EXCYTIN-BENCH: EVALUATING LLM AGENTS ON CYBER THREAT INVESTIGATION

Anonymous authors
Paper under double-blind review

ABSTRACT

We present **ExCyTIn-Bench**, the first benchmark to **E**valuate an LLM agent **x** on the task of **Cy**ber **T**hreat **In**vestigation through security questions derived from investigation graphs. Real-world security analysts must sift through a large number of heterogeneous alert signals and security logs, follow multi-hop chains of evidence, and compile an incident report. With the developments of LLMs, building LLM-based agents for automatic thread investigation is a promising direction. To assist the development of LLM agents, we construct a benchmark from a controlled Azure tenant including a SQL environment covering 57 log tables from Microsoft Sentinel and related services, and 589 automatically generated test questions. We leverage security logs extracted with expert-crafted detection logic to build threat investigation graphs, and then generate questions with LLMs using paired nodes on the graph, taking the start node as background context and the end node as answer. Anchoring each question to these explicit nodes and edges not only provides automatic, explainable ground truth answers but also makes the pipeline reusable and readily extensible to new logs. This also enables the automatic generation of procedural tasks with verifiable rewards, which can be naturally extended to training agents via reinforcement learning. Our comprehensive experiments with different models confirm the difficulty of the task: with the base setting, the average reward across all evaluated models is 0.249, and the best achieved is 0.368, leaving substantial headroom for future research.

1 INTRODUCTION

The growing reliance on digital services for critical functions worldwide underscores the need to secure our digital future. Meanwhile, cyberattacks are growing in quantity, variety, and sophistication. For example, cloud environment intrusions increased by 75% from 2022 to 2023 [8]. Although traditional defenses like behavioral analysis, malware signature matching, and anomaly detection can mitigate threats [15; 12], attackers continue to develop tactics to evade them [31]. Thus, human-led threat investigations have become critical, requiring analysts to manually go through system and network logs, apply reasoning, and leverage domain expertise to detect and respond to threats [2].

Meanwhile, advancement of Large Language Models (LLMs) has enabled astonishing achievements in complex tasks [54; 14; 45; 47; 46], that LLM agents can understand observations and select actions in complex environments such as code interpretation and database interaction to perform sequential actions [53; 63; 50; 49; 25]. Also, LLMs trained with enormous corpora of text provide them a wealth of knowledge across a range of domains [61], including cybersecurity knowledge. Thus, Cyber-Security threat investigation is a promising area for the application of LLM-based autonomous agents, as previous works have shown that LLMs are capable of multi-step observation, reasoning, and actions, which are key components for successful investigation and detection of potential threat actors and indicator of compromises (IoCs) [13; 23; 62].

To assist development, a rigorous, standardized benchmark is needed to evaluate the cybersecurity investigation capability of LLM agents. The benchmark should resemble the real-world threat investigation scenario with a critical mass of security event logs, heterogeneity across real-world multi-stage security incident types. However, existing literature focuses on evaluating LLMs from a knowledge memorization perspective [56; 22], instead of targeting the security investigation and reasoning ability of LLM agents. For example, CTIBench [1] constructs a multi-choice Q&A to

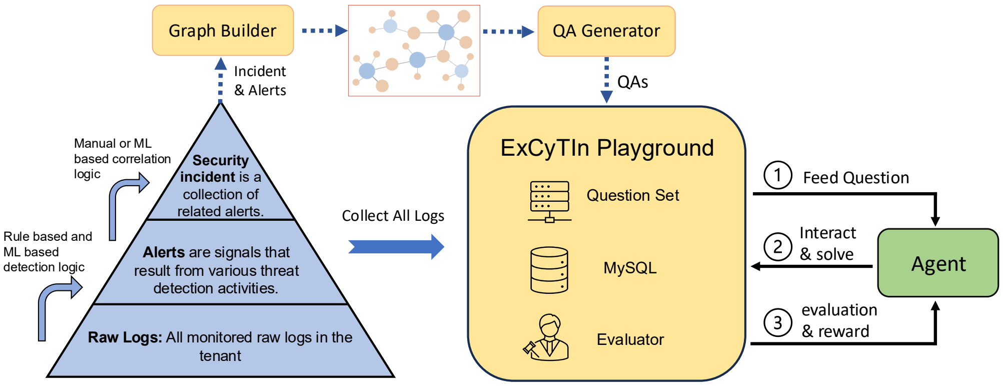

Graph Builder
QA Generator
Incident & Alerts
QAs
Manual or ML based correlation logic
Security incident is a collection of related alerts.
Rule based and ML based detection logic
Alerts are signals that result from various threat detection activities.
Raw Logs: All monitored raw logs in the tenant
Collect All Logs
ExCyTIn Playground
Question Set
MySQL
Evaluator
1 Feed Question
2 Interact & solve
3 evaluation & reward
Agent

Figure 1: Overview of the ExCyTin Bench. 1. (Left Triangle) We collect the raw logs and security alerts & incidents from original tenant. 2. We construct graphs with the incidents and alerts data, then use the graphs to generate question-answer pairs. 3. We build an MySQL environment for all the logs. 4. (Middle Square) Finally, we build an environment that that agents can interact with to solve questions and get evaluated.

evaluate LLMs on Cyber Threat Intelligence(CTI) knowledge, SECURE [5] evaluates LLMs on security extraction and understanding.

To fill this gap, we build ExCyTIn-Bench, to evaluate LLM agents on cybersecurity threat investigation. Construction of the benchmark consisted of three steps: *1. Data Collection.* From a fictional Microsoft Azure tenant, we collect 59 distinct tables of logs consisting of 8 different cyberattacks. *2. Question Generation.* We propose a principled method to construct bipartite incident graphs from the alerts and entities involved in the attacks, then use it to generate QA pairs, resulting in a test dataset of 589 questions. *3. Environment Construction.* We construct a MySQL Docker image to be a reinforcement learning (RL) environment in which agents can submit queries and receive feedback similar to InterCode [53]. The database queries are treated as actions, and the execution result as observations. We use an LLM as an evaluator by default, but deterministic checking of the answer is also available. Since our questions are generated from paths in the incident graphs, we can assign partial rewards if the agents find any intermediate information along the path.

We test with a wide range of current language models, including proprietary and open-source models, and models with different sizes and types (chat, reasoning, etc) in Section 4. We include a detailed analysis from the perspective of performance, behavior, and efficiency. We found that our benchmark is challenging even among the latest, highest-performing models, with `o4-mini` achieving the highest reward of 0.368. We also test different methods (e.g., ReAct [55], Expel [64], Best-of-N, Self-Reflection [38]) to help understand how different prompting and test-time scaling strategies affect performance on our benchmark. To summarize, our contributions are the following:

- We release ExCyTIn-Bench, which, to the best of our knowledge, is the first benchmark to evaluate LLM agents on threat investigation tasks. The benchmark is built on real-world security logs generated from simulated real-world attacks, and requires the agents to query logs to investigate.
- We propose a new QA generation method with LLM from bipartite incident investigation graphs, where each question is non-repetitive and anchored to explicit nodes and edges.
- We conduct a comprehensive experiment on the proposed benchmark to provide understanding and insights for future directions.

2 Background and Related Work

2.1 Background

Cybersecurity threat investigation systematically probes digital environments to detect, analyze, and mitigate malicious activity [18; 7]. Threat analysts serve as cyber-detectives: they parse vast logs, link evidence from diverse sources, and judge potential threats. Since modern infrastructures produce vast volumes of data daily, a central challenge is efficiently identifying and correlating threat signals, with other key skills required (e.g., *evidence gathering and synthesis*, more in Appendix C). Our benchmark is designed to assess LLM agents on all of these capabilities.

Threat Investigation Graphs. Incident graphs portray multi-stage attacks by linking alerts, events, and indicators of compromise (IoCs) into a unified view. Nodes denote alerts (e.g, suspicious file downloads) or entities (e.g, user accounts) while edges capture their relationships (e.g., a phishing email that triggers a malicious download). Sequencing nodes along the kill chain (reconnaissance, intrusion, persistence, etc.) exposes adversary tactics, surfaces patterns, and clarifies next steps for responders. In our benchmark, we utilize these threat investigation (or incident graphs) for grounding LLMs during question generation and evaluation.

2.2 Related Work

LLMs have become a promising foundation to build agents for various complicated tasks [50; 64; 14; 6; 59]. Our goal is to create a cybersecurity threat investigation benchmark for LLM agents, which is closely related to LLM-based agents and cybersecurity: this requires LLM agents to have cybersecurity domain expertise and knowledge to be able to explore system logs, analyze suspicious behavior, and answer security-related questions. On the other hand, we build a SQL environment for LLM agents to interact with, to test models' ability on effective and efficient SQL generation.

LLMs in Cybersecurity. Most recent LLM-based cybersecurity efforts focus on knowledge memorization and information extraction [30; 24; 29; 40; 56]. CTIBench [1] evaluates LLMs' understanding of the threat landscape via the MITRE ATT&CK framework [32], while Crimson [21] fine-tunes an LLM to map CVEs to MITRE ATT&CK techniques and generate actionable insights. SECURE [5] benchmarks models on security extraction, comprehension, and reasoning. [39] aggregates public CTI reports with structured intelligence, and [11] extracts threat behaviors from unstructured OSCTI text. Perrina et al.'s tool [35] produces CTI reports from entity graphs, and [37] uses reinforcement learning to simulate LLM-driven attacks on network topologies. Finally, CyBench [60] focuses on capture-the-flag (CTF) tasks. Although some prior work also employs graphs [11; 15], our graph-based approach differs in both concept and construction.

Benchmark LLMs in Interactive Environments. [16] benchmarks LLM-based agents on data analysis tasks through an execution environment. [33] introduces a database question answering system that LLMs need to interact with a SQL interpreter, reason, and organize the results. [53] creates interactive code environments (Bash, SQL, and Python) based on static datasets [57; 28; 4] for LLM to act on. [17] build a dataset of machine learning tasks that LLMs need to perform actions like reading/writing files, executing code. SWE-Bench [20] builds a dataset on real-world software engineering problems. [52] introduces a benchmark that supports cross-environment GUI tasks over websites, desktop computers, or mobile phones.

LLMs in Text-to-SQL Text-to-SQL benchmarks [57; 26] are proposed to test models on generating SQL queries given a question. Many works have been proposed to solve this task [3; 41]. [44] proposes a multi-agent framework, including a decomposer agent for Text-to-SQL generation and two auxiliary agents for tool execution and error refinement. [10] provides a systematic review of prompt engineering for Text-to-SQL generation. C3-SQL [9], StructGPT [19], Din-SQL [36] propose frameworks targeting SQL generation that consist of several stages with different strategies such as self-consistency [48] or query decomposition. StateFlow [51] introduces a framework with state and transitions to control the data exploration and selection in SQL tasks.

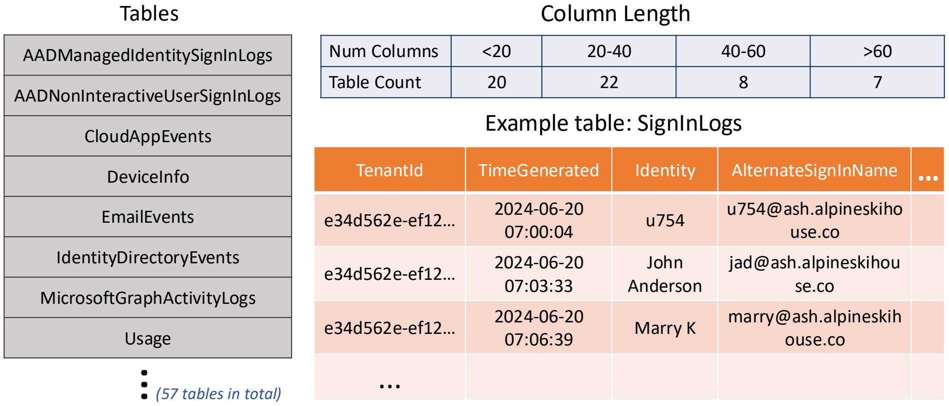
3 ExCyTIn-Bench

See Figure 1 For an overview of constructing the benchmark. Below we dive into details of data collection, question generation and environment setup in Section 3.1, 3.2 and 3.3.

