CNCF Project Focus

Arc #1

Infrastructure Foundations

DISCIPLINED COMPOSITION

Building Cloud Native Infrastructure that scales

Christian Dussol

The problem isn't the **number** of tools, it's the lack of **discipline** in **composing** the toolbox.

*Inspired by **Kelsey Hightower***
(JetBrains Interview)

THE JOURNEY

Three CNCF projects. One question.

Episode #1 — Knative
Serverless / Scale-to-zero

Episode #2 — Crossplane
Infrastructure / Universal control plane

Episode #3 — Cilium
Networking / Kernel-speed security

THE MINDSET SHIFT

"What can this tool do?"

"How do these tools compose into a coherent platform?"

4 KEY PRINCIPLES

1 **Separated Responsibilities**

Each tool owns one domain. Strong contracts.
No overlap.

2 **Composition Over Collection**

Value isn't in individual tools. it's in how they compose.

3 **Governance As Foundation**

Policy-as-Code alongside capabilities, not after.

4 **Containers As The Steady Core**

Serverless for the right patterns. Containers by default.

The correlation

The same **composition** principles that work for **technology** work for **team organization**.

TECHNOLOGY		**TEAMS**
Separated domains	⟷	Platform teams
Strong APIs	⟷	Clear contracts
Reduced coupling	⟷	Reduced cognitive load

Team Topologies

THE COMPOSED PLATFORM

APPLICATION LAYER

KNATIVE
Auto-scaling • Cost-optimized

K8s API

NETWORK LAYER

CILIUM
L7 Security • Observability

Kyverno
Across all layers

K8s API

INFRASTRUCTURE LAYER

CROSSPLANE
Multi-cloud • Self-service

aws

Google Cloud

HONEST LESSONS

What I would tell you before you start.

Knative

Serverless isn't for everything.

Scale-to-zero shines for idle workloads but containers remain your default.

Crossplane

The learning curve is steep.

Multi-cloud abstraction comes at the cost of serious upfront investment in Compositions and CRDs.

Cilium

eBPF is powerful but opaque.

When networking breaks, debugging requires expertise your team may not have yet.

Build skills before you build platforms.

WHAT'S NEXT

ARC #1
DISCIPLINED COMPOSITION / INFRASTRUCTURE FOUNDATIONS
✅ COMPLETE

The foundation is solid.
But how do we know what's happening inside?

- Application performance?
- Distributed tracing?
- Cost attribution?
- Proactive alerting?

ARC #2: OBSERVABILITY

You can't optimize what you can't see

THE FULL STORY

Read my complete synthesis on **Medium**

Includes

- Architecture patterns
- Real-world scenarios
- Source code
- Kyverno governance policies
- Team Topologies applications

Medium article: **https://bit.ly/3ZZRVQg**