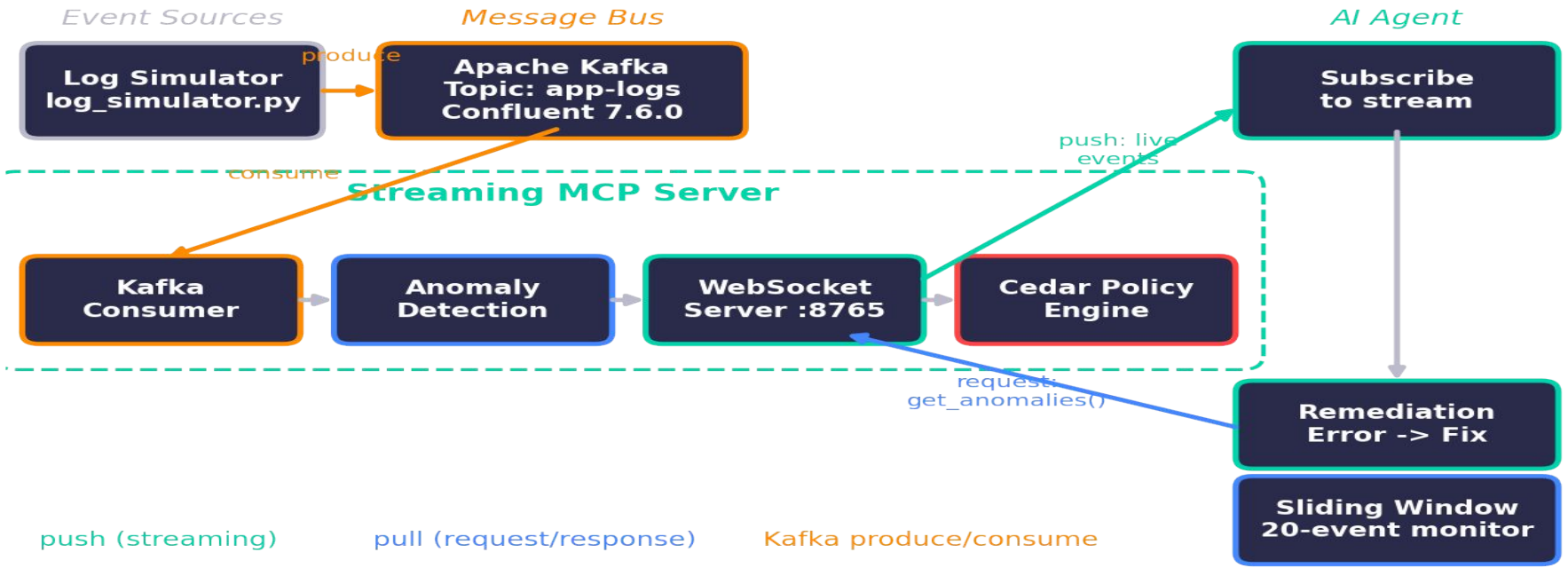
MCP Dev Summit North America

MCP Live: Streaming Context to AI Agents

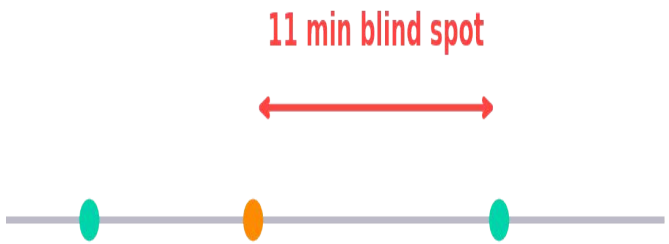
Speaker: Harshit Kohli
Sr Technical Account Manager
Amazon Web Services

System Architecture



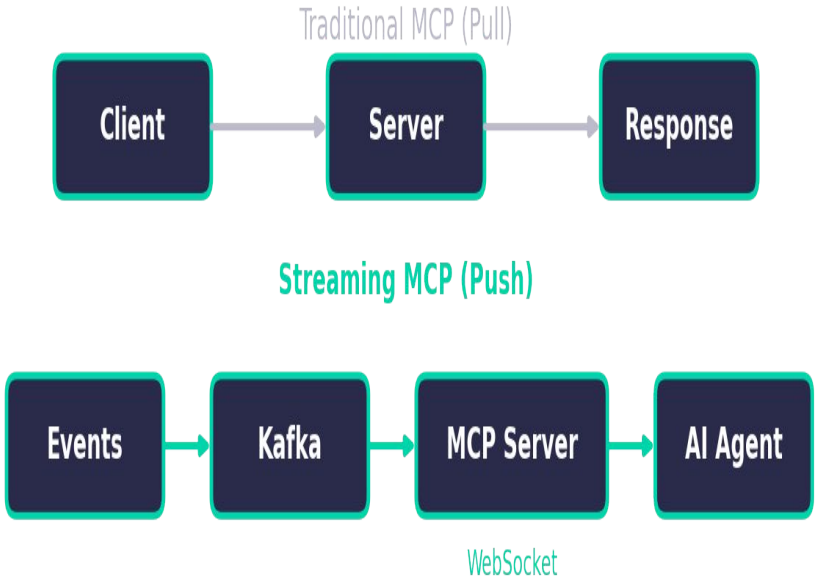
The Stale Context Problem

- Traditional MCP = request/response snapshots
- Agent asks for logs → gets point-in-time dump → stale data
- Deploy at 2:03pm, errors at 2:04pm, agent polls at 2:15pm
- What if context came TO the agent, as it happened?