3.1 Data collection

We collected data from the Azure tenant "Alpine Ski House", which is a fictional Microsoft company used for demonstration of security products. This tenant is a complete SIEM (Security Information and Event Management) environment with all necessary event logs from security products like MS Sentinel and Defender. For example, "EmailEvents" records email events, including receivers and senders, and other tables such as "EmailAttachmentInfo" provide additional details. We collect 57 tables in total (Figure 2) with the columns containing different data types. We note that one tenant is enough, as it is to provide the schema of the database neeeded for security investigation.

From the tenant, we simulate 8 distinct, non-repetitive incidents (Table 1). Each incident corresponds to a complex real-world attack kill chain used against Azure clients that have happened be-



Tables

AADManagedIdentitySignInLogs
AADNonInteractiveUserSignInLogs
CloudAppEvents
DeviceInfo
EmailEvents
IdentityDirectoryEvents
MicrosoftGraphActivityLogs
Usage

⋮ *(57 tables in total)*

Column Length

Num Columns	<20	20-40	40-60	>60
Table Count	20	22	8	7

Example table: SignInLogs

TenantId	TimeGenerated	Identity	AlternateSignInName	...
e34d562e-ef12...	2024-06-20 07:00:04	u754	u754@ash.alpineskihouse.co	
e34d562e-ef12...	2024-06-20 07:03:33	John Anderson	jad@ash.alpineskihouse.co	
e34d562e-ef12...	2024-06-20 07:06:39	Marry K	marry@ash.alpineskihouse.co	
...				

Figure 2: Overview of the database. We collect a total of 57 tables. The number of columns from these tables vary from 8 to 139.

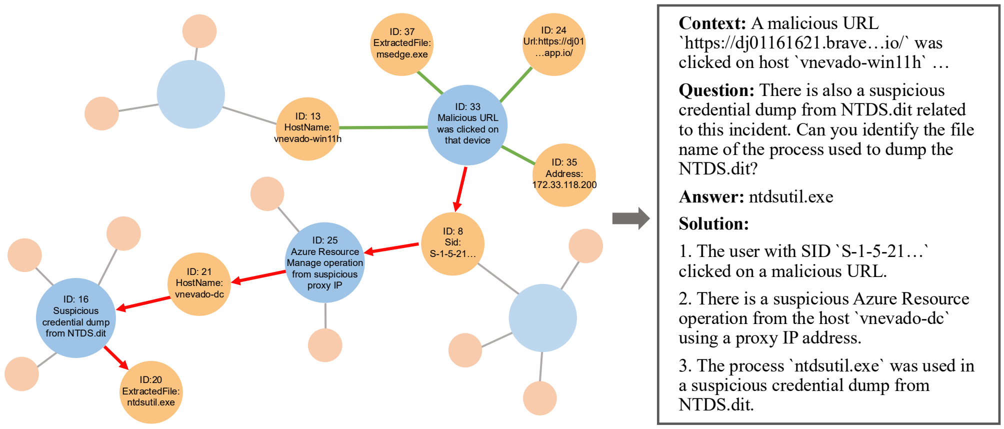
ID	Title	Time	#Alerts	#Qs	Labels
5	Operation Alpine Lockbit: Multi-Stage Manatee Tempest Ransomware Campaign	47	2770	98	Ransomware, Credential Theft, Lateral Movement
34	Macro-Enabled Document Dropper with PowerShell Backdoor Deployment	80	430	82	Backdoor, Persistence
38	Multi-Stage Fileless Attack	25	157	11	Process Injection, Covert C2
39	Operation Alpine Storm: Human-Operated intrusion chain	475	1873	98	Phish URL, Domain Compromise, Credential Harvest, Defense Evasion
55	Phishing-Enabled ADFS Key Exfiltration and Lateral Movement Campaign	7739	1093	100	Spear-Phish Email, Lateral Movement, Persistence
134	Multi-Stage Business Email Compromise and Account Takeover Attack	17	352	57	BEC Fraud, Compromised Credentials, Password Spray
166	SAP Financial Manipulation Attack	88	430	87	BEC Fraud, SAP Access, Data Exfiltration
322	Domain Credential Harvest Attack	11	352	56	Proxy Evasion, Credential Phish, Domain Compromise

Table 1: Table of collected incidents, including time span (in minutes), number of alerts, number of questions generated, and labels.

fore, including various attack stages and many attack techniques (See Figure 27-41 in appendix for detailed report). These known attacks have been studied, and their behavior is tracked and recorded in table"SecurityIncident" and "SecurityAlert". For each incident, we segment the corresponding logs based on time ranges, collecting logs from one hour before the first event to one hour after the last. These time windows vary between 2 hours and 5 days, with no temporal overlap between attacks. In addition, we also compiled a continuous log stream covering the entire sequence of incidents. This unified database spans 44 days from the first activity in the earliest incident to the last activity of the most recent incident. This configuration mirrors real-world conditions, where SOC analysts typically do not have prior knowledge of attack timelines, making detection and analysis considerably more challenging. We also processed all data to replace Personally Identifiable Information (PII) of security analysts who use this tenant for various research purposes like understanding threats, simulating attacks, etc., see Appendix D.4 for details.

3.2 Question Generation

We want to create questions that can measure LLM investigation skill: to answer a question, the LLM agent needs to probe into the log data, analyze, and link related events to find the source of the malicious activity. While manual creation of these questions is expensive and does not scale, LLMs have proven useful in QA generation [5; 1]. A straightforward way is to ask an LLM to read all the details of an incident and draft Q&As. However, we found that this method produced generic questions that ignore the key concepts used to link the alerts, events, and entities together



Context: A malicious URL \`https://dj01161621.brave…io/\` was clicked on host \`vnevado-win11h\` …

Question: There is also a suspicious credential dump from NTDS.dit related to this incident. Can you identify the file name of the process used to dump the NTDS.dit?

Answer: ntdsutil.exe

Solution:

1. The user with SID \`S-1-5-21…\` clicked on a malicious URL.

2. There is a suspicious Azure Resource operation from the host \`vnevado-dc\` using a proxy IP address.

3. The process \`ntdsutil.exe\` was used in a suspicious credential dump from NTDS.dit.

Figure 3: Example Question Generation. The start alert and entities will be used as context, and the question asks about the end entity (ID 20). A solution path is also based on the graph.

into a cohesive investigation. Moreover, these questions can lack a deterministic answer or ask about security knowledge not present in the database.

We further investigated the behavior of human SOC analysts and the collected data to solve this issue. We gain two insights that motivate us to build a bipartite threat investigation graph, and then use the graph for QA generation: 1. Manual incident reviews confirm that SOC investigations are inherently relational: start with a seed alert or entity, analysts pivot related IoCs, pull connected events, then iterate to identify suspicious patterns. 2. In our data, we observe that each step of the attack is recorded in the "SecurityIncident" or "SecurityAlert" tables (explained in Section 3.1), which are suitable grounding sources for question generation.

Bipartite Graph Construction. We create a $G = (U, V, E)$ for each incident, where the vertices set is partitioned into `Alert` vertices (U) and `Entity` vertices (V), and every edge from a vertex $u \in U$ connects one in V. The alert table contains an entity column that documents a list of entities associated with these alerts, and each entity can also appear in different alert entries. Thus, we can easily build such a graph from the incident and alert table.

The bipartite graph models the investigation process closely, and we can view the investigation of a SOC analyst as walking on the graph. The analyst starts with one given alert u_s and a set of related entities V_s (connected to u_s). Using this information and entities of interest (IoCs), the analyst can query the database to find related events/alerts u_i and entities V_i. Note that the alert-entity graph is only a small sub-graph traced by the analyst during their investigation. This sub-graph is part of a much larger alert-entity graph of the environment, which includes all alerts and entities in the environment not relevant to the particular incident.

Algorithm 1 Question–Answer–Solution Generation

```
Require: Bipartite Graph $G = (U, V, E)$, start entity count $k$, LLM $M$, question prompt $P_g$, solution prompt $P_s$, Set of question–answer–solution triples $Q$
$Q \leftarrow \emptyset$
for all distinct pairs $(u_s, u_t) \in U \times U$ do
    $V_s \leftarrow \textsc{Sample}(\textsc{GetFarthestEntities}(u_s, u_t), k)$    ▷ Get start entities
    $v_e \leftarrow \textsc{Sample}(\textsc{GetFarthestEntities}(u_t, u_s), 1)$    ▷ Get answer
    $(q, a) \leftarrow M(P_g, u_s, V_s, u_t, v_e)$    ▷ Generate question and answer
    $s \leftarrow M(P_s, q, \textsc{ShortestPath}(u_s, v_e))$    ▷ Generate solution path
    $Q \leftarrow Q \cup \{(q, a, s)\}$
return $Q$
```

QA Generation From Graph. We then use LLMs to generate questions from the graph (See Algorithm 1). We pick any two `Alert` vertices u_s and u_e as the starting and ending points to construct one question. We extract the entities connected to one alert that is farthest from another alert, so that we can have a longer path (GETFARTHESTENTITIES shown in Algorithm 3). We select

k entity vertices V_s as the starting vertices ($k = 2$) and one entity v_e connected to u_e, since we want to give more context to the agent and some entities might not be useful. We instruct the LLM to use u_s and V_s as the background context, and write a question using v_e and u_e, with u_e as the final answer to this question (prompts in Figure 17 & 18). The question tests whether an agent can start with one alert and do investigations towards the goal.

For each question, we can obtain the shortest path between the selected alerts. We take this path in our graph as one optimal solution, but we note that there can be many different paths reaching the ending alert (some might not even be in the graph). We note that the entities in the path can be viewed as IoCs, since they are used to discover new events (alerts) and gather information. We pass the path to LLM to generate a step-by-step solution. The proposed QA generation strategy is a principled way to generate security questions from a bipartite alert-entity graph, and has the following benefits: 1. The questions are non-repetitive, and they test LLMs' abilities on querying the database to perform investigation. 2. We can obtain a clear answer and a solution path, which allows us to have a fine-grained and accurate evaluation. 3. The length of the shortest path can also be used as a measure of the difficulty of the question, so a longer path indicates a harder question. Based on this, we can acquire a total of 7542 questions from the generated graphs, and we generate a total of 589 questions for testing, which is used in experiments, see Appendix E for full details of question logistics and train/test split.

3.3 Environment Setup

Algorithm 2 Reward Calculation

```
Require: solution steps $\mathcal{S}$, submitted answer $a_{\text{sub}}$
1: $r_{\text{sum}} \leftarrow 0,\ \gamma \leftarrow 0.4,\ discount \leftarrow 1$
2: if check_answer($\mathcal{S}_{|\mathcal{S}|}$, $a_{\text{sub}}$) then
3:     return 1                                        ▷ Perfect match on final answer
4: for $i \leftarrow |\mathcal{S}| - 1$ down to 1 do
5:     if check_step($\mathcal{S}_i$, $a_{\text{sub}}$) then
6:         $r_{\text{sum}} \leftarrow r_{\text{sum}} + discount$            ▷ Add discounted reward for this step
7:     $discount \leftarrow discount \times \gamma$                 ▷ Update discount for next older step
8: return $r_{\text{sum}}$
```

To conduct a security investigation, the agent needs to interact with a given database. We set up a MYSQL docker environment following [53] to execute queries. In the environment, the agent can choose to output a query to be executed or submit the answer. The solving process ends when the agent submits the answer or a maximum number of steps is reached. We set the maximum number of entries and the maximum character length that can be returned to avoid context overflow from queries. By default, we employ LLM to evaluate the submitted answers (we use GPT-4o), which is shown to be more robust and align with human judgment (See Appendix D.5 for analysis). The evaluation consists of two steps (See Algorithm 2): 1. match the submitted answer a_{sub} with the ground-truth. 2. If not correct, we check if the submitted answer contains any intermediate step solutions, which shows how much progress the agent makes and useful information it acquires. We assign a decayed reward starting from the last step. This reward not only checks the final answer but also performs a fine-grained evaluation of the agent's intermediate steps. This also evaluates the agent's ability to identify and extract key information (IoCs) from its investigation process. Since each step of the solution contains an unambiguous string, deterministic string comparison is allowed. Thus, the environment can provide reliable rewards for intermediate steps, which is a source of process supervision that is rarely available in other reasoning tasks such as math or coding [27]. This makes our environment particularly well-suited for future work on training agents with RL.

