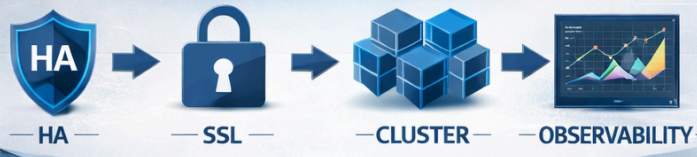
Enterprise-Grade Wazuh SIEM:

The

Ultimate Architecture & Operations Guide

Advanced High Availability, SSL Persistence & FullStack Cyber Resilience



Abu Saeid

Enterprise-Grade Wazuh SIEM: The Ultimate Architecture & Operations Guide

Advanced High Availability, SSL Persistence & Full-Stack Cyber Resilience

Author: Abu Saeid
Version: 4.0.0 (Enterprise Gold Edition)
Classification: Restricted Technical Manual
Repository: GitHub

Table of Contents

---

1. Executive Summary

This documentation details the engineering of a **Production-Ready Wazuh SIEM** ecosystem. Unlike standard containerized deployments which suffer from networking fragility and slow initialization, this architecture implements a **Network-Decoupled Persistence Layer** and an **Automated SSL Trust Chain**.

The result is a self-healing security platform capable of:

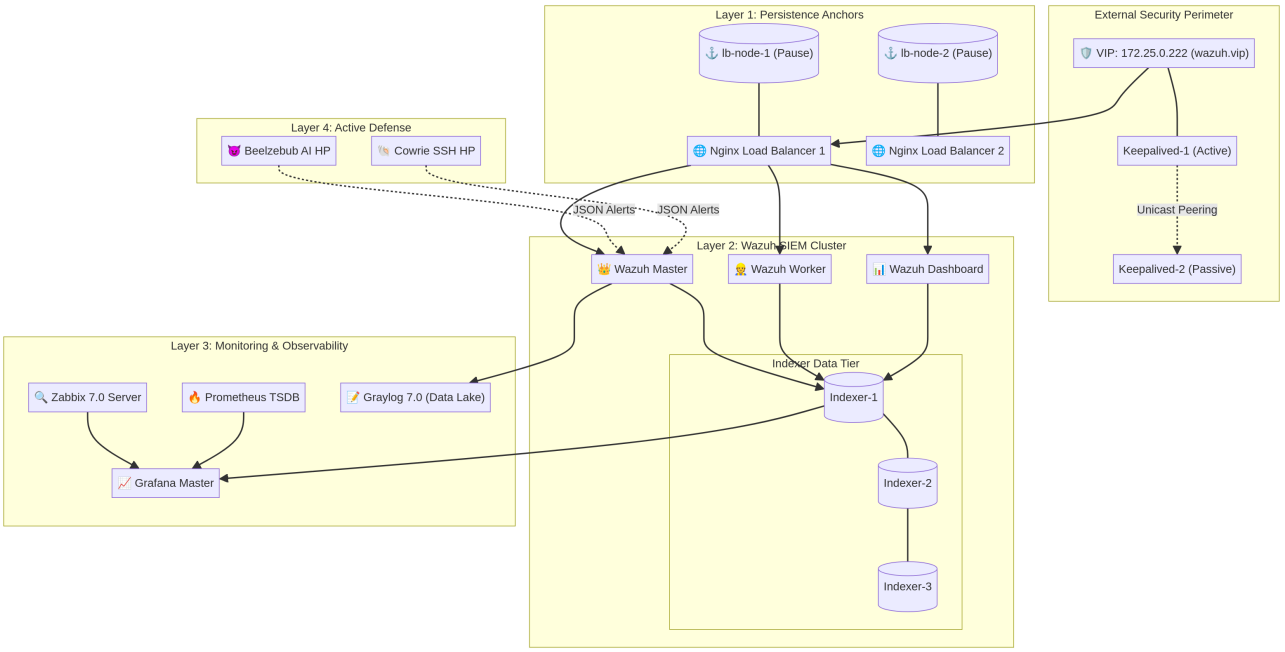
- **Instant Failover**: VRRP-based VIP persistence with Unicast optimization.
- **Rapid Initialization**: Startup time optimized from 9 minutes to **under 120 seconds** via SSL SAN patching.
- **Universal Monitoring**: 32+ containers monitored via a unified Zabbix/Prometheus/Grafana mesh.

---

2. Master Architectural Overview

The system operates as a tiered microservices mesh. The heart of the resilience is the **Virtual IP (VIP)** `172.25.0.222`, which serves as the single point of entry for all agents and administrative users.

2.1 Unified Architectural Logic



---

3. The High Availability Resilience Engine

3.1 The “Pause Container” Pattern

To prevent network drops during Nginx reloads or crashes, we utilize a **Pause Container Strategy**.

