# Amazon Bedrock AgentCore

## A Field Guide for AWS Builders

Rowan Udell - AWS Security Hero & Consultant

AWS Brisbane Usergroup, April 2026

linkedin.com/in/rowanu

# Hi, I'm Rowan 👋

- **AWS Security Hero**
- Independent consultant
- Using AWS for **15+ years**
- Helping businesses **build and secure agents**

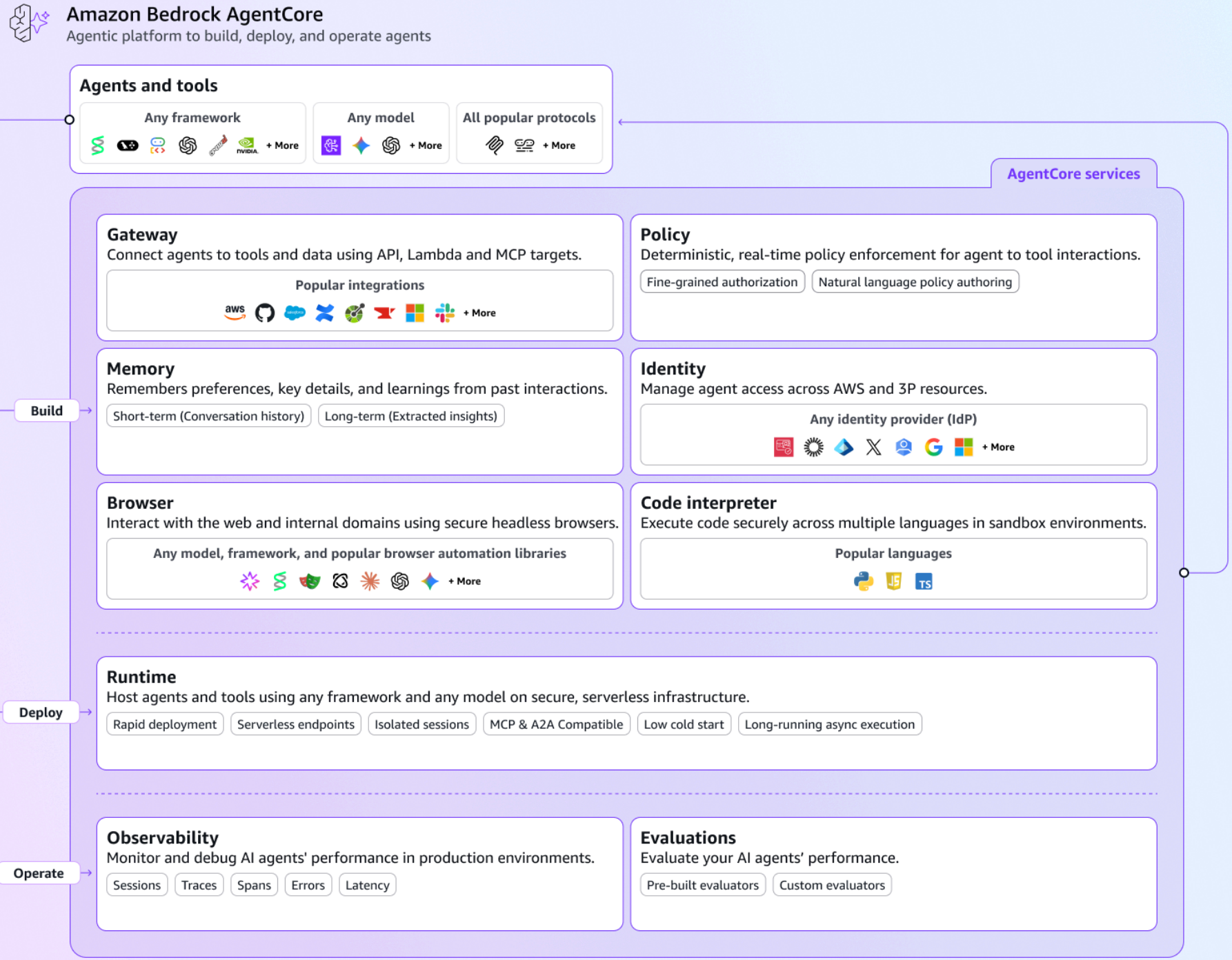
# The Problem with AI Agents Today

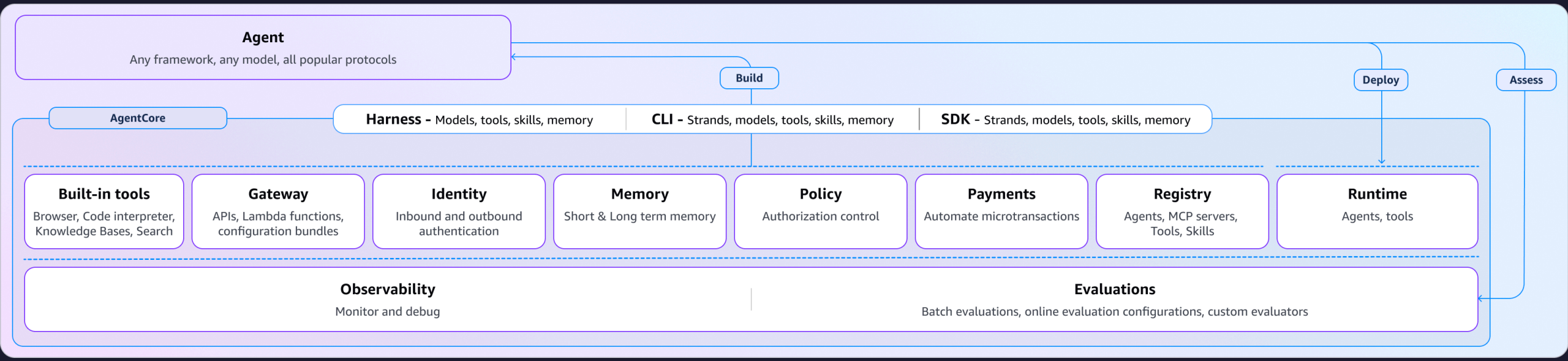
- Prototypes are easy. **Production is hard.**
- Every team re-invents the same things:
  - Hosting and scaling agent code
  - Memory and session management
  - Authentication and authorization
  - Tool integration
  - Observability and evaluation
- AgentCore is the **platform layer** between your agent code and production.

# What is Amazon Bedrock AgentCore?

A suite of **10 composable services** for building, running, and governing AI agents.

- Not a new framework. It's **infrastructure for agents**
- **Framework-agnostic**: LangGraph, CrewAI, Strands, custom code
- **Model-agnostic**: Bedrock, Claude, OpenAI, Gemini, whatever
- Use what you need, skip what you don't


Amazon Bedrock AgentCore
Agentic platform to build, deploy, and operate agents
Agents and tools
Any framework
+ More
Any model
+ More
All popular protocols
+ More
AgentCore services
Gateway
Connect agents to tools and data using API, Lambda and MCP targets.
Popular integrations
+ More
Policy
Deterministic, real-time policy enforcement for agent to tool interactions.
Fine-grained authorization
Natural language policy authoring
Build
Memory
Remembers preferences, key details, and learnings from past interactions.
Short-term (Conversation history)
Long-term (Extracted insights)
Identity
Manage agent access across AWS and 3P resources.
Any identity provider (IdP)
+ More
Browser
Interact with the web and internal domains using secure headless browsers.
Any model, framework, and popular browser automation libraries
+ More
Code interpreter
Execute code securely across multiple languages in sandbox environments.
Popular languages
Deploy
Runtime
Host agents and tools using any framework and any model on secure, serverless infrastructure.
Rapid deployment
Serverless endpoints
Isolated sessions
MCP & A2A Compatible
Low cold start
Long-running async execution
Operate
Observability
Monitor and debug AI agents' performance in production environments.
Sessions
Traces
Spans
Errors
Latency
Evaluations
Evaluate your AI agents' performance.
Pre-built evaluators
Custom evaluators


