Let's Go Pulsar

Developing Cloud Native Apache Pulsar Applications with Go

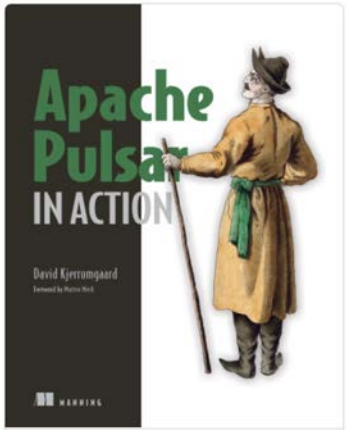
David Kjerrumgaard
Developer Advocate

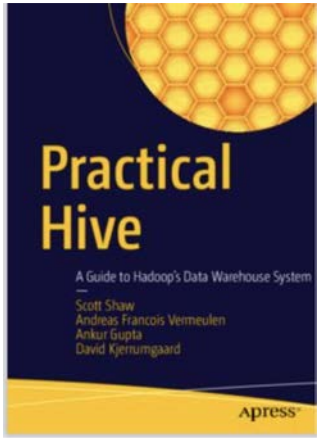
- Apache Pulsar Committer
- Former Principal Software Engineer on Splunk's internal Pulsar-as-a-Service platform.
- Former Director of Solution Architecture at Streamlio.
- Global practice director of Professional Services at Hortonworks.

