Plan de revue – Architecture DAAS Teranga (archi12.html)

1. Périmètre et objectifs

- Cas d’usage couverts, SLA/SLO visés, volumes/latences cibles.
- Contraintes sécurité/conformité, dépendances externes (Keycloak, OPA, Elastic) et attentes de disponibilité.

2. Lecture structurée du diagramme

- Vérifier cartographie des 3 VMs, rôles (ingress/compute/storage/security/governance/monitoring) et cohérence légende Mermaid.
- Confirmer répartition HA et affinités/anti-affinités implicites.

Questions

- Les trois VMs sont-elles de même taille et disposent-elles d'anti-affinités pour les rôles critiques (Kafka, Redis, PG, KDC) ?
- Quels services sont réellement actifs/actifs vs actif/passif ? Quelles probes (liveness/readiness) les gouvernent ?
- Le diagramme reflète-t-il la nomenclature réelle (FQDN, IPs, ports) et la correspondance avec l'infra cible ?

3. Réseau, accès, exposition

- Chemin d’entrée VIP MetalLB → Nginx Ingress: HA, annonces, healthchecks, timeouts.
- TLS/Cert-Manager: couverture FQDN, renouvellement, stockage clés, politiques de chiffrement.
- Segmentation: namespaces, network policies, pare-feu inter-VM, multi-tenant.

Questions

- VIP MetalLB: quel protocole d’annonce (BGP/ARP), combien de speakers par VM, stratégies d’élection ?
- Ingress: quelles classes, quelles limites de connexion/timeouts, quelle politique de retry ?
- Cert-Manager: ACME ou PKI interne ? Où sont stockées les clés privées ? Politique de rotation ?
- NetworkPolicies: quelles sont les communications explicitement autorisées entre namespaces/services ? Y a-t-il un default-deny ?
- Pare-feu inter-VM: quelles règles minimalistes sont nécessaires pour le cluster (et seulement celles-ci) ?
- Gestion multi-tenant: quelles isolations (namespace, RBAC, quotas) sont prévues ?

4. Identité, authentification, autorisation

- OIDC Keycloak: flux, durées de jetons/refresh, scopes/claims, propagation.
- OPA: points d'application (Ingress, apps), politique par défaut, logs de décision.
- Kerberos KDC: synchro KDC1→KDC2, intégration Kafka/Trino, rotation des clés.

Questions

- Keycloak: quelles durées d'access/refresh tokens ? Quelles claims sont utilisées pour l'authz applicative ? Politique de rotation des secrets clients ?
- OPA: où sont placés les points d'application (Ingress, sidecar, middleware applicatif) ? Quel fallback si OPA est indisponible ? Les policies sont-elles testées et versionnées ?
- Kerberos: comment est gérée la réplication KDC1→KDC2 ? Où sont stockées les clés principales ? Quelle procédure de rotation et de recovery ?

5. Données, stockage, persistance

- MinIO (réplication 1⇔2), Longhorn (1-2-3): politiques de réplication, anti-affinité, RPO/RTO, sauvegardes/restores.
- Postgres primary/replica: mode de réplication, bascule, cohérence avec Airflow/Superset.
- Kafka (3 brokers): facteur de réplication, ISR, quotas, schémas/compatibilité, monitoring du lag.

Questions

- MinIO: mode de réplication (site/site ou distributed erasure sur les 2 instances) ? Tests de restore récents ? Politique de versioning et de lifecycle ?
- Longhorn: factor de réplication, anti-affinité des réplicas, politique de snapshots/backups hors cluster ?
- Postgres: réplication synchrone ou asynchrone ? Temps de bascule cible ? Comment Airflow/Superset tolèrent la bascule ?
- Kafka: RF par topic, minISR configuré ? Quotas producteurs/consommateurs ? Gestion du schéma (Schema Registry ?) et compatibilité ?

6. Cache et état éphémère

- Redis master/replica: bascule, persistance AOF/RDB, partitionnement, usages (cache Superset, sessions Airflow).
- Spark executors / Airflow workers: gestion éphémère, montée/descente en charge, isolation.

Questions

- Redis: mode de failover (sentinel/operator) ? AOF activé ? Politique de sauvegarde/restore ? Nombre de shards si partitionnement ?
- Quels TTL pour les caches Superset et les sessions Airflow ? Comportement en cas de perte du cache (dégradation acceptable ?).
- Pour les workloads éphémères (Spark, Airflow K8sExecutor): quotas, limites, priorités, mécanismes d'auto-scaling ?