Agent
Any framework, any model, all popular protocols
Build
Deploy
Assess
AgentCore
Harness - Models, tools, skills, memory
CLI - Strands, models, tools, skills, memory
SDK - Strands, models, tools, skills, memory
Built-in tools
Browser, Code interpreter, Knowledge Bases, Search
Gateway
APIs, Lambda functions, configuration bundles
Identity
Inbound and outbound authentication
Memory
Short & Long term memory
Policy
Authorization control
Payments
Automate microtransactions
Registry
Agents, MCP servers, Tools, Skills
Runtime
Agents, tools
Observability
Monitor and debug
Evaluations
Batch evaluations, online evaluation configurations, custom evaluators

## What AgentCore Is Not

- Not the `agentcore` CLI
- Not **Bedrock Agents**
  - AgentCore = infrastructure; Bedrock Agents = opinionated orchestration
- Not a new agent **framework**: bring your own
- Not **limited to Bedrock models**: works with any LLM
- Not a monolith: each service is **independently useful**

## Release Timeline

- **Jul 2025**: Preview launch (4 regions)
  Runtime, Memory, Gateway, Browser, Code Interpreter, Observability, Identity
- **Oct 2025**: GA (9 regions)
  VPC support, A2A protocol, MCP server connectivity
- **Dec 2025**: Policy & Evaluations *(preview)*
  Episodic memory, bidirectional streaming
- **Mar 2026**: Policy GA *(13 regions)*, Evaluations GA *(9 regions)*
- **Apr 2026**: Registry *(preview, 5 regions)*

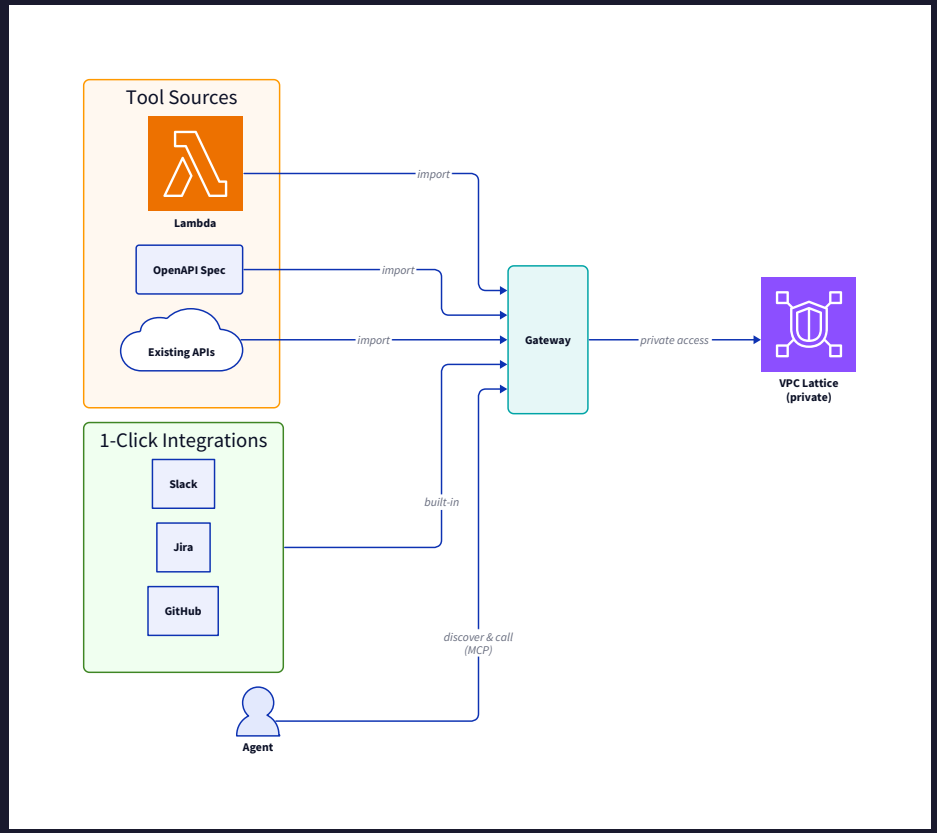
# Build Your Agent

Gateway · Policy · Memory · Identity · Browser · Code Interpreter

# Gateway

Turn APIs into **MCP-compatible tools**, without code.

