# Security Operations Center (SOC) Lab

# Complete Installation & Configuration Guide

Automated Threat Detection and Response Platform

Splunk SIEM • Shuffle SOAR • Suricata IDS • Zeek Network Monitor

OpenAI AI Analysis • Jira Ticketing • Slack Notifications

Author: Davi Lal

December 2025

## Table of Contents

1. Executive Summary

2. Prerequisites and System Requirements

3. Part 1: Suricata IDS Installation and Configuration

4. Part 2: Zeek Network Monitor Installation and Configuration

5. Part 3: Splunk Enterprise Installation and Configuration

6. Part 4: Shuffle SOAR Installation and Workflow Configuration

7. Part 5: OpenAI API Configuration

8. Part 6: Jira Integration Configuration

9. Part 7: Slack Integration Configuration

10. Part 8: Testing and Validation

11. Part 9: Troubleshooting Guide

12. Part 10: Best Practices and Maintenance

## Executive Summary

This guide provides comprehensive documentation for building an enterprise-grade Security Operations Center (SOC) lab environment. The infrastructure integrates multiple security tools to create an automated threat detection and incident response platform that achieves a 98% reduction in mean time to respond (MTTR).

### Architecture Overview

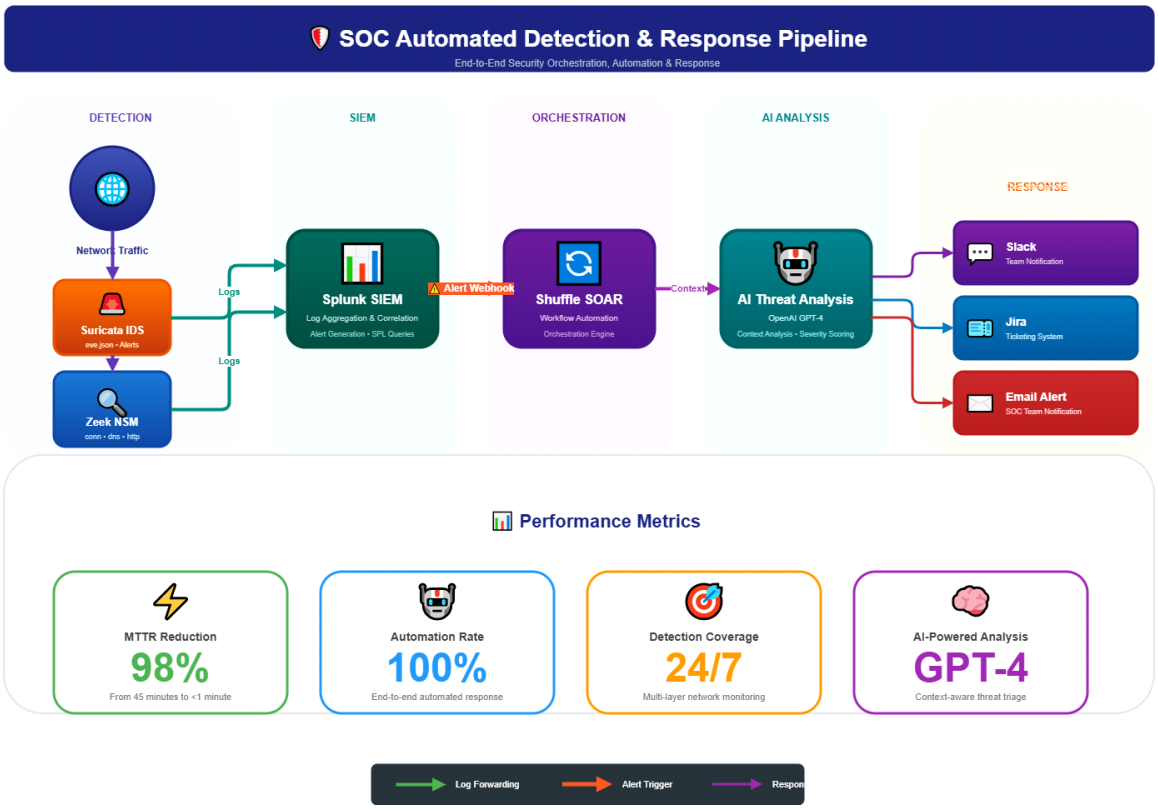
The SOC lab implements a layered security architecture:

- Network Detection Layer: Suricata IDS and Zeek network monitoring capture and analyze network traffic
- SIEM Layer: Splunk Enterprise aggregates, correlates, and analyzes security events
- Orchestration Layer: Shuffle SOAR automates incident response workflows
- AI Analysis Layer: OpenAI GPT-4 provides intelligent threat analysis and recommendations
- Ticketing Layer: Jira automatically creates and tracks security incidents
- Communication Layer: Slack delivers real-time notifications to the SOC team

### Key Features

- Automated threat detection using signature-based and anomaly-based methods
- AI-powered threat analysis with executive summaries and technical details
- Automated incident ticket creation with MITRE ATT&CK framework mapping
- Real-time SOC team notifications via Slack
- Comprehensive logging and forensic capabilities
- 98% reduction in mean time to respond (MTTR)

(1) Whole Automation Walkflow




## Prerequisites and System Requirements

### Hardware Requirements

| Component | Minimum Specification |
|---|---|
| CPU | 4 cores (8 cores recommended for production) |
| RAM | 16 GB (32 GB recommended) |
| Storage | 100 GB SSD (500 GB recommended for extended log retention) |
| Network | 1 Gbps network interface |

### Software Requirements

- Operating System: Kali Linux 2024 or Ubuntu 22.04 LTS
- Splunk Enterprise 9.x (Free license available)
- Suricata 7.x or later
- Zeek (formerly Bro) 6.x or later

- Shuffle SOAR (cloud or self-hosted)
- Python 3.8 or later

### External Services Required

- OpenAI API account (GPT 4 access recommended)
- Jira Cloud or Server instance
- Slack workspace with incoming webhook capability

## Part 1: Suricata IDS Installation and Configuration

### 1.1 Suricata Overview

Suricata is an open-source network Intrusion Detection System (IDS) and Intrusion Prevention System (IPS) that provides real-time threat detection capabilities. It uses signature-based detection rules to identify malicious network traffic and security threats.

### 1.2 Installation Steps

#### Update System Packages

```
sudo apt update && sudo apt upgrade -y
```

#### Install Suricata

```
sudo apt install suricata -y
```

#### Verify Installation

```
sudo systemctl status suricata

suricata --version
```

#### 📸 Screenshot: Suricata Status

*Insert Suricata_status.png - Shows Suricata service running with version 8.0.2, active (running) status*

### 1.3 Suricata Configuration

#### Edit Main Configuration File

```
sudo nano /etc/suricata/suricata.yaml
```

### Configure Network Interface

Locate the af-packet section and configure your network interface (typically eth0):

```
af-packet:

  - interface: eth0

    cluster-id: 99

    cluster-type: cluster_flow

    defrag: yes
```

### Configure Home Network

Set your home network CIDR range in the vars section:

```
vars:

  address-groups:

    HOME_NET: "[192.168.1.0/24]"

    EXTERNAL_NET: "!$HOME_NET"
```

### Enable EVE JSON Logging

Configure EVE logging for Splunk integration:

```
outputs:

  - eve-log:

      enabled: yes

      filetype: regular

      filename: /var/log/suricata/eve.json

      types:

        - alert:

            payload: yes

            metadata: yes

            http: yes
```

## 1.4 Suricata Rules Management

### Update Suricata Rules

```
sudo suricata-update
```

### Enable Specific Rule Sources

List available rule sources:

```
sudo suricata-update list-sources
```

Enable ET/OPEN ruleset (recommended):

```
sudo suricata-update enable-source et/open

sudo suricata-update
```

### Test Configuration

```
sudo suricata -T -c /etc/suricata/suricata.yaml -v
```

You should see a message confirming the configuration is valid.

## 1.5 Start Suricata Service

1. Enable Suricata to start on boot:

```
sudo systemctl enable suricata
```

1. Start the Suricata service:

```
sudo systemctl start suricata
```

1. Verify Suricata is running:

```
sudo systemctl status suricata
```

## 1.6 Verify Log Generation

Check that Suricata is generating logs:

```
sudo tail -f /var/log/suricata/eve.json

sudo tail -f /var/log/suricata/fast.log
```

Generate test traffic (e.g., browse websites) and confirm logs are being created.

