

Threat Explorer: A Comparative Study of Agentic Architectures and Visualization Strategies for Conversational Cybersecurity Analytics

Kostadin Devedzhiev
University of Cambridge
Cambridge, United Kingdom
kgd26@cam.ac.uk

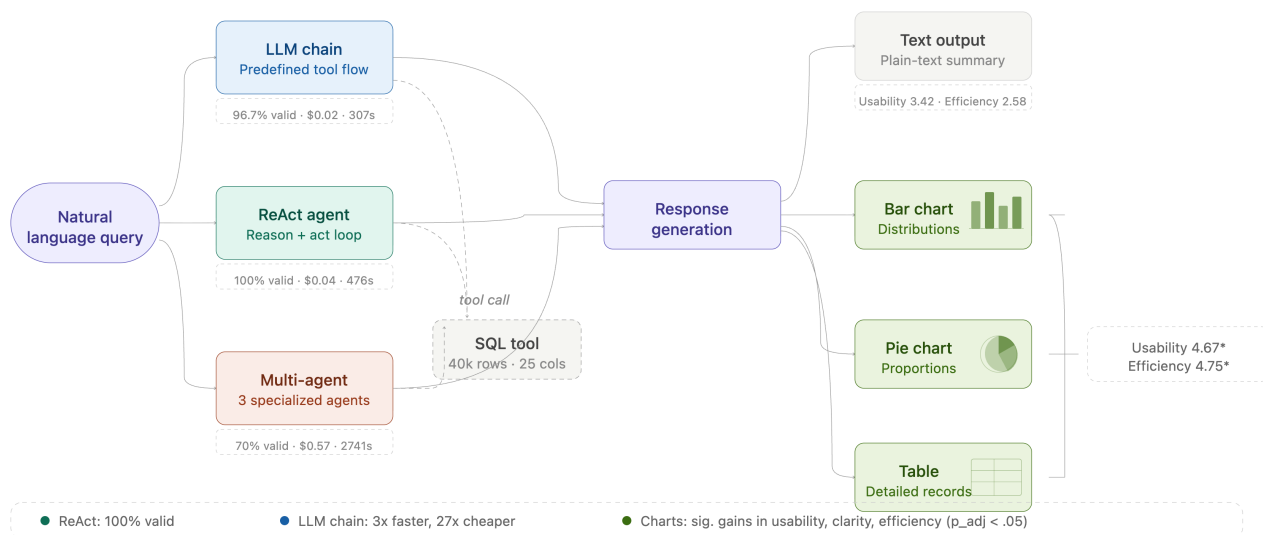


Figure 1: System overview of Threat Explorer. A natural language query is routed to one of three agent architectures—LLM Chain, ReAct, or Multi-Agent—each invoking an SQL tool over a 40k-row cybersecurity dataset before generating a response. Key performance metrics are annotated per agent.

Abstract

This paper presents **Threat Explorer**, a conversational AI system for cybersecurity threat analysis that translates natural language queries into SQL over a 40,000-record attack dataset. We evaluate three agentic architectures—an LLM Chain, a ReAct agent, and a multi-agent system—across retrieval accuracy, latency, cost, and perceived quality. A separate within-subjects user study ($N=12$) compares text-only and chart-augmented responses using Likert-scale surveys with Holm–Bonferroni-corrected Wilcoxon tests. The ReAct agent achieves the highest query validity (100%) while the LLM Chain offers the best cost–speed trade-off; chart-based output yields statistically significant improvements in usability, clarity, and efficiency. We discuss the socio-technical risks of visually compelling but potentially inaccurate LLM output and describe transparency mechanisms to support human-in-the-loop validation. The source code is available at <https://github.com/kostadindev/Threat-Explorer>.

CCS Concepts

• **Human-centered computing** → **Interactive systems and tools**; • **Computing methodologies** → **Natural language processing**; • **Security and privacy** → **Intrusion/anomaly detection and malware mitigation**.

Keywords

Retrieval-Augmented Generation, Agentic AI, Natural Language to SQL, Cybersecurity Analytics, Human–AI Interaction, Data Visualization

1 Introduction

Threat Explorer is a chatbot for analyzing cybersecurity threats in a database using natural language, inspired by Stellar Cyber’s AI Investigator [10]. The system lets security experts analyze data quickly, without deep knowledge of the schema or query language, in collaboration with AI. The database uses Inscibo’s dataset from Kaggle [4], containing 40,000 records and 25 columns (e.g., Timestamp, Source IP Address, Attack Type, Anomaly Score, IDS/IPS

Alerts). Threat Explorer can answer queries such as “show the last 10 attacks with an anomaly score over 75” or “show the number of high-severity attacks by type.”

The system implements three agent architectures: a custom LLM chain, a ReAct agent [13], and a multi-agent orchestration (Figure 1). This paper investigates the system along two dimensions:

- (1) **Technical Robustness:** Evaluating the retrieval performance, speed, cost efficiency, and perceived usefulness of different agent architectures for response generation in multi-turn dialogues.
- (2) **Design Effectiveness:** Comparing structured output strategies (plain text vs. charts) in terms of usability, helpfulness, and cognitive load through a controlled user study.

2 Baseline System Design

Threat Explorer is a **Retrieval-Augmented Generation (RAG)** [6] system powered by OpenAI’s **GPT-4o mini** [7] for its cost-effectiveness. The workflow consists of four stages: (1) receiving a natural language prompt, (2) constructing an SQL query via an agent, (3) executing the query to retrieve relevant records, and (4) generating a natural language report of the results. The baseline agent is a pre-defined LLM chain equipped with tools for executing SQL queries and inspecting the database schema. The backend is built with FastAPI [2] and uses LangChain [5] and CrewAI [1] for agent orchestration. The frontend is built with React [9], and the database is SQLite [8].

3 Technical Experimental Study

3.1 Research Question

To what extent do different agentic architectures affect the accuracy, speed, cost, and perceived utility of the RAG-based dialogue system?

3.2 Setup and Evaluation Metrics

A test set of 10 dialogues with 30 system turns was annotated, each containing a rubric and ground-truth SQL queries of varying complexity across use cases including attack analysis, protocol investigation, severity analysis, and temporal analysis. An example rubric structure is shown in Figure 2.

Each turn is evaluated on the following metrics:

- **Retrieval Performance:** *Query Validity* (percentage of executable queries) and *Pattern Match Accuracy* (correctness of query results against the ground truth).
- **Cost:** Token consumption and total price in USD.
- **Speed:** Wall-clock time in seconds to produce a response.
- **Generation Quality (LLM-as-a-Judge):** Scores on a 1–5 scale for factuality, helpfulness, and overall quality, following the LLM-as-a-Judge framework [3].

3.3 Results

3.4 Discussion

The **ReAct agent** achieved 100% query validity: every SQL query produced through its iterative reasoning-and-acting loop was executable. It tied with the LLM Chain at 86.7% pattern match accuracy. The **LLM Chain** delivered the fastest response times and the lowest

Table 1: Retrieval performance across agents ($N=30$ turns).

Metric	LLM Chain	ReAct	Multi-Agent
Query Validity	96.7%	100.0%	70.0%
Pattern Match	86.7%	86.7%	43.3%

Table 2: Efficiency and token distribution across agents ($N=30$ turns).

Metric	LLM Chain	ReAct	Multi-Agent
Total Time (s)	306.9	475.8	2741.4
Input Tokens	84,265	190,517	2,047,357
Output Tokens	13,275	23,259	444,435
Total Tokens	97,540	213,776	2,491,792
Total Cost (\$)	\$0.021	\$0.043	\$0.574

Table 3: Generation quality (LLM-as-a-Judge, 1–5 scale).

Metric	LLM Chain	ReAct	Multi-Agent
Factuality	4.67	4.53	3.70
Helpfulness	4.40	4.57	4.17
Overall Quality	4.37	4.47	3.67

token consumption and cost, making it the most efficient option for budget-constrained deployments.