4 Experiments

4.1 Base Models Comparison

Setup. We first compare the performances of different LLMs in Table 2 using a base prompt (See Figure 11). Since the reasoning model like `o1-mini` struggles with the output format, we've added extra instructions to reinforce it. We set `temperature` = 0 and `max_step` = 25. We use `GPT-4o` as our evaluator for all experiments. To fully understand how the capabilities of base models perform on our benchmark, we evaluate a wide range of the latest LLMs. Our evaluation covers proprietary and open-source models, chat and reasoning models, and models of different sizes. See Appendix F.1 for more details of the setup, the models, results, and analysis.

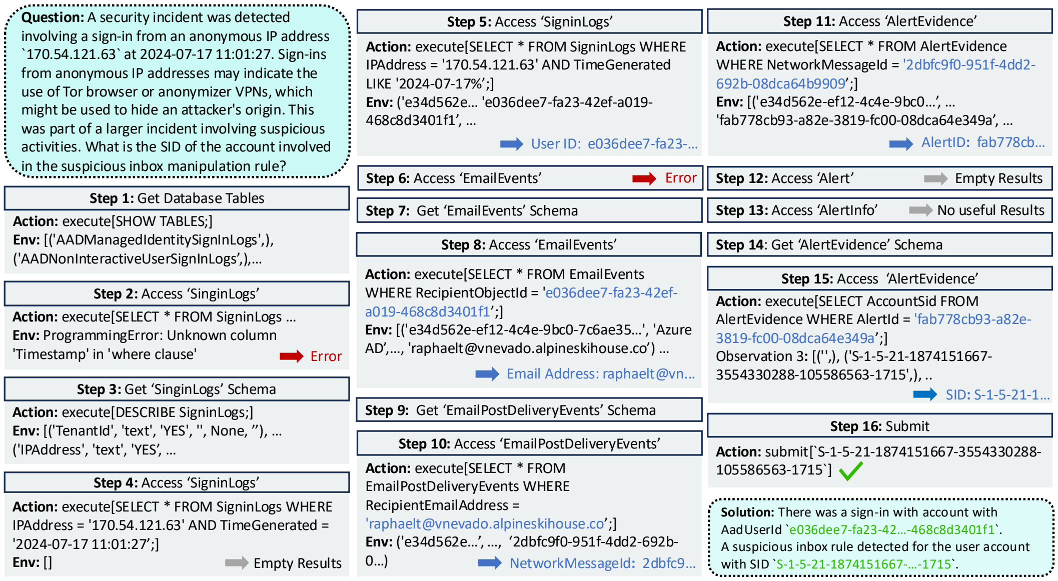
	Incident Number								Avg
	5	34	38	39	55	134	166	322	reward
GPT-4o	0.338	0.293	0.364	0.273	0.249	0.491	0.166	0.315	0.293
GPT-4o-mini	0.163	0.195	0.273	0.185	0.174	0.228	0.163	0.276	0.192
o1-mini†	0.147	0.244	0.091	0.230	0.160	0.333	0.189	0.382	0.222
Phi-4-14B	0.086	0.037	0.182	0.082	0.066	0.130	0.085	0.125	0.085
Llama4-17b-Mav	0.259	0.302	**0.545**	0.324	0.216	0.421	0.189	0.371	0.290
Llama4-17b-Scout	0.216	0.285	0.182	0.228	0.220	0.453	0.193	0.367	0.262
GPT-4.1	0.356	0.315	0.364	0.295	0.258	0.474	**0.292**	0.489	0.338
GPT-4.1-mini	0.324	0.210	0.182	0.248	0.248	0.333	0.216	0.387	0.271
GPT-4.1-nano	0.164	0.185	0.091	0.118	0.136	0.077	0.097	0.179	0.136
o3-mini†	0.350	0.293	0.273	0.257	0.227	0.404	0.253	0.360	0.296
o4-mini†	0.312	**0.383**	0.545	**0.362**	0.284	**0.568**	0.269	**0.517**	**0.368**
Gemini 2.5 Flash	0.312	0.329	0.364	0.248	0.224	0.491	0.260	0.375	0.305
Qwen-3-32b	0.191	0.229	0.091	0.207	0.116	0.2	0.133	0.25	0.182

Table 2: Evaluation of different models. The average per incident and the total average reward are shown (Total average reward = sum of reward / total question count). The model is sorted by release date (first being the earliest). † indicates the agent uses enhanced format prompt.

Result Analysis. We observe the following: **(1)** `o4-mini` delivers the best mean reward (0.368), surpassing the next-best `GPT-4.1` by 0.03 (9 % relative); the explicit-reasoning line shows a steady progression (o1-mini → o3-mini → o4-mini). **(2)** `Phi-14B` barely solves any tasks, and smaller open-source models such as `Qwen-2.5-7B` and `Llama-3-8B` behave similarly, so we omit them. In contrast, a more recent `Llama4-Mav-17B` reaches 0.29, making it competitive with proprietary chat models like `GPT-4o` and `Gemini 2.5 Flash`. **(3)** Incidents 55 and 166 are the most challenging, as no model exceeds 0.3. However, on incidents 38, 134, and 322, the highest model performance exceeds 0.5. Interestingly, these are incidents that have the fewest alerts. **Takeaways:** Recent models are achieving higher rewards overall, open-source models are rapidly closing the gap with proprietary ones, and explicit-reasoning models are improving quickly. Because our benchmark is new and absent from any publicly available training data, test-set leakage is unlikely, showing genuine progress in LLM capabilities. More analysis on path length and rewards is in Appendix F.1.

Behavior Analysis. Agents must first explore and infer the structure of the tables since the schema is not provided. To arrive at the correct answer, they have to combine information from several tables. Figure 4 shows a representative trajectory for the baseline agent: it gradually discovers the schema and refines its SQL queries whenever an error or empty result occurs. Like a human analyst, the agent bootstraps on intermediate findings to steer subsequent exploration. In the reference (gold) solution, the answer is produced in two concise steps: (1) retrieve the user ID, then (2) use that ID to obtain the account SID. The agent pursues a longer, alternative route yet still converges on the correct answer, demonstrating that our benchmark supports multiple viable search strategies. The main difficulty remains in modelling the database accurately enough to construct valid queries. Across models, the Pearson correlation between query success rate and reward is 0.86, indicating a strong positive association. Further analysis of factors linked to high reward is provided in Appendix F.1.

Efficiency Analysis. Turns: In Figure 5a, we plot the change in reward by increasing max turns allowed to interact with the database. We can see the reward spikes from 5 to 15 turns, then plateaus between 15 and 25. `o4-mini` scales well with increased turns, from around 0.07 to 0.37 at 25 turns. In comparison, chat models like `GPT-4o`, `Gemini-2.5-flash` start at around the same reward at 5 turns, but can only reach 0.3 with the turns allowed increased to 25. The comparison of `GPT-4o`, `GPT-4.1` with `GPT-4o-mini`, `GPT-4.1-nano` shows that smaller models have less gain with more increase. **Cost:** In Figure 5b, we plot the reward versus cost for each model and draw the Pareto front line (Pricing in Table 6a). `Gemini-2.5-flash` and `Llama-4-Mav` are the most efficient models, with a competitive reward of around 0.3 while keeps its cost low. `GPT-4o`



Question: A security incident was detected involving a sign-in from an anonymous IP address `170.54.121.63` at 2024-07-17 11:01:27. Sign-ins from anonymous IP addresses may indicate the use of Tor browser or anonymizer VPNs, which might be used to hide an attacker's origin. This was part of a larger incident involving suspicious activities. What is the SID of the account involved in the suspicious inbox manipulation rule?

Step 1: Get Database Tables

Action: execute[SHOW TABLES;]
Env: [('AADManagedIdentitySignInLogs',), ('AADNonInteractiveUserSignInLogs',),...

Step 2: Access 'SinginLogs'

Action: execute[SELECT * FROM SigninLogs ...
Env: ProgrammingError: Unknown column 'Timestamp' in 'where clause' ➡ Error

Step 3: Get 'SinginLogs' Schema

Action: execute[DESCRIBE SigninLogs;]
Env: [('TenantId', 'text', 'YES', '', None, ''), ... ('IPAddress', 'text', 'YES', ...

Step 4: Access 'SigninLogs'

Action: execute[SELECT * FROM SigninLogs WHERE IPAddress = '170.54.121.63' AND TimeGenerated = '2024-07-17 11:01:27';]
Env: [] ➡ Empty Results

Step 5: Access 'SigninLogs'

Action: execute[SELECT * FROM SigninLogs WHERE IPAddress = '170.54.121.63' AND TimeGenerated LIKE '2024-07-17%';]
Env: ('e34d562e... 'e036dee7-fa23-42ef-a019-468c8d3401f1', ...
➡ User ID: e036dee7-fa23-...

Step 6: Access 'EmailEvents' ➡ Error

Step 7: Get 'EmailEvents' Schema

Step 8: Access 'EmailEvents'

Action: execute[SELECT * FROM EmailEvents WHERE RecipientObjectId = 'e036dee7-fa23-42ef-a019-468c8d3401f1';]
Env: [('e34d562e-ef12-4c4e-9bc0-7c6ae35...', 'Azure AD',..., 'raphaelt@vnevado.alpineskihouse.co') ...
➡ Email Address: raphaelt@vn...

Step 9: Get 'EmailPostDeliveryEvents' Schema

Step 10: Access 'EmailPostDeliveryEvents'

Action: execute[SELECT * FROM EmailPostDeliveryEvents WHERE RecipientEmailAddress = 'raphaelt@vnevado.alpineskihouse.co';]
Env: ('e34d562e...', ..., '2dbfc9f0-951f-4dd2-692b-0...) ➡ NetworkMessageId: 2dbfc9...

Step 11: Access 'AlertEvidence'

Action: execute[SELECT * FROM AlertEvidence WHERE NetworkMessageId = '2dbfc9f0-951f-4dd2-692b-08dca64b9909';]
Env: [('e34d562e-ef12-4c4e-9bc0...', ... 'fab778cb93-a82e-3819-fc00-08dca64e349a', ...
➡ AlertID: fab778cb...

Step 12: Access 'Alert' ➡ Empty Results

Step 13: Access 'AlertInfo' ➡ No useful Results

Step 14: Get 'AlertEvidence' Schema

Step 15: Access 'AlertEvidence'

Action: execute[SELECT AccountSid FROM AlertEvidence WHERE AlertId = 'fab778cb93-a82e-3819-fc00-08dca64e349a';]
Observation 3**:** [('',), ('S-1-5-21-1874151667-3554330288-105586563-1715',), ..
➡ SID: S-1-5-21-1...

Step 16: Submit

Action: submit[`S-1-5-21-1874151667-3554330288-105586563-1715`] ✓

Solution: There was a sign-in with account with AadUserId `e036dee7-fa23-42...-468c8d3401f1`. A suspicious inbox rule detected for the user account with SID `S-1-5-21-1874151667-...-1715`.

