System Architecture: The "Go-CSPM" Framework

The system is designed to bridge the gap between static configuration security and dynamic runtime behavior. By using Go, you create a high-performance orchestrator that manages five critical layers:

1. Static Policy Engine (Shift-Left)

The Go CLI acts as a gatekeeper in the CI/CD pipeline.

- **Action:** Scans Kubernetes manifests and Dockerfiles for security "red flags" like privileged containers or missing resource limits.
- **Goal:** Catch misconfigurations before they reach the cloud.

2. Live Stream Monitor (Runtime)

Once the application is running in the local Kubernetes cluster (Minikube/Kind), the tool switches to monitoring mode.

- **Action:** It consumes real-time security events from **Falco**.
- **Go Advantage:** You can use Go routines to concurrently process high volumes of network, file, and system call data without performance lag.

3. AI Behavioral Analyser

This layer moves beyond simple rules to detect "unknown" threats.

- **Action:** The Go tool sends extracted behavioral features to a **Scikit-learn** model.
- **Goal:** Compare live activity against a baseline of "normal" behavior to identify active attacks or anomalies.

4. Smart Forensic Vault

Instead of dumping massive log files, the system uses "Policy-Aware Retention".

- **Action:** When the AI flags an anomaly, the Go tool selectively captures high-fidelity forensic evidence (like exact system calls and network headers).
- **Benefit:** Reduces noise and preserves only what is necessary for investigation.

5. Automated Investigator

The final output is a human-readable forensic report.

- **Action:** Generates structured summaries that explain what happened, the severity, and suggested remediation steps.