LF NETWORKING
Developer & Testing Forum

ONAP NF Package Onboarding Use Cases

ONAP Streamlining – The Process

November 2023

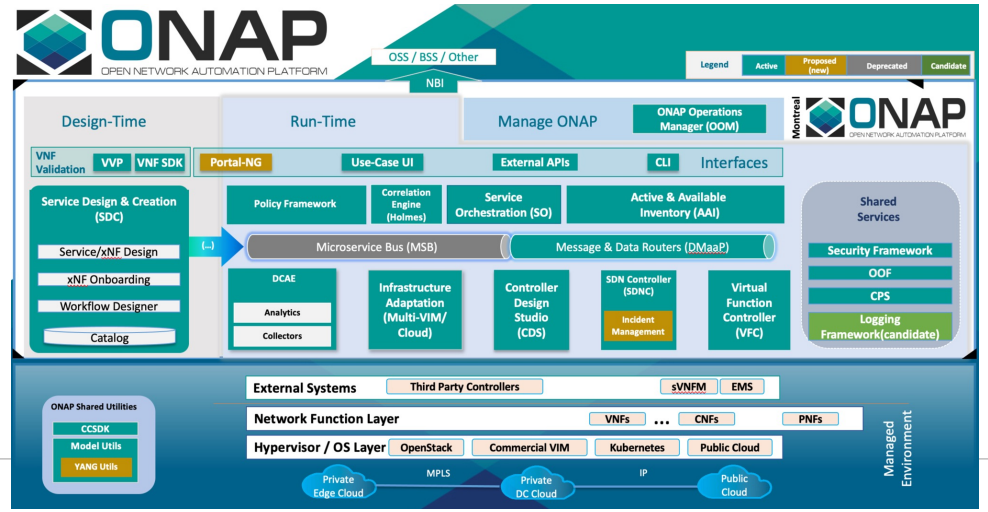
Presenter:
Kamel Idir (Ericsson)
Byung-Woo Jun (Ericsson) – ONAP ARCCOM Chair, TSC

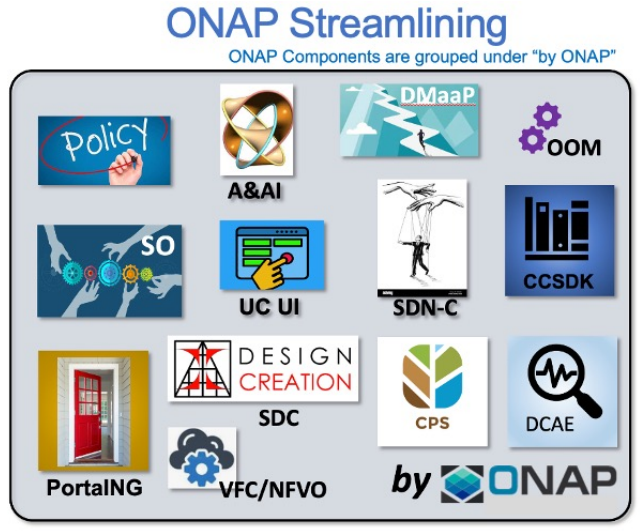
https://lfnetworking.org

ONAP Streamlining – Transformation

LFN Developer & Testing Forum

- Thru ONAP Streamlining, **ONAP is no longer a platform**, rather it provides various network automation functions, and security reference configuration in LFN.
- ONAP enables individual ONAP function build, and component deployment thru CD
- **Build use cases for repository-based E2E service, NS, CNF and CNA onboarding, and CD-based ONAP component triggering mechanisms with abstracted interfaces for choreography**
  - Standard-based abstracted interfaces with declarative APIs
  - Each component is autonomous and invoked from any level of Network Automation, by leveraging CD mechanisms – e.g., GITOps and CD readiness
- ONAP will become more intent-based and declarative, and bring in more AI, conforming to standards such as 3GPP, TMForum, ETSI, IETF, ORAN, etc.
  - Extend UUI User Intent support and AI-based natural language translation, by applying coming 3GPP and TMForum models and APIs
  - Delegate resource-level orchestration to functions from the external community
- ONAP continues to support the Service Mesh, Ingress, OAuth2, IdAM-based authentication and authorization, and consider sidecar-less solution for NF security.





