**Day 3: Mastering Logs for Site Reliability Engineering**

**Section 1: 🔍 Beginner Level - The Foundation of Logging**

* **Introduction & Recap:**
* Welcome & Agenda Overview for Day 3.
* **Connect the Dots:** Briefly revisit Day 1 (Observability Pillars) and Day 2 (Metrics). How do Logs fit in? (Metrics tell you *what* happened, Logs often tell you *why*).
* Acknowledge their existing experience: "You've all used logs for troubleshooting. Today, we'll look at them through an SRE lens – focusing on structure, scale, and proactive use."
* **What Are Logs & Why SREs Care:**
* **Definition:** Beyond simple text files – event streams capturing discrete events over time.
* **Purpose (SRE Perspective):**
* Debugging & Root Cause Analysis (RCA) - The familiar ground.
* Monitoring system health & behavior (beyond just errors).
* Auditing & Security Analysis (Who did what, when?).
* Performance Analysis (Identifying bottlenecks, slow operations).
* Understanding user behavior.
* Basis for Alerting.
* **Types of Logs:** Application, System (syslog, journald), Infrastructure (cloud provider logs), Network, Security/Audit, Database, Event Logs. Show examples of each.
* **The Problem with Unstructured Logs:**
* Show examples of messy, inconsistent, free-text logs.
* Dispaly the challenges: Hard to parse reliably, difficult to query precisely, slow searching, inconsistent fields, impossible to aggregate meaningfully.
* **Activity (Short):** Give a small unstructured log snippet. Ask pairs to find specific information (e.g., "How many times did user X log in?"). Discuss the difficulty.
* **Introduction to Structured Logging:**
* **The Solution:** Logging in a consistent, machine-readable format (Key-Value, JSON).
* **Why it's Critical for SRE:** Enables reliable parsing, filtering, aggregation, correlation, and automation.
* Show simple examples: transforming an unstructured log message into JSON.
* Benefits: Faster querying, powerful filtering (e.g., where user\_id=123 and response\_time\_ms > 500), easy aggregation (e.g., count errors by service), easier correlation.
* **Key Takeaway:** Structure is Non-Negotiable for effective logging at scale.

**2: 🧩 Intermediate Level - Collection, Centralization & Analysis**

* **Log Collection & Shipping:**
* **The Need:** Getting logs from where they are generated (servers, containers, apps) to a central place.
* **Common Mechanisms:**
* Log Files (tail, agents reading files).
* Standard Output/Error (stdout/stderr - common in containers).
* Direct API calls (logging libraries sending directly).
* Syslog protocol.
* **Introduction to Log Shippers/Agents:** Tools like Fluentd, Fluent Bit, Logstash, Splunk Forwarders, CloudWatch Agent, Datadog Agent. Explain their role (collect, parse, buffer, route).
* **Concept:** Sidecars vs. DaemonSets vs. Standalone Agents.
* **Centralized Logging Platforms:**
* **Why Centralize?** Single pane of glass, correlation across services, consistent querying interface, retention management, access control.
* **Conceptual Overview of Components:**
* **Storage/Indexing:** (e.g., Elasticsearch, Loki, Splunk Indexers, CloudWatch Log Groups) - How logs are stored for searching.
* **Query Engine:** (e.g., Kibana, Grafana, Splunk Search, CloudWatch Insights Query Language) - How you interact with the data.
* **Visualization:** Dashboards, charts based on log data.
* **Demo:** Perform basic searches and filtering in the chosen demo platform using structured log data. (e.g., find all logs for a specific request ID, filter for ERROR level logs, show logs for a specific service instance).
* **Effective Querying Techniques (Splunk and Grafana):**
* Moving beyond simple keyword searches.
* Filtering by fields (leveraging structured logs).
* Using boolean operators (AND, OR, NOT).
* Time range selection.
* Basic regular expressions (if applicable to the tool).

**Section 3: 💡 Advanced Level - Optimization, Automation & Integration**

* **Logs as a Source for Metrics & Alerting:**
* **Connecting to Day 2:** Deriving metrics *from* logs (e.g., counting specific log messages like HTTP 500 errors, extracting latency values from log fields).
* **Tools:** Show how logging platforms can generate metrics (e.g., Kibana Lens/TSVB, Loki LogQL aggregations, CloudWatch Metric Filters, Datadog Log-Based Metrics).
* **Log-Based Alerting:** Setting up alerts based on log query results (e.g., "Alert if more than 10 'payment failed' logs appear in 5 minutes"). Discuss trade-offs vs. metric-based alerting (granularity vs. latency/cost).
* **Discussion:** When would you use log-based metrics/alerts vs. traditional metrics?
* **Correlation & Observability:**
* **Connecting to Day 1:** Tying it all together.
* **Trace IDs:** The power of having a unique ID across logs from different services for a single request. Show how to filter logs based on a Trace ID.
* **Linking Logs, Metrics, and Traces:** Demonstrate (conceptually or in a tool) how you might see a metric spike, drill down to relevant logs using time correlation, find a trace ID in the logs, and then view the entire distributed trace.
* **Log Volume, Cost Management & Sampling:**
* Logs can be voluminous and expensive!
* **Strategies:**
* **Log Levels:** Using appropriate levels (DEBUG, INFO, WARN, ERROR) and adjusting them dynamically. Don't log DEBUG in production usually.
* **Filtering at Source:** Agents dropping noisy/unimportant logs.
* **Sampling:** Logging only a percentage of certain high-volume events (e.g., log 10% of successful HTTP 200 requests but 100% of HTTP 500 errors). Discuss implications.
* **Retention Policies:** Setting appropriate lifecycles for log data (hot vs. cold storage, archival, deletion).
* **Cost Optimization:** Choosing the right tools and storage tiers.
* **Security & Compliance Considerations & Wrap-up:**
* **PII & Sensitive Data:** The importance of *not* logging sensitive information (passwords, credit cards, PII). Scrubbing techniques.
* Audit Logs: Importance for security and compliance.
* **Recap:** Key takeaways from the day. Logs as a proactive tool, importance of structure, centralization, correlation, and cost management.
* **Q&A:** Final questions.