# Part 2: Zeek Network Monitor Installation and Configuration

## 2.1 Zeek Overview

Zeek (formerly Bro) is a powerful network monitoring framework that provides comprehensive visibility into network traffic. Unlike signature-based IDS systems, Zeek

focuses on network analysis and generates detailed logs about network activity, protocols, and behaviors.

## 2.2 Installation Steps

### Install Dependencies

```
sudo apt install cmake make gcc g++ flex bison libpcap-dev \

  libssl-dev python3 python3-dev swig zlib1g-dev -y
```

### Install Zeek from Package

For Debian/Ubuntu systems:

```
echo 'deb
http://download.opensuse.org/repositories/security:/zeek/xUbuntu_22.04/ /' |
\

  sudo tee /etc/apt/sources.list.d/security:zeek.list



curl -fsSL
https://download.opensuse.org/repositories/security:zeek/xUbuntu_22.04/Releas
e.key | \

  gpg --dearmor | sudo tee /etc/apt/trusted.gpg.d/security_zeek.gpg >
/dev/null



sudo apt update

sudo apt install zeek -y
```

### Add Zeek to PATH

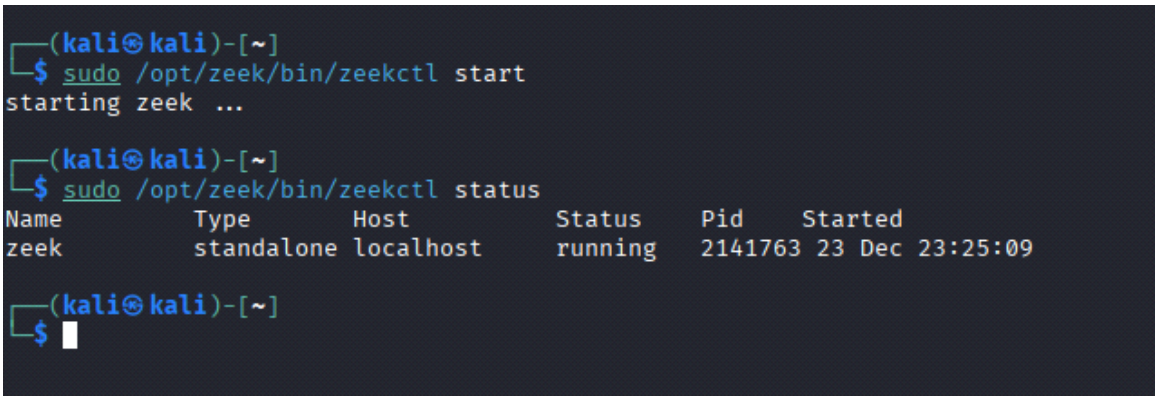
```
echo 'export PATH=/opt/zeek/bin:$PATH' >> ~/.bashrc

source ~/.bashrc
```

### Verify Installation

```
zeek --version
```

### 📸 Screenshot: Zeek Status




## 2.3 Zeek Configuration

### Configure Network Interface

Edit the node configuration file:

```
sudo nano /opt/zeek/etc/node.cfg
```

Configure the interface (replace eth0 with your interface):

```
[zeek]

type=standalone

host=localhost

interface=eth0
```

### Configure Local Networks

Edit the networks configuration:

```
sudo nano /opt/zeek/etc/networks.cfg
```

Add your local network:

```
192.168.1.0/24    Private IP space

10.0.0.0/8        Private IP space
```

### Enable JSON Logging

Create or edit the local site policy file:

```
sudo nano /opt/zeek/share/zeek/site/local.zeek
```

Add the following line to enable JSON logging:

```
@load policy/tuning/json-logs.zeek
```

### 2.4 Deploy and Start Zeek

1. Deploy Zeek configuration:

```
sudo /opt/zeek/bin/zeekctl deploy
```

1. Check Zeek status:

```
sudo /opt/zeek/bin/zeekctl status
```

1. If Zeek is not running, start it:

```
sudo /opt/zeek/bin/zeekctl start
```

### 2.5 Verify Log Generation

Check that Zeek is generating logs:

```
ls -la /opt/zeek/spool/zeek/
```

View connection logs:

```
tail -f /opt/zeek/spool/zeek/conn.log
```

You should see network connection data being logged in real-time.

### 2.6 Configure Log Rotation

Set up automated log rotation in zeekctl.cfg:

```
sudo nano /opt/zeek/etc/zeekctl.cfg
```

Configure log rotation settings:

```
LogRotationInterval = 3600    # Rotate logs every hour

LogExpireInterval = 0         # Never delete logs automatically
```

## Part 3: Splunk Enterprise Installation and Configuration

### 3.1 Splunk Overview

Splunk Enterprise is a powerful Security Information and Event Management (SIEM) platform that aggregates, indexes, and analyzes machine data from multiple sources. It serves as the central correlation and analysis engine for the SOC lab.

### 3.2 Download and Install Splunk

1. Download Splunk Enterprise from splunk.com (requires free account)
2. Install Splunk:

```
sudo dpkg -i splunk.deb
```

1. Start Splunk and accept the license:

```
sudo /opt/splunk/bin/splunk start --accept-license
```

1. Enable Splunk to start on boot:

```
sudo /opt/splunk/bin/splunk enable boot-start
```

### 3.3 Initial Splunk Configuration

- Access Splunk web interface at: http://localhost:8000
- Default credentials: admin / changeme (change immediately)
- Complete the initial setup wizard

### 3.4 Configure Suricata Data Input

#### Create Index

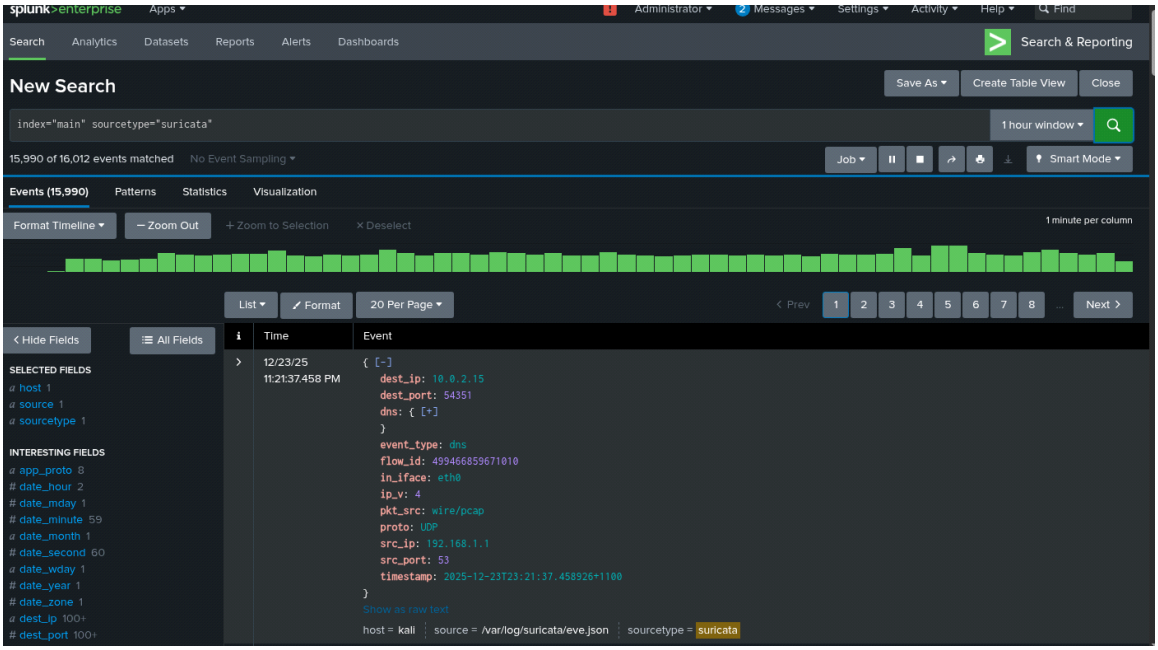
1. In Splunk Web, navigate to Settings → Indexes
2. Click 'New Index'
3. Configure: Index Name: main, Index Data Type: Events, Max Size: 500000 MB