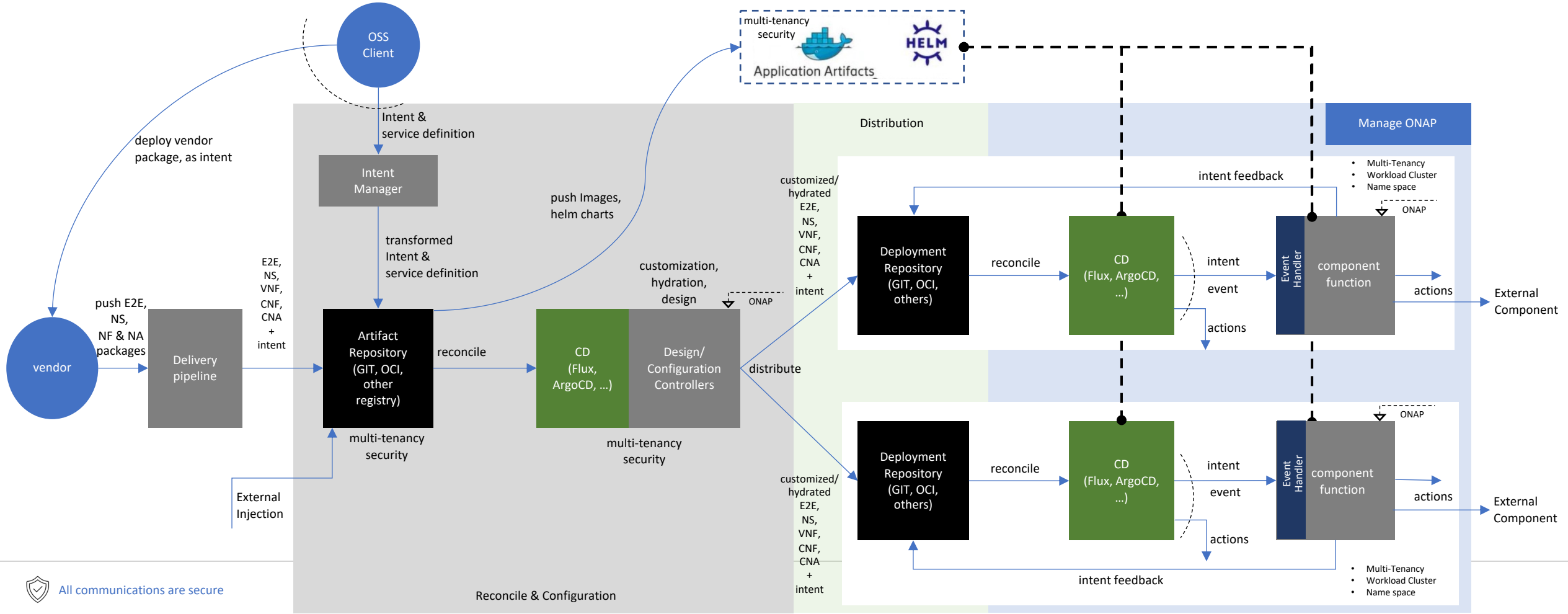
ONAP focuses on Network Automation Functions

- Modular
- individual
- interface abstraction
- loose coupling
- Extensibility
- Interchangeability
- Autonomous
- Declarative
- CI / CD ONAP components and E2E Service, NS, CNF & CAN handling

ONAP Repository-based Software Package + LCM Use Case – an idea

LFN NETWORKING
LFN Developer & Testing Forum

- Package onboarding to ONAP thru repository can also trigger SW LCM flows (deploying packages as intents).
- Applications, packages and intents are worked in the multi-tenancy, multi-workload cluster and multi-namespace runtime environment.