amazon FedEx splunk> HORTONWORKS LOCKHEED MARTIN Zappos.com

David Kjerrumgaard
Author

- Author of **Pulsar In Action.**
- Co-Author of ***Practical Hive***





https://streamnative.io/download/manning-ebook-apache-pulsar-in-action

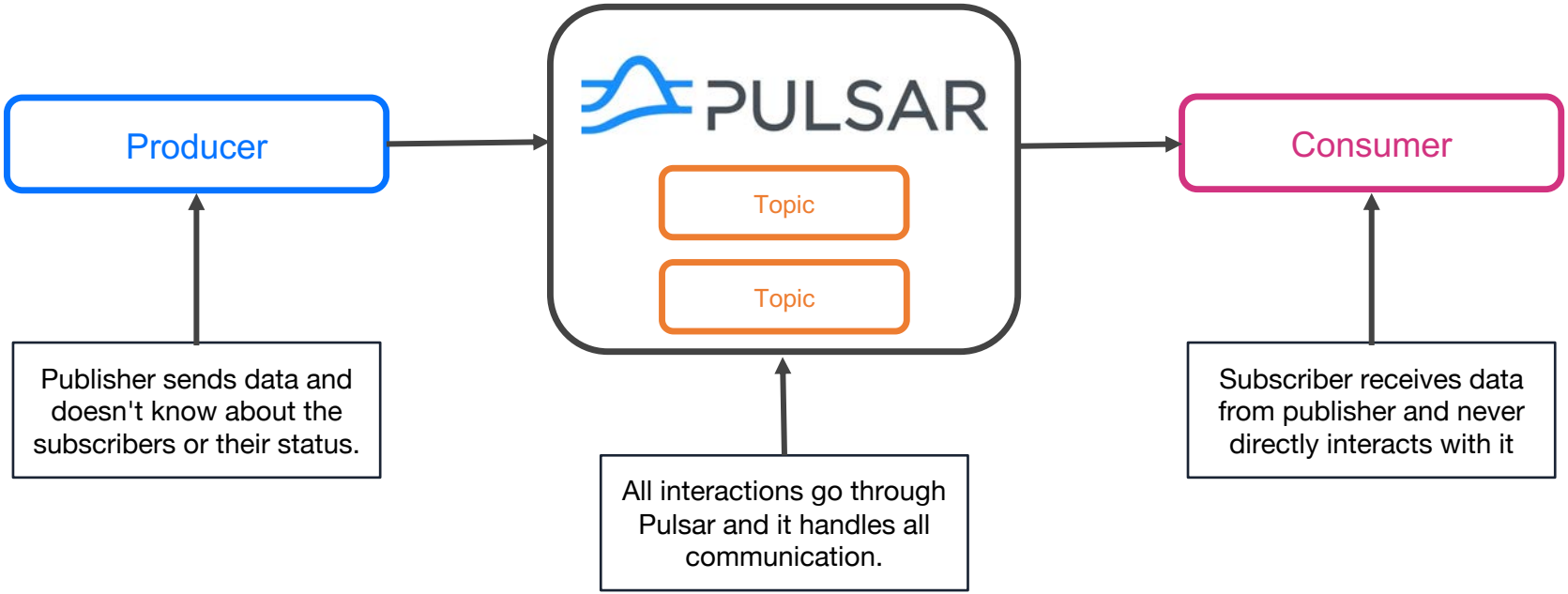
Agenda

1. What is Apache Pulsar?
2. Why Go is a good fit for Cloud-Native Pulsar Apps
3. Building Cloud-Native Pulsar Apps with Go
4. Deploying Cloud-Native Pulsar Apps with Go

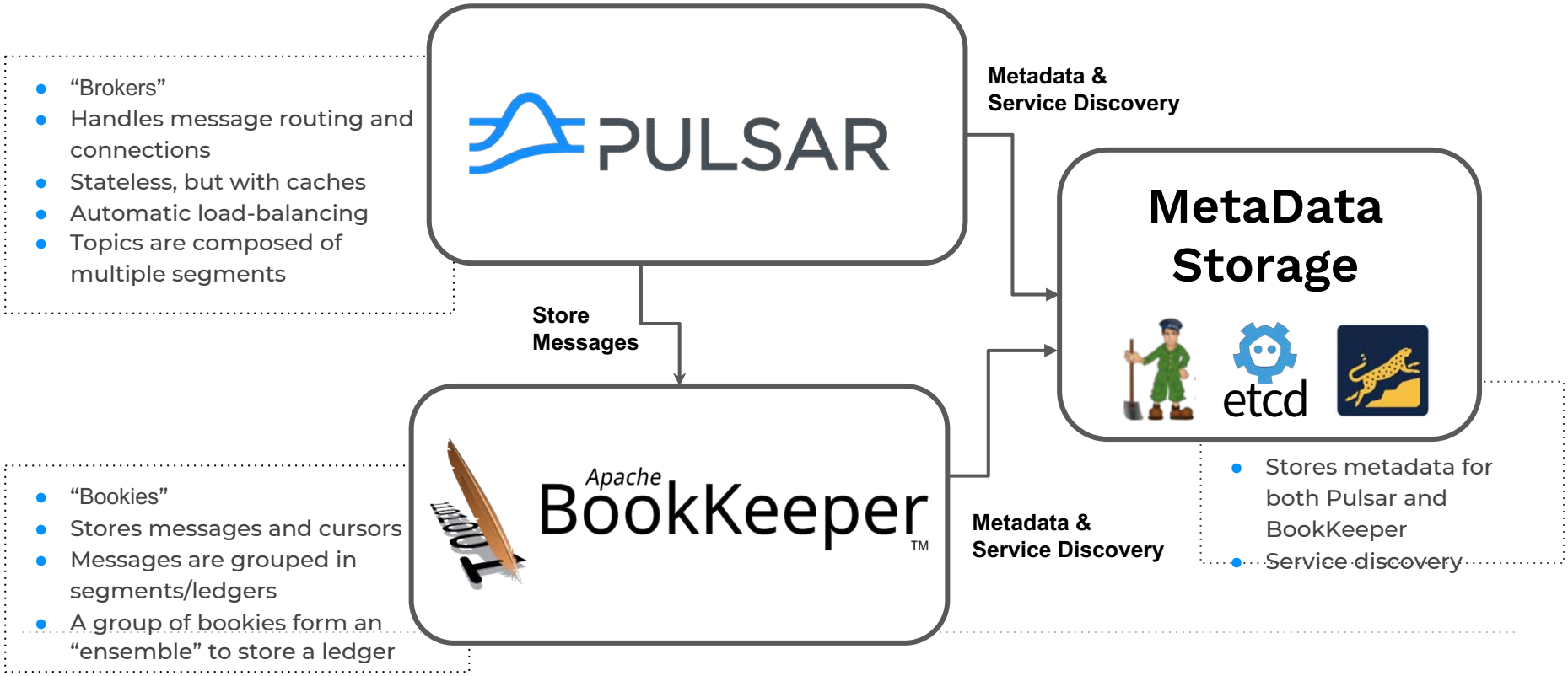
What is Apache Pulsar?