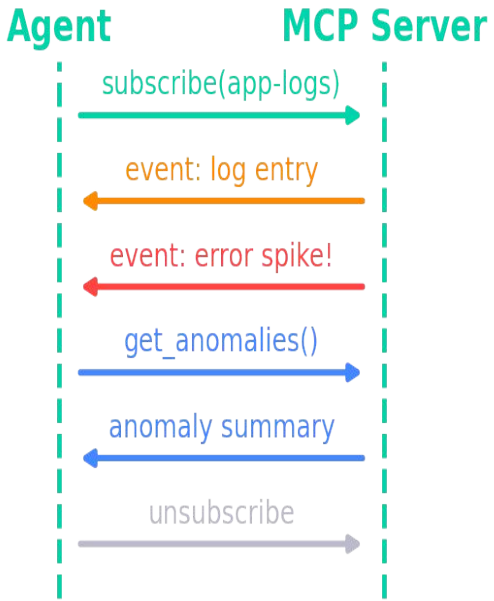
Streaming MCP Architecture

- Standard MCP: Client → Server → Response (pull model)
- Streaming MCP: Server → Client via subscriptions (push model)
- Why Kafka: durable, replayable, backpressure handling
- Hybrid protocol: streaming + request/response on same connection



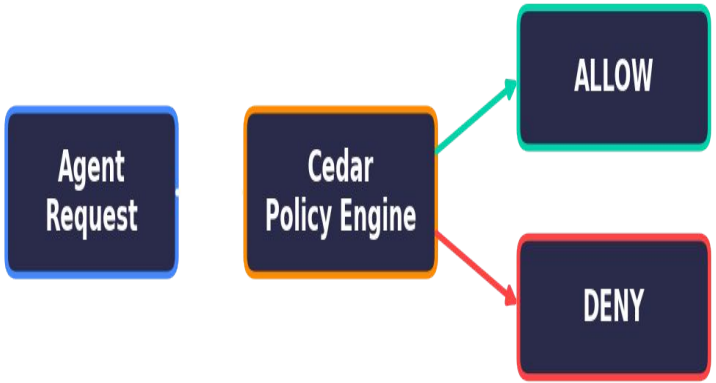
Key Protocol Extension: subscribe

- New method: subscribe / unsubscribe for continuous push
- Still supports get_anomalies, get_context (request/response)
- Bounded queues (500 events) handle backpressure automatically



Cedar Authorization for MCP

- Open-source policy language by AWS
- Declarative: permit/forbid(principal, action, resource)
- Role-based access control for MCP actions
- Default deny — no matching policy = request denied



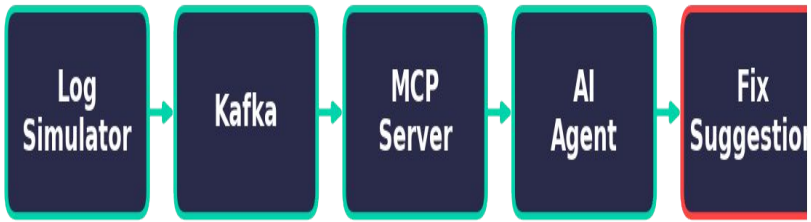
Default Deny — no matching policy = blocked

Cedar Policy Examples

- permit(principal, action == Action::"subscribe", resource == Stream::"app-logs");
- permit(principal, action == Action::"get_anomalies", resource)
  when { principal.role == "ops-agent" };
- forbid(principal, action == Action::"subscribe", resource == Stream::"audit-logs")
  when { principal.role == "readonly" };
- Forbid always wins over permit — secure by default