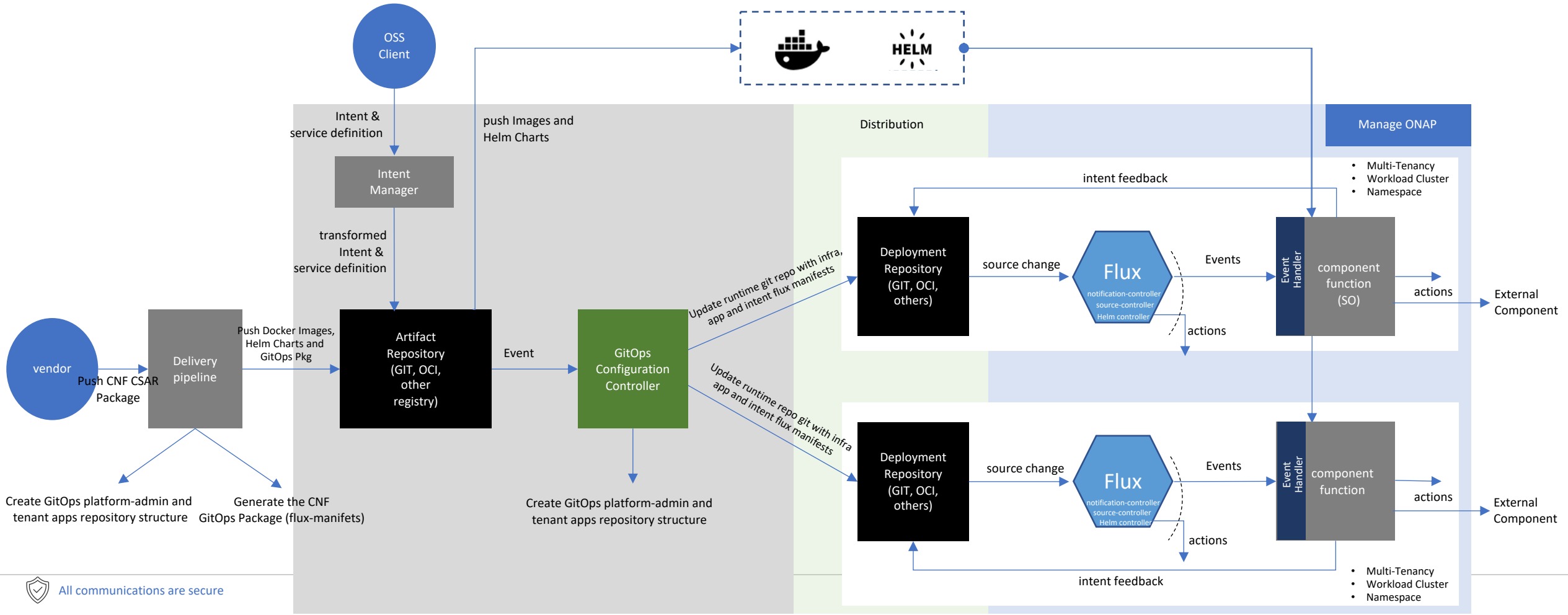


All communications are secure

ONAP CNF CSAR Package Onboarding Use Case

LFN Developer & Testing Forum

- ONAP repository-based SW management enables smaller imperative actions that can be triggered by different events in the orchestration and SW LCM flow.
- Applications, packages and intents are worked in multi-tenancy, multi-workload cluster and multi-namespace runtime environment



ONAP CNF CSAR Package Onboarding and FluxCD Deployment Use Case

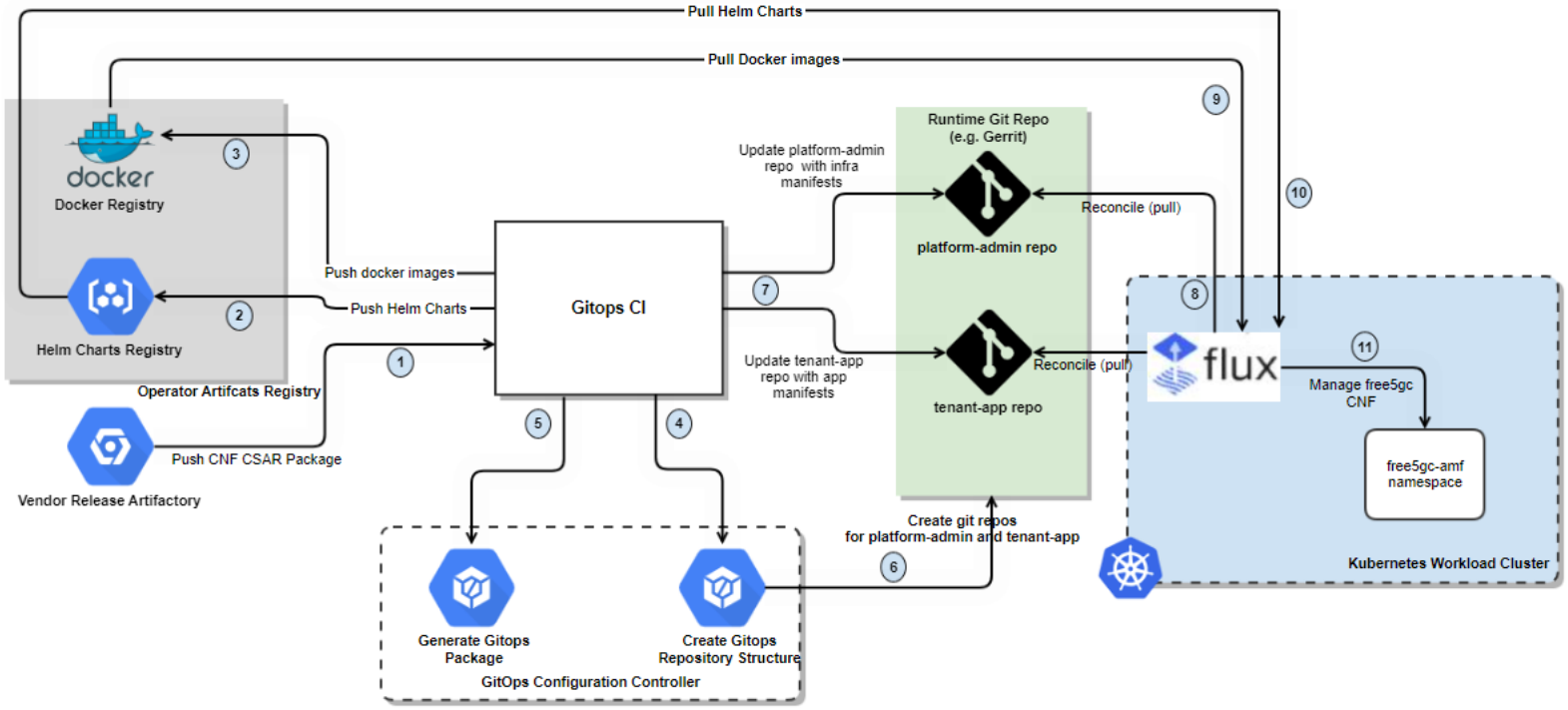
LFN Developer & Testing Forum

Use Cases:

1. Download CNF CSAR package / Vendor pushes CNF CSAR package.
2. Gitops CI pushes Helm charts to helm registry.
3. Gitops CI pushes docker images to Docker registry.
4. Gitops CI creates Gitops repository structure.
5. Gitops CI generates Gitops package (flux manifests required to create app desired state in runtime git repo).
6. GitOps Config controller creates runtime git repos for platform-admin tenant and tenant-app (multi-tenancy).
7. GitOps CI updates runtime git repos with infra and application manifests (app desired state).
8. Flux reconciles.
9. Flux pulls docker images from docker registry.
10. Flux pulls helm charts from helm repository
11. Deploys CNF in Kubernetes Workload Cluster.

Pre-requisites:

- Workload Cluster deployment
- Flux Bootstrap



Git Repository Structure (Flux)

- Mono Repo
  - Single tenant clusters
  - Single repo for all clusters and applications
- Repo per environment.
  - In this approach, we can have different repositories for Lab/Staging/Production, OR different repo for each cluster.
- Repo per team
  - This approach is for multi-tenancy clusters if we assume that an organization has a dedicated platform admin team that provides Kubernetes-as-a-service for other teams. For example, we can have
    - One repository for platform-admin team responsible for setting up the staging and production environments, maintains the cluster-wide resources (CRDs, controllers, admission webhooks, etc.) and configurations of tenant's namespaces and Git repos
    - One repo per tenant responsible for setting up applications definitions (K8s deployments, Helm releases) and configures how the apps are reconciled on each environment (Kustomize overlays, Helm values).
- Repo per application
  - Like repo per team, but in this approach each tenant is a single app instead of a team (handling one or more apps).

Ways of structuring your repositories | Flux (fluxcd.io)