- **Centralized & secure**: VPC Lattice, built-in auth
- **Semantic tool discovery**: find the right tool
- **Import from anywhere**: Lambda, OpenAPI, Smithy
- **Credential injection**: per-tool auth
- **Composition**: multiple APIs in a single MCP
- **1-click integrations**: Slack, Jira, GitHub, Salesforce, Zendesk

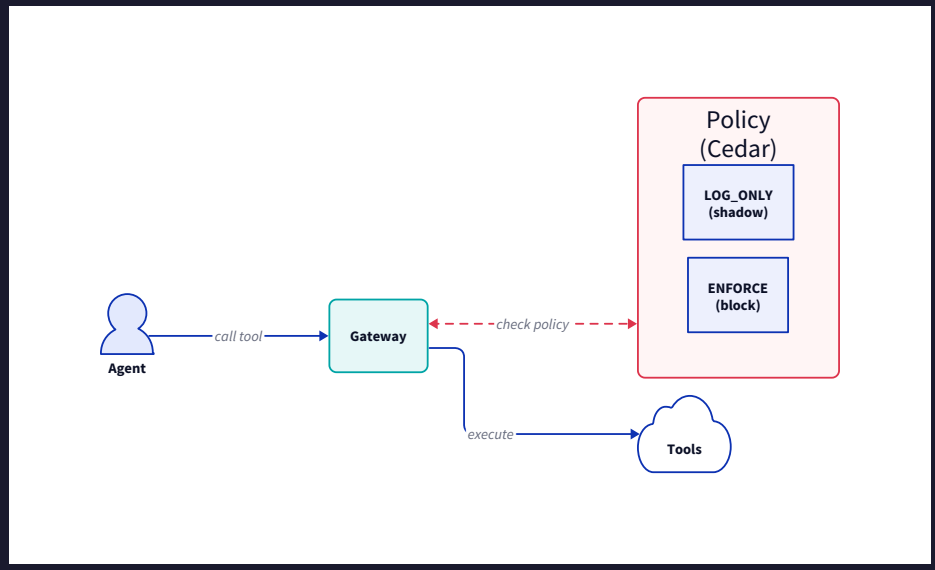


# Policy

Fine-grained access control using **Cedar**, enforced *outside* agent code.

```
forbid(
  principal,
  action == AgentCore::Action::"PaymentTools__transfer_funds",
  resource
) when { context.amount > 10000 };
```

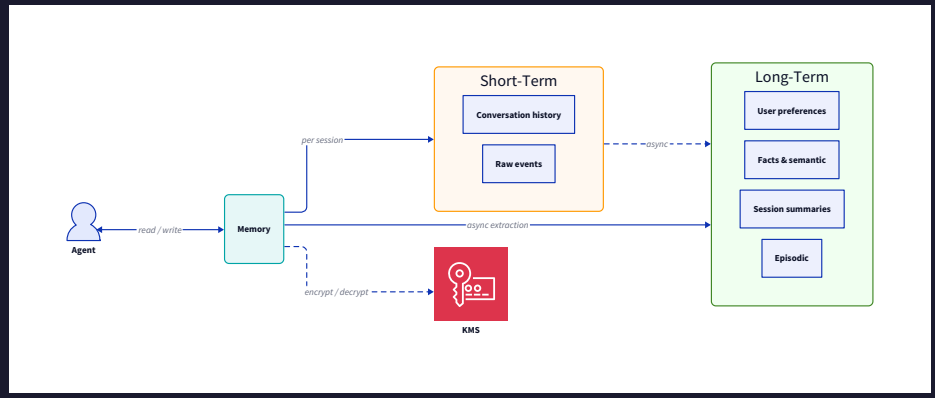
- **Declarative**: outside prompts and agent logic
- **Granular interception**: gateway/tool/operation/etc
- **Safe rollout**: LOG_ONLY, then ENFORCE
- **Schema validation**: generates from Gateway tools
- **Natural language authoring**



# Memory

Give agents the ability to **remember**.

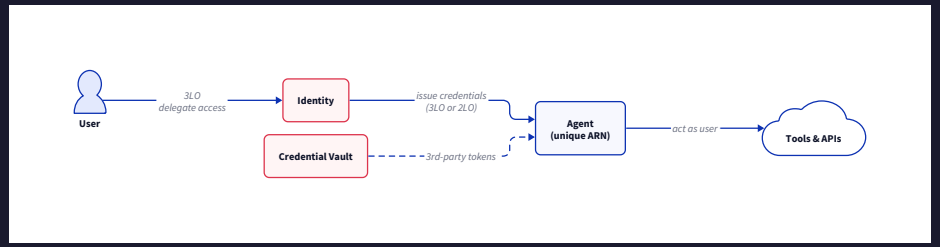
- **Managed**: no DDB tables, encrypted with KMS
- **Short-term**: session context, conversation history
- **Long-term**: preferences, facts, semantic (async)
- **Multi-agent sharing**



# Identity

First-class identity for agents, not just IAM roles.

- **Multi-credential**: OAuth2 tokens, API keys, client certs, SAML, custom tokens
- **OAuth flows**: client credentials (2LO) + authorization code (3LO)
- **Helpful SDK**: annotations reduce boilerplate, automatic token refresh
- **Request verification**: signature/expiry/scope validated on every call
- **Audit trail**: credential access logged



## Browser

Sandboxed web browsing agents can use at runtime.

- **Isolated Chromium**: one microVM per session, full isolation
- **Playwright-based**: automation via WebSocket streaming API
- **Live View**: real-time monitoring powered by AWS DCV — embed in your app
- **Session recording**: replay saved to S3 for audit and debugging
- **Extensible**: browser extensions, profiles, proxies, enterprise policies
- **Configurable TTL**: sessions auto-terminate after timeout

# Code Interpreter

Sandboxed code execution agents can use at runtime.

- **Multi-language**: Python, JavaScript, TypeScript — common libraries pre-installed
- **Sandboxed**: containerized execution, isolated per session
- **Long-running**: default 15 min, extendable up to 8 hours
- **Data processing**: CSV, Excel, JSON — cleaning, analysis, visualisation
- **Internet access**: agents can fetch external data during execution (opt-in)
- **Framework integration**: Strands, LangChain, or direct SDK/boto3
- **CloudTrail logging**: code execution is auditable

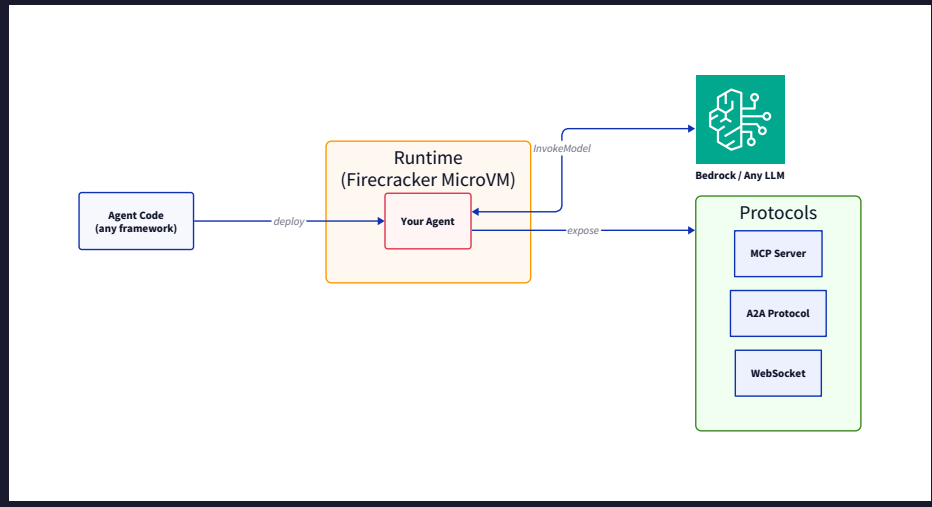
# Deploy Your Agent

Runtime · Registry

# Runtime

Serverless hosting for AI agents. No infrastructure to manage.

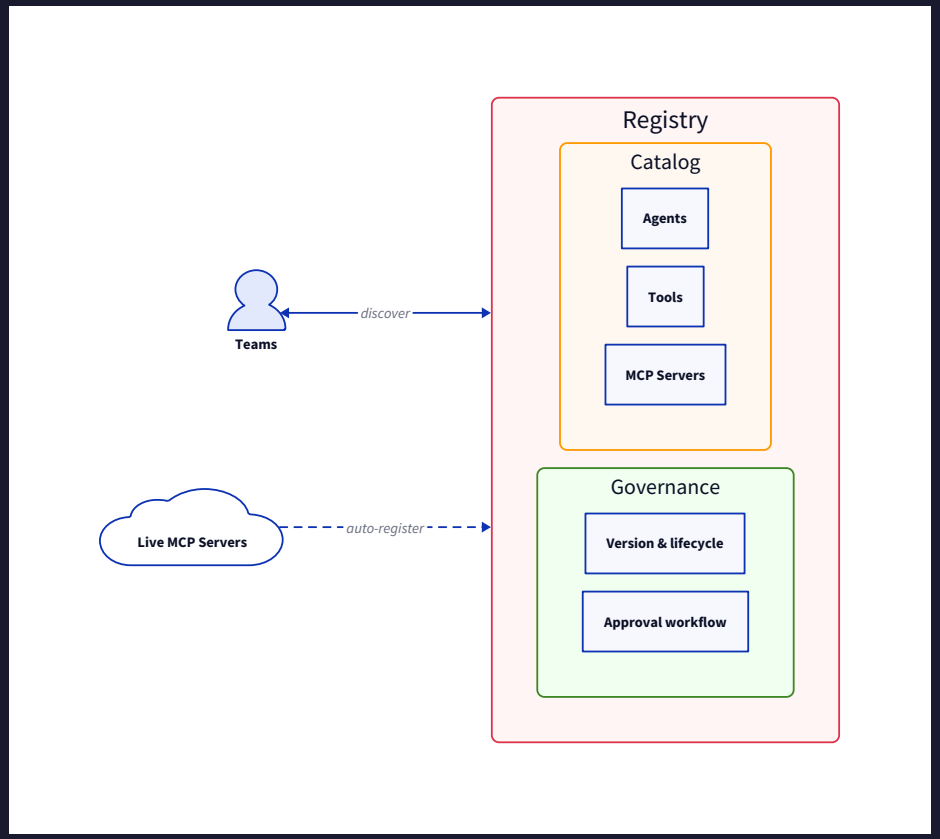
- **Consumption-based pricing**: pay only for resources used
- **MicroVM isolation**: Firecracker, up to **8 hours**
- **Multi-protocol**: MCP, A2A, HTTP, WebSockets
- **Shell execution**: same container as agent
- **Persistent filesystem**



# Registry

Centralized discovery and governance for your agent estate.

- **Catalog**: agents, tools, MCP servers, custom resources, with MCP-native access
- **Auto-discovery**: URL-based sync from live MCP servers and agent endpoints
- **Hybrid search**: natural language + keyword matching
- **Governance**: ownership, versioning, lifecycle metadata, approval workflows (+ EventBridge)
- **Flexible auth**: IAM or custom JWT for search and MCP invoke



