Kennedy Mosoti

Observability Platform Engineer · Infrastructure Automation · AI Tooling Experiments

Dallas-Fort Worth, TX
kennedy.rmosoti@gmail.com
github.com/kmosoti

> I work closest to the platform layer: Splunk, Logstash, SaltStack, Linux, RCA/remediation, and DR readiness. I am also interested in AI agents because I think the useful version is not magic. It is better tooling, better context, and fewer vague handoffs.

Fit Map

OBSERVABILITY PLATFORM OPERATIONS

Splunk Enterprise, SHC/indexer support, Logstash ingestion paths, Kafka topic cleanup, RCA/remediation

INFRASTRUCTURE AUTOMATION

Python, SaltStack, Bash, Git, Terraform, REST APIs, Linux, release-oriented support work

DR / RESILIENCE WORK

CM/deployer parity investigation, NAS bottleneck analysis, MEPC support, SHC readiness signal work

AI-ADJACENT TOOLING

AI code evaluation, local LLM experiments, tool-wrapper ideas, RAG/context retrieval research

Core Skills

Observability / Telemetry: Splunk Enterprise, Search Head Clustering, indexer clusters, Logstash, Kafka ingestion workflows, Cribl, ServiceNow, RCA, platform health analysis

Infrastructure / Automation: Python, SaltStack, Bash, Terraform, Git/GitHub, REST APIs, Linux, AWS, VMware/vSphere exposure, HA/DR readiness

AI / Agent Tooling: AI code evaluation, prompt/context engineering, local LLM experimentation, RAG/reranking concepts, FastAPI-style tool wrappers, VS Code/GitHub Copilot workflow research

Current Work

Netbuilder - Associate Observability Engineer | JPMC LogA Platform Contractor

Plano, TX | Aug 2024 - Present

- Support and automate a centralized observability platform spanning Splunk, Logstash, SaltStack, and Linux-based infrastructure for multi-tenant logging environments.
- Work with Kafka-to-Logstash-to-Splunk ingestion paths, including onboarding/sync analysis, stale Kafka topic cleanup PRs, and Logstash consumer remediation.
- Automated search-filter updates across **3,000+ Splunk roles** with Python, replacing a risky manual update path with a repeatable rollout process.
- Supported Salt-based Splunk automation, including login-banner orchestration across **100+ search heads** and SHC repave design support for **15 on-prem nodes**.
- Built duplicate-detection logic for onboarding and SaltStack configuration data to reduce drift and lower deployment risk.
- Investigated MEPC/DR readiness gaps around CM/deployer workflows, including NAS-based app parity bottlenecks, non-HA CM posture, and SHC readiness signals.
- Performed RCA and remediation for farm-break compliance/security findings; completed a **40-host reboot remediation target** and tracked follow-up work.
- Contributed to Step-3 repave failure analysis, SELinux context drift planning, and release-oriented support.

Other Work Experience

Data Annotation Tech - AI Code Evaluation Contractor

Jan 2024 - Present

- Evaluate AI-generated code and reasoning for correctness, efficiency, maintainability, edge-case handling, and instruction following across Python, JavaScript, and SQL.

Amazon Web Services - Associate Solutions Architect

Jan 2023 - Nov 2023

- Assisted enterprise customers with AWS architecture guidance across availability, security, scalability, and performance concerns.
- Built a cloud-migration cost estimation tool and cloud-native prototype work using AWS services and JavaScript.

Earlier Technical Experience

NCR Voyix - Software Engineering Intern | ServiceLink - IT Help Desk Support

- Created Python automation scripts for mobile testing workflows and supported web application, account-access, SQL reporting, and production-support requests.

Education & Certifications

B.S. Software Engineering
University of Texas at Arlington

Certifications: Splunk Certified Admin; Splunk Power User; Splunk User; Cribl Certified User; AWS Certified Solutions Architect - Associate

My Engineering Principles

01 Prefer boring automation.
Clear inputs, clear outputs, and logic that can be tested without guessing what the tool is doing.

02 Separate config from behavior.
I like the Salt pattern for a reason: pillar/state style separation makes workflows easier to reason about.

03 Do not confuse a spike with a system.
If it only worked once, or only exists as an investigation, I will call it that.

04 Use evidence instead of polish.
PRs, logs, commands, validation output, screenshots, and failure notes beat vague status writing.

05 Build tools like black boxes.
The caller should not need to know the inside of the machine to use it correctly.

Observability & Infrastructure / Platform Engineering

- **Telemetry ingestion:** Kafka topic cleanup, Logstash consumer behavior, ingestion troubleshooting, and Splunk indexing/search behavior.
- **Splunk platform support:** Search head, SHC, indexer, deployer, and cluster-manager related work across large managed environments.
- **Automation and drift control:** Python, SaltStack, Git, and validation-oriented workflows to make changes easier to review.
- **DR and outage readiness:** CM/deployer parity investigation, app sync bottlenecks, SHC readiness signals, and MEPC support workflows.
- **Operational remediation:** RCA, farm breaks, host remediation, SELinux drift, repave failure investigation, and release-oriented support tasks.

AI Tooling Experiments

Not my main lane yet. This is where I keep pulling my platform work.

- Evaluate AI-generated code and technical reasoning professionally across Python, JavaScript, and SQL.
- Experiment with local LLM setup using Python, `uv`, Hugging Face Transformers, and local model downloads.
- Research custom agents, MCP constraints, tool registries, RAG/reranking, and context retrieval for infrastructure workflows.
- Prefer explicit API/tool contracts over vague agent behavior, especially for systems work where failure needs to be inspectable.

Projects / Personal R&D

- **TEA-style engineering doctrine:** Personal repo for typed, encapsulated architecture, automation patterns, observability thinking, and agent workflow standards.
- **Observability automation concepts:** Notes and experiments for wrapping Splunk, Salt, VMware, and internal platform APIs behind deterministic tooling.
- **Terraform/AWS automation:** Terraform-based AWS provisioning experiments with Python support scripts for repeatable infrastructure deployment.
- **Second-brain workflow:** Markdown-first notes and diagrams for connecting work across Splunk, Salt, DR, automation, and AI tooling.

Why this format

This resume is intentionally more like an engineering map than a normal keyword sheet. The first page gives the recruiter the basic signal. The later pages give a technical manager a clearer view of how I think about platform work, automation, and AI-assisted tooling.

Source style preference: markdown-first, dark-mode, evidence-oriented, and easy to keep updated as a living work artifact.