#### Add Suricata Data Input

1. Navigate to Settings → Data Inputs
2. Click 'Files & Directories' → 'New Local File & Directory'
3. Configure:

- File or Directory: /var/log/suricata/eve.json
- Sourcetype: _json
- Index: main

## Screenshot: Suricata Logs in Splunk

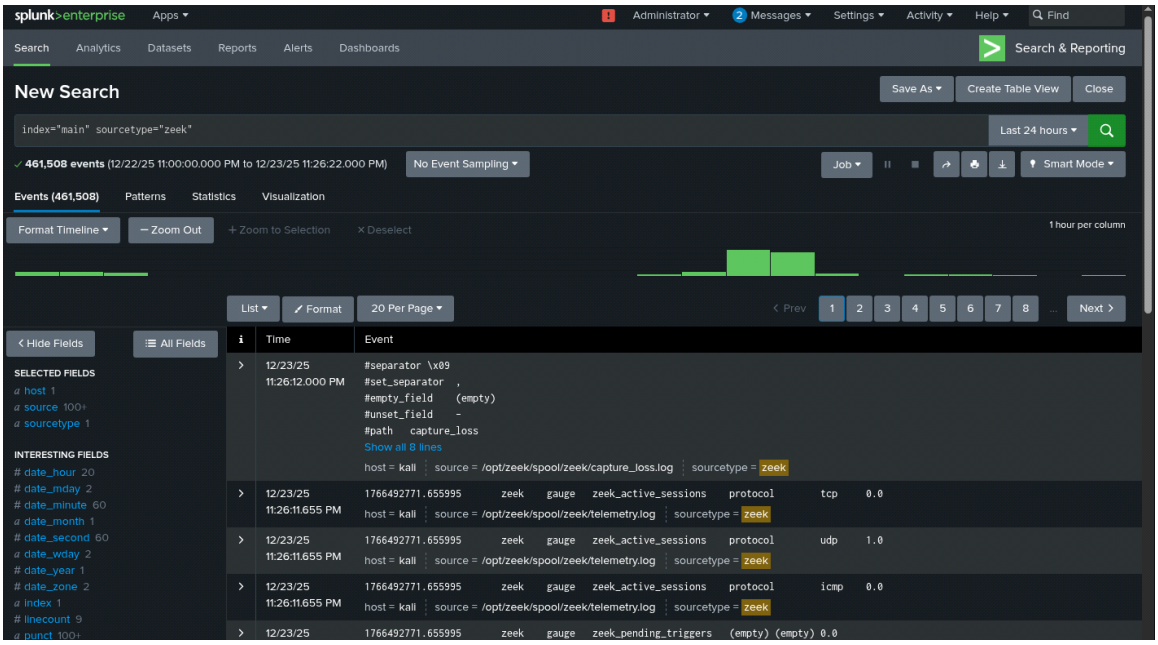



## 3.5 Configure Zeek Data Input

1. Navigate to Settings → Data Inputs
2. Click 'Files & Directories' → 'New Local File & Directory'
3. Configure:

- File or Directory: /opt/zeek/spool/zeek/*.log
- Whitelist: .*.log$
- Sourcetype: _json
- Index: main

### Screenshot: Zeek Logs in Splunk




## 3.6 Create Custom Splunk Alerts

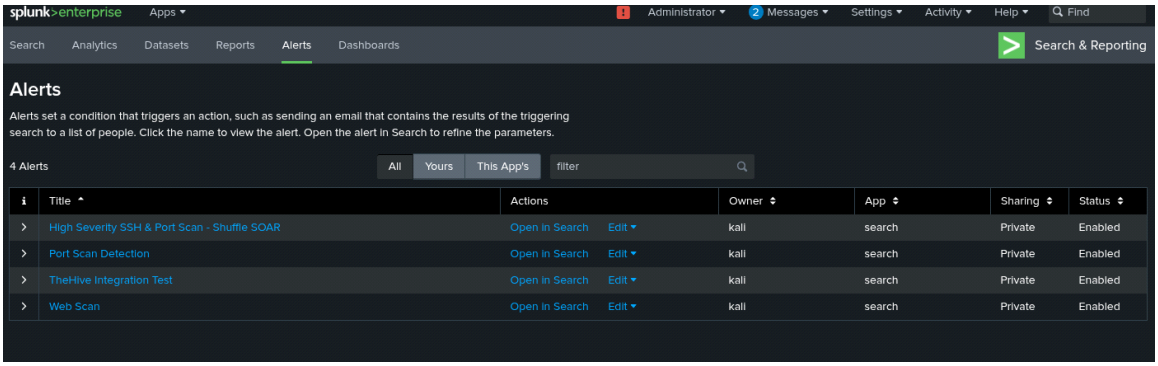
### Example: High Severity SSH & Port Scan Alert

1. Navigate to Settings → Searches, Reports, and Alerts
2. Click 'New Alert'
3. Configure the search query (SPL):

```
index="main" sourcetype="suricata" event_type="alert"
| eval severity_level=case(
    severity==1, "high",
    severity==2, "medium",
    severity==3, "low",
    1=1, "unknown")
| search severity_level="high"
```

1. Set alert conditions: Trigger: Number of Results, Condition: is greater than 0, Throttle: 5 minutes

## Screenshot: Custom SPL Query

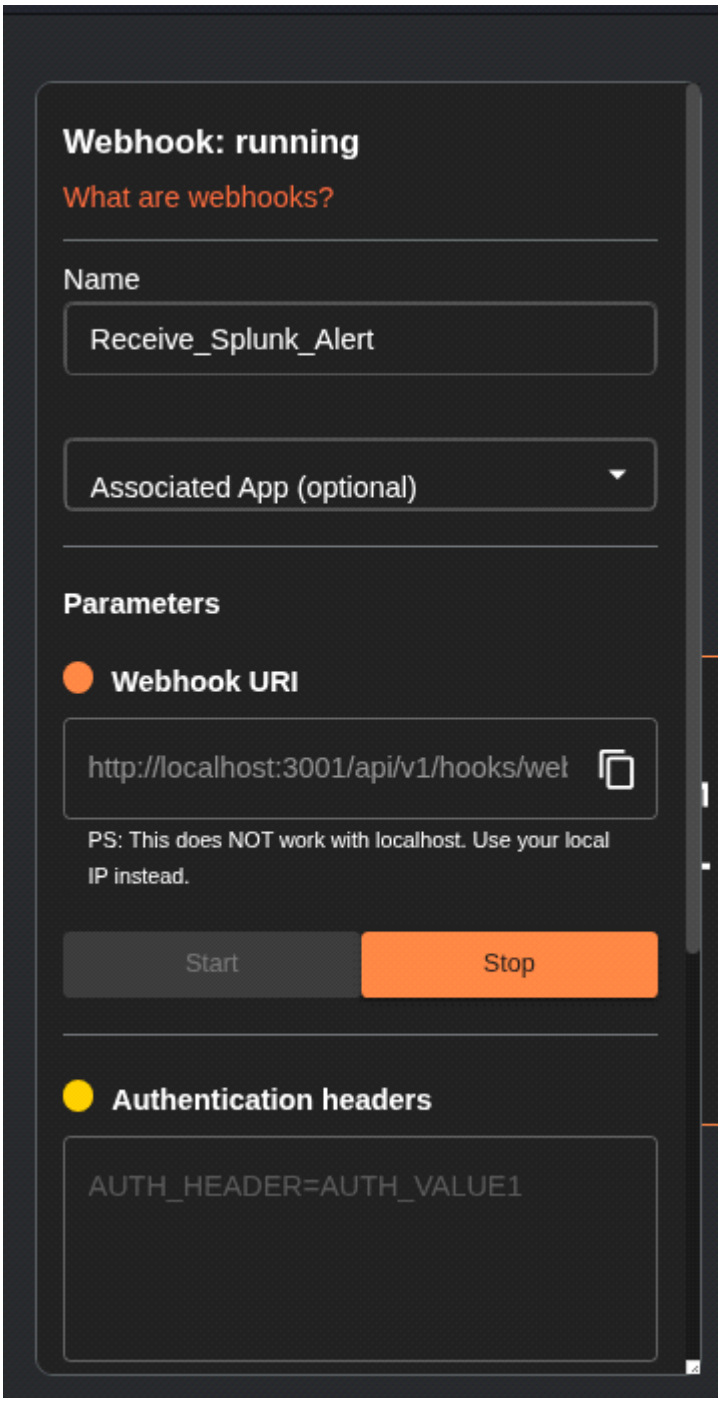



## 3.7 Configure Webhook Alert Action

1. In the alert configuration, click 'Add Actions'
2. Select 'Webhook'
3. Configure webhook settings:

- URL: http://localhost:3001/api/v1/hooks/webhook_[_id]
- Method: POST
- Content-Type: application/json

## 📸 Screenshot: Webhook Configuration



## Part 4: Shuffle SOAR Installation and Configuration

### 4.1 Shuffle Overview

Shuffle is an open-source Security Orchestration, Automation and Response (SOAR) platform that enables automated incident response workflows. It serves as the orchestration layer connecting Splunk, OpenAI, Jira, and Slack.