Figure 4: An example trajectory of Baseline Agent (with GPT−4o) solving a question. The agent goes through several steps to reaching the answer: After getting the user id from 'SignInLogs', it starts exploring two different email-related logs to get a network message id, and finally use it to find the SID. Since the agent also explores schema of tables as it progresses. Full example in Figure 15.

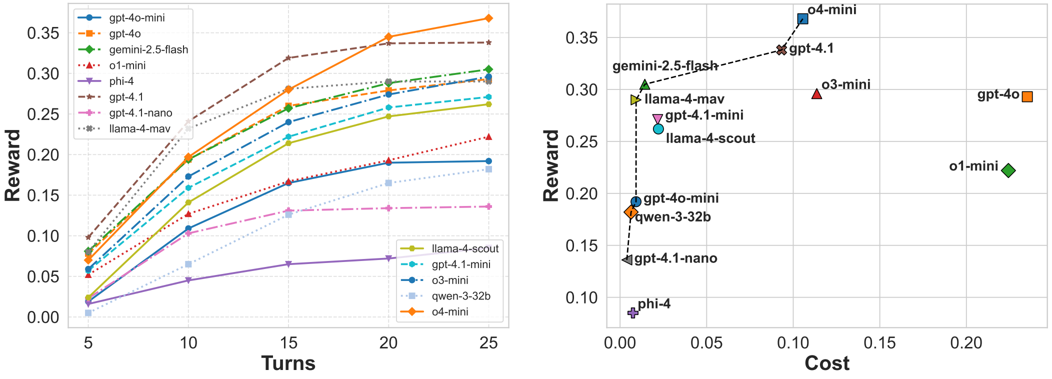


Figure 5: (a) Reward vs. Number of Turns. (b) Reward vs. Cost.

and o1-mini are the least efficient, which is expected since prices fall with each new release and they're the oldest models.

4.2 Comparison of different methods

Setup. To disentangle how different methods (e.g, prompting, test-time scaling) influence agent behavior, we evaluate six different configurations (Prompts in Figure 11 to 14). **(1) Base Prompt**. **(2) Strategy** adds additional notes on using the alert tables as a reference to investigate, which is usually how a SOC analysts find information. **(3) ReAct** We follow ReAct [55] to add 3 few-shot examples to base, which are selected from runs of the train set. **(4) Expel** [64] first distills rules from a small training split, then invokes this external memory, as well as retrieving similar examples during inference. **(5) Best-of-N** retries at most three times and returns the highest-reward trajectory. This is an oracle setting in which the agent needs the reward to determine if a problem needs to be rerun. **(6) Reflection** [38] extends Best-of-N by letting the agent criticize its failed attempt, append the learned rule to the prompt, and retry. We run all methods on **GPT-4o**, **GPT-4o-mini**, and **o3-mini** (temp = 0, max_step = 15) and report average reward, interaction turns, and API cost in Table 3. (Also see a preliminary fine-tuning result on **GPT-4o** in Appendix F.2.)

k	Agent	GPT-4o			GPT-4o-mini			o3-mini		
		reward	turn	cost	reward	turn	cost	reward	turn	cost
1	Base	0.26	11	0.24	0.165	11	0.009	0.219	11	**0.073**
	Strategy	0.273	11	**0.18**	0.290	12	0.010	0.259	12	0.077
	ReAct	0.354	11	0.24	0.274	10	0.016	0.25	11	0.075
	Expel	**0.390**	**9**	0.38	**0.311**	**9**	0.023	**0.265**	**9**	0.191
3	Strategy+BoN	0.473	27	**0.37**	0.418	27	**0.028**	0.382	29.6	0.192
	Strategy+Reflect	0.505	25	0.47	0.440	26	**0.028**	0.394	**26.1**	**0.172**
	ReAct + BoN	**0.563**	21	0.49	0.423	24	0.036	0.378	28.6	0.197
	ReAct+Reflect	**0.563**	**21**	0.50	**0.452**	**24**	0.035	**0.414**	28.2	0.19

Table 3: Evaluation results for GPT-4o, GPT-4o-mini, and o3-mini across methods. Reward, turn, and cost are reported, grouped by number of trials.

Results. For a *single* trial ($k = 1$), Expel attains the best reward among all models, while also finishing in the fewest turns (9). Its performance comes at a higher price: Expel costs 1.6x more than Strategy on GPT-4o because the learned knowledge block and retrieved examples inflate the prompt. o3-mini can hardly benefit from methods like ReAct and Expel, with even a slight drop in accuracy when switching to ReAct, but GPT-4o and GPT-4o-mini have significant gains when switching from base to ReAct and Expel (+ around 0.1). For $k = 3$, we only test and compare Strategy and React prompting due to cost constraints. ReAct+Reflect achieves the best among the models, and we can see that Reflect can almost always help with the performance with different models and prompting strategies. We note that no improvement is made switching from ReAct+BoN to ReAct+Reflect, tested with GPT-4o, which may indicate that this setting has reached an upper bound. We also did a pass-10 experiment to investigate the scaling bounds in Appendix F.3.

4.3 Ablation on DB Scope and Time Window (Figure 6)

DB Scope	Time Window	Reward
raw + alert	Per incident	0.260
raw	Per incident	0.213
alert	Per incident	0.459
raw + alert	Full history	0.248
raw	Full history	0.184
alert	Full history	0.382

Figure 6: Ablation on database setup.

By default, one database is set up per incident with both raw and alert logs. **DB Scope:** In the real world, zero-day and sophisticated attacks may evade security detections and not be shown in alert logs. We remove alert logs to simulate this and observe an obvious drop in performance. This indicates that these alert logs created with rules, heuristics, and ML-based detections are crucial for investigation. We also set up an alert-only database for comparison and found a substantial increase in reward. This is expected since our questions are built from security tables. This also shows that unrelated noise from the database can impact performance. **Time Window:** We also test with a full version of the database (explained in Section 3.1). Moving from the per-incident slices to the full database lowers average reward to 0.248, which is expected since a longer horizon introduces extra noise. We note that degradation from using a longer time span is mild compared to switching the DB scope. Since the questions constructed by LLMs tends to include time information in the question, which relieves the effect of a noisier environment (Tested with GPT-4o).

5 Conclusion

In this paper, we create ExCyTIn-Bench, the first benchmark to evaluate LLM agents on cybersecurity threat investigations based on real-world setup. It includes an open-source Azure security database, a QA dataset, and a standardized environment. We also introduce an automated, structured approach that leverages LLMs to generate high-quality questions from bipartite alert–entity graphs, enabling fine-grained evaluation of an agent's intermediate steps. We evaluate various LLMs and agent systems on the benchmark. **Future Directions.** Open-source models are catching up with propriety models in our environment, and they can be used for further distillation and training to boost performance. Our environment offers fine-grained process rewards that allow precise credit assignment across steps, which is an uncommon but valuable feature for our environment. Overall, this makes it a promising testbed for training LLM agents with reinforcement learning [65].

REFERENCES

[1] Md Tanvirul Alam, Dipkamal Bhushl, Le Nguyen, and Nidhi Rastogi. Ctibench: A benchmark for evaluating llms in cyber threat intelligence. *arXiv preprint arXiv:2406.07599*, 2024.

[2] Lampis Alevizos and Martijn Dekker. Towards an ai-enhanced cyber threat intelligence processing pipeline. *Electronics*, 13(11):2021, 2024.

[3] Arian Askari, Christian Poelitz, and Xinye Tang. Magic: Generating self-correction guideline for in-context text-to-sql. *arXiv preprint arXiv:2406.12692*, 2024.

[4] Jacob Austin, Augustus Odena, Maxwell Nye, Maarten Bosma, Henryk Michalewski, David Dohan, Ellen Jiang, Carrie Cai, Michael Terry, Quoc Le, et al. Program synthesis with large language models. *arXiv preprint arXiv:2108.07732*, 2021.

[5] Dipkamal Bhusal, Md Tanvirul Alam, Le Nguyen, Ashim Mahara, Zachary Lightcap, Rodney Frazier, Romy Fieblinger, Grace Long Torales, and Nidhi Rastogi. Secure: Benchmarking generative large language models for cybersecurity advisory. *arXiv preprint arXiv:2405.20441*, 2024.

[6] Weize Chen, Yusheng Su, Jingwei Zuo, Cheng Yang, Chenfei Yuan, Chen Qian, Chi-Min Chan, Yujia Qin, Yaxi Lu, Ruobing Xie, et al. Agentverse: Facilitating multi-agent collaboration and exploring emergent behaviors in agents. *arXiv preprint arXiv:2308.10848*, 2023.

[7] CrowdStrike. What is cyber threat hunting?, 2023. URL `https://www.crowdstrike.com/cybersecurity-101/threat-hunting/`. Accessed: 14 March 2024.

[8] CrowdStrike. 2024 global threat report, 2024. URL `https://www.crowdstrike.com/global-threat-report/`.

[9] Xuemei Dong, Chao Zhang, Yuhang Ge, Yuren Mao, Yunjun Gao, Jinshu Lin, Dongfang Lou, et al. C3: Zero-shot text-to-sql with chatgpt. *arXiv preprint arXiv:2307.07306*, 2023.

[10] Dawei Gao, Haibin Wang, Yaliang Li, Xiuyu Sun, Yichen Qian, Bolin Ding, and Jingren Zhou. Text-to-sql empowered by large language models: A benchmark evaluation. *arXiv preprint arXiv:2308.15363*, 2023.

[11] Peng Gao, Fei Shao, Xiaoyuan Liu, Xusheng Xiao, Zheng Qin, Fengyuan Xu, Prateek Mittal, Sanjeev R Kulkarni, and Dawn Song. Enabling efficient cyber threat hunting with cyber threat intelligence. In *2021 IEEE 37th International Conference on Data Engineering (ICDE)*, pages 193–204. IEEE, 2021.

[12] Wajih Ul Hassan, Adam Bates, and Daniel Marino. Tactical provenance analysis for endpoint detection and response systems. In *2020 IEEE Symposium on Security and Privacy (SP)*, pages 1172–1189. IEEE, 2020.

[13] Mohammed Hassanin and Nour Moustafa. A comprehensive overview of large language models (llms) for cyber defences: Opportunities and directions. *arXiv preprint arXiv:2405.14487*, 2024.

[14] Sirui Hong, Xiawu Zheng, Jonathan Chen, Yuheng Cheng, Jinlin Wang, Ceyao Zhang, Zili Wang, Steven Ka Shing Yau, Zijuan Lin, Liyang Zhou, et al. Metagpt: Meta programming for multi-agent collaborative framework. *arXiv preprint arXiv:2308.00352*, 2023.

[15] Md Nahid Hossain, Sadegh M Milajerdi, Junao Wang, Birhanu Eshete, Rigel Gjomemo, R Sekar, Scott Stoller, and VN Venkatakrishnan. {SLEUTH}: Real-time attack scenario reconstruction from {COTS} audit data. In *26th USENIX Security Symposium (USENIX Security 17)*, pages 487–504, 2017.

[16] Xueyu Hu, Ziyu Zhao, Shuang Wei, Ziwei Chai, Guoyin Wang, Xuwu Wang, Jing Su, Jingjing Xu, Ming Zhu, Yao Cheng, et al. Infiagent-dabench: Evaluating agents on data analysis tasks. *arXiv preprint arXiv:2401.05507*, 2024.

[17] Qian Huang, Jian Vora, Percy Liang, and Jure Leskovec. Mlagentbench: Evaluating language agents on machine learning experimentation. In *Forty-first International Conference on Machine Learning*, 2024.

[18] IBM. What is threat hunting?, 2023. URL `https://www.ibm.com/topics/threat-hunting`. Accessed: 1 Oct 2024.

[19] Jinhao Jiang, Kun Zhou, Zican Dong, Keming Ye, Wayne Xin Zhao, and Ji-Rong Wen. Structgpt: A general framework for large language model to reason over structured data. *arXiv preprint arXiv:2305.09645*, 2023.