Apache Pulsar is a Cloud-Native Messaging and Event-Streaming Platform.

Producer-Consumer



Pulsar Cluster



Pulsar's Distinguishing Features

Tiered storage

Enable historical data to be offloaded to cloud-native storage and store event streams for indefinite periods of time.

Multi-tenancy

Enable multiple user groups to share the same cluster, either via access control, or in entirely different namespaces.

Scalability

Decoupled data computing and storage enable horizontal scaling to handle data scale and management complexity.

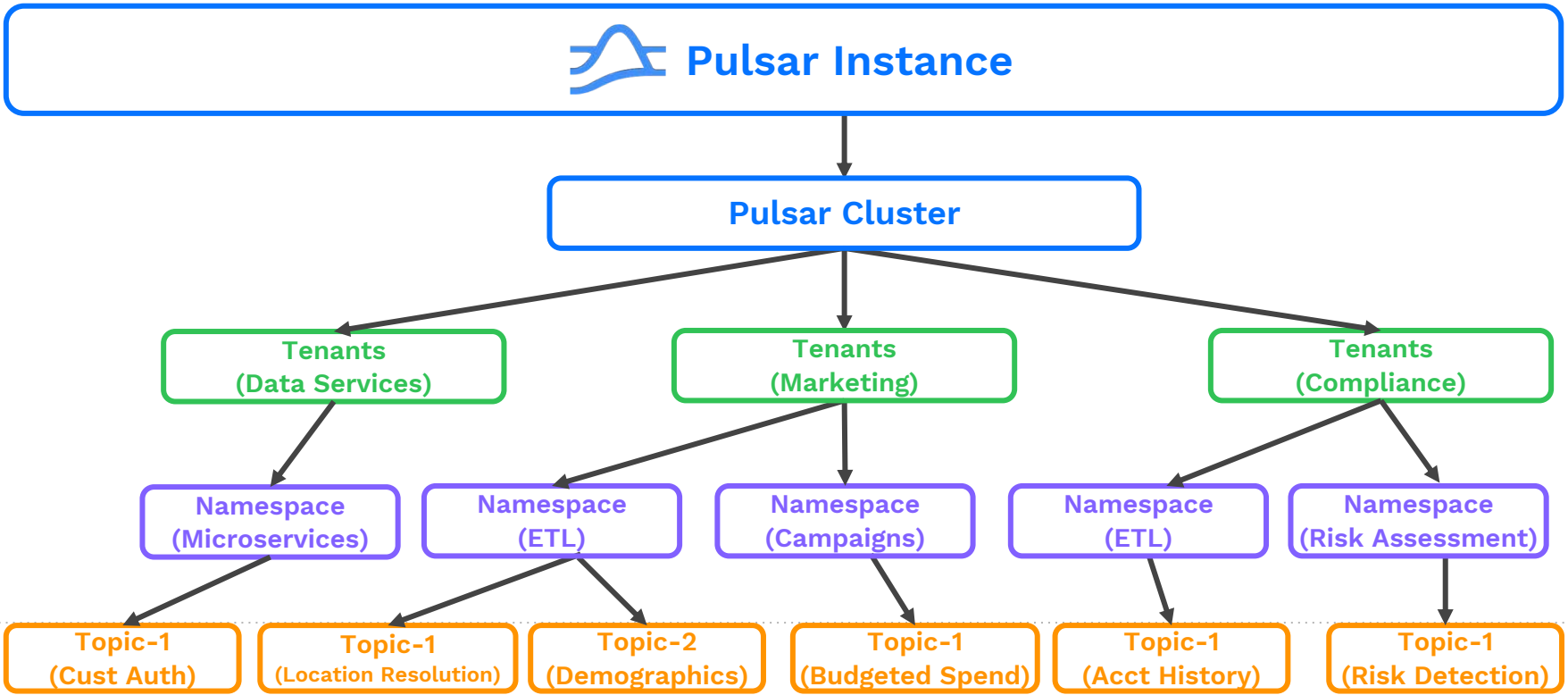
Geo-replication

Support for multi-datacenter replication with both asynchronous and synchronous replication for built-in disaster recovery.