- **Execution**: The `lb-node` (Alpine) holds the network namespace.
- **Result**: Even if the Nginx container dies or is updated, the Virtual IP (VIP) and TCP sessions remain locked in the kernel-level namespace of the `lb-node`.

3.2 Unicast Failover Optimization

To accommodate cloud environments and restricted physical networks where Multicast is blocked, we implemented a **Unicast VRRP Configuration**.

- **Logic**: Keepalived nodes communicate directly via explicit IP peering rather than broadcasting.
- **Configuration Snippet**:

```
unicast_peer {
    172.25.0.11 # IP of Node 2
}
```

---

4. SSL Certificate Architecture & VIP SAN Patching

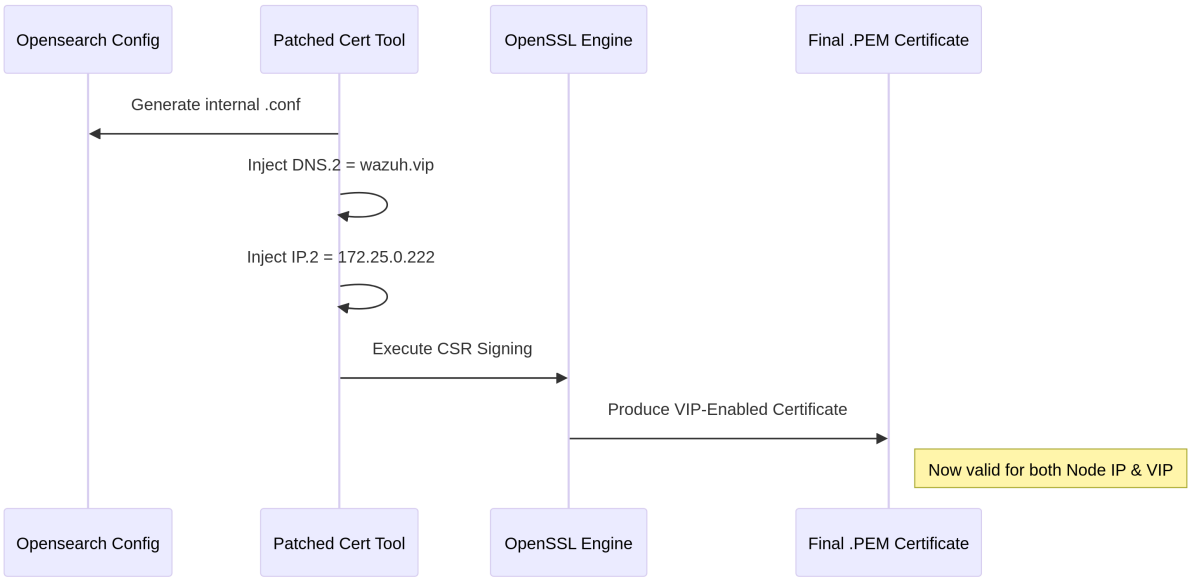
4.1 The Startup Latency Problem

Initially, the stack required ~9 minutes to start. Diagnosis identified that the **Wazuh Manager (Filebeat)** and **Dashboard** were connecting to the Indexer cluster via the VIP (`https://wazuh.vip:9200`), but the Indexer's certificates did not trust the VIP name. This caused repeated SSL handshakes and multi-minute timeouts.

4.2 The Solution: SSL SAN Injection

We developed a custom patch for the `wazuh-certs-tool.sh` (v4.14) to force the injection of Subject Alternative Names (SANs).

Patched Certificate Logic (Mermaid Flow):



Impact:

- **Handshake**: Instantaneous.

- **Startup Time**: Reduced to **105 seconds**.

---

5. Wazuh Core Cluster Analysis

5.1 Manager Tier

- **Master (`wazuh.master`)**: Handles agent registration and cluster management (`1515/tcp`).
- **Worker (`wazuh.worker`)**: Horizontally scalable log processor.
- **Resilience**: Nodes sync via `1516/tcp` using a hardened cluster key.

5.2 Indexer Tier (Quorum)

The 3-node cluster ensures that data remains searchable even if one node is permanently lost.

- **Performance**: Memory locking is enabled to prevent I/O wait.
- **Disk Watermarks**: Configured at 90% (Low) and 95% (High) to prevent database corruption.