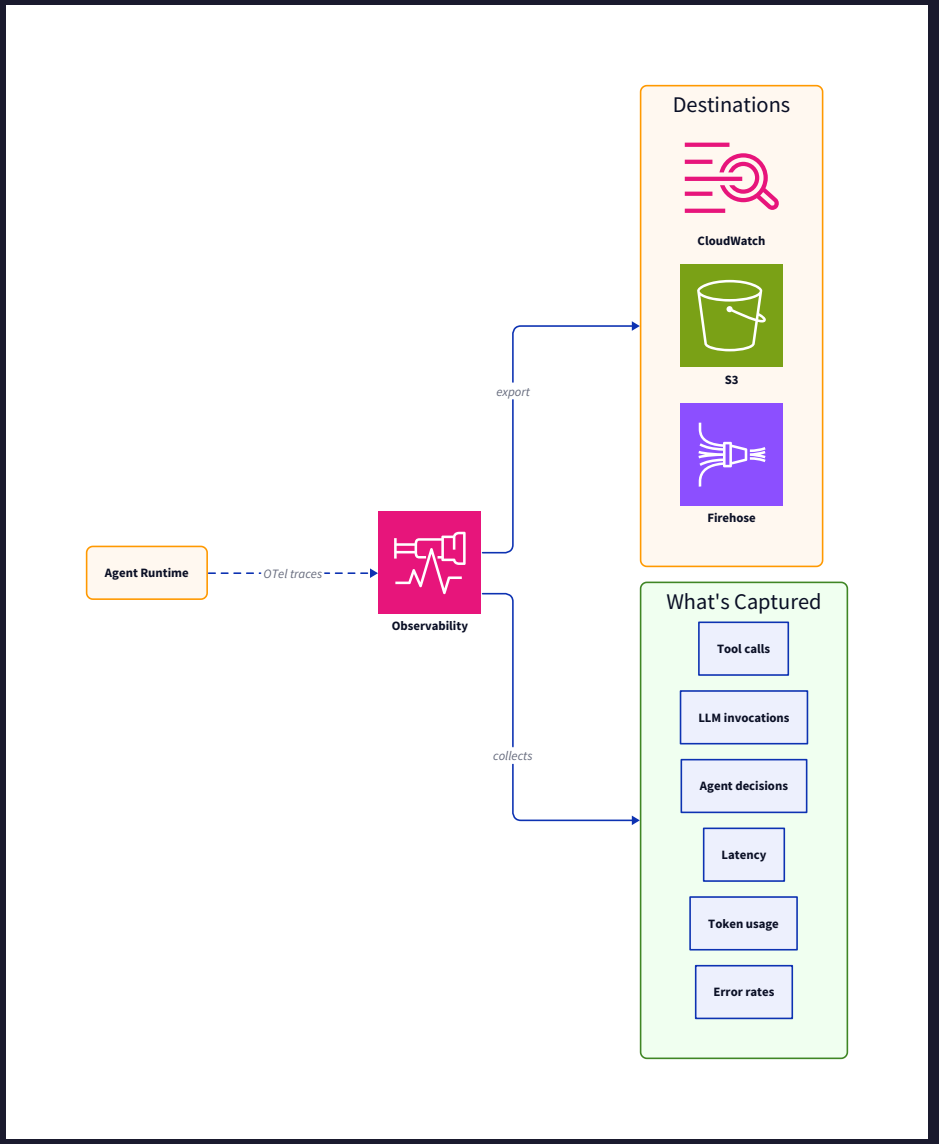
# Operate Your Agents

Observability · Evaluations

# Observability

See what your agents are **actually doing**.

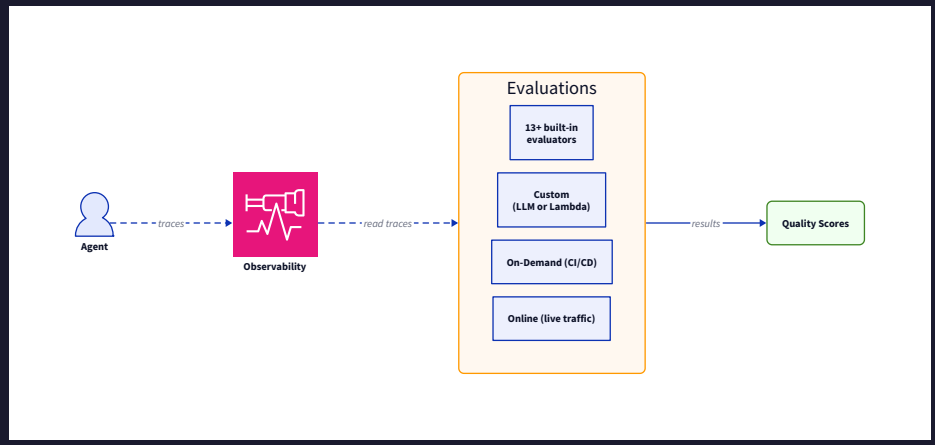
- **OpenTelemetry-compatible**: tracing to Amazon CloudWatch
- **Agent-aware**: tool calls, LLM invocations, latency, token usage, error rates
- **Auto-instrumented + extensible**: spans for Gateway, Memory, Tools, Policy out of the box
- **Multi-destination**: CloudWatch, S3, or Firehose
- **Not just Runtime**: agents hosted anywhere
- **ADOT SDK**: for custom instrumentation