Runtime Git Repo Structure Example for Flux Multi-Tenancy

Platform admin repo

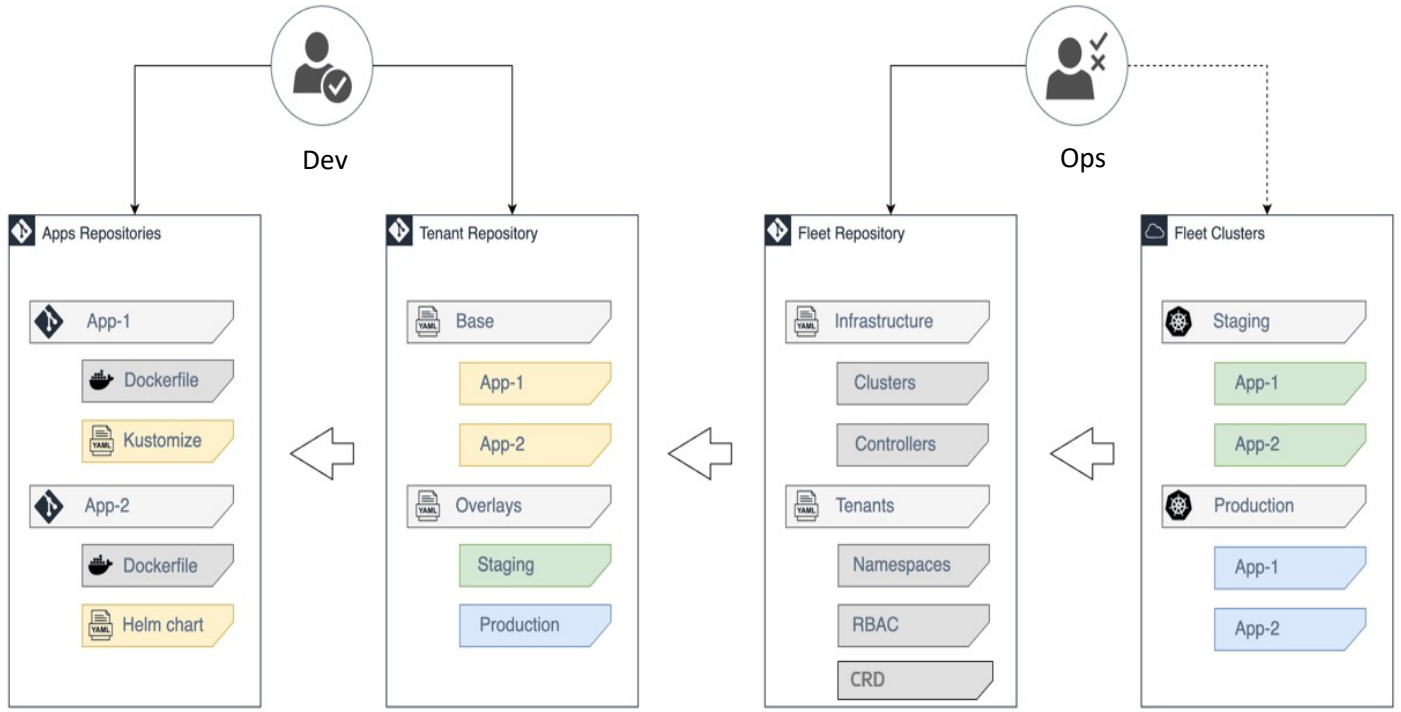
```
├── clusters
│   └── cluster-01
│       ├── flux-system
│       │   ├── gotk-components.yaml
│       │   ├── gotk-sync.yaml
│       │   └── kustomization.yaml
│       ├── infrastructure.yaml
│       └── tenants.yaml
├── infrastructure
│   ├── base
│   └── cluster-01
│       └── kustomization.yaml
└── tenants
    ├── base
    │   └── dep-team
    │       ├── helm-repo.yaml
    │       ├── kustomization.yaml
    │       ├── rbac.yaml
    │       └── sync.yaml
    └── cluster-01
        └── dep-team
            └── free5gc-amf
                └── kustomization.yaml
```

Tenants' repo

```
└── apps
    └── cluster-01
        └── free5gc-amf
```

Flux Multi-Tenancy: Platform admin vs Tenant Roles

- Platform Admin
  - Has cluster admin access to the fleet of clusters.
  - Has maintainer access to the fleet Git repository.
  - Manages cluster wide resources (CRDs, controllers, cluster roles, etc).
  - Onboards the tenant's main GitRepository and Kustomization.
  - Manages tenants by assigning namespaces, service accounts and role binding to the tenant's apps.
- Tenant
  - Has admin access to the namespaces assigned to them by the platform admin.
  - Has maintainer access to the Git repository and apps repositories.
  - Manages app deployments with GitRepositories and Kustomizations.
  - Manages app releases with HelmRepositories and HelmReleases.



fluxcd/flux2-multi-tenancy: Manage multi-tenant clusters with Flux (github.com)

Q & A

LFN NETWORKING
LFN Developer & Testing Forum

Thank you !