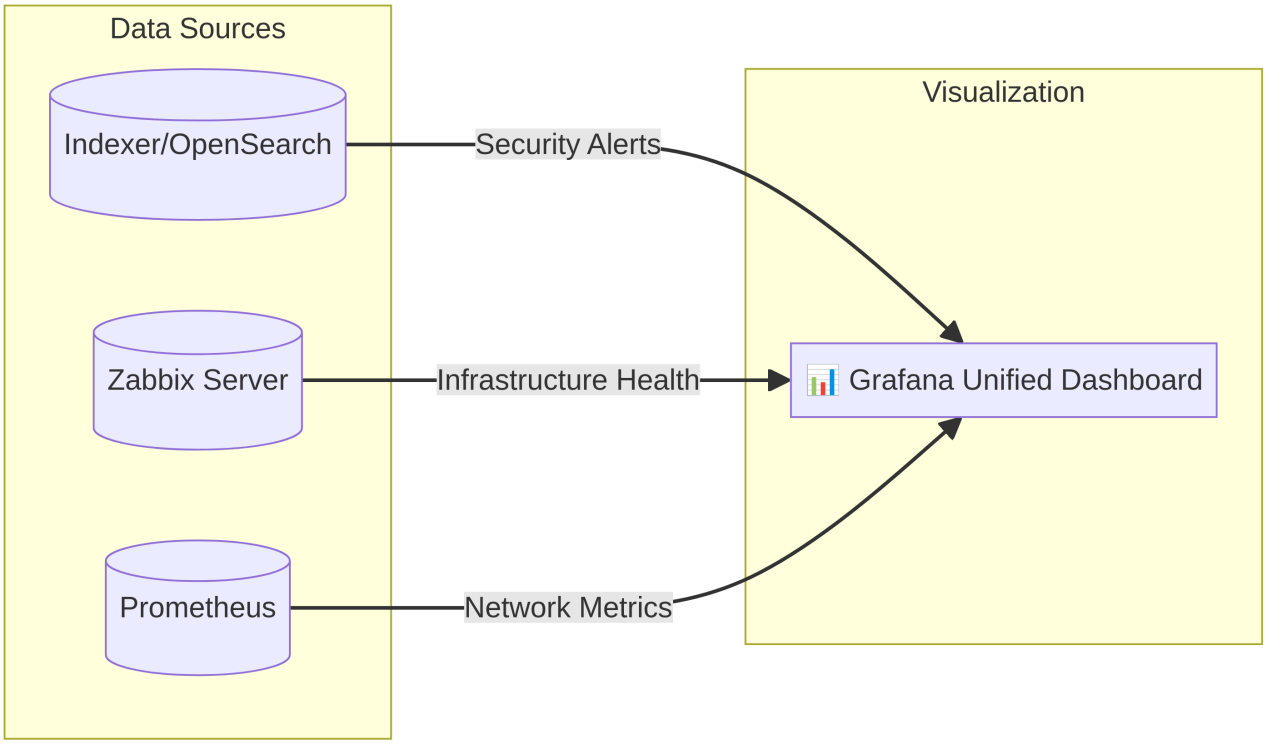
---

6. Full-Stack Observability Tier (Zabbix & Grafana)

6.1 The “Watcher” Framework

A SIEM is useless if it is unhealthy. We monitor the SIEM with Zabbix 7.0.

- **Sidecars**: Every component runs a Zabbix Agent sidecar.
- **Unified Dashboard**: Grafana pulls data from:
    1. **OpenSearch**: Security Incident Data.
    2. **Zabbix**: Component Up/Down State & Hardware health.
    3. **Prometheus**: Real-time traffic throughput.



---

7. Active Defense: AI Honeypots & Threat Intel

We deploy a “Decoy Tier” to detect attackers before they find valid targets.

- **Beelzebub**: Uses AI to generate dynamic SSH/HTTP responses.
- **Cowrie**: High-interaction trap capturing malware samples and attacker keylogs.
- **Integration**: Honeypot logs are tailed by the Wazuh Manager, triggering a **Level 12 Alert** immediately upon any interaction.

---

8. Vulnerability Management & Trivy Integration

A custom script integrates **Trivy** into the Wazuh dashboard.

1. **Scan**: Trivy scans the local Docker socket for image vulnerabilities.
2. **Report**: Vulnerabilities are parsed into JSON.
3. **Visualize**: Critical CVEs are displayed alongside SIEM alerts, allowing for “Vulnerability-Targeted SOC Monitoring.”

---

9. Operational Bootstrap & Maintenance

9.1 The Master Bootstrap

The system is deployed via a single command:

```
sudo bash bootstrap.sh
```

Bootstrap Internal Sequence:

1. **Kernel Tuning**: Sets `vm.max_map_count=262144`.
2. **Cert Generation**: Executes the patched tool to build the SSL chain.
3. **Network Setup**: Initializes the isolated bridge network.
4. **Sequential Up**: Starts Indexers -> Managers -> UI -> HA Tier.