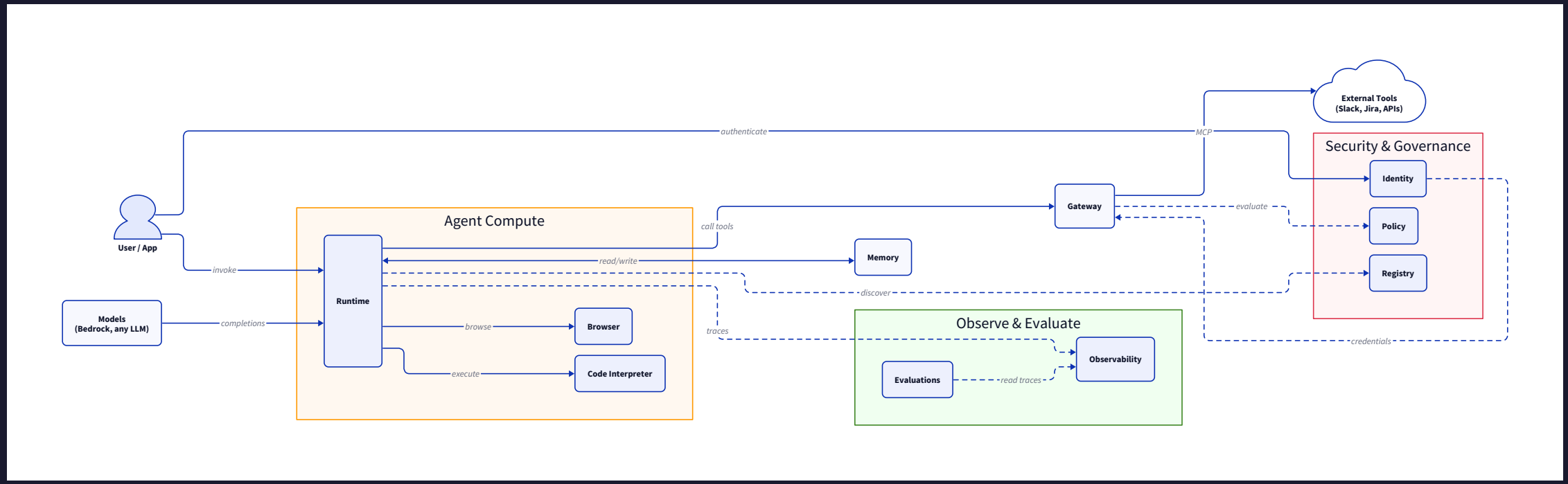
# Evaluations

Measure agent quality **systematically**.

- **LLM-as-a-Judge**
- **13+ built-in evaluators**: correctness, faithfulness, relevance, toxicity
- **Two modes**: on-demand (CI/CD) and online
- **Custom evals**: your own LLM or Lambda based
- **Reference answers**: compare outputs against expected answers and tool sequences
- **Session + trace level**: score individual turns or entire conversations
- **Framework integration**: Strands



# How the Services Fit Together



## Getting Started

Start small. You don't need all 10 services on day one.

Minimum viable agent stack:

1. **Runtime** to deploy your existing agent code
2. **Memory** when you need cross-session context
3. **Gateway** to connect it to tools

Then layer on:

- **Identity** when users need delegated access
- **Observability** to see what it's doing
- **Policy** when you need tool access control (start with LOG_ONLY)
- **Evaluations** before you promote to production

## Key Takeaways

1. AgentCore is **infrastructure for agents**, not another framework
2. **10+ services** that are individually useful and composable
3. Start with **Runtime + Memory + Gateway**
4. Use **Policy in LOG_ONLY** mode before enforcing
5. **Identity propagation** avoids the "agent uses a shared service account" anti-pattern
6. Treat agent security like application security, because it is (with some other stuff)

# Thanks!

Any questions?

**Slides** slider.rowanudell.com 🍔
**Amazon Bedrock Agentcore** Documentation
**Cedar playground** cedarpolicy.com
**Strands SDK** on GitHub
**AWS re:Post** AgentCore tag