



id: SI_007

model: google/gemma-4-26b-a4b-it

seeds:

04_risiko: mulighet

03_relasjon: makt

05_handling: implementer

01_perspektiv: strategisk

02_tidhorisont: syklisk

Instans SI_007

Seeds: strategisk | syklisk | makt | mulighet | implementer

Analyse: Kontroll av agentisk autonomi

Å innføre agentisk KI i et heterogent landskap er ikke et IT-prosjekt; det er en **strategisk** omfordeling av **makt** fra menneskelige operatører til algoritmiske aktører. Dere står overfor en risiko for å delegere kontroll til systemer som opererer i en annen tidsskala enn deres eksisterende kontrollmekanismer.

Arkitektur og kontroll: Fra statisk til syklisk styring Glem tradisjonell perimeter-sikkerhet. Dere må bygge en **syklisk** kontroll-loop: Observer → Evaluer → Isoler → Reager. Siden dere mangler OPA, må dere **implementere** en “Sidecar Proxy Architecture” for agenter. Hver agent må tvinges gjennom en kontroll-node (en “Policy Enforcement Point”) som fungerer som en digital sandkasse.

- **Execution Graph:** Modeller agentens vei som en rettet asyklisk graf (DAG). Hvis en agent prøver å skape en loop eller en uforutsett kjedereaksjon, må graf-analysen trigge en umiddelbar circuit breaker.

IAM: Dynamisk maktutøvelse Tradisjonell RBAC er dødt i møte med agenter. Dere må bevege dere mot **Just-in-Time (JIT) Token Scoping**. Agenter skal ikke ha “service accounts” med faste rettigheter, men motta ephemeral credentials som kun eksisterer i validitetstiden for den spesifikke oppgaven (f.eks. 5 minutter). Dette begrenser skadeomfanget ved en “runaway agent”.

Integrasjon og Blindsoner: Risikoen ved latent kobling Den største **muligheten** ligger i automatisering, men den største blindsonen er latent coupling. Agenter som opererer mot legacy-monolitter uten idempotens, vil skape uopprettelige datakorupsjoner. Hvis et API-kall feiler halvveis, og agenten “prøver igjen” uten transaksjonell garanti, skaper dere en destruktiv feedback-loop.



Strategisk implementeringsplan: Ikke prøv å løse alt i piloten. Fokuser på å bygge “skilleveggene” først.

Neste steg:

1. **Etabler en “Gatekeeper Proxy”:** Implementer en enkel sentralisert proxy (f.eks. basert på Envoy) som alle agent-kall må gå gjennom, uavhengig av legacy-status.
2. **Definer “Blast Radius”-profiler:** Kategoriser alle API-er i tre nivåer: Read-only (Lav risiko), Transactional (Middels), og System-altering (Høy). Begrens piloten strengt til nivå 1 og 2.
3. **Audit-sentralisering:** Opprett en dedikert hendelsesstrøm (f.eks. via Kafka eller lignende) som logger både Prompt → Decision → Action, slik at dere kan rekonstruere agentens “tankerekke” post-mortem.