### 4.2 Installation Options

Shuffle can be deployed in two ways:

- Cloud Version: Free hosted version at shuffler.io (recommended for testing)
- Self-Hosted: Docker-based deployment for full control

#### Option 1: Using Shuffle Cloud

- Visit https://shuffler.io
- Create a free account
- No installation required

#### Option 2: Self-Hosted Shuffle with Docker

1. Install Docker and Docker Compose:

```
sudo apt install docker.io docker-compose -y

sudo systemctl enable docker

sudo systemctl start docker
```

1. Clone Shuffle repository:

```
git clone https://github.com/Shuffle/Shuffle

cd Shuffle
```

1. Start Shuffle:

```
sudo docker-compose up -d
```

1. Access Shuffle at http://localhost:3001

### 4.3 Creating the Automated Response Workflow

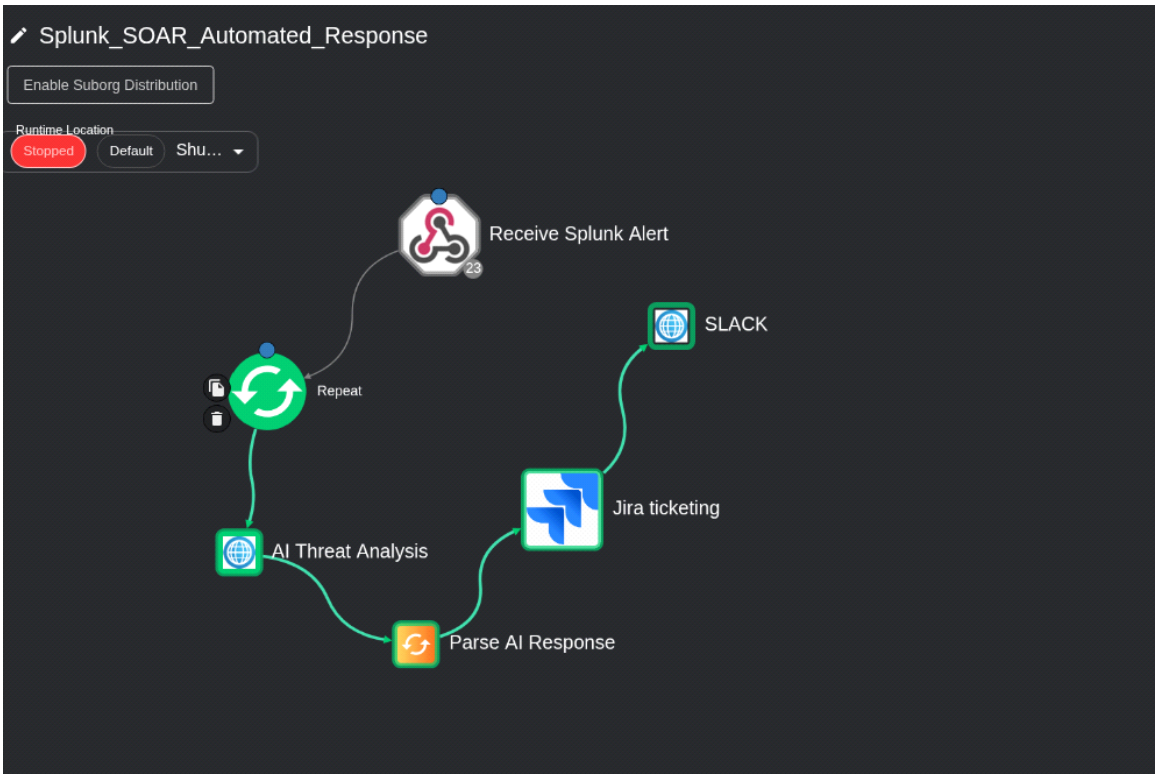
#### Workflow Overview

The automated response workflow consists of 5 main components:

1. Receive Splunk Alert - Webhook trigger from Splunk

2. AI Threat Analysis - OpenAI GPT-4 analyzes the alert
3. Parse AI Response - Extract structured data from AI output
4. Create Jira Ticket - Automated incident ticket creation
5. Send Slack Notification - Alert SOC team in real-time

### 📸 Screenshot: Complete Workflow




### Step 1: Configure Webhook Trigger

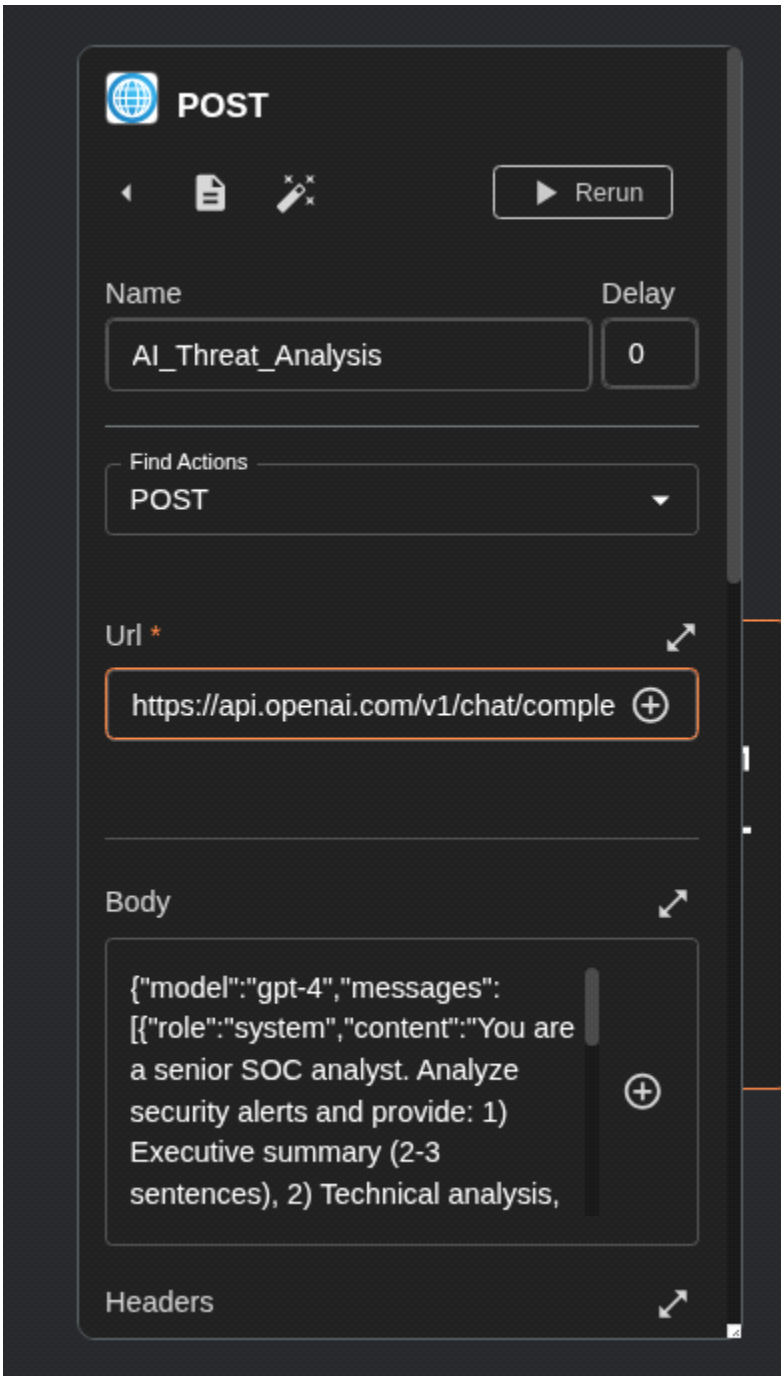
1. In Shuffle, create a new workflow
2. Add a 'Webhook' node
3. Configure: Name: Receive_Splunk_Alert, Start node: Enabled
4. Copy the webhook URL for use in Splunk