[20] Carlos E Jimenez, John Yang, Alexander Wettig, Shunyu Yao, Kexin Pei, Ofir Press, and Karthik Narasimhan. Swe-bench: Can language models resolve real-world github issues? *arXiv preprint arXiv:2310.06770*, 2023.

[21] Jiandong Jin, Bowen Tang, Mingxuan Ma, Xiao Liu, Yunfei Wang, Qingnan Lai, Jia Yang, and Changling Zhou. Crimson: Empowering strategic reasoning in cybersecurity through large language models. *arXiv preprint arXiv:2403.00878*, 2024.

[22] Pengfei Jing, Mengyun Tang, Xiaorong Shi, Xing Zheng, Sen Nie, Shi Wu, Yong Yang, and Xiapu Luo. Secbench: A comprehensive multi-dimensional benchmarking dataset for llms in cybersecurity. *arXiv preprint arXiv:2412.20787*, 2024.

[23] Matan Levi, Yair Allouche, Daniel Ohayon, and Anton Puzanov. Cyberpal. ai: Empowering llms with expert-driven cybersecurity instructions. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 24402–24412, 2025.

[24] Guancheng Li, Yifeng Li, Wang Guannan, Haoyu Yang, and Yang Yu. Seceval: A comprehensive benchmark for evaluating cybersecurity knowledge of foundation models. https://github.com/XuanwuAI/SecEval, 2023.

[25] Guohao Li, Hasan Abed Al Kader Hammoud, Hani Itani, Dmitrii Khizbullin, and Bernard Ghanem. Camel: Communicative agents for ”mind” exploration of large scale language model society, 2023.

[26] Jinyang Li, Binyuan Hui, Ge Qu, Jiaxi Yang, Binhua Li, Bowen Li, Bailin Wang, Bowen Qin, Ruiying Geng, Nan Huo, et al. Can llm already serve as a database interface? a big bench for large-scale database grounded text-to-sqls. *Advances in Neural Information Processing Systems*, 36:42330–42357, 2023.

[27] Hunter Lightman, Vineet Kosaraju, Yura Burda, Harri Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let’s verify step by step, 2023. URL `https://arxiv.org/abs/2305.20050`.

[28] Xi Victoria Lin, Chenglong Wang, Luke Zettlemoyer, and Michael D Ernst. Nl2bash: A corpus and semantic parser for natural language interface to the linux operating system. *arXiv preprint arXiv:1802.08979*, 2018.

[29] Zefang Liu. Secqa: A concise question-answering dataset for evaluating large language models in computer security. *arXiv preprint arXiv:2312.15838*, 2023.

[30] Zefang Liu, Jialei Shi, and John F Buford. Cyberbench: A multi-task benchmark for evaluating large language models in cybersecurity, 2024.

[31] Arash Mahboubi, Khanh Luong, Hamed Aboutorab, Hang Thanh Bui, Geoff Jarrad, Mohammed Bahutair, Seyit Camtepe, Ganna Pogrebna, Ejaz Ahmed, Bazara Barry, et al. Evolving techniques in cyber threat hunting: A systematic review. *Journal of Network and Computer Applications*, page 104004, 2024.

[32] MITRE. Mitre att&ck, 2025. URL `https://attack.mitre.org/`. A knowledge base of adversary tactics and techniques.

[33] Linyong Nan, Ellen Zhang, Weijin Zou, Yilun Zhao, Wenfei Zhou, and Arman Cohan. On evaluating the integration of reasoning and action in llm agents with database question answering. *arXiv preprint arXiv:2311.09721*, 2023.

[34] Joshua Nordine. OSINT Framework. `https://github.com/lockfale/osint-framework` (commit 68c904c), 2024. Accessed 2025-05-10.

[35] Filippo Perrina, Francesco Marchiori, Mauro Conti, and Nino Vincenzo Verde. Agir: Automating cyber threat intelligence reporting with natural language generation. In *2023 IEEE International Conference on Big Data (BigData)*, pages 3053–3062. IEEE, 2023.

[36] Mohammadreza Pourreza and Davood Rafiei. Din-sql: Decomposed in-context learning of text-to-sql with self-correction. *Advances in Neural Information Processing Systems*, 36, 2024.

[37] Maria Rigaki, Ondřej Lukáš, Carlos A Catania, and Sebastian Garcia. Out of the cage: How stochastic parrots win in cyber security environments. *arXiv preprint arXiv:2308.12086*, 2023.

[38] Noah Shinn, Beck Labash, and Ashwin Gopinath. Reflexion: an autonomous agent with dynamic memory and self-reflection. *arXiv preprint arXiv:2303.11366*, 2023.

[39] Giuseppe Siracusano, Davide Sanvito, Roberto Gonzalez, Manikantan Srinivasan, Sivakaman Kamatchi, Wataru Takahashi, Masaru Kawakita, Takahiro Kakumaru, and Roberto Bifulco. Time for action: Automated analysis of cyber threat intelligence in the wild. *arXiv preprint arXiv:2307.10214*, 2023.

[40] Madeena Sultana, Adrian Taylor, Li Li, and Suryadipta Majumdar. Towards evaluation and understanding of large language models for cyber operation automation. In *2023 IEEE Conference on Communications and Network Security (CNS)*, pages 1–6. IEEE, 2023.

[41] Shayan Talaei, Mohammadreza Pourreza, Yu-Chen Chang, Azalia Mirhoseini, and Amin Saberi. Chess: Contextual harnessing for efficient sql synthesis. *arXiv preprint arXiv:2405.16755*, 2024.

[42] The MITRE Corporation. Common Vulnerabilities and Exposures (CVE) Program. `https://www.cve.org/`, 2025. Accessed 2025-05-10.

[43] The MITRE Corporation. MITRE ATT&CK® Knowledge Base. `https://attack.mitre.org/`, 2025. Version 17.1. Accessed 2025-05-10.

[44] Bing Wang, Changyu Ren, Jian Yang, Xinnian Liang, Jiaqi Bai, Linzheng Chai, Zhao Yan, Qian-Wen Zhang, Di Yin, Xing Sun, et al. Mac-sql: A multi-agent collaborative framework for text-to-sql. *arXiv preprint arXiv:2312.11242*, 2024.

[45] Guanzhi Wang, Yuqi Xie, Yunfan Jiang, Ajay Mandlekar, Chaowei Xiao, Yuke Zhu, Linxi Fan, and Anima Anandkumar. Voyager: An open-ended embodied agent with large language models. *arXiv preprint arXiv:2305.16291*, 2023.

[46] Lei Wang, Chen Ma, Xueyang Feng, Zeyu Zhang, Hao Yang, Jingsen Zhang, Zhiyuan Chen, Jiakai Tang, Xu Chen, Yankai Lin, et al. A survey on large language model based autonomous agents. *Frontiers of Computer Science*, 18(6):186345, 2024.

[47] Shenzhi Wang, Chang Liu, Zilong Zheng, Siyuan Qi, Shuo Chen, Qisen Yang, Andrew Zhao, Chaofei Wang, Shiji Song, and Gao Huang. Avalon's game of thoughts: Battle against deception through recursive contemplation. *ArXiv*, abs/2310.01320, 2023. URL `https://api.semanticscholar.org/CorpusID:263605971`.

[48] Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, and Denny Zhou. Self-consistency improves chain of thought reasoning in language models. *arXiv preprint arXiv:2203.11171*, 2022.

[49] Qingyun Wu, Gagan Bansal, Jieyu Zhang, Yiran Wu, Shaokun Zhang, Erkang Zhu, Beibin Li, Li Jiang, Xiaoyun Zhang, and Chi Wang. Autogen: Enabling next-gen llm applications via multi-agent conversation framework. *arXiv preprint arXiv:2308.08155*, 2023.

[50] Yiran Wu, Feiran Jia, Shaokun Zhang, Hangyu Li, Erkang Zhu, Yue Wang, Yin Tat Lee, Richard Peng, Qingyun Wu, and Chi Wang. Mathchat: Converse to tackle challenging math problems with llm agents. In *ICLR 2024 Workshop on Large Language Model (LLM) Agents*, 2024.

[51] Yiran Wu, Tianwei Yue, Shaokun Zhang, Chi Wang, and Qingyun Wu. Stateflow: Enhancing llm task-solving through state-driven workflows. *arXiv preprint arXiv:2403.11322*, 2024.

[52] Tianqi Xu, Linyao Chen, Dai-Jie Wu, Yanjun Chen, Zecheng Zhang, Xiang Yao, Zhiqiang Xie, Yongchao Chen, Shilong Liu, Bochen Qian, et al. Crab: Cross-environment agent benchmark for multimodal language model agents. *arXiv preprint arXiv:2407.01511*, 2024.

[53] John Yang, Akshara Prabhakar, Karthik Narasimhan, and Shunyu Yao. Intercode: Standardizing and benchmarking interactive coding with execution feedback. *arXiv preprint arXiv:2306.14898*, 2023.

[54] John Yang, Carlos E Jimenez, Alexander Wettig, Kilian Lieret, Shunyu Yao, Karthik Narasimhan, and Ofir Press. Swe-agent: Agent-computer interfaces enable automated software engineering. *arXiv preprint arXiv:2405.15793*, 2024.

[55] Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. React: Synergizing reasoning and acting in language models. *arXiv preprint arXiv:2210.03629*, 2022.

[56] Javier Yong, Haokai Ma, Yunshan Ma, Anis Yusof, Zhenkai Liang, and Ee-Chien Chang. Attackseqbench: Benchmarking large language models' understanding of sequential patterns in cyber attacks. *arXiv preprint arXiv:2503.03170*, 2025.

[57] Tao Yu, Rui Zhang, Kai Yang, Michihiro Yasunaga, Dongxu Wang, Zifan Li, James Ma, Irene Li, Qingning Yao, Shanelle Roman, et al. Spider: A large-scale human-labeled dataset for complex and cross-domain semantic parsing and text-to-sql task. *arXiv preprint arXiv:1809.08887*, 2018.

[58] Yang Yue, Zhiqi Chen, Rui Lu, Andrew Zhao, Zhaokai Wang, Yang Yue, Shiji Song, and Gao Huang. Does reinforcement learning really incentivize reasoning capacity in llms beyond the base model?, 2025. URL `https://arxiv.org/abs/2504.13837`.

[59] Yifan Zeng, Yiran Wu, Xiao Zhang, Huazheng Wang, and Qingyun Wu. Autodefense: Multi-agent llm defense against jailbreak attacks. *arXiv preprint arXiv:2403.04783*, 2024.

[60] Andy K Zhang, Neil Perry, Riya Dulepet, Eliot Jones, Justin W Lin, Joey Ji, Celeste Menders, Gashon Hussein, Samantha Liu, Donovan Jasper, et al. Cybench: A framework for evaluating cybersecurity capabilities and risk of language models. *arXiv preprint arXiv:2408.08926*, 2024.

[61] Jie Zhang, Haoyu Bu, Hui Wen, Yu Chen, Lun Li, and Hongsong Zhu. When llms meet cybersecurity: A systematic literature review. *arXiv preprint arXiv:2405.03644*, 2024.

[62] Jie Zhang, Haoyu Bu, Hui Wen, Yongji Liu, Haiqiang Fei, Rongrong Xi, Lun Li, Yun Yang, Hongsong Zhu, and Dan Meng. When llms meet cybersecurity: A systematic literature review. *Cybersecurity*, 8(1):1–41, 2025.

[63] Yunjia Zhang, Jordan Henkel, Avrilia Floratou, Joyce Cahoon, Shaleen Deep, and Jignesh M Patel. Reactable: Enhancing react for table question answering. *arXiv preprint arXiv:2310.00815*, 2023.

[64] Andrew Zhao, Daniel Huang, Quentin Xu, Matthieu Lin, Yong-Jin Liu, and Gao Huang. Expel: Llm agents are experiential learners. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 19632–19642, 2024.

