



id: SI_007

model: mistralai/mistral-large-2512

seeds:

04_risiko: sårbarhet

03_relasjon: makt

05_handling: utforsk

01_perspektiv: strategisk

02_tidhorisont: kortsiktig

Instans SI_007

Agentisk KI i et sårbart maktlandskap: En utforskende strategi for kortsiktig kontroll

Dere står foran en **maktforskyvning** – autonome agenter vil operere som uoffisielle systemadministratorer uten menneskelig mellomledd. I et landskap preget av **sårbarheter** (legacy-systemer, svak IAM, manglende policy-motor) blir denne makten en **kortsiktig risiko** med potensielt irreversible konsekvenser. Min analyse fokuserer på å **utforske** grenseflatene mellom autonomi og kontroll, der hver teknisk beslutning må balansere **strategisk langsiktighet** med umiddelbare tiltak.

Arkitektur: Sårbarhet som designprinsipp

1. Zero Trust som nødbrems, ikke arkitektur

- Implementer **runtime-policyer** (f.eks. OPA) som midlertidig stopper agenter ved mistenkelig aktivitet (f.eks. 5+ API-kall/sekund, endringer i sensitive databaser).
- Eksempel: Hvis en agent forsøker å oppdatere personopplysninger utenfor definerte tidsvinduer (08-16), blokkeres handlingen og et varsel sendes til et menneskelig team.

2. Execution Graph som sårbarhetskart

- Modellér agentens handlinger som en **graf med risikonoder** (f.eks. "endring i økonomisystem" = høy risiko, "lesing av værdata" = lav risiko).



- Konkret: Bruk **OpenTelemetry** til å spore agentens sti gjennom systemet. Hvis grafen viser uventede hopp (f.eks. fra HR-system til betalings-API), utløses en “pause-og-godkjenn”-mekanisme.

3. Safe Execution Environments (SEE) for kortsiktig isolasjon

- Kjør agenter i **ephemeral containers** (f.eks. Kubernetes + gVisor) med:
 - **Tidsbegrensning** (max 30 min per oppgave)
 - **Ressursbegrensning** (CPU/memory)
 - **API-gateway som proxy** (alle kall går via en gateway som logger og validerer før videreformidling)

IAM: Maktbegrensning gjennom dynamisk sårbarhet

- **Ephemeral credentials med “sårbarhetsvindu”**
 - Gi agenter **tidsbegrensede tokens** (f.eks. 15 min) som automatisk roteres.
 - Eksempel: Bruk **HashiCorp Vault** til å utstede tokens med begrenset scope (f.eks. kun lesetilgang til én database).
- **ABAC-policyer som maktbalanse**
 - Definer **kontekstuelle regler** (f.eks. “agent X kan kun endre data i system Y mellom kl. 12–13 hvis Z-sensor er aktiv”).
 - Konkret: Bruk **OPA/Rego** til å skrive policyer som evaluerer både hva agenten gjør og når det skjer.

API-sikkerhet: Utforskning av uforutsigbarhet

- **API-gateway som sårbarhetsfilter**
 - Implementer **rate-limiting** (f.eks. 10 kall/minutt per agent) og **circuit breakers** (stopp agenten hvis 3 feilkall på rad).
 - Eksempel: Hvis en agent gjør et ikke-idempotent kall (f.eks. “send faktura”), krev **menneskelig godkjenning** via en Slack-bot.
- **Syntetiske API-tester for emergent behavior**
 - Kjør **chaos engineering**-tester (f.eks. med **Gremlin**) for å simulere agentfeil (f.eks. uendelige loops, ugyldige dataendringer).



Logging: Sporbarhet som maktredskap

- **Audit-pipeline med “sårbarhetsmarkører”**

- Logg alt (prompts, API-kall, beslutningsgrunnlag) i en **immuterbar ledger** (f.eks. **Amazon QLDB**).
- Konkret: Merk hver handling med:
 - **Risikonivå** (lav/medium/høy)
 - **Menneskelig godkjenningsstatus** (godkjent/avvist/ventende)
 - **Tidsstempel + agent-ID**

- **Automatisk isolering ved avvik**

- Hvis en agent bryter policyer (f.eks. endrer data utenfor scope), **frys den umiddelbart** og send varsel til et **SOC-team**.

Neste steg: Utforsk sårbarhetene før de utforsker dere

1. Pilot med “sikkerhetsbur”

- Velg én lavrisiko-prosess (f.eks. automatisk henting av værdata) og kjør agenten i et **isolert miljø** med:
 - OPA for policy-evaluering
 - Vault for ephemeral credentials
 - OpenTelemetry for logging
- Mål: Identifiser 3 uforutsette sårbarheter innen 4 uker.

2. Modellér worst-case-scenarier

- Lag en **risikomatrix** for emergent behavior (f.eks. “agent A endrer priser i system B → system C trigger automatiske bestillinger”).
- Eksempel: Bruk **Monte Carlo-simulering** til å estimere sannsynligheten for kjedereaksjoner.

3. Bygg en “kill switch”

- Implementer en **sentralisert “pause”-knapp** som stopper alle agenter umiddelbart ved mistanke om eskalering.
- Krav: Knappen må være tilgjengelig for minst 3 personer og testes ukentlig.

Avslutning: Dere står ikke bare foran en teknisk utfordring, men en **maktkamp** mellom autonomi og kontroll. Den kortsiktige strategien må være å **utforske sårbarhetene før**



agentene gjør det – ved å designe systemer som forventer feil, ikke unngår dem. Makt er ikke noe dere gir agentene; det er noe dere låner ut, med strenge vilkår.