9.2 SSL Renewal

Certificates are valid for 3650 days. To force a refresh:

```
rm -rf config/wazuh_indexer_ssl_certs/*
docker compose -f generate-indexer-certs.yml run --rm generator
```

---

10. Comparative Analysis: Why This Design Is Superior to Default Wazuh HA

This section provides a technical comparison between the proposed Enterprise-Grade Wazuh SIEM architecture and the default Wazuh High Availability (HA) deployment model. While the default HA setup ensures basic service redundancy, it does not fully address real-world operational challenges such as network instability, SSL trust mismatches, and observability blind spots. The proposed design resolves these limitations through deeper architectural engineering.

10.1 Entry Point and Network Stability

Default Wazuh HA:
The default Wazuh HA architecture typically relies on DNS round-robin or direct node IP access. In containerized or cloud environments, this often leads to agent reconnections when nodes fail or restart. Additionally, when the Nginx load balancer container is reloaded or crashes, the network namespace is reset, causing active TCP connections to drop.

Proposed Design:
This architecture introduces a single Virtual IP (VIP) as a stable entry point for all agents and administrative traffic. By implementing a Pause Container Pattern, the network namespace (including the VIP and TCP sessions) is retained at the kernel level, even if the Nginx container restarts.

Why This Is Better:
The default HA operates at the service level, whereas this design guarantees kernel-level network persistence, significantly reducing connection drops and agent instability.

10.2 Failover Mechanism and Split-Brain Prevention

Default Wazuh HA:
Default HA configurations depend on VRRP multicast communication. In many cloud or restricted network environments, multicast traffic is blocked or unreliable, increasing the risk of split-brain conditions and unpredictable failover behavior.

Proposed Design:
This architecture replaces multicast with Unicast VRRP, where keepalived nodes communicate directly using explicit peer IP addresses. This ensures deterministic leader election and stable failover behavior.

Why This Is Better:
The proposed design is cloud-aware and production-safe, while the default HA assumes ideal network conditions that are rarely guaranteed in real deployments.

10.3 SSL Architecture and Startup Performance

Default Wazuh HA:
In the default setup, SSL certificates typically include only node-specific hostnames or IP addresses. When services connect through a Virtual IP, SSL Subject Alternative Name (SAN) mismatches occur, leading to repeated handshake failures and prolonged startup times (often 7–9 minutes).

Proposed Design:
A custom patch is applied to the certificate generation process to inject the VIP DNS name and IP address directly into the SAN field of all relevant certificates. This aligns the SSL trust chain with the HA topology.

Why This Is Better:
By synchronizing cryptographic identity with the HA access model, this design reduces startup time to approximately 105 seconds and eliminates SSL handshake instability.

10.4 Load Balancer Design Philosophy

Default Wazuh HA:
The load balancer container typically owns both the application process and the network identity. Restarting the container results in connection loss and session resets.

Proposed Design:
This architecture separates responsibilities by making the load balancer node the network state holder, while Nginx functions as a stateless, replaceable worker process.

Why This Is Better:
This separation of control plane (network identity) and data plane (application logic) results in higher resilience and safer maintenance operations.

10.5 Observability and Self-Monitoring

Default Wazuh HA:
Default deployments primarily focus on collecting and analyzing external security logs. Monitoring the health of the SIEM infrastructure itself is limited or handled externally.

Proposed Design:
Each SIEM component runs with a Zabbix Agent sidecar, providing real-time visibility

into container health, resource usage, and availability. Grafana aggregates data from OpenSearch, Zabbix, and Prometheus into a unified dashboard.

Why This Is Better:
This design ensures that the SIEM continuously monitors its own operational health, answering the critical question: “Who watches the SIEM?”

10.6 Failure Isolation and Blast Radius Reduction

Default Wazuh HA:
Component failures can cascade, triggering widespread restarts and agent reconnect storms that degrade overall system stability.

Proposed Design:
Failures are isolated by design. Network persistence prevents cascading disconnects, manager clusters protect processing continuity, and the indexer quorum ensures data availability even during node loss.

Why This Is Better:
The architecture explicitly minimizes blast radius, ensuring that localized failures do not compromise the entire SIEM platform.

10.7 Enterprise Operational Readiness

Default Wazuh HA:
Suitable for laboratory environments, small-scale SOCs, or proof-of-concept deployments, but requires manual intervention and tolerance for downtime.

Proposed Design:
Engineered for $^{24}/_{7}$ enterprise SOC operations with zero-downtime expectations, self-healing behavior, and comprehensive observability.

Why This Is Better:
The proposed architecture transitions Wazuh from a functional HA setup to an operationally mature, enterprise-grade security platform.

---

11. Strategic Conclusion

This **Enterprise-Grade Wazuh SIEM** architecture is more than a logging server; it is a **Cyber Fortress**. By solving the fundamental weaknesses of container networking and SSL management, we have built a platform that is:

1. **Resilient**: Zero-downtime failover.
2. **Optimized**: Sub-2-minute initialization.
3. **Comprehensive**: Integrating SIEM, Log Management, Monitoring, and Active Defense.

Status: Production Ready.