Cyber-Physics Integration Guide

SEAMLESS INTEROPERABILITY &
ARCHITECTURE OVERVIEW VERSION

CYBER-PHYSICS

+9647712926825

Prepared by Munther Kazem

V2.2.0 Enterprise Edition

Executive Summary

Cyber-Physics (CP) is designed as an **Overlay Intelligence Layer**, not a replacement for existing infrastructure. It integrates with your current security stack (SIEM, EDR, NDR) via a standard **RESTful API**, providing deterministic, physics-based decision-making in real-time.

Key Integration Features:

- **Zero-Friction Deployment:** Docker-based containerization.
- **Universal Compatibility:** JSON/HTTP standard protocols.
- **Low Latency:** Millisecond processing time (O(1) complexity).
- **Secure:** Baked-in license keys and offline validation.

Integration Architecture

Data Flow:

1. **Ingest (Sensors):** Existing systems (SIEM/NDR) forward metadata streams to the CP Core.
2. **Process (Physics Engine):** The CP Core calculates kinetic metrics (Momentum, Energy, Mass) locally.
3. **Response (Actuators):** CP triggers immediate actions via Webhooks to SOAR or EDR platforms.

Ingestion Methods (Input)

CP accepts data via the POST /ingest endpoint. Below are configuration examples for common enterprise data pipelines.

A. Logstash / Fluentd (Universal Forwarder)

Ideal for forwarding logs from **Splunk, ELK, or QRadar:**

```
# Logstash HTTP Output Configuration
output {
  http {
    url => "http://cp-core-internal:8081/ingest"
    http_method => "post"
    format => "json"
    headers => {
      "client-id" => "TJDEED-NODE-01"
      "Content-Type" => "application/json"
    }
    mapping => {
      "len_new" => "%{bytes_out}"
      "len_old" => "%{bytes_in}"
      "is_bot" => "%{is_automated}"
      "is_minor" => false
      "comment" => "Real-time stream from SIEM"
    }
  }
}
```

B. Python Middleware (Custom Integration)

For connecting custom internal tools or NDR solutions:

```
import requests
import json

CP_ENDPOINT = "http://localhost:8081/ingest"
HEADERS = {"client-id": "TJDEED-NODE-01", "Content-Type": "application/json"}

def send_to_physics_engine(traffic_data):
    payload = {
        "len_new": traffic_data['packet_size'],
        "len_old": traffic_data['historical_avg'],
        "is_bot": traffic_data['heuristic_flag'],
        "is_minor": False,
        "comment": "high_velocity_burst"
    }

    try:
        response = requests.post(CP_ENDPOINT, json=payload, headers=HEADERS)
        if response.status_code == 200:
            print(f"Physics Metrics: {response.json()}")
    except Exception as e:
        print(f"Connection failed: {e}")
```

Kinetic Response (Output)

Cyber-Physics does not just "alert"; it decides. You can configure the system to trigger actions when specific physical thresholds (e.g., High Momentum Attack) are breached.

A. SOAR Webhook (Cortex XSOAR / UiPath)

Trigger a playbook automatically.

Trigger Logic:

- **Condition:** Anomaly Score > 0.95 AND Velocity > 1000 events/s
- **Action:** Send POST request to SOAR.

Payload Example:

```json
{
  "incident_type": "KINETIC_DDOS",
  "severity": "CRITICAL",
  "source_ip": "192.168.1.50",
  "physics_data": {
    "momentum": 98.5,
    "mass_accumulation": "High"
  },
  "recommended_action": "BLOCK_IP_IMMEDIATELY"
}
```

B. Dashboard Visualization

CP provides a /metrics endpoint for Prometheus/Grafana integration to visualize the "Health Physics" of the network.

- **Endpoint:** GET http://localhost:8081/metrics
- **Metrics:** events_per_second, accuracy, anomaly_score.

Technical Specifications

Requirement	Specification
Deployment Format	Docker Container (Linux/Windows)
CPU	4 CPUs or higher (No GPU required)
RAM	4 GB Minimum (8 GB Recommended)
Network Ports	8081 (API), 6379 (Internal Redis)
Security	Offline License Validation, Header Authentication
Scalability	Horizontal Scaling (Stateless API Design)