RFC – AI Agent for Network Intrusion Detection & Response

Team: CAAVe (Alexandra Viorel, Catalin Albu)

1. Project Description

The proposed system is a Network Defense Agent that uses AI to detect, classify, and respond to cyber threats in real time. The agent integrates multiple AI models and processes of reasoning to adapt dynamically to network behavior, compared with the traditional Intrusion Detection Systems (IDS) that depend on fixed rule sets.

It continuously ingests network telemetry from simulated traffic, analyzes it using specialized models, and interacts with external tools: a firewall manager, an IP Address reputation checker, incident management systems using Jira API, and an alerting system using Slack. The agent will act as a "brain" that autonomously determines the best response plan: whether to block, limit, report, or escalate to a human analyst for review. Using an LLM-based reporting model, it will create incident summaries for further review.

The project's goal is to provide a realistic autonomous cybersecurity co-pilot, that can protect network environments using intelligent reasoning and integrating multiple tools.

2. Problem Definition

There are several modern challenges to network intrusion detection, such as:

- Changing Attack Patterns: Systems that rely on signatures are vulnerable to zero-day or polymorphic attacks;
- High False Positive Rates: Static thresholds and one-size-fits-all rules cause alert fatigue;
- Scalability and Data Volume: Real-time network traffic can generate thousands of events per second;
- Contextual Decision Making: The inability of legacy systems to reason about severity, intent or business impact prevents them from making contextual decisions;
- Automation Gaps: Human analysts must manually respond to the detected threats, which causes delays in mitigation

Moreover, in addition to identifying malicious activity, the technical challenge is to reason and act in a dynamic, context-aware manner. The agent must continuously learn from fresh data and human feedback while coordinating several AI models and tools in real time.

3. State of the Art

3.1. Traditional IDS/IPS solutions such as Snort, Suricata and Zeek rely on signature-based detection and rule engines.

- **Pros**: They have low computational costs, are mature ecosystems, and are fast, reliable and accurate for known threats.
- **Cons**: They cannot adapt to zero-day or polymorphic attack, require frequent manual rule updates and produce a high number of false positives when dealing with unknown attack vectors.

3.2. UNSW-NB15 and CICDS2017 datasets enabled supervised training of classifiers (using CNN, Random Forest or SVM Machine Learning algorithms).

- **Pros**: They are better at detecting unknown patterns.
- **Cons**: They are still static, and don't use multi-model reasoning or contextual decision-making.

3.3. Anomaly detection with AutoEncoder in Network Traffic.

- **Pros**: They can detect unsupervised the unusual behavior.
- **Cons**: They do not classify or respond to attacks autonomously.

3.4. New technologies such as Cortex XSOAR or Splunk SOAR show the potential of autonomous AI agents that communicate with various tools and systems.

- **Pros**: They enable automated and context-aware actions by integrating detection, reasoning and response into a single workflow.
- **Cons**: Most of them are difficult to integrate and not open for further research.

4. Proposed Solution

4.1 Data Sources:

The system architecture is focused on an AI agent that continuously monitors and analyzes over two data sources.

- **Data Source 1**: a real-time simulated network traffic, which includes packets, flows, IPs and ports.
- **Data Source 2**: Threat Intelligence or Reputation Database: contain known malicious IPs, domains, scores.

Raw traffic is consumed by an ETL (Extract Transform Load) Pipeline, which then converts it into feature vectors and stores them into a database. Then, depending on the state of the network, the agent coordinates a series of AI models and tools.

4.2. AI Models:

The system integrates several AI models, each of them serving a unique role in the detection of attacks and response process.

4.2.1. Anomaly Detector:

- It identifies unusual or suspicious network flows by comparing real-time traffic patterns with statistical foundations or learned normal behavior.
- **Type**: Unsupervised model (Autoencoder or Isolation Forest)

4.2.2. Threat Classifier:

- Classifies the detected anomalies into distinct attack categories, such as DDos, port scanning or brute-force.
- **Type**: Supervised learning model (CNN)

4.2.3. Risk Scoring Model:

- Combines the anomaly level, threat classification results and the targeted asset's criticality to compute a contextual severity score.
- **Type**: Regression model

4.2.4. LLM Incident Summarizer:

- It creates a summary and suggested actions for the detected incidents.
- **Type**: Generative AI model (GPT)

4.3. Tool Interactions:

The agent interacts dynamically with multiple tools, each of them having a specific role. When events occur, these tools enable the system to respond autonomously and contextually.

4.3.1. Firewall Rule Manager: Decides automatically whether to block/unblock suspicious IP addresses. It allows active threats to be immediately restricted.

4.3.2. Threat Intelligence Lookup: Enhance detection results by searching for IPs/domains in global reputation data.

4.3.3. Jira Ticket Creator: By integrating the Jira API, it will create incidents.

4.3.4. Slack Bot: Helps for a fast response to high-risk events, by sending real-time alerts on a dedicated channel.

4.4. Decision Flow:

4.4.1. Perception: use reputation data to enhance live traffic.

4.4.2. Analysis: perform anomaly detection, threat classification and risk scoring.

4.4.3. Reasoning: evaluate severity, confidence and business context.

4.4.4. Planning: select the right tools and actions: block, throttle or escalate.

4.4.5. Action: execute the plan using APIs: firewall, Jira or Slack.

4.4.6. Explain: create reports using LLM.

4.5. End-to-End Stack:

- Data Simulation: Python script
- Backend API: Django REST API
- Database: PostgreSQL
- Frontend: React + TS
- AI Models & Agent Orchestrator: Python + specific Machine Learning libraries (Scikit-learn, TensorFlow)