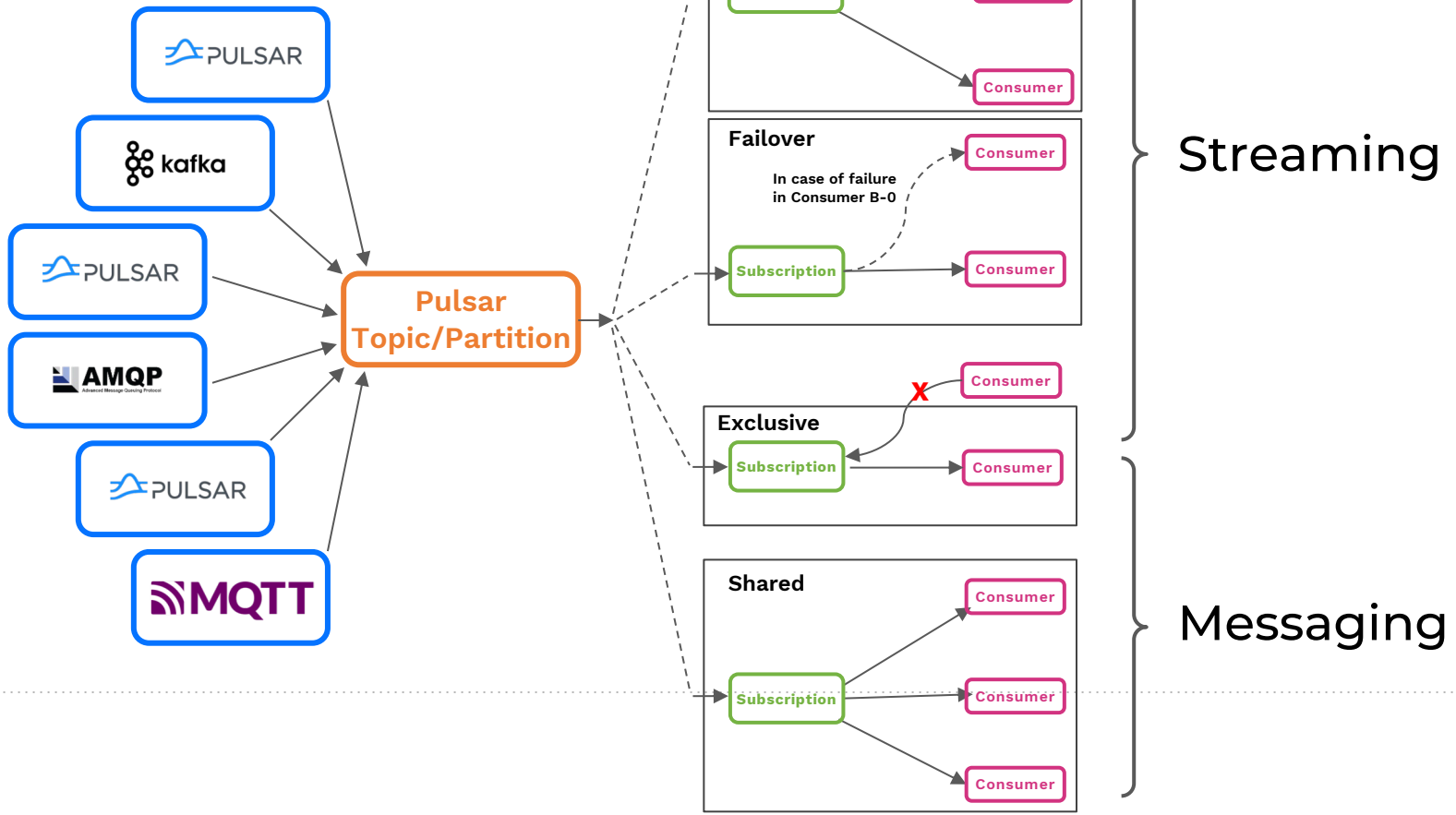
Durability

Provides strong data durability guarantees by fsync-ing data to multiple disks on each write.

