Baseline Document for Thesis Writing

I. Latar Belakang

Perkembangan teknologi informasi membawa konsekuensi meningkatnya ancaman siber terhadap infrastruktur digital. Organisasi menggunakan **Security Information and Event Management (SIEM)** seperti **Wazuh** untuk mengumpulkan dan menganalisis log keamanan. Namun, pendekatan berbasis aturan (rule-based detection) memiliki keterbatasan: sulit mendeteksi ancaman baru (zero-day), menghasilkan jumlah peringatan yang sangat besar (alert fatigue), serta membutuhkan pembaruan manual yang berkelanjutan.

Machine Learning (ML) menawarkan pendekatan alternatif melalui **deteksi anomali**, yaitu mengidentifikasi aktivitas yang menyimpang dari pola normal. Thesis ini bertujuan untuk mengintegrasikan algoritma **Isolation Forest** pada SIEM Wazuh guna memprioritaskan alert berdasarkan skor anomali. Dengan demikian, SOC (Security Operation Center) dapat lebih fokus pada peringatan yang paling relevan.

II. Rumusan Masalah

1. Bagaimana cara mengurangi ketergantungan penuh pada rule-based detection dalam SIEM?
2. Bagaimana metode machine learning dapat digunakan untuk **memprioritaskan alert** berdasarkan tingkat anomali?
3. Bagaimana cara memastikan bahwa model dapat dievaluasi dengan adil meskipun data nyata sangat imbang (anomali jarang terjadi)?
4. Bagaimana cara merancang pipeline data dari log mentah → pembersihan → rekayasa fitur → pelatihan model → inferensi real-time?

III. Tujuan Penelitian

1. Mengembangkan model **deteksi anomali berbasis Isolation Forest** untuk log keamanan Wazuh.
2. Mendesain sistem pipeline **end-to-end** mulai dari pengumpulan log hingga notifikasi SOC.
3. Mengevaluasi model dengan metrik yang relevan untuk deteksi anomali (Average Precision, Precision@k, PR-curve, WSS@1%).
4. Memberikan rekomendasi desain untuk penerapan sistem serupa pada SIEM produksi.

IV. Desain Sistem dan Pipeline

A. Arsitektur Sistem

1. **Data Collection**: Wazuh Manager mengumpulkan log dari agen (≈5 host).

2. **Ingestion**: Log dikirimkan ke Logstash → Kafka.
3. **Warehouse**: Kafka menyimpan data ke ClickHouse untuk keperluan historis.
4. **ML Service**: Python-based API untuk inference (Isolation Forest).
5. **Orchestrator**: n8n menangani workflow (alert → Telegram/Email).
6. **Visualization**: Wazuh Dashboard tetap digunakan sebagai konsol utama.

(Gambar arsitektur dapat ditambahkan sebagai Gambar IV.1 dalam laporan)

B. Pipeline Data

1. **Raw Logs** → Wazuh JSON events.
2. **Cleaning** → Hilangkan kolom tidak relevan, normalisasi tipe data, buang duplikasi. Output: `cleaned_wazuh.csv`.
3. **Splitting** → Bagi berdasarkan waktu:
4. T1: Training (s/d 2025-08-02)
5. T2: Validation (2025-08-03 → 2025-08-07)
6. T2_synth: Validation dengan anomali injeksi
7. T3: Testing (≥ 2025-08-08)
8. **Feature Engineering** → Derived features meliputi:
9. Off-hours flag (jam kerja vs di luar jam kerja)
10. Rule rarity per agent (TF-IDF)
11. Decoder rarity per agent
12. Severity deviation (z-score)
13. Pair/triple features (Agent×Rule, Agent×Decoder, Rule×Hour)
14. **Modeling** → Isolation Forest dilatih pada T1, divalidasi dengan T2_synth, diuji pada T3.
15. **Evaluation** → Hitung AP, P@k, PR-curve, WSS, dan dokumentasi kasus normal vs anomali.

V. Transformasi Data

- **Input:** Wazuh raw logs (JSON, 400+ kolom awal).
- **Cleaning:** Disederhanakan menjadi inti (event_id, timestamp, agent, rule_id, rule_level, decoder).
- **Feature Engineering:** Tambahkan derived features untuk memperkuat konteks.
- **Injection:** Pada T2, anomali sintetis diinjeksikan (off-hours, rare rule, novel decoder, severity outlier, recency). Digunakan untuk validasi model.

VI. Evaluasi Model

1. **T2 Clean** → Menunjukkan bagaimana model menangani data asli.
2. **T2_synth** → Validasi terkontrol dengan ground truth.
3. **Metrics**:
4. Average Precision (AP)
5. Precision@50, Precision@1%
6. Work Saved over Sampling (WSS@1%)
7. PR Curve visualisasi

8. **T3** → Blind test untuk simulasi real-world SOC triage.

VII. Luaran Penelitian

- **Model anomaly detection** terlatih berbasis Isolation Forest.
- **Pipeline implementasi** (Wazuh → Logstash → Kafka → ClickHouse → ML → n8n → SOC).
- **Dokumentasi lengkap** data cleaning, feature engineering, anomaly injection, training, evaluasi.
- **Hasil validasi** berupa tabel, grafik, dan contoh kasus normal vs anomali.
- **Rekomendasi praktis** untuk penerapan lebih luas pada SIEM enterprise.

VIII. Rencana Pengembangan (Saran)

- Tambahan algoritma (LOF, Autoencoder) untuk perbandingan.
- Penggunaan metode explainability (SHAP/LIME) untuk menjelaskan skor anomali.
- Integrasi real-time alert prioritization langsung ke SOC workflow.
- Perluasan cakupan data (lebih banyak agen/log tipe jaringan).