The multi-agent architecture performed worst across all metrics, primarily due to poor inter-agent orchestration. Improving it was deprioritized because the simpler agents already achieved high accuracy at lower cost. The LLM judge scored the LLM Chain highest for factuality but favored ReAct for helpfulness and overall quality.

These trade-offs motivate a system design that lets users switch agents based on their requirements and budget. Figure 3 illustrates an example where the multi-agent system produced a convincing but hallucinated output, whereas the ReAct agent returned accurate results for the same prompt. The multi-agent orchestration consists of three specialized agents: an SQL Query Analyst, a Cybersecurity Threat Analyst, and a Report Formatter. System prompts specifying tools, schema, reasoning strategies, roles, and expected output formats are available in the project repository.

4 Design-Focused Experimental Study

4.1 Research Question

Does augmenting responses with LLM-generated visualizations, compared to text-only output, improve the usability, helpfulness, clarity, trust, and efficiency of a cybersecurity analytics chatbot?

4.2 Setup

A toggle button in the interface switches between two output modes: **text** (plain-text summaries) and **chart** (visual data presentations). Twelve postgraduate students at the University of Cambridge used the chatbot with each mode for five minutes in a within-subjects

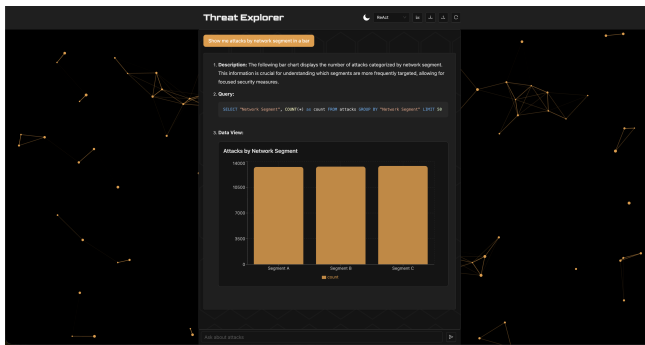
```
# ===== DIALOGUE 9: IDS/IPS ALERTS ANALYSIS =====
{
  "id": "d9",
  "title": "IDS/IPS Alert Investigation",
  "category": "alert_analysis",
  "description": "User investigates IDS/IPS alert patterns",
  "turns": [
    {
      "turn_id": 1,
      "user_message": "Show me attacks that triggered IDS/IPS alerts",
      "expected_query_pattern": r'SELECT.*"IDS/IPS Alerts".*LIMIT',
      "expected_visualization": "db-table",
      "rubric": {
        "query_validity": "Must filter for non-empty IDS/IPS alerts",
        "correctness": "Must return attacks with IDS/IPS alerts",
        "visualization": "Should use table for detailed records"
      }
    },
    {
      "turn_id": 2,
      "user_message": "What attack types generated these alerts?",
      "expected_query_pattern": r'SELECT.*"Attack Type".*COUNT',
      "expected_visualization": "db-chart",
      "rubric": {
        "query_validity": "Must count attack types",
        "context_awareness": "Should maintain IDS/IPS alert filter from turn 1",
        "visualization": "Should use chart for attack type distribution"
      }
    },
    {
      "turn_id": 3,
      "user_message": "Show me the network segments where these alerts occurred",
      "expected_query_pattern": r'SELECT.*"Network Segment".*COUNT',
      "expected_visualization": "db-chart",
      "rubric": {
        "query_validity": "Must count by network segment",
        "context_awareness": "Must maintain IDS/IPS alert context",
        "visualization": "Should use chart for segment distribution"
      }
    }
  ]
},
```

Figure 2: Example of a test rubric with ground-truth SQL query and expected results.

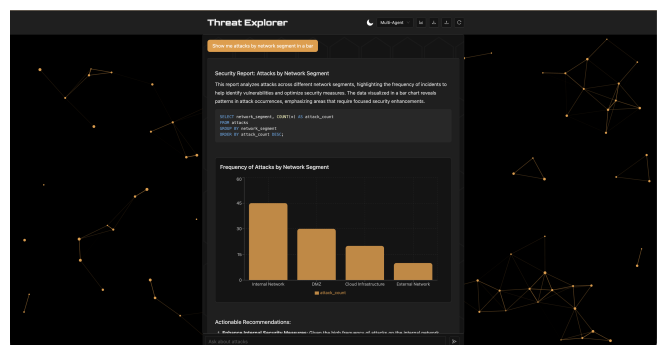
design. Half of the participants held undergraduate degrees in computer science; none had formal cybersecurity experience. A post-interaction survey (Table 4) collected responses on five dimensions using a 5-point Likert scale, plus a forced-choice preference question.

4.3 Results and Statistical Analysis

Likert-scale responses were analyzed using one-sided Wilcoxon Signed-Rank Tests ($\alpha = 0.05$), chosen given the a priori hypothesis of improvement and the small sample size. After Holm–Bonferroni correction for multiple comparisons, the chart version scored higher across all five dimensions, with statistically significant differences ($p_{adj} < 0.05$) in usability, clarity, and efficiency (Table 5).



(a) ReAct agent: correct result.

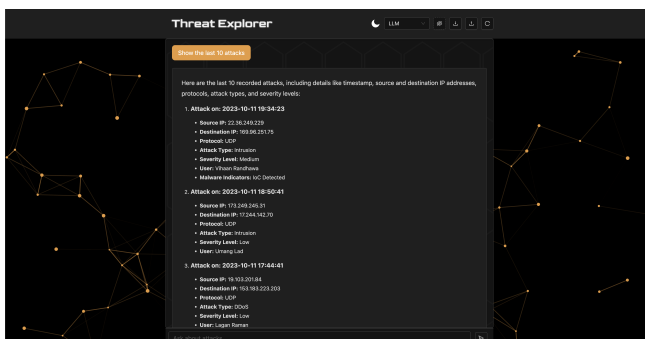


(b) Multi-agent: hallucinated output.

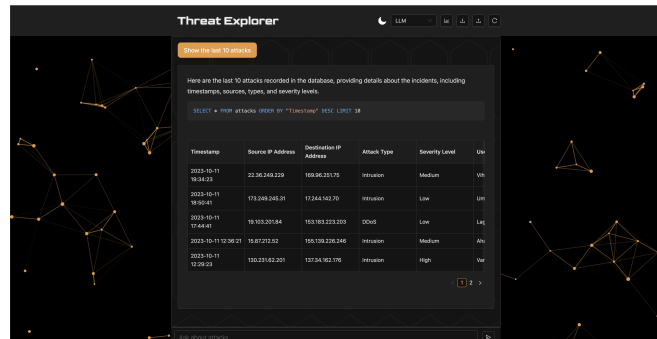
Figure 3: Comparison of ReAct vs. multi-agent outputs for the same query.

Table 4: Post-interaction survey questions.

Dimension	Survey Question	Scale
A. Usability	The responses were presented in a well-organized way.	Likert 1–5
B. Helpfulness	The system gave me the right level of detail.	Likert 1–5
C. Clarity	I could identify the key evidence supporting the conclusion.	Likert 1–5
D. Trust	I trust the chatbot’s output for the tasks I performed.	Likert 1–5
E. Efficiency	This version helped me understand the output quickly.	Likert 1–5
F. Preference	Which version do you prefer?	Chart / Text



(a) Text-only response.



(b) Chart-augmented response.

Figure 4: Comparison of text-only vs. chart-based responses.

Table 5: Design experiment results with Holm–Bonferroni correction ($N=12$, Likert 1–5).

Dim.	\bar{x}_T	\bar{x}_C	s_T	s_C	p	p_{adj}
A. Usab.	3.42	4.67	0.79	0.49	.002	.008*
B. Help.	3.50	4.42	1.09	0.67	.045	.078
C. Clar.	3.17	4.50	0.83	0.67	.002	.008*
D. Trust	3.83	4.58	0.94	0.67	.039	.078
E. Effic.	2.58	4.75	1.31	0.45	.001	.005*

\bar{x}_T/\bar{x}_C = Text/Chart mean; s_T/s_C = SD. * $p_{adj} < 0.05$.