7. Traitement et orchestration

- Airflow HA: mode scheduler (actif/actif ?), backend DB unique, broker Redis, reprise après incident.
- NiFi + Registry: réplication des flows, persistance d'état, S2S/SSL.
- Trino: coordonnateur unique, workers 1/2; affinité, élasticité.
- Superset: scaling horizontal, politiques de cache.

Questions

- Airflow: nombre de schedulers et webservers, healthchecks, mode de bascule ? Comment est géré le broker Redis (HA, persistance) ?
- NiFi: configuration du cluster, persistance du state management, chiffrement S2S, réplication/backup du Registry ?
- Trino: haute dispo du coordonnateur (unique ?) ; comment est gérée la perte du coordonnateur ? Workers auto-scalables ?
- Superset: stratégie de cache (clé, TTL, invalidation), scaling horizontal, gestion des connexions DB.

8. Observabilité et opérations

- Prometheus principal/HA: fédération, rétention, stockage, alerting.
- Logs/trace: pipeline manquant sur le schéma (Elastic ? autre), couverture applicative.
- Tableaux de bord SLI/SLO, runbooks incidents, tests de reprise.

Questions

- Prometheus: comment sont gérées la rétention et le stockage durable ? Y a-t-il une fédération ou un pair de secours ? Alertmanager configuré ?
- Logs: quelles sources sont centralisées ? Format commun ? Rétention et accès restreint ? Existe-t-il une corrélation traces/logs/metrics ?
- SLI/SLO: quels indicateurs par domaine (ingress, auth, Kafka, DB, jobs batch) ? Quelles alertes et seuils ? Runbooks documentés et testés ?

9. Résilience, disponibilité, reprise

- Modes de panne: perte VM, réseau, dépendance externe.
- Procédures de bascule (Redis, PG, Kafka, KDC, Ingress), tests de chaos.
- Capacité vs charge: sizing RAM/CPU, marge, stratégie d'auto-scaling si K8s.

Questions

- Quelles procédures documentées de bascule pour Redis/PG/Kafka/KDC/Ingress ? Derniers tests réalisés ?
- Quels scénarios de chaos/DR ont été exécutés (perte VM complète, perte réseau inter-VM, perte service externe Keycloak/OPA/Elastic) ?
- Capacité: quel headroom actuel vs pic estimé ? Existe-t-il une stratégie d'auto-scaling ou au moins d'alerting sur saturation ?

10. Sécurité élargie

- Chiffrement en transit (Ingress↔services, services↔externes) et au repos (Longhorn/MinIO/PG/Kafka).
- Gestion des secrets (stockage, rotation, audits).
- Durcissement images/OS, scans vulnérabilités, segmentation réseau.

Questions

- Chiffrement: TLS partout ? Mutual TLS interne ? Cipher suites/policies ? Certificats pour Kafka, NiFi, Trino ?
- Secrets: où sont stockés (Vault/SealedSecrets/autre) ? Politique de rotation et d'audit ? Gestion des credentials DB/Kafka/Keycloak ?
- Sécurité hôte et images: niveau de durcissement OS, scanner d'images, fréquence des scans vulnérabilités, correctifs ?
- Segmentation: RBAC Kubernetes détaillé ? Accès admin restreints et tracés ?

11. Gouvernance et données

- OpenMetadata/SODA: couverture des sources, qualité, lineage, fréquence des scans.
- Elastic: indexation/search OpenMetadata, capacité, ILM, sécurité.

Questions

- OpenMetadata: quelles sources sont onboardées ? Qui maintient le catalogue ? Quelle fréquence de synchronisation et de profiling ?
- SODA: quelles règles de qualité ? Où sont stockés les résultats et alertes ? Intégration avec Trino coordonnateur ?
- Elastic: indices concernés, politique ILM, authentification/autorisation, capacité et backups ?

12. Déroulé de revue

- Kickoff 30–45 min pour cadrer objectifs/contraintes.
- Revue statique du diagramme + questionnaires par domaine (réseau, sécurité, données, ops).
- Session risques/écarts et priorisation (haut/moyen/bas).
- Restitution: fiche risques + plan d’actions court/long terme; demandes éventuelles de tests de bascule/charge.

Questions pour le déroulé

- Quels livrables attendus (fiche risques, plan d’actions, backlog) et horizon de mise en œuvre ?
- Qui sont les référents par domaine (réseau, sécurité, données, ops) et leurs disponibilités ?
- Quel calendrier pour d’éventuels tests complémentaires (bascule, charge, chaos) ?

13. Artéfacts à collecter

- Config MetalLB/Ingress, Cert-Manager, policies OPA, realms/clients Keycloak.
- Topologie Kafka/PG/Redis/Longhorn/MinIO, plans de backup/restore.
- Dashboards Prometheus/alerting, runbooks existants, SLA/SLO formalisés.