### Step 2: Configure OpenAI Integration

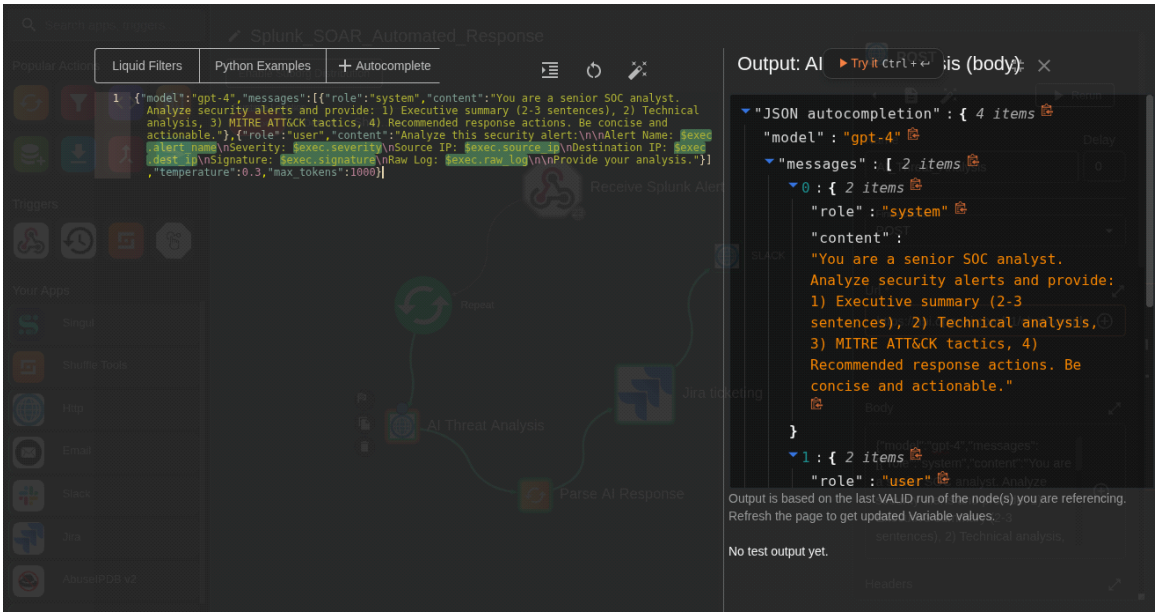
1. Add an 'HTTP' or 'OpenAI' app node
2. Name: AI_Threat_Analysis

3. Configure authentication (see Part 5 for API setup)
4. Set the prompt to analyze alerts

**📸 Screenshot: OpenAI Node Settings**



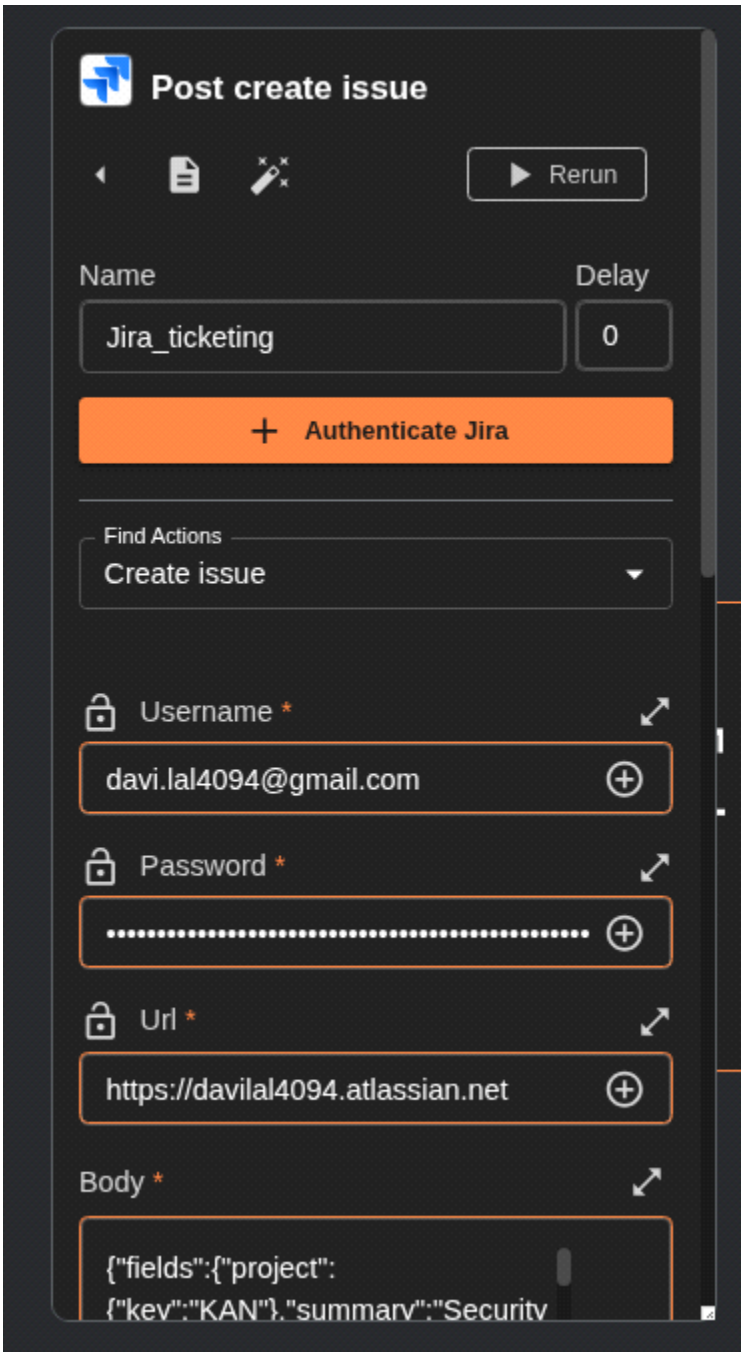
## 📸 Screenshot: OpenAI JSON Body



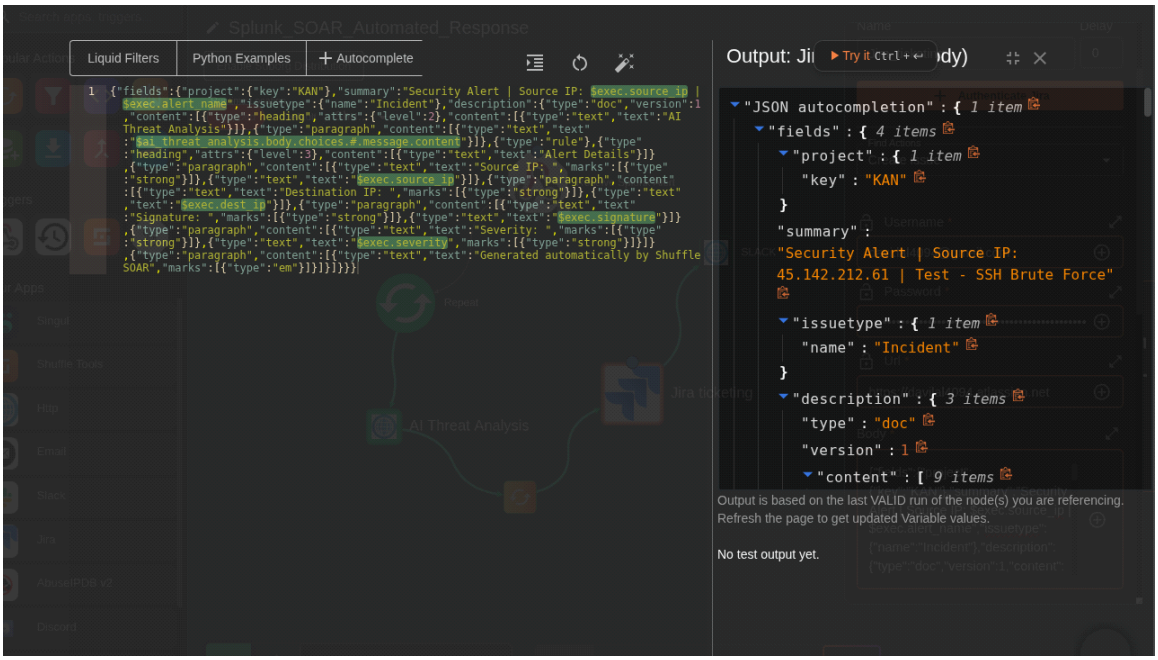

### Step 3: Configure Jira Integration

1. Add a 'Jira' app node
2. Name: Jira_ticketing
3. Action: Create issue
4. Configure authentication (see Part 6)

## 📸 Screenshot: Jira Node Settings



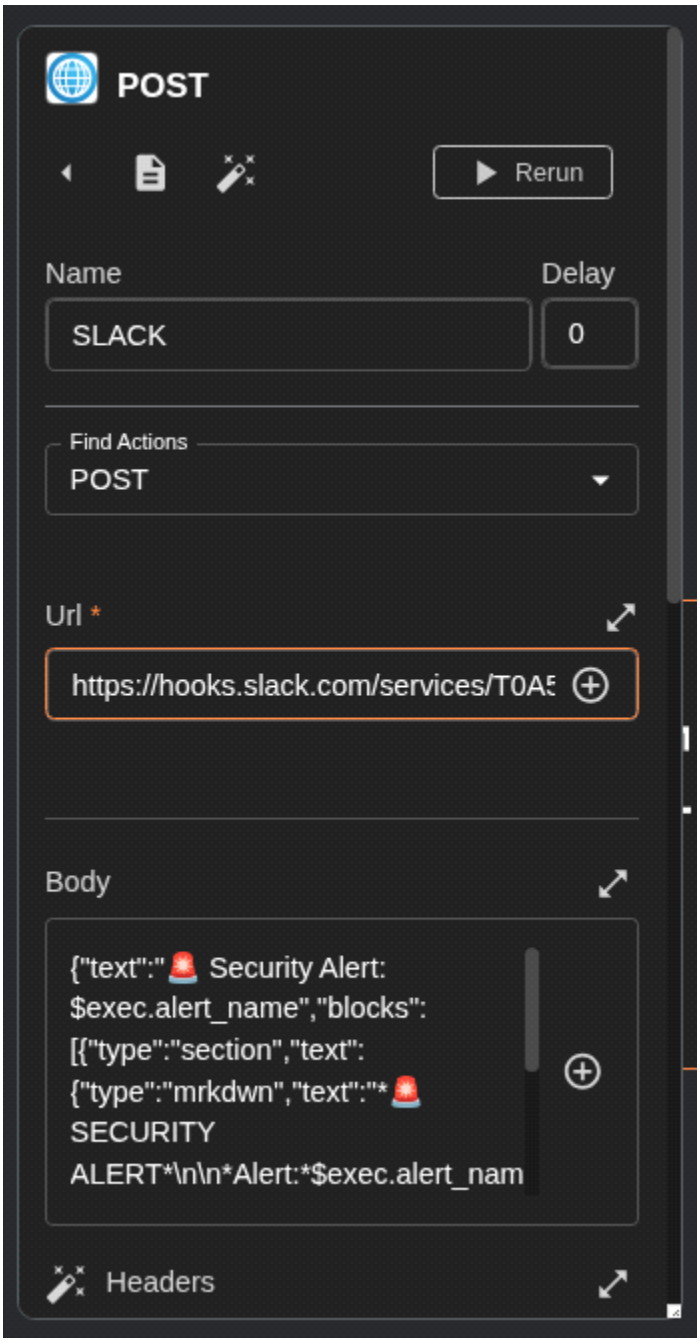
### 📸 Screenshot: Jira JSON Body



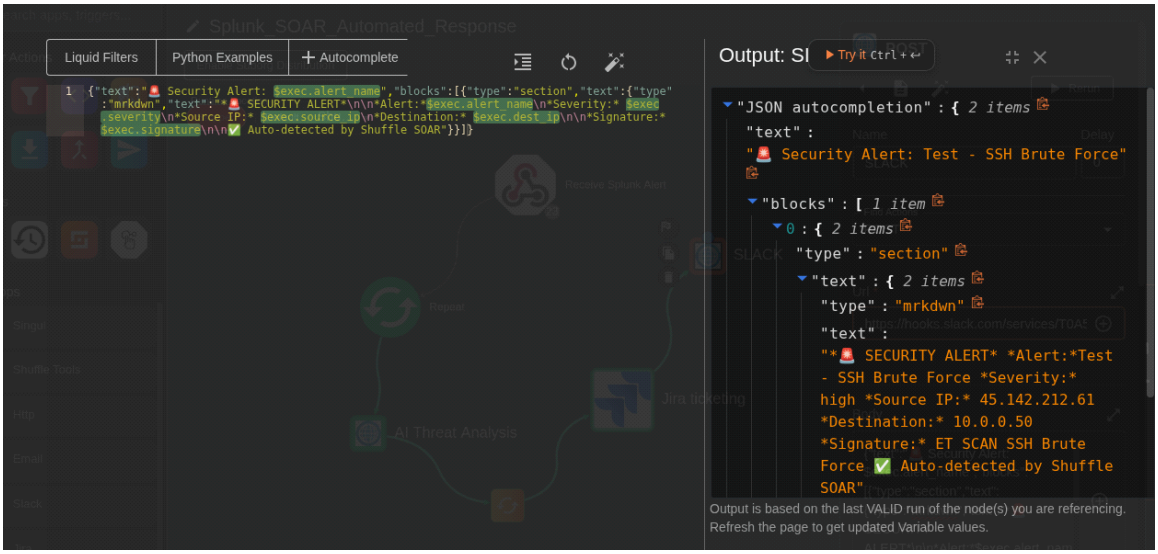

## Step 4: Configure Slack Integration

1. Add an 'HTTP' or 'Slack' app node
2. Name: SLACK
3. Action: POST
4. Configure webhook URL (see Part 7)

## 📸 Screenshot: Slack Node Settings



### 📸 Screenshot: Slack JSON Body




### 4.4 Connect Workflow Nodes

1. Connect nodes in sequence:

- Receive Splunk Alert → AI Threat Analysis
- AI Threat Analysis → Parse AI Response
- Parse AI Response → Jira ticketing
- Jira ticketing → SLACK

1. Add a 'Repeat' node after Receive Splunk Alert for continuous processing
2. Save and activate the workflow

## Part 5: OpenAI API Configuration

### 5.1 Create OpenAI API Account

1. Visit https://platform.openai.com
2. Create an account or sign in

3. Navigate to API Keys section
4. Click 'Create new secret key'
5. Copy and securely store the API key

## 5.2 Configure API in Shuffle

### Authentication Settings

- URL: https://api.openai.com/v1/chat/completions
- Method: POST
- Headers:

```
Authorization: Bearer YOUR_API_KEY

Content-Type: application/json
```

### Request Body Configuration

Configure the JSON body to send alert data to GPT-4:

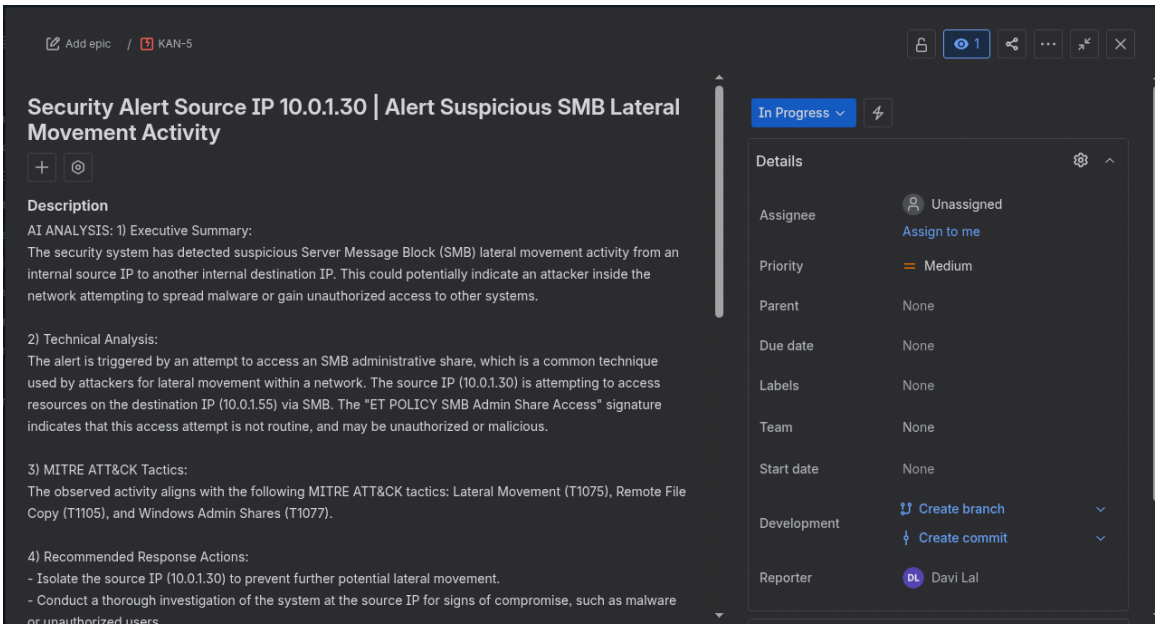
```
{
  "model": "gpt-4",
  "messages": [
    {
      "role": "system",
      "content": "You are a senior SOC analyst..."
    },
    {
      "role": "user",
      "content": "Analyze this security alert..."
    }
  ],
  "temperature": 0.3,
  "max_tokens": 1000
}
```

### 5.3 AI Analysis Output

The AI analysis provides:

1. Executive Summary: High-level overview for management
2. Technical Analysis: Detailed security assessment
3. MITRE ATT&CK Tactics: Framework mapping
4. Recommended Response Actions: Immediate and investigative steps

### 📸 Screenshot: AI Analysis Summary




## Part 6: Jira Integration Configuration

### 6.1 Jira Setup Requirements

- Jira Cloud or Server instance
- Admin or project permissions
- API token (for Cloud) or password (for Server)

### 6.2 Create Jira API Token

1. Go to https://id.atlassian.com/manage-profile/security/api-tokens
2. Click 'Create API token'
3. Provide a label (e.g., 'Shuffle SOAR Integration')
4. Copy and securely store the token

### 6.3 Create Security Incidents Project

1. In Jira, navigate to Projects → Create project
2. Select 'Kanban' template
3. Name: Security_Incidents
4. Key: KAN
5. Create the project

### 6.4 Configure Jira in Shuffle

#### Authentication Settings

- Username: your_email@example.com
- Password/API Token: YOUR_API_TOKEN
- URL: https://your-domain.atlassian.net

#### Action Configuration

Body configuration for creating issues:

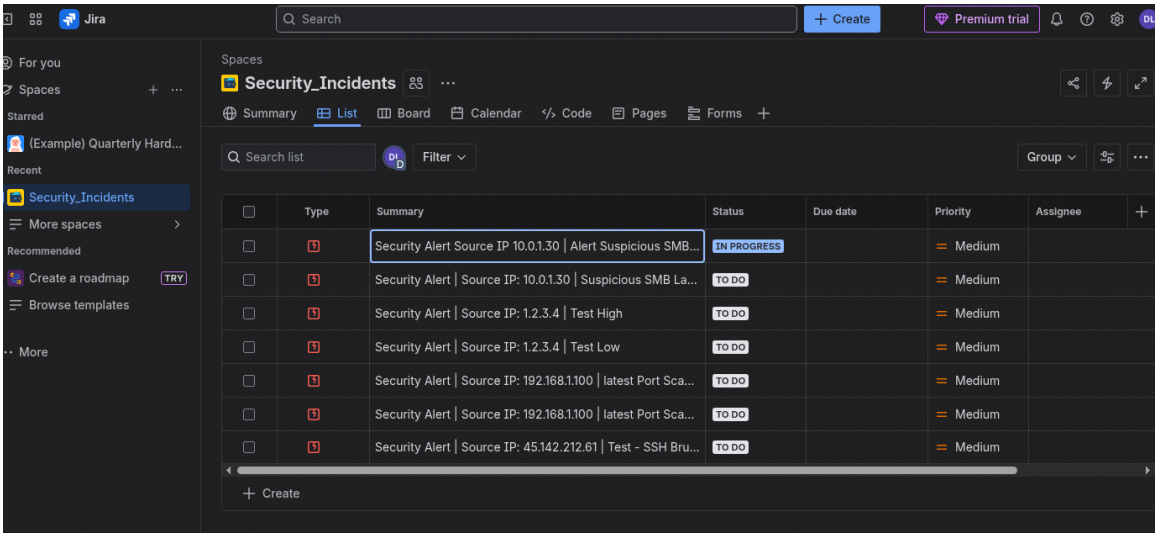
```
{
  "fields": {
    "project": { "key": "KAN" },
    "summary": "Security Alert | Source IP: $exec.source_ip",
    "issuetype": { "name": "Incident" },
    "description": { ... AI analysis content ... }
  }
}
```

### 6.5 Verify Ticket Creation

After the workflow runs, verify tickets are created with:

- Descriptive summary with source IP and alert name
- Complete AI analysis in description
- Appropriate priority (Medium by default)
- Status: In Progress or To Do

#### Screenshot: Jira Ticket List




## Part 7: Slack Integration Configuration

### 7.1 Slack Setup Requirements

- Slack workspace (free or paid)
- Permission to add apps and incoming webhooks
- Dedicated SOC channel (recommended)

### 7.2 Create Slack Incoming Webhook

1. Visit https://api.slack.com/apps
2. Click 'Create New App' → 'From scratch'

3. App Name: 'SOC Alerts', select your workspace
4. Navigate to 'Incoming Webhooks' → Toggle to On
5. Click 'Add New Webhook to Workspace'
6. Select channel (e.g., #soc-alerts)
7. Copy the Webhook URL

### 7.3 Create SOC Channel

1. In Slack, create a new channel: soc_analysis
2. Make it private if handling sensitive data
3. Add relevant team members
4. Invite the SOC Alerts app to the channel

### 7.4 Configure Slack in Shuffle

#### HTTP POST Settings

- URL: YOUR_SLACK_WEBHOOK_URL
- Method: POST
- Headers: Content-Type: application/json

#### Message Body Configuration

```
{
  "text": "🚨 SECURITY ALERT",
  "blocks": [
    {
      "type": "section",
      "text": {
        "type": "mrkdwn",
        "text": "*Alert:* $exec.alert_name..."
      }
    }
  ]
```