4.4 Discussion

The results indicate that structured visual output provides substantial user experience benefits, supporting the decision to make visualizations the default output mode in Threat Explorer.

4.4.1 Socio-Technical Reflection on Trust. The elevated trust and helpfulness scores for chart-based output warrant careful interpretation. Users may perceive data visualizations as inherently more credible than plain text—a concern in an LLM-powered system that can generate incorrect queries, hallucinate explanations, or misinterpret data. This risk is amplified for users without domain expertise, who may over-rely on visually compelling output and accept results without scrutiny.

4.4.2 Mitigation and Transparency. To improve transparency, agents always include the SQL queries used in each response with syntax highlighting, making them inspectable (see Figure 5). Agents also provide explanations of their reasoning process. In future iterations, queries should be user-editable so that analysts can refine data exploration even when agents fail to fulfill requirements. Such interactive features would shift the user from a passive recipient to an active validator, promoting human-in-the-loop machine learning [12] and human-AI collaboration [11].

5 Final System Design

The final system supports switching between all three agent architectures at runtime. Key features include conversation logging (Figure 6), uploading and downloading dialogues in JSON format, and toggling between text and chart output modes. Figure 5 shows example dialogues from the system. The full source code, documentation, evaluation scripts, and survey data are available at <https://github.com/kostadindev/Threat-Explorer>.

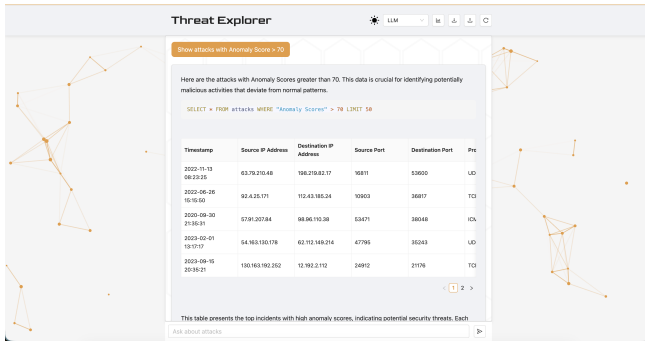
6 Conclusion

This work explored technical and design aspects of Threat Explorer, a conversational AI system for cybersecurity data analysis. The technical study showed that the ReAct agent is the most reliable (100% query validity), while the LLM Chain is the most efficient (3× faster, 27× cheaper than the multi-agent system), motivating the design decision to let users switch between agents. The design experiment yielded statistically significant results demonstrating that chart-augmented responses improve usability ($p_{adj} = 0.008$), clarity ($p_{adj} = 0.008$), and efficiency ($p_{adj} = 0.005$). However, visually compelling output carries the risk of masking inaccurate LLM-generated content; Threat Explorer mitigates this by exposing its reasoning process and SQL queries.

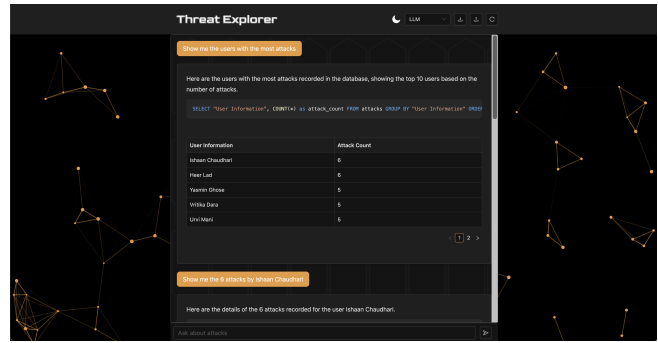
Future work includes stronger guardrails against adversarial prompts, editable inline SQL queries and charts, support for additional chart types, streaming partial responses over WebSockets, adaptive agent routing, improved prompting and orchestration strategies, short- and long-term conversational memory, and observability tooling.

