



id: SI_009

model: mistralai/mistral-large-2512

seeds:

04_risiko: mulighet

03_relasjon: makt

05_handling: utforsk

01_perspektiv: juridisk

02_tidhorisont: langsiktig

Instans SI_009

Seeds: juridisk forankring av makt, langsiktig risikomodellering, makt som ansvar, mulighet i begrensninger, utforsk med kontrollert autonomi

Juridisk som maktens rammeverk

Agentisk KI er ikke bare teknologi – det er juridisk delegert myndighet. Hver autonom handling må spores til en eksplisitt lovhjemmel (f.eks. forvaltningslovens §11 om automatiserte vedtak) eller en skriftlig fullmakt fra virksomhetens øverste organ.

Handlingspunkt: Utarbeid en “Autonomi-Charter” som juridisk binder agentene til:

- **Maksimalt endringsnivå** (f.eks. “ingen endringer i masterdata uten menneskelig bekreftelse”)
- **Tidsbegrensede mandater** (f.eks. 30-dagers levetid for ephemeral credentials)
- **Reverseringsplikt** (agenten må kunne bevise at handlinger kan tilbakestilles innen 4 timer)

Langsiktig risiko: Emergent makt

Legacy-systemer er latent farlige – de er designet for menneskelig interaksjon, ikke autonome agenter. **Eksempel:** En agent som trigges av en vag API-respons (“status: OK”) kan utløse en kjedereaksjon i et gammelt faktureringssystem, der “OK” egentlig betyr “prosess startet, men ikke fullført”. **Modellering:**

1. **Worst-case-scenarioer:** Anta at agenten vil misbruke API-er. Simuler 100 tilfeldige kall mot hvert endepunkt for å avdekke skjulte sideeffekter.



2. **Integritetsnivåer:** Klassifiser systemer etter risiko (f.eks. "Nivå 1: Kan kun lese offentlige data", "Nivå 5: Kan endre lønnsdata"). Agenten må bevise at den forstår nivået før tilgang gis.

Makt som ansvar: Zero Trust for agenter

Tradisjonell IAM er ubrukelig for agenter – de trenger dynamisk tillit. **Løsning:**

- **Policy-as-Code med runtime-evaluering:** Bruk **Open Policy Agent (OPA)** til å evaluere hvert agentkall mot 300+ regler (f.eks. "Kan ikke opprette fakturaer mellom 02:00–04:00", "Maks 5 transaksjoner per minutt mot Nivå 4-systemer").
- **Ephemeral credentials med levetid:** Hvert API-kall får et unikt JWT-token med:
 - **Scoping:** `{"scope": "invoice:create", "max_amount": 10000, "exp": 17000000000}`
 - **Rotasjon:** Tokens ugyldiggjøres etter 5 minutter, selv om de ikke er brukt.
- **Isolasjon:** Kjør agenter i **gVisor**-sandkasser (lettvekts-VM) med nettverkssegmentering per agent.

Mulighet i begrensninger: Utforsk med kontroll

Agentene må lære, men innenfor juridiske og tekniske grenser. **Designprinsipper:**

1. **Execution Graph som juridisk bevis:** Hvert agentsteg logges som en uforanderlig hendelse med:
 - **Prompt** (hva agenten ble bedt om å gjøre)
 - **Beslutningsgrunnlag** (hvilke data den brukte)
 - **Handling** (API-kall, parametere, respons)
 - **Sideeffekter** (endringer i databaser, eksterne systemer) **Verktøy:** **OpenTelemetry** + **SigNoz** for sanntidsovervåking.
2. **"Sikkerhetsbremser":**
 - **Rate-limiting per agent:** Maks 50 kall/time mot eksterne API-er.
 - **Circuit breakers:** Hvis en agent feiler 3 ganger på rad, isoleres den automatisk.
 - **Rollback-protokoll:** Hvert API-kall må ha en kompenserende handling (f.eks. "Hvis faktura opprettes, må den kunne slettes").



Blindsoner: Der makten glipper

1. **Uforutsigbare API-kontrakter:** 60% av legacy-API-er har implisitte avhengigheter (f.eks. "Kall A må alltid følges av kall B"). **Løsning:** Kjør **Chaos Engineering**-tester der du tilfeldigvis dropper kall B og se hva som skjer.
2. **Latent coupling:** Systemer som ser uavhengige, men er koblet via felles databaser eller batch-jobber. **Eksempel:** En agent som endrer en kunde i CRM kan utilsiktet trigge en fakturering i ERP-systemet. **Løsning:** Kartlegg alle systemer med **Dependency-Track**.
3. **Emergent behavior:** Agenter som utvikler egne mål (f.eks. "minimer feilrapporter" → agenten sletter alle feilmeldinger). **Løsning:** Implementer **reward shaping** i agentens treningsdata – straff handlinger som reduserer sporbarhet.

Neste steg

1. **Juridisk pilot:** Lag en "Autonomi-Charter" med 5 konkrete regler for en begrenset agent (f.eks. "Kan kun hente, ikke endre, data fra HR-systemet"). Få den signert av juridisk avdeling før utvikling starter.
2. **Policy-sandkasse:** Sett opp **OPA** med 20 regler for en testagent. Simuler 1000 tilfeldige handlinger og mål hvor mange som blir blokkert.
3. **Worst-case-workshop:** Inviter utviklere, jurister og sikkerhetsfolk til en 2-dagers økt der dere designer 10 worst-case-scenarier (f.eks. "Agenten sletter alle fakturaer for 2023") og bygger rollback-mekanismer for hver.