Topics


Pulsar Instance
Pulsar Cluster
Tenants (Data Services)
Tenants (Marketing)
Tenants (Compliance)
Namespace (Microservices)
Namespace (ETL)
Namespace (Campaigns)
Namespace (ETL)
Namespace (Risk Assessment)
Topic-1 (Cust Auth)
Topic-1 (Location Resolution)
Topic-2 (Demographics)
Topic-1 (Budgeted Spend)
Topic-1 (Acct History)
Topic-1 (Risk Detection)

Multi-Protocol



Why Go is a good fit for Cloud-Native Pulsar Apps

Cloud-Native Advantages

- Go has become a popular choice for building microservices and cloud-native applications. This is due in part to its efficiency and scalability, as well as its support for concurrency and parallelism.
- Go also has strong support for building HTTP servers and clients, making it well-suited for building RESTful APIs and other web services.
- Go has several third-party libraries and frameworks that are specifically designed for cloud-native development. K8s itself is written in Go.

Pulsar Ecosystem Support

- Apache Pulsar provides a Golang client library that allows developers to interact with Pulsar clusters from their Golang applications.
  - This client library provides a simple and intuitive API that allows developers to publish and consume messages from Pulsar topics, as well as perform administrative tasks like creating and managing topics.
- Pulsar provides a Golang Function API that allows developers to write stream processing applications in Golang.
  - This API provides a simple and efficient way to process data streams in real-time. Developers can easily create custom stream processing functions in Golang, which can be deployed and run in Pulsar clusters.

Building Cloud-Native Pulsar Apps with Go

The Pulsar Go Client

- You can use a Pulsar Go client to create Pulsar producers, consumers, and readers in Golang

- You can install the pulsar library by using either **`go get`** or **`go module`**

```
go get -u "github.com/apache/pulsar-client-go/pulsar"
```

- API docs are available on the Godoc page; https://pkg.go.dev/github.com/apache/pulsar-client-go/pulsar

Live Local Dev & Test Walkthrough

Packaging Cloud-Native Pulsar/Go Applications

Cloud-Native = Containerization

- The first step in deploying a cloud-native application is to containerize the application.
- Containerization is a method of virtualization that allows you to package an application and its dependencies into a single unit, called a container.
- Containers provide an isolated environment for an application to run in, ensuring that it has access to all the resources it needs without interfering with other applications running on the same system.

Containerization with Buildpacks

- Buildpacks are a technology for building container images that automate the process of compiling and packaging application code and its dependencies.
- Buildpacks are used to create container images for cloud-native applications without needing to create an maintain a dockerfile.
- Buildpacks detect the language and framework used by the application, and then automatically download and install the necessary dependencies and configurations.

Container Tagging & Publication

- After generating the containers with buildpacks.io, they only exist on your development machine.
- They must be tagged and published to a container repository in order to be deployed outside of your local development environment.

Live Containerization Walkthrough

Deploying Cloud-Native Pulsar/Go Applications

Deployment Manifest

- Kubernetes uses YAML files called manifests to define the configuration for the application deployment.

- Manifests include details about the container image to use, how many replicas of the application to run, and other configuration details such as;
  - Deployment
  - ConfigMap /Secret
  - PersistentVolumeClaim

ConfigMaps

- ***ConfigMaps*** are Kubernetes objects that provide a way to store configuration data in key-value pairs.
- They are typically used to store configuration data that is required by an application at runtime, such as environment variables or configuration files.
- They can be used to store configuration files that are required by an application at runtime. The configuration files can be mounted as a volume in a container using the volumes field in the pod manifest.

Live Deployment Demonstration

Summary

Key Takeaways

1. Apache Pulsar is a cloud-native messaging and event streaming platform
2. Go is a good fit for developing cloud-native applications that use Pulsar
3. Apache Pulsar's native Go client library makes developing cloud-native Pulsar applications easy.
4. Buildpacks are a great tool for containerizing your Go applications
5. Walked through the process of packaging and deploying a cloud-native Go application that interacts with Apache Pulsar

https://github.com/david-streamlio/lets-go-pulsar