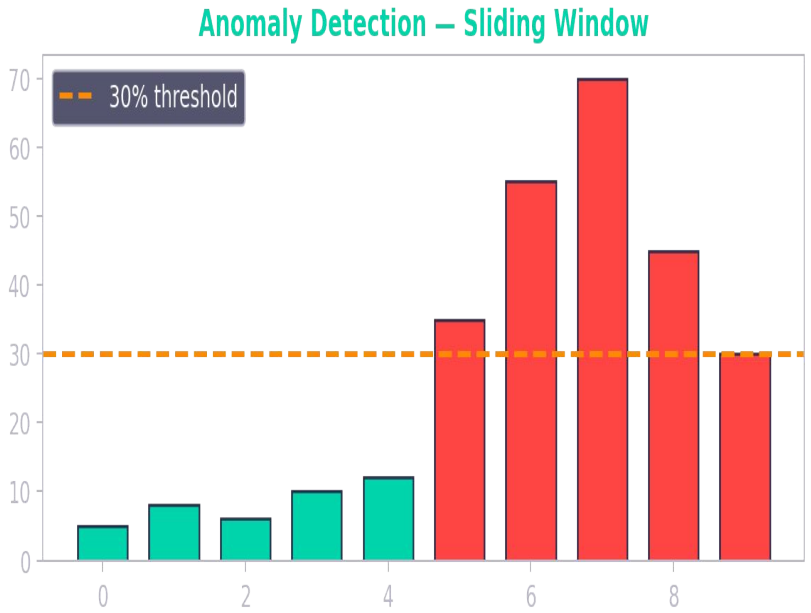
Live Demo: Real-time Log Monitoring

- Log simulator → Kafka → MCP Server → AI Agent
- Error spike → agent detects anomaly within 1 second
- Agent queries for anomaly summary mid-stream
- Cedar blocks unauthorized agents from sensitive streams



Anomaly Detection Flow

- Rolling window of last 100 events maintained server-side
- Error rate > 30% triggers anomaly alert
- Agent maps error patterns to fix suggestions automatically

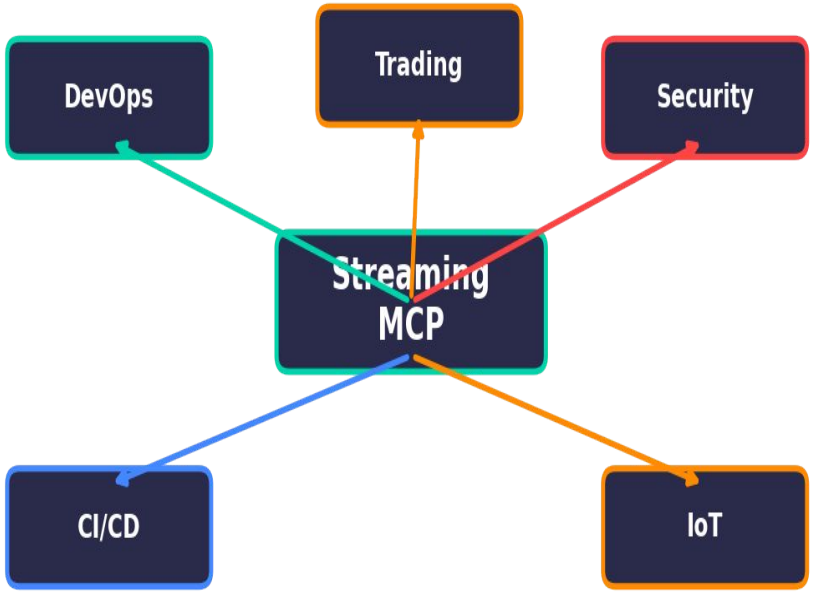


Production Patterns & Gotchas

- 1. Backpressure: bounded queues, drop slow subscribers
- 2. Hybrid protocol: streaming AND on-demand on same WebSocket
- 3. Kafka consumer groups: same = load balance, different = fan-out
- 4. Reconnection: exponential backoff + context snapshot
- 5. Be selective: stream errors & anomalies, batch normal metrics

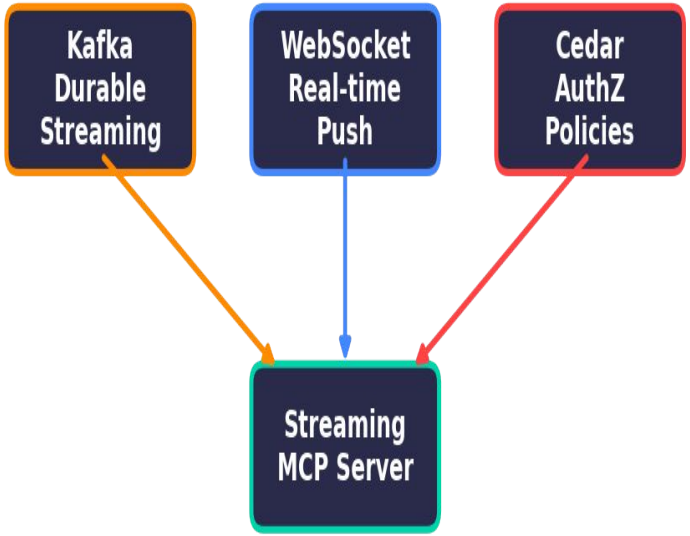
What Streaming MCP Unlocks

- DevOps – real-time incident detection and auto-remediation
- Trading – live market data feeding AI decision agents
- Security – streaming audit logs with Cedar-authorized access
- CI/CD – file watchers notifying agents of code changes
- IoT – sensor data streaming to predictive maintenance agents



Key Takeaways

- MCP doesn't have to be request/response only
- Kafka (Confluent 7.6.0) + WebSocket = durable, scalable streaming bridge
- Cedar provides declarative, auditable authorization for MCP
- Hybrid protocol (push + pull) on single connection is the sweet spot
- All open source — Python 3.9+, Docker, ~200 lines of code



Hybrid Push + Pull on a Single Connection

MCP
Dev Summit
North America

Demo

MCP Dev Summit North America

Thank You

Email: kohli6@gmail.com

LinkedIn: