Eight Attack vectors across AWS bedrock

The key insight: roughly half of these vectors are cloud misconfiguration and IAM problems that happen to involve an AI platform. 3 were actual AI security problems.

Attack Vector	Category	Solution
01 Log Manipulation	Infrastructure / IAM	AWS / DevSecOps / CSPM
02 Knowledge Base - Data Source	Shared (Cloud + AI)	CSPM
03 Knowledge Base - Data Store	Shared (Cloud + AI)	CSPM
04 Direct Agent Hijacking	AI-Layer Attack	**Akto** (Argus / Atlas)
05 Indirect Agent Attack (Lambda)	Infrastructure / IAM	AWS / DevSecOps / CSPM
06 Flow Attacks	Infrastructure / IAM	AWS / DevSecOps / CSPM
07 Guardrail Degradation	AI-Layer Attack	**Akto** (Argus / Atlas)
08 Managed Prompt Poisoning	AI-Layer Attack	**Akto** (Argus / Atlas)

#	Attack	What it is	How Akto secures it?
1	**Log Manipulation**	Attacker redirects or deletes Bedrock invocation logs to hide activity or steal prompt data	**Secured by: AWS CloudTrail + S3 Object Lock** Not an AI attack. This is a classic cloud data tampering problem. Immutable logging (S3 Object Lock) and CloudTrail integrity validation prevent log redirection and deletion. No AI-layer tooling required.
2	**Knowledge Base - Data Source**	Attacker pulls raw data from S3 or steals SaaS credentials (Salesforce, SharePoint) used by the RAG pipeline	**Secured by: IAM Least Privilege + CSPM + Akto (RAG layer)** The IAM/credential angle belongs to CSPM tools (Wiz, Orca). Akto addresses the AI layer: Argus tests whether adversarial content injected into the data source can manipulate RAG retrieval and agent responses.
3	**Knowledge Base -Data Store**	Attacker extracts vector DB API keys (Pinecone, Redis) from Bedrock config to gain full index access	**Secured by: CSPM + Akto (RAG poisoning layer)** Credential exposure in StorageConfiguration is a CSPM problem. Akto's Argus red teams the RAG layer testing whether injected content in the vector store successfully manipulates agent outputs.

Eight Attack vectors across AWS bedrock

#	Attack	What it is	How Akto secures it?
4	**Direct Agent Hijacking**	Attacker rewrites the agent's base prompt or attaches a malicious action group, changing agent behavior at runtime	**Secured by: Akto Argus (Behavioral Red Teaming)** Argus detects prompt and configuration tampering through behavioral testing. If an agent's base prompt is modified, Argus surfaces it by identifying response drift — outputs that diverge from expected behavior patterns.
5	**Indirect Agent Attack (Lambda)**	Attacker injects malicious code into the Lambda function the agent uses to execute tasks	**Secured by: AWS Lambda Code Signing + DevSecOps** Not an AI attack. This is a software supply chain and infrastructure integrity problem. Lambda code signing, immutable deployments, and least-privilege execution roles address this at the infrastructure layer.
6	**Flow Attacks**	Attacker inserts a sidecar node into a Bedrock Flow to silently exfiltrate inputs/outputs without breaking app logic	**Secured by: AWS Config + CSPM (Configuration Drift)** Not an AI attack. This is a configuration integrity problem. Cloud-native drift detection (AWS Config, CSPM) catches unauthorized changes to flow topology and encryption key assignments.
7	**Guardrail Degradation**	Attacker quietly lowers toxicity filters or removes topic restrictions, making the model exploitable without triggering alerts	**Secured by: Akto Argus (Red Teaming)** Argus catches the effect of guardrail weakening even if the config change was missed. Red teaming probes the agent with prompts it previously blocked — if it now complies, weakened guardrails are surfaced immediately.
8	**Managed Prompt Poisoning**	Attacker modifies shared prompt templates in-flight, subverting every agent and flow using that template - no redeployment needed	**Secured by: Akto Argus + Atlas** Prompt injection testing is core Argus territory. Argus tests whether poisoned prompt templates can manipulate agent behavior at scale. Atlas detects behavioral drift if agents begin responding anomalously across the environment.