[65] Andrew Zhao, Yiran Wu, Yang Yue, Tong Wu, Quentin Xu, Matthieu Lin, Shenzhi Wang, Qingyun Wu, Zilong Zheng, and Gao Huang. Absolute zero: Reinforced self-play reasoning with zero data. *arXiv preprint arXiv:2505.03335*, 2025.

A LLM Usage

In this paper, we have minor usage of LLMs to help polish writing, which includes the following: 1. check for grammar. 2. Shorten and refine sentences. 3. Provide word choices.

B Limitations and Broader Impacts

Limitations. While ExCyTIn-Bench represents a significant step toward evaluating LLM agents on realistic threat investigations, it remains tied to a controlled, simulated Azure tenant and covers only eight attack scenarios. This constrained scope may not capture the full diversity of attack techniques, log schemas, and scale encountered in enterprise environments. Moreover, the automatic QA generation—anchored to explicit graph paths—could bias questions toward neatly structured reasoning steps and may underrepresent the ambiguity and noise present in real-world investigations. Finally, our evaluation currently relies on SQL-based interactions and LLM or deterministic scoring, which may not fully reflect the varied toolchains, data sources, or human-in-the-loop workflows used by security teams in practice.

Broader Impacts. By open-sourcing both the benchmark and its underlying environment, ExCyTIn-Bench aims to accelerate research on autonomous LLM agents capable of navigating complex, multi-step security analyses. A shared, reproducible evaluation platform can foster more rapid iteration on prompting strategies, model architectures, and reinforcement-learning techniques tailored to cyber-defense tasks. In the long run, progress driven by this benchmark could lead to more effective automated triage and incident response, lowering the barrier to advanced threat hunting for under-resourced organizations. At the same time, publicly available benchmarks must be balanced against the risk that malicious actors might study agent behaviors to identify weaknesses or craft evasion techniques, underscoring the need for responsible disclosure and continued collaboration between defenders and the research community.

C Additional Background

Below are key tasks and skills that cybersecurity threat analysts typically leverage in their day-to-day work:

- Understanding Logs and Data Sources: Familiarity with the wide range of log formats (e.g., system event logs, network traffic logs, web server logs) and how each source can reveal indicators of compromise.
- Triage and Data Analysis: Rapidly filtering large datasets to identify potential leads, such as unusual account behavior or abnormal traffic spikes.
- Querying and Coding Skills: Ability to write SQL or other specialized query languages to extract the exact data needed, enabling deeper inspection of suspicious events or user activity.
- Evidence Correlation and Synthesis: Combining data from multiple sources—such as SIEM (Security Information and Event Management) alerts, intrusion detection systems, endpoint security suites, and threat intelligence feeds—to construct a complete picture of the incident. This also involves recognizing patterns and drawing relationships between events, timestamps, and potential attackers' tactics, techniques, and procedures (TTPs).
- Hypothesis Testing: Formulating and testing possible explanations for alert signals—for instance, whether peculiar activity might stem from a misconfiguration or a targeted attack. Iterating through multiple hypotheses, gathering more evidence until one scenario best explains the observed behaviors.
- Noise Filtering: Distinguishing benign anomalies (e.g., legitimate system updates, authorized organization-wide password resets) from malicious behaviors. Employing data normalization and enrichment techniques to reduce extraneous signals and highlight true threats.
- Leveraging Cyber-security Domain Expertise: Applying deep knowledge of security frameworks and best practices (e.g., MITRE ATT&CK [43], CVE (Common Vulnerabilities and Exposures) [42], OSINT (Open Source Intelligence) [34], etc.) to guide investigation processes and validate findings. Drawing on historical context about common threat actor tactics and industry-specific threats to anticipate potential entry points or attack vectors.

D Additional Benchmark Details

D.1 Access

Please access the code and dataset on `https://github.com/kaebvcidn/Excytin-Bench`.

D.2 Features

Real-world security database Although the tenant is for a fictional company, the log data we use is from real-world tenant environments. Our database consists of security log data, and the volume of the database is much bigger than previous works [53; 57]. We further note that it is extremely challenging to acquire actual client data for research, and it is nearly impossible to open-source such data. The dataset we are releasing is rare: a tenant built for a fictional company with many properties similar to real-world customer tenants.

Real-world attacks The attacks are complete replications of real-world kill chains used against Azure clients. All of them actually occurred and are well documented in cybersecurity news and blogs. For example, Incident 5 is a simulation of "Manatee Tempest Ransomware" (2021), and Incident 134 covers the "BEC and Account Take-over" (2024) attack.

Require multi-hop data exploration and analysis. Our questions are constructed so that the agent has to interact with the database to reach the final answer, utilizing new information gained to continue the investigation.

Domain Specific Our benchmark is built on security log data. So it requires the agent to have a strong knowledge in the security domain to understand the data and conduct meaningful reasoning.

Fine-grained Rewards The decayed reward calculation adopted from RL allows us to evaluate agents' performances with greater granularity, instead of only success and failures. This metric also enables better evaluation of the intermediate steps.

D.3 Database Setup

Our questions are constructed from alert and incident tables. In a common scenario, the security analysts will be given these tables to serve as starting points to conduct analysis. This is also the current setup for conducting experiments. However, there may be new attacks that analytic rules cannot detect and summarize them into security alerts. This requires the security analysts to analyze and find IoCs from the rest of the logs. We also support the setting to remove the security logs from the database to simulate this scenario.

D.4 Personally Identifiable Information (PII) Anonymization

Below we explain how we do PII Anonymization on our dataset through a joint effort of manual and LLM-based examination.

1. Identification of PII Columns Each table is scanned column-by-column. For every column we draw a random sample of five values and prompt a (LLM) to decide whether the column contains PII. Columns provisionally flagged in this first pass are examined once more with three focused prompts:

1. Confirm whether the column indeed holds PII.
2. Decide whether the column stores a dictionary/`JSON` structure.
3. If it does, enumerate which keys inside the structure contain PII.

The union of both LLM passes is then reviewed by domain experts, yielding a curated list of PII-bearing columns that serves as ground truth for the remainder of the pipeline.

2. Creation of PII Value Mappings For every confirmed PII column we gather its set of unique values. If the column encodes a dictionary, only the keys identified in the previous stage are considered.

- **Regex-based substitution.** We manually go througth the tables to recognize common PII patterns, and each candidate value is matched against them (IPv4/IPv6 addresses, e-mail addresses, UUIDs, MAC addresses, latitude/longitude pairs, *etc.*). Matches are replaced by randomly generated surrogates that obey the same syntax.
- **LLM-based substitution.** Values that do not match any pattern are batched (ten values per batch) and passed to the LLM, which returns semantically plausible yet fictitious substitutes (e.g. “John” → “Javier”).

All substitutions are cached in a dictionary so that a source value is always mapped to the same surrogate. The resulting mappings are classified into coarse categories (`ip`, `email`, `other`) and briefly inspected to remove spurious or already anonymised tokens. Empirically, IP addresses account for roughly 95% of all distinct PII values encountered.

3. Dataset-wide Replacement In the final stage we stream every table in the dataset, globally replacing each source PII value with its surrogate. This guarantees *referential consistency*—queries that join on an anonymised IP address still succeed—and eliminates residual PII leakage while preserving analytical utility.

Sample 1: Several wrong IPs are listed

Ground Truth: 170.54.121.63
Submitted: The IP addresses associated with the suspicious login activity for Nina Park are ‘191.109.229.245‘, ‘36.193.21.220‘, and ‘237.7.81.122‘.
LLM judgment: Incorrect
Analysis: The submitted answer does not contain the golden answer’s IP address (170.54.121.63), which is crucial to identifying the suspicious login activity for Nina Park. None of the submitted IP addresses match the golden answer.

Sample 2: Comparing “dp5hn” with dp5hn@vnevado.alpineskihouse.co

Ground Truth: dp5hn
Submitted: The account name involved in the Command and Control behavior is ”dp5hn@vnevado.alpineskihouse.co”.
LLM judgment: Correct
Analysis: The submitted answer contains the golden answer ”dp5hn” with additional context in the form of a domain, which still identifies the account accurately.

Sample 3: One of two submitted answers matches the ground truth

Ground Truth: S-1-5-21-1440151660-4837083288-105596163-1120
Submitted: The SID values associated with the affected accounts are ‘S-1-5-21-1440151660-4837083288-105596163-1120‘ and ‘S-1-5-21-1846150160-3535302852-1555536564-2111‘.
LLM judgment: Correct
Analysis: The submitted answer includes the golden answer ‘S-1-5-21-1440151660-4837083288-105596163-1120‘ along with an additional SID value. It is still relevant and within the context of the question.

Figure 7: Examples of LLM judgments matching human heuristics.

D.5 LLM AS JUDGE

The main reason we use **GPT-4o as the judge** is that the agents we evaluated often fail to follow the required answer format. We observed the following issues when we relied on string matching:

- **The agent does not follow the answer format.** For example, when the correct answer is `18.27.43.343`, the agent writes: “The final answer is 18.27.43.343” instead of providing only the IP address.
- **The agent returns the answer in an alternative form.** For example, if the correct answer is the user name `user1`, the agent might output an email containing the same string—`user1@alpineskihouse.co`—which should still be regarded as correct.
- **The agent offers multiple answers, including the correct one.** For example, when only one IP address is correct, the LLM may supply two IPs, one of which is correct. After discussion with security experts, they still think it is pretty helpful to narrow down the final answer to 2 IP addresses using a LLM assistant, and can be judged as “correct”.

We believe string matching is a valid evaluation method and can easily be enabled in our system. However, because current LLMs already struggle with our benchmark, we adopt the *LLM-as-judge* approach to measure agents’ ability to solve tasks rather than secondary abilities such as strict formatting. We design the judging prompt with techniques to minimize hallucination (task decomposition, provision of all necessary information) and with the self-reflection technique. We also include explicit rules to make the judgments more closely aligned with human heuristics.

To verify the effectiveness of using an LLM as the judge, we randomly selected 160 answered questions and manually reviewed them; all were evaluated correctly:

	Total questions	Submitted questions	TP	FP	TN	FN
Combined	163	132	56	0	76	0

In most cases, the agent submits a single string that can be judged easily. We also observe the following scenarios in which the LLM’s judgment aligns with human judgment, please see Figure 7 for 3 examples.

E ADDITIONAL QUESTION GENERATION DETAILS

Algorithm 3 GETFARTHESTENTITIES (From Algorithm 1)

```
Require: Graph $G = (V, E)$ where each node has attribute type, start alert $a_s \in V$, end alert $a_e \in V$
1: $S \leftarrow \{ v \in \text{NEIGHBORS}(G, a_s) \mid \texttt{type}(v) = \text{“entity”} \}$    ▷ entities adjacent to $a_s$
2: $D \leftarrow$ empty map    ▷ keys: path length, values: entity lists
3: for all $e \in S$ do
4:     $d \leftarrow \text{SHORTESTPATHLENGTH}(G, e, a_e)$
5:     $D[d] \leftarrow D[d] \cup \{e\}$    ▷ bucket entity by distance
6: $d_{\max} \leftarrow \max\{d \mid d \in D\}$    ▷ greatest distance observed
7: return $D[d_{\max}]$    ▷ all entities farthest from $a_e$
```

E.1 GRAPHS AND REPORTS

In Figure 19-26, we put overview plots of graphs for all incidents. Note that the graphs are for illustrative purposes and some details might be hard to read due to the large graph size. The bigger blue nodes represent alerts, and the smaller red nodes represent entities. In Figure 27- 41, we put summarized reports of these graphs using LLM.

