modules for core services like VPC, EKS, and IAM.
- Ensured the stability, security, and performance of mission-critical Kubernetes clusters, handling cluster upgrades, advanced troubleshooting, and security policy implementation.
- Utilized Saltstack for configuration management of legacy systems and bare-metal servers, ensuring consistent integration with cloud-native infrastructure.
- Implemented and managed a centralized secrets management strategy using HashiCorp Vault, integrating it with both the Terraform IaC pipeline and Kubernetes workloads.
- Designed and maintained a comprehensive observability stack using Prometheus, Thanos, and Grafana to provide a unified, long-term view of system health across distributed environments.

DevOps Administrator | SysOps | CHEOPS TECHNOLOGY
| Jun, 2021 - Oct, 2022
As a key member of a dedicated DevOps squad, I was tasked with the design and construction of an internal Kubernetes-as-a-Service (KaaS) platform. The goal was to deliver a fully self-service, automated, and secure platform for internal development teams to build, deploy, and manage their applications.

- Contributed to the construction of the KaaS platform from the ground up, using Ansible to automate the provisioning of servers and the deployment of Kubernetes clusters.
- Implemented a GitOps workflow using ArgoCD, ensuring all platform and application configuration changes were version-controlled, auditable, and automatically deployed.
- Designed and managed robust CI/CD pipelines using GitLab CI/CD to automate the build, testing, and deployment of containerized applications.
- Secured the platform by integrating HashiCorp Vault for secrets management and Keycloak for centralized OIDC authentication and authorization (IAM).
- Maintained the health and performance of the underlying Linux servers, storage, and networking infrastructure that supported the KaaS platform.

## PROJECTS

Home Cluster
A comprehensive, personal Kubernetes-as-a-Service (KaaS) platform designed to replicate a professional cloud-native environment for home use. Built on Proxmox for virtualization, the entire infrastructure is declaratively managed with NixOS and a GitOps workflow powered by FluxCD. This ensures perfect reproducibility and automated configuration management. The architecture achieves secure multi-tenancy by using Kamaji to run Kubernetes control planes as pods within a management cluster, providing strong isolation. Identity and access are centralized through Zitadel, while HashiCorp Vault manages secrets securely. The platform serves as a powerful environment for hosting personal media servers, home automation, and for experimenting with advanced cloud-native technologies.

Netwatch
A powerful security and operations tool for Kubernetes that automates the management of temporary network access. It addresses the common operational need for granting developers or SREs short-term access to specific services (e.g., a database for debugging) without leaving permanent security holes. Netwatch introduces a 'NetworkAccess' Custom Resource (CRD) where a user can request access with a specified duration. The Netwatch controller then automatically generates a corresponding, temporary Kubernetes NetworkPolicy. Once the defined duration expires, the controller automatically removes the NetworkPolicy, ensuring the principle of least privilege is maintained and the cluster remains secure.

MaxTac
A Kubernetes controller created to abstract away the complexity of writing and managing native Kubernetes NetworkPolicy resources. It solves the problem of NetworkPolicies being verbose, difficult to reason about, and prone to misconfiguration. MaxTac introduces two intuitive Custom Resource Definitions (CRDs)—'Access' for internal traffic and 'ExternalAccess' for ingress/egress. These CRDs provide a simpler, higher-level abstraction for developers to define connectivity rules based on application logic rather than low-level pod and port selectors. The controller translates these simple CRDs into the appropriate, more complex NetworkPolicy YAML, reducing cognitive load and improving the security posture of the cluster.

Songbird
A command-line interface (CLI) tool that acts as a linter and simulator for Kubernetes network connectivity, designed to prevent network-related outages. In a complex cluster with numerous NetworkPolicies, it's challenging to predict the outcome of a change. Songbird solves this by allowing operators to proactively evaluate their NetworkPolicy configuration. It can simulate traffic flow between any two points in the cluster and report whether the current policy set would allow or deny the connection.

This enables teams to validate their network security rules in a CI/CD pipeline or locally before applying them to a live cluster, thereby preventing accidental service disruptions.

NixBook
A declarative and modular NixOS configuration framework specifically tailored for laptops. It addresses the classic 'it works on my machine' problem by allowing a developer's entire operating system—from the kernel and system packages to application configurations and dotfiles—to be defined as code using Nix Flakes. This ensures a perfectly reproducible, version-controlled environment that can be easily shared and deployed across multiple machines. The framework is designed with customizable profiles and modules, making it simple to manage different setups (e.g., for work and personal use) from a single, clean codebase.

NixOS-Server
A server-focused NixOS configuration that applies declarative and GitOps principles to manage bare-metal infrastructure. This repository contains a complete, code-defined setup for servers that host Kubernetes clusters. Every aspect of the server environment, including networking, storage, system packages, and security hardening, is defined in Nix code and managed in a Git repository. Changes to the infrastructure are made via a git push, which triggers an automated deployment process. This provides a powerful audit trail, enables atomic updates and rollbacks, and ensures that the entire server fleet remains in a consistent, predictable, and reproducible state.