SENTRY-LOGIC: Deployment Blueprint

This document provides a deployment schema and system blueprint for SENTRY-LOGIC, enabling a DevOps or system architect to set up a prototype environment.

1. Component Overview:

The following modules are deployed as separate services:

* **Input Proxy:** Intercepts LLM input and output.
* **Symbol Mapper:** Processes input/output to generate symbolic logs.
* **Log Storage:** Stores symbolic log entries.
* **Alert Engine:** Analyzes logs and generates alerts.
* **UI Frontend:** Provides a web interface for viewing logs and alerts.
* **(Optional) Reverse Proxy:** Acts as a gateway for external access.
* **(Optional) Encryption Gateway:** Handles encryption/decryption of data in transit.

2. Service Configuration Basics:

Service	Communication Flow	Suggested Ports	Volumes	Environment Variables/API Tokens

| Input Proxy | User/App → Input Proxy → LLM → Input Proxy → Symbol Mapper → Log Storage | 8080 (HTTP/HTTPS) | - | `LLM_API_URL`, `SENTRY_MAPPER_URL`, `SENTRY_LOGSTORE_URL` |

| Symbol Mapper | Input Proxy → Symbol Mapper → Log Storage → Alert Engine | 8081 (HTTP) | - | `SENTRY_LOGSTORE_URL`, `SENTRY_ALERTENGINE_URL` |

| Log Storage | Symbol Mapper → Log Storage; Alert Engine → Log Storage; UI Frontend → Log Storage | 5432 (PostgreSQL) | /logs | `POSTGRES_USER`, `POSTGRES_PASSWORD`, `POSTGRES_DB` |

| Alert Engine | Log Storage → Alert Engine → UI Frontend | 8082 (HTTP) | - | |

| UI Frontend | User → UI Frontend → Log Storage, Alert Engine | 80 (HTTP/HTTPS) | - | |

| Reverse Proxy | External → Reverse Proxy → All Services | 443 (HTTPS) | - | `SSL_CERTIFICATE_PATH`, `SSL_PRIVATE_KEY_PATH`, `PROXY_PASS_*_URL` |

| Encryption Gateway| User/App <-> Encryption Gateway <-> All Services | 443 (HTTPS) | - | `ENCRYPTION_KEY`, `DECRYPTION_KEY`, `TARGET_SERVICES_PUBLIC_KEYS` |

3. Example Deployment Structure (YAML/Compose):

```yaml
version: '3.8'
services:
  input_proxy:
    image: sentry-logic/input-proxy:latest
    ports:
      - "8080:8080"
    environment:
```

```
      LLM_API_URL: "http://llm:8000/api"
      SENTRY_MAPPER_URL: "http://symbol_mapper:8081/map"
      SENTRY_LOGSTORE_URL: "http://log_storage:5432/logs"  # Hypothetical HTTP endpoint for logs
    depends_on:
      - symbol_mapper
      - log_storage
    # Security: Use a network policy to restrict access to only necessary services.
    networks:
      - sentry_net

  symbol_mapper:
    image: sentry-logic/symbol-mapper:latest
    ports:
      - "8081:8081"
    environment:
      SENTRY_LOGSTORE_URL: "http://log_storage:5432/logs" # Hypothetical HTTP endpoint
      SENTRY_ALERTENGINE_URL: "http://alert_engine:8082/alerts"
    depends_on:
      - log_storage
    networks:
      - sentry_net

  log_storage:
    image: postgres:14-alpine
    volumes:
```

```
      - sentry_logs:/var/lib/postgresql/data
    environment:
      POSTGRES_USER: "sentry_user"
      POSTGRES_PASSWORD: "sentry_password"
      POSTGRES_DB: "sentry_db"
    # Security: Database should only be accessible from within the Sentry network.
    networks:
      - sentry_net
    # No ports exposed to the host

  alert_engine:
    image: sentry-logic/alert-engine:latest
    ports:
      - "8082:8082"
    depends_on:
      - log_storage
    networks:
      - sentry_net

  ui_frontend:
    image: sentry-logic/ui-frontend:latest
    ports:
      - "80:80" # Exposed to the host, or behind a reverse proxy
    environment:
      LOG_STORAGE_URL: "http://log_storage:5432/logs" # Hypothetical HTTP
      ALERT_ENGINE_URL: "http://alert_engine:8082/alerts"
```

```
    depends_on:
      - log_storage
      - alert_engine
    networks:
      - sentry_net

  reverse_proxy: # Optional
    image: nginx:latest
    volumes:
       - ./nginx.conf:/etc/nginx/nginx.conf # configure to proxy requests to internal services
    ports:
       - "443:443"
    networks:
      - sentry_net  # Place in front of the other sentry services

  encryption_gateway:  # Optional
    image: sentry-logic/encryption-gateway:latest  # An image for handling encryption
    ports:
      - "4443:4443"  # A different port
    # Environment variables
    environment:
       ENCRYPTION_KEY: "your_encryption_key"  # Key
       DECRYPTION_KEY: "your_decryption_key" # Key
    # Security: Secure handling of keys
    networks:
```

```
      - sentry_net
volumes:
  sentry_logs: # Named volume for storing logs


networks:
  sentry_net:
    driver: bridge
```

4. Security Considerations:

* TLS or Secure Channels: All communication between SENTRY-LOGIC services should use TLS (HTTPS) to prevent eavesdropping. For inter-service communication within the deployment (like input_proxy to symbol_mapper), consider using internal certificates or mutual TLS. The optional encryption gateway service can be used to handle this.

* Role-Based Access Control (RBAC): Implement RBAC at the Log Storage and UI Frontend layers to restrict access to logs and system functions based on user roles (e.g., "analyst," "administrator").

* Prevent Symbolic Log Leaks: The log_storage service should be isolated from external access. The optional reverse proxy is important in preventing direct access. Network policies should prevent direct access to the storage service. Consider using a separate, secure logging network if possible.

* Secure Separation: The log_storage and alert_engine services should be deployed on separate containers (or even separate machines) to minimize the impact of a potential compromise of one service. The database should be firewalled.