E.2 Question Logistics

Since the deterministic analytic rules are run at an interval, there are many repetitive alerts generated. We first process each incident to remove repetitive alerts. If there are disjoint graphs in an incident, we will keep the larger one. Since we can generate one question from any two alerts, the number of questions is bounded by the 2nd exponential of the number of alerts, where we can generate at most 7542 questions. However, a number of alerts for each incident is extremely unbalanced, resulting in several questions from 4624 to 16 questions. We also want to split the data into a training and test set, with a main focus on the diversity and quality of the test set. To this end, we use the following split strategy: if an incident has less than 150 questions, we will take around 70% of the questions as test data. If the number of questions is bigger than 150, we will cap the number of questions to 100. Under these criteria and filtering after question generation, we collected a total of 589 questions as the test set (See Figure 4). We also created a strategy for sampling questions to split the training and test. Since we are building questions from the graph, and the train and test sets are all from one graph, we want the train samples to have less overlap in paths with the tested sets. When performing training, the model might remember knowledge over the path information and could ”cheat” from these. Thus, we create an overlap score and use it to guide the random sampling. We will random split k times and select the split with the highest overlap score.

Incident	**Path Length**					**Total**
	1	3	5	7	9	
38	4	7	0	0	0	11
34	9	73	0	0	0	82
5	3	74	15	6	0	98
39	4	57	34	3	0	98
134	7	50	0	0	0	57
322	5	19	23	9	0	56
166	11	76	0	0	0	87
55	3	57	26	13	1	100

Table 4: Number of questions generated from different path length for each incident.

E.3 Split Train Test Set

We want to split the train and test set so that they have fewer overlaps. For example, with a simple graph $\{A-B-C, D-B-E\}$, it is best to split the paths in the subgraph $A-B-C$ and $D-B-E$ into two different sets, instead of $A-B$, $D-E$ to train, $A-C$, $D-E$ to test. For this purpose, we randomly split the training or test set and compute a customized total overlap score between every two paths from the train and test sets. We run this for T trials and select the split with the lowest overlap score.

Overlap Score Calculation Given two paths $P_1 = (v_0, \ldots, v_m)$ and $P_2 = (u_0, \ldots, u_n)$ in the same graph, convert each path to its ordered set of directed edges:

$$E_1 = \{(v_i, v_{i+1}) \mid 0 \le i < m\}, \qquad E_2 = \{(u_j, u_{j+1}) \mid 0 \le j < n\}.$$

Let

- $E_{\text{shared}} = E_1 \cap E_2$ — edges appearing in both paths;
- $E_{\text{unshared}} = E_1 \triangle E_2$ — edges that appear in exactly one path (the symmetric difference).

We reward overlap with a positive weight $\alpha > 0$ and penalize divergence with a cost factor $\beta > 0$, scaling the penalty by the combined edge count so the two terms are on comparable footing. To guarantee that the score is zero whenever the paths share no edge, we define the overlap score piecewise:

$$S(P_1, P_2; \alpha, \beta) = \begin{cases} 0, & |E_{\text{shared}}| = 0, \\ \alpha\,|E_{\text{shared}}| \;-\; \beta\,\dfrac{|E_{\text{unshared}}|}{|E_1| + |E_2|}, & \text{otherwise.} \end{cases}$$

	Incident Number								Avg
	5	34	38	39	55	134	166	322	reward
GPT-4o	0.338	0.293	**0.364**	0.273	0.249	**0.491**	0.166	0.315	0.293
o1-mini†	0.147	0.244	0.091	0.230	0.160	0.333	0.189	0.382	0.222
o3-mini†	0.350	0.293	0.273	0.257	0.227	0.404	**0.253**	0.360	0.296
MM o1-mini*	0.304	0.256	0.273	0.238	0.296	0.316	0.211	0.379	0.279
MM o1*	0.398	**0.317**	0.091	0.265	**0.297**	0.474	0.228	**0.391**	**0.323**
MM o3-mini*	**0.404**	0.310	**0.364**	**0.274**	0.264	0.333	0.218	0.375	0.308
Finetune GPT-4o	0.345	*0.241*	*0.091*	0.262	0.299	0.418	0.246	0.355	0.298

Table 5: **Results with Master-Slave testing and finetuning.** Models related to the MM testing is also included for comparison. * denotes a special mixture of model use (MM).The average per incident and the total average reward are shown (Total average reward = sum of reward / total question count). We only should the related models in the table (Full result in Table 2). * indicates the agent is instructed with additional format notes. † indicates the agent is instructed with additional format notes. For finetune (Appendix F.2), incident 34 and 38 are the hold-out (test) incidents, and the performance drops significantly.

Thus, S ranges from $-\beta$ (complete mismatch) to α (identical edge sets), and is exactly 0 when the two paths have no common edges at all.

F Additional Experiments and Details

F.1 Additional Details on Experiment with Different Models

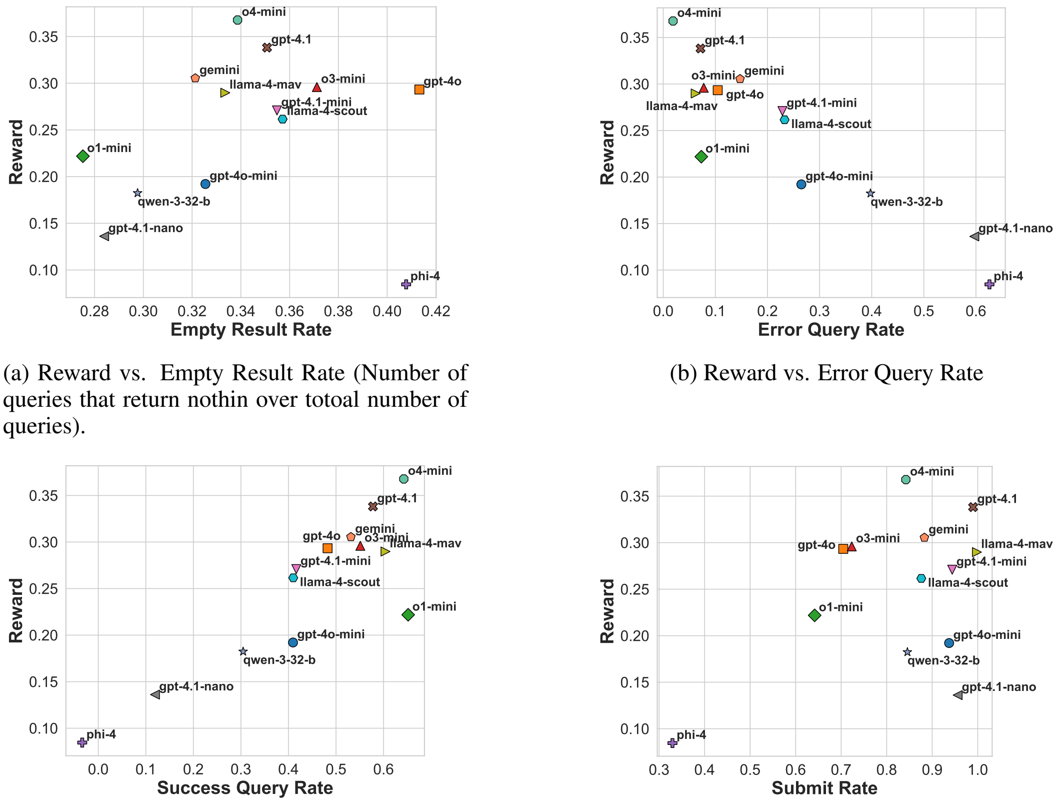
Setup For the first experiment to test different LLMs, we set `max_step` = 25, `max_num_entry` = 15, `max_char_len` = 100000. We use `GPT-4o` as our evaluator for all experiments. We set the temperature to 0 for all LLMs. For the rest of experiements, we maintain the same setting, execept that we reduce the `max_step` to 15.

We test with the following models: (1) `GPT-4o:` (May 2024, OpenAI) multimodal model that reasons across audio, vision, and text; `GPT-4o-mini` (July 2024, OpenAI): distilled variant. (2) `o1-mini:` (Sept 2024, OpenAI) cost-efficient small reasoning model optimised for math and coding. (3) `Phi-4-14B:` (Dec 2024, Microsoft) 14 B-parameter model trained largely on synthetic data. (4) `Llama4-Maverick:` (Apr 2025, Meta) open-weight Mixture-of-Experts multimodal model with 128 experts; `Llama4-Scout` (Apr 2025, Meta): lighter variant with 16 experts. (5) `GPT-4.1:` (Apr 2025, OpenAI) successor to GPT-4o with stronger coding/instruction following; `GPT-4.1-mini` and `GPT-4.1-nano`: smaller, cheaper versions. (6) `o3-mini:` (Apr 2025, OpenAI) upgraded small reasoning model. (7) `o4-mini:` (Apr 2025, OpenAI) latest reasoning model by OpenAI. (8) `Gemini 2.5 Flash:` (Apr 2025, Google DeepMind) budget multimodal model that "thinks" before responding. (9) `Qwen3-32b:` (Apr 2025, Alibaba) Open-source model with hybrid reasoning capabilities.

If the agent doesn't submit answer before reaching the max step, we take it as failure (reward = 0). We use Azure services for OpenAI models and AI Foundry for Phi-4. We use Google service for Gemini. For other open-sourced models `Llama-4-Maverick`, `Llama-4-Scout`, `Qwen-3-32b`, we use cloud service DeepInfra.

Master-Slave Testing With Base Agent (See Table 5) Since the o1 model is very expensive and time-consuming, we set up a special master-slave model switching for it (displayed as MM o1), which will use GPT-4o for 4 steps and switch to o1 every 5th step. For comparison, we also test this setting with o1-mini and o3-mini. The "master–slave" interleaving strategy boosts performance over a single reasoning model, with larger gains when the master is stronger.

Additional Behavior Analysis In Figure 9, we plot reward versus different rates for each model to help understand the behavior of baseline agents with different models. From Figure 9b and 9c,



(a) Reward vs. Empty Result Rate (Number of queries that return nothin over totoal number of queries).

(b) Reward vs. Error Query Rate

(c) Reward vs. Success Query Rate (successful queries that get non-empty returns / total queries) for each model.

(d) Reward vs. Submit Rate (Number of questions submitted / Total number of questions).

Figure 9: Reward versus different query performance metrics and submit rate for each model.

we find there is a strong relation between reward and how well the agent can give queries. From Figure 9a, we find that models that are more likely to get higher rewards with higher empty results. A higher empty rate can indicate that the agent has better fundamental capabilities to understand the table schema to give correct queries. However, it may still struggle to get meaningful results even if it can understand the query well. In Figure 9d, we plot submit rate versus reward. Achieving a lower accuracy with a high submit rate indicates that the agent cannot access their progress correctly, and is overconfident in submitting their results. Smaller models like `gpt-4o-mini` and `gpt-4.1-nano` tend to submit their answer more often, but stronger models like `o4-mini` achieve high results without a relatively low submit rate.

Path Len	# Questions	Avg. Reward
1	46	0.347
3	413	0.249
5	98	0.237
7	31	0.328

Figure 8: Counts and average rewards by path length.

Results of questions generated from different length (Figure 8). Our questions are generated from a different path. We count the questions generated from different path lengths, and averaged the model rewards in Table 2. From the question generated from path length 1-5, the reward is decreasing as expected. However, the reward suddenly rises when length is 7 (Since there is only 1 question generated from path length 9, we didn't show it here). We hypothesize that while we can have an explicit representation of the hops based on our constructed graph, this might not be a full view. As the hop grows larger, there may be another easier and unknown path that is not shown in our graph. Also note that there are fewer data points for path 7, so this result might be biased.

Model	Input Price	Output Price
gpt-4o-mini	0.15	0.6
gpt-4o	2.5	10.0
o1-mini	1.1	4.4
o3-mini	1.1	4.4
phi-4	0.07	0.14
gpt-4.1	2.0	8.0
gpt-4.1-mini	0.4	1.6
gpt-4.1-nano	0.1	0.4
llama-4-mav	0.17	0.6
llama-4-scout	0.17	0.6
o4-mini	1.1	4.4
gemini-2.5-flash	0.15	0.6
qwen-3-32b	0.1	0.3

(a) Pricing per million tokens (input vs. output).