References

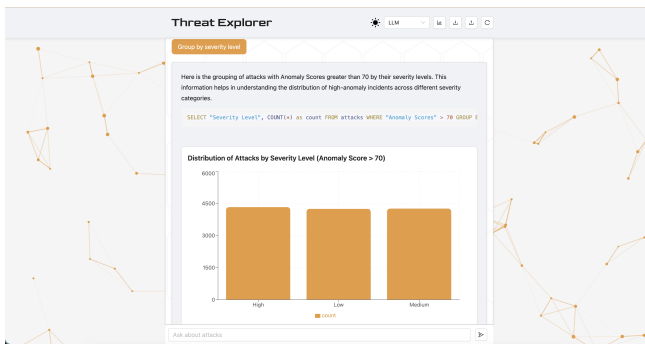
- [1] CrewAI. 2024. CrewAI – Platform for Multi AI Agents Systems. <https://www.crewai.com/>
- [2] Sebastián Ramírez. 2023. FastAPI Documentation. <https://fastapi.tiangolo.com/>
- [3] Jiawei Gu, Xuhui Jiang, Zhichao Wang, Taiping Li, and Shuai Zhao. 2024. A Survey on LLM-As-a-Judge. *arXiv preprint arXiv:2411.15594*.
- [4] Inciribo. 2024. Cyber Security Attacks. Kaggle. <https://www.kaggle.com/datasets/teaminciribo/cyber-security-attacks>
- [5] Harrison Chase. 2022. LangChain. <https://github.com/langchain-ai/langchain>
- [6] Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. 2020. Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks. In *Advances in Neural Information Processing Systems*, Vol. 33. 9459–9474.
- [7] OpenAI. 2024. GPT-4o Mini: Advancing Cost-Efficient Intelligence. <https://openai.com/index/gpt-4o-mini-advancing-cost-efficient-intelligence/>
- [8] Python Software Foundation. 2024. sqlite3 – DB-API 2.0 Interface for SQLite Databases. *Python 3 Documentation*. <https://docs.python.org/3/library/sqlite3.html>
- [9] Meta Open Source. 2025. React. <https://react.dev/>
- [10] Stellar Cyber. 2025. AI Investigator – Natural Language Threat Hunting with XDR. <https://stellarcyber.ai/ai-investigator-natural-language-threat-hunting/>
- [11] Vanshika Vats, Marzia Hoque Tania, Dolly Sharma, and Akshi Kumar. 2024. A Survey on Human-AI Teaming with Large Pre-Trained Models. *arXiv preprint arXiv:2403.04931*.
- [12] Xingjiao Wu, Luwei Xiao, Yixuan Sun, Junhang Zhang, Tianlong Ma, and Liang He. 2022. A Survey of Human-In-The-Loop for Machine Learning. *Future Generation Computer Systems* 135 (Oct. 2022), 364–381. 10.1016/j.future.2022.05.014
- [13] Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. 2022. ReAct: Synergizing Reasoning and Acting in Language Models. *arXiv preprint arXiv:2210.03629*.