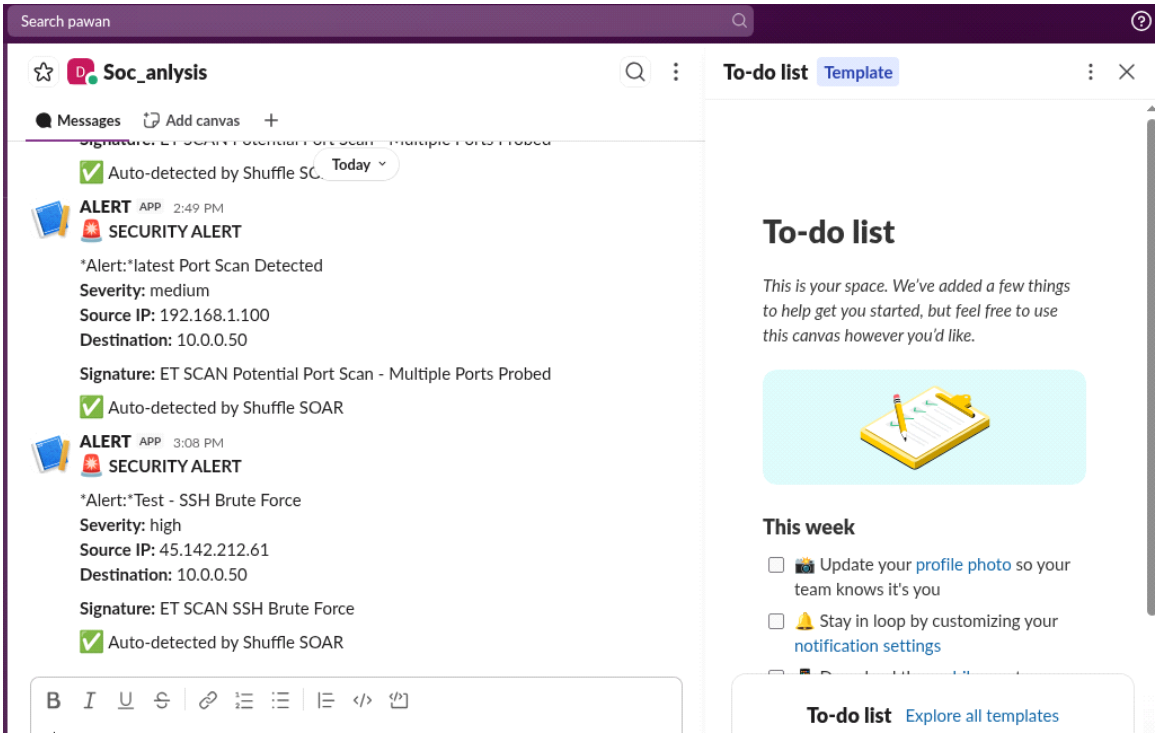
}

## 7.5 Verify Slack Notifications

Verify notifications include:

- Alert emoji (🚨)
- Alert name and severity
- Source and destination IPs
- Signature information
- Auto-detection confirmation

### 📸 Screenshot: Slack Notification



## Part 8: Testing and Validation

### 8.1 End-to-End Workflow Test

#### Generate Test Traffic

1. Port scan simulation:

```
nmap -sS 192.168.1.100
```

1. SSH brute force simulation:

```
hydra -l test -P /usr/share/wordlists/rockyou.txt ssh://192.168.1.100
```

1. Web vulnerability scan:

```
nikto -h http://192.168.1.100
```

#### Verify Alert Detection

1. Check Suricata logs:

```
sudo tail -f /var/log/suricata/fast.log
```

1. Check Zeek logs:

```
tail -f /opt/zeek/spool/zeek/conn.log
```

1. Verify logs appear in Splunk:

```
index="main" sourcetype="suricata" event_type="alert"
```

### 8.2 Validate Automation Workflow

| Stage | Component | Validation Method |
|---|---|---|
| 1 | Splunk Alert | Check Splunk Alerts page |
| 2 | Webhook Delivery | Check Shuffle execution logs |
| 3 | AI Analysis | Verify OpenAI node executed |
| 4 | Jira Ticket | Check Security_Incidents project |
| 5 | Slack Notification | Confirm alert in SOC channel |

### 8.3 Performance Metrics

The completed SOC lab achieves:

- Mean Time to Detect (MTTD): < 5 seconds

- Mean Time to Respond (MTTR): < 30 seconds (98% reduction)
- Alert to Ticket Creation: < 20 seconds
- Alert to SOC Notification: < 25 seconds
- Automation Success Rate: > 99%

## Part 9: Troubleshooting Guide

### 9.1 Suricata Issues

#### Suricata Not Detecting Traffic

1. Verify network interface:

```
ip addr show
```

1. Check Suricata is running on correct interface:

```
sudo cat /var/log/suricata/suricata.log | grep interface
```

1. Ensure interface is in promiscuous mode:

```
sudo ip link set eth0 promisc on
```

#### No Alerts Generated

- Update rules: sudo suricata-update
- Check rule loading:

```
sudo grep 'rule reload complete' /var/log/suricata/suricata.log
```

- Verify HOME_NET is configured correctly

### 9.2 Zeek Issues

#### Zeek Not Starting

- Check configuration:

```
sudo /opt/zeek/bin/zeek -C -i eth0
```

- View crash reports:

```
sudo /opt/zeek/bin/zeekctl diag
```

- Redeploy:

```
sudo /opt/zeek/bin/zeekctl deploy
```

## 9.3 Splunk Issues

### Data Not Indexing

- Check data input status in Settings → Data Inputs
- Verify file permissions:

```
sudo chmod 644 /var/log/suricata/eve.json
```

- Check internal logs:

```
index="_internal" ERROR
```

### Webhook Not Triggering

- Test webhook manually:

```
curl -X POST http://localhost:3001/api/v1/hooks/webhook_ID
```

- Verify webhook URL in Splunk alert action
- Check Shuffle is running and accessible

## 9.4 Shuffle SOAR Issues

### Workflow Not Executing

- Check workflow is activated
- Review execution logs for errors
- Verify all node credentials are valid
- Test each node individually

### API Authentication Errors

- OpenAI: Verify API key is valid and has credits
- Jira: Ensure API token hasn't expired
- Slack: Regenerate webhook if needed

## Part 10: Best Practices and Maintenance

### 10.1 Security Best Practices

#### Credential Management

- Store all API keys securely (use environment variables)
- Rotate credentials every 90 days
- Use least-privilege access for service accounts
- Enable multi-factor authentication

#### Network Security

- Restrict Splunk web interface to internal networks
- Use TLS/SSL for all API communications
- Implement firewall rules to limit service exposure
- Consider network segmentation for SOC infrastructure

### 10.2 Regular Maintenance Tasks

| Task | Description | Frequency |
|---|---|---|
| Update IDS Rules | Run suricata-update | Weekly |
| Log Rotation | Archive old logs | Daily |
| System Updates | Update OS and tools | Monthly |
| Alert Tuning | Reduce false positives | Monthly |
| Backup Configuration | Backup configs and workflows | Weekly |
| Performance Review | Check system resources | Weekly |

### 10.3 Monitoring and Alerting

- Monitor Suricata and Zeek service health
- Track Splunk indexing rates and license usage
- Review Shuffle execution logs for failures
- Set up alerts for critical system failures
- Monitor OpenAI API usage and costs

### 10.4 Continuous Improvement

- Regularly review false positive rates

- Expand detection coverage with new signatures
- Enhance AI prompts for better analysis
- Add new integrations as needed
- Document lessons learned from incidents

## Conclusion

This SOC lab demonstrates a complete, enterprise-grade security operations infrastructure. By integrating network detection, SIEM correlation, automated orchestration, and AI-powered analysis, the platform achieves rapid threat detection and response capabilities.

The 98% reduction in mean time to respond represents a transformative improvement in security operations efficiency. Automated workflows eliminate manual processes, reduce human error, and ensure consistent incident handling.

Key achievements:

- Real-time threat detection across network layers
- Automated AI-powered threat analysis
- Seamless integration of detection, analysis, ticketing, and notification
- Comprehensive logging and forensic capabilities
- Scalable architecture ready for production

## Appendix A: Quick Reference Commands

### Suricata Commands

```
# Check status

sudo systemctl status suricata


# Update rules
```

```
sudo suricata-update


# Restart service
sudo systemctl restart suricata


# View logs
sudo tail -f /var/log/suricata/eve.json
```

### Zeek Commands

```
# Check status
sudo /opt/zeek/bin/zeekctl status


# Deploy configuration
sudo /opt/zeek/bin/zeekctl deploy


# Restart Zeek
sudo /opt/zeek/bin/zeekctl restart


# View logs
tail -f /opt/zeek/spool/zeek/conn.log
```

### Splunk Commands

```
# Start Splunk
sudo /opt/splunk/bin/splunk start


# Stop Splunk
sudo /opt/splunk/bin/splunk stop


# Restart Splunk
sudo /opt/splunk/bin/splunk restart
```

## Useful Splunk SPL Queries

```
# View all Suricata alerts

index="main" sourcetype="suricata" event_type="alert"


# High severity only

index="main" sourcetype="suricata" severity=1


# Top source IPs

index="main" sourcetype="suricata" | top src_ip
```