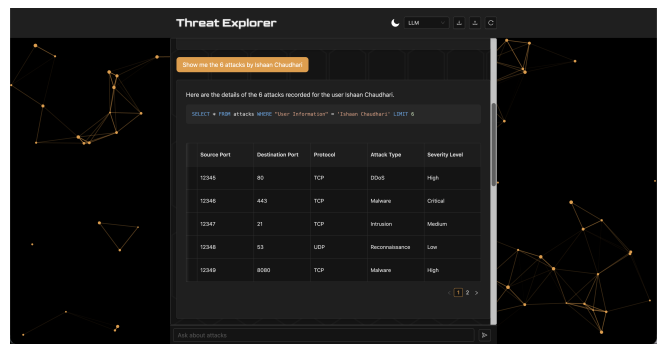
(a) Query 1



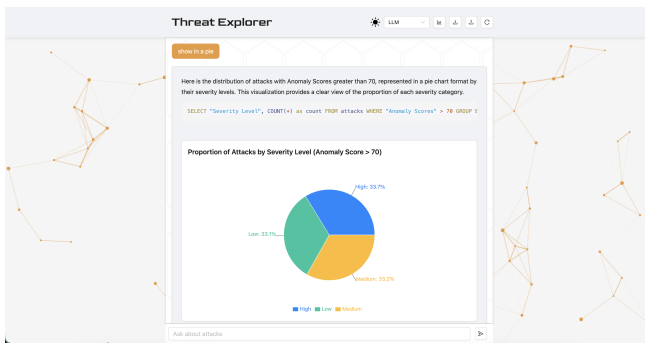
(b) Response 1



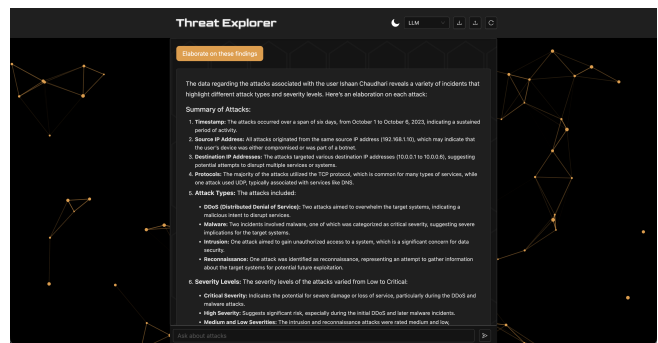
(c) Query 2



(d) Response 2



(e) Query 3



(f) Response 3

Figure 5: Example dialogues: user queries (left) and system responses with charts and SQL transparency (right).

```
backend > logs > {} conversation_conv_1765414802243_z0cosm4ne_20251211_010105.json > ...
1
2 {
3   "conversation_id": "conv_1765414802243_z0cosm4ne",
4   "agent_type": "llm",
5   "started_at": "2025-12-11T01:01:05.792966",
6   "messages": [
7     {
8       "role": "user",
9       "content": "Show attacks with Anomaly Score > 70",
10      "timestamp": "2025-12-11T00:51:14.2472"
11    },
12    {
13      "role": "assistant",
14      "content": "Here are the recent attacks with an Anomaly Score greater than 70, highlighting significant incidents that may require attention.\n\n```\nSELECT * FROM attacks WHERE \\'Anomaly Scores
15      "timestamp": "2025-12-11T00:51:14.2842",
16      "agent_type": "llm"
17    },
18    {
19      "role": "user",
20      "content": "Now filter by attack type DDoS",
21      "timestamp": "2025-12-11T00:52:13.5852"
22    },
23    {
24      "role": "assistant",
25      "content": "Here are the recent DDoS attacks with an Anomaly Score greater than 70, highlighting significant incidents that may require attention.\n\n```\nSELECT * FROM attacks WHERE \\'Anomaly S
26      "timestamp": "2025-12-11T00:52:13.6222",
27      "agent_type": "llm"
28    },
29    {
30      "role": "user",
31      "content": "Filter by high severity level",
32      "timestamp": "2025-12-11T00:53:42.3622"
33    },
34    {
35      "role": "assistant",
36      "content": "Here are the recent DDoS attacks with an Anomaly Score greater than 70 and a Severity Level classified as High.\n\n```\nSELECT * FROM attacks WHERE \\'Anomaly Scores\' > 70 AND \\'Atta
37      "timestamp": "2025-12-11T00:53:42.4802",
38      "agent_type": "llm"
39    },
40    {
41      "role": "user",
42      "content": "Tell me about these users",
43      "timestamp": "2025-12-11T00:55:28.3142"
44    },
45    {
46      "role": "assistant",
47      "content": "Here are the details of the users associated with the recent high-severity attacks:\n\n## 1. Akarsh Khurana\n- **Attack Timestamp:** 2023-02-24 06:39:25\n- **Source IP Address:** 57.7.
48      "timestamp": "2025-12-11T00:55:28.3652",
49    }
50  ]
51 }
```

Figure 6: Conversation log output from Threat Explorer.