Model	Query Count	Return Length
gpt-4o-mini	11.9	30k
gpt-4o	13.4	36k
o1-mini	16.6	33k
o3-mini	13.6	27k
phi-4	12.0	18k
gpt-4.1	9.2	24k
gpt-4.1-mini	11.5	18k
gpt-4.1-nano	5.7	18k
llama-4-mav	7.9	28k
llama-4-scout	13.4	57k
o4-mini	12.4	28k
gemini-2.5-flash	9.1	26k
qwen-3-32b	13.5	12k

(b) Average query count and query return length (in thousands of chars).

F.2 Fine-Tuning

We conducted preliminary fine-tuning experiments on GPT-4o variants to assess whether it could improve accuracy (See Table 5). From our logs, we extracted successful trajectories produced by GPT-4o, GPT-4o-mini, o1-mini, and o3-mini, withholding those from incidents 28 and 34 as a held-out test set. The remaining 253 trajectories were used to fine-tune each model via Azure Training Service, training only on the assistant's responses. Although overall accuracy on the training incidents remained essentially unchanged, performance on the hold-out incidents suffered markedly: accuracy dropped from 0.293 to 0.241 on incident 34 and from 0.364 to 0.091 on incident 38. This suggests that fine-tuning amplified the model's bias toward the training incidents, degrading its ability to generalize. Given our small sample size, additional studies are needed to characterize the impact of fine-tuning more precisely. However, these initial results imply that naïve fine-tuning may be ill-suited to this task and motivate exploring alternatives, such as reinforcement learning with value regularization (RLVR), to bolster performance on unseen incidents.

F.3 Additional Experiment on Scaling

To further explore the limits of our test-time scaling method, we apply Best-of-N sampling to the baseline GPT-4o agent over 10 independent trials (see Figure 10). Across these trials, the mean reward rose from approximately 0.29 to 0.62. Although the reward has not fully converged by trial 10, the rate of improvement diminishes, indicating a flattening slope. Meanwhile, the average number of rounds continues to grow steadily. This follows that all our test-time in-context learning methods perform worse than baseline models in pass@k at high k values. A notion very similarly observed for RLVR approaches in [58].

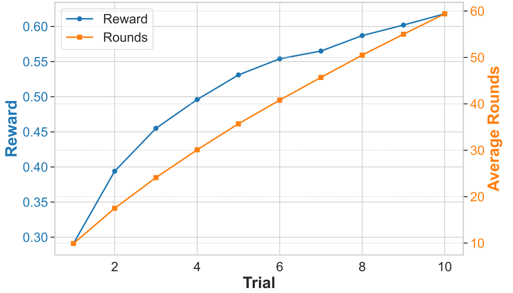


Figure 10: Average rounds and average reward with increasing trials. Tested with base agent + GPT-4o.

F.4 Prompts and Example

In Figure 15, we show the full processed details of an example run. At each step, we instruct LLM to give a thinking and an action in an ReACT style. The question generation prompt and the solution generation prompt are in Figure 17 and 18. We show the prompt for baseline model in Figure 11 and the enhanced version for reasoning models (e.g., `o1-mini`) in Figure 12. The strategy prompt is shown in Figure 14. We show five rules extracted with Expel in 13 and one example used in ReAct in Figure 16.

You are a security analyst.
You need to answer a given security question by querying the database.
The logs are stored in a MySQL database, you can use SQL queries to retrieve entries as needed. Note there are more than 20 tables in the database, so you may need to explore the schema or check example entries to understand the database structure.

Your response should always be a thought-action pair:
Thought: your reasoning
Action: your SQL query

In Thought, you can analyze and reason about the current situation,
Action can be one of the following:
(1) execute[your query], which executes the SQL query
(2) submit[your answer], which is the final answer to the question

Figure 11: Base Prompt for Baseline Agent.

BASE_PROMPT +
You should only give one thought-action per response. The action from your response will be executed and the result will be shown to you.
Follow the format ”Thought: ....
nAction: ....” exactly.
Do not include any other information in your response. Wait for the response from one action before giving the next thought-action pair. DO NOT make assumptions about the data that are not observed in the logs.

Figure 12: Additional prompt added when testing with `o3-mini`, `o4-mini`, `o1-mini`.

- Ensure that extracted IP addresses or other critical data directly aligns with the cybersecurity incident’s timeframe and context. This avoids errors in connecting unrelated entities or events and ensures actionable insights.
- Align queries explicitly with essential relational identifiers (e.g., AlertId, ProcessId, DeviceName) relevant to the cybersecurity incident to ensure precise evidence extraction. Avoid unnecessary queries to broader tables unless relational data is exhausted, ensuring query efficiency and high investigation value.
- Refine queries to focus specifically on the relevant timeframe, user accounts, and IP details tied to suspicious activity to avoid noise from unrelated data. Leverage alert-specific metadata like Alert IDs or IP addresses directly linked to the cybersecurity incident to maintain query precision and deliver actionable insights.
- Focus queries specifically on the timeframe and context of the suspicious activity, ensuring alignment with the provided incident timeline to maintain investigation precision and actionable insights. This prevents errors and avoids unnecessary focus on unrelated data.
- Balancing initial exploration with leveraging primary attributes such as DeviceId, AccountUpn, or NetworkMessageId is essential. Avoid premature scope expansion before thoroughly investigating relevant relational links and structured data to target accessible evidence effectively while minimizing errors and misalignments.
...

Figure 13: **Five example rules extracted with Expel.** An Expel consists of the base prompt, all the extracted rules, and 1 demonstration trajectory.

You are a security analyst working on investigating a security incident.
You need to answer a given question about the security incident by querying the database of security logs provided to you.
The logs are stored in a MySQL database, you can use SQL queries to retrieve entries as needed.
Note there are more than 20 tables in the database, so you may need to explore the schema or check example entries to understand the database structure.

A security incident is composed of a group of related alerts connected by data elements or entities that are shared between the alerts such as User accounts, Hosts, Mailboxes, IP addresses, Files, Cloud applications, Processes, URLs etc. Alerts are signals that result from various threat detection activities. These signals indicate the occurrence of malicious or suspicious events in your environment. Besides the security alert signals, you can also find additional information in the raw logs like device events, user activities, network traffic, etc.

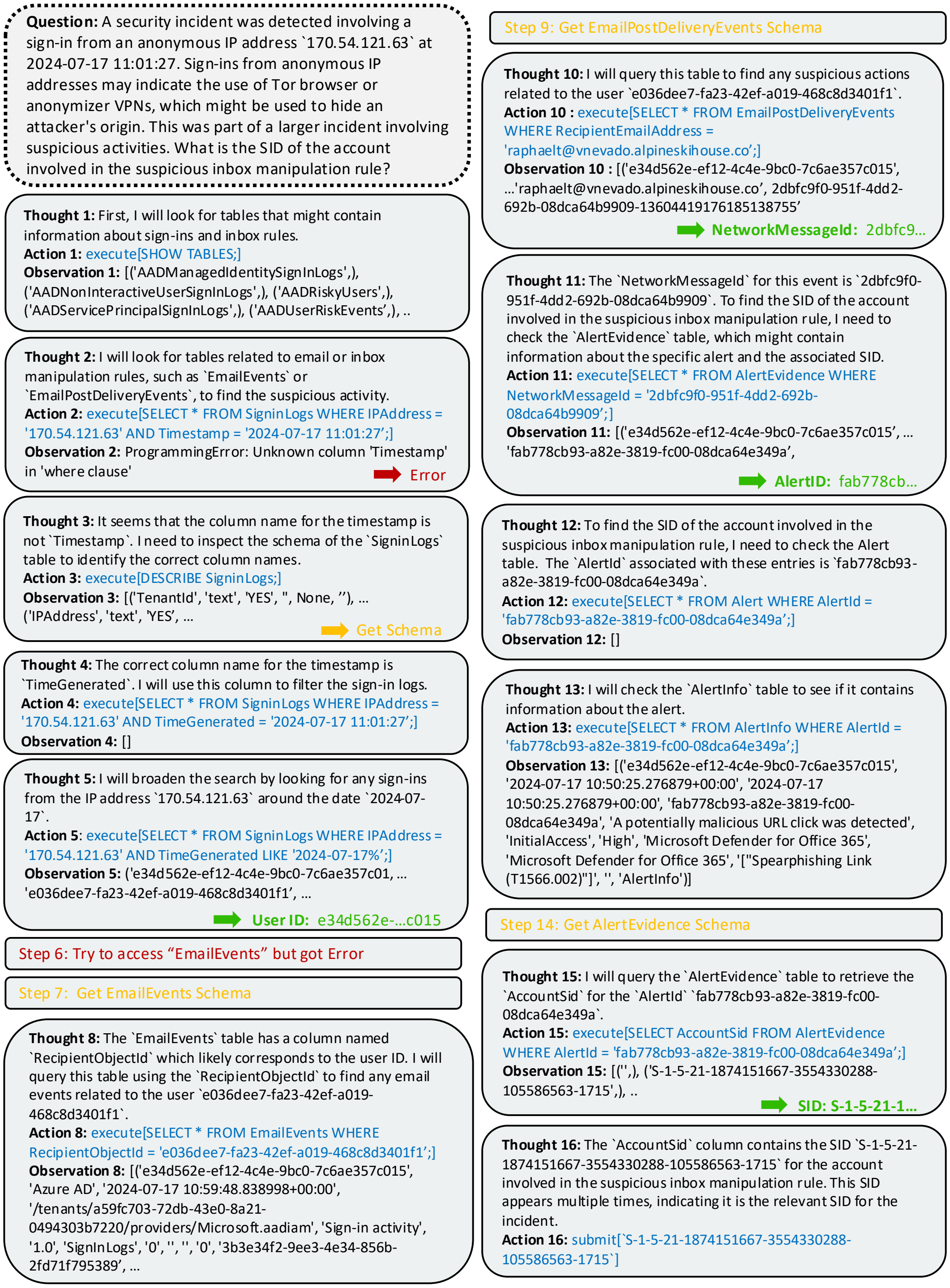
The best way to approach the question is to start from the alert(s) that are part of the security incident and then explore the related logs to understand the context of the incident. If the security incident table is available to you, first thing to do should be to find all the information about the given incident from the security incident table.

Other important tables to look at are the alert tables, which contains alerts and other additional information that was generated by the security monitoring tools. Once you find the alerts that are part of the security incident, you can explore the entities that are shared between the alerts to understand the context of the incident. These entities can be further used to find additional information from the logs and even alerts that you might have missed.

Your response should always be a thought-action pair:
Thought: ¡your reasoning¿
Action: ¡your SQL query¿

In Thought, you can analyze and reason about the current situation,
Action can be one of the following:
(1) execute[¡your query¿], which executes the SQL query
(2) submit[¡your answer¿], which is the final answer to the question

Figure 14: Strategy Prompt.



Question: A security incident was detected involving a sign-in from an anonymous IP address `170.54.121.63` at 2024-07-17 11:01:27. Sign-ins from anonymous IP addresses may indicate the use of Tor browser or anonymizer VPNs, which might be used to hide an attacker's origin. This was part of a larger incident involving suspicious activities. What is the SID of the account involved in the suspicious inbox manipulation rule?

Thought 1: First, I will look for tables that might contain information about sign-ins and inbox rules.
Action 1: execute[SHOW TABLES;]
Observation 1: [('AADManagedIdentitySignInLogs',), ('AADNonInteractiveUserSignInLogs',), ('AADRiskyUsers',), ('AADServicePrincipalSignInLogs',), ('AADUserRiskEvents',), ..

Thought 2: I will look for tables related to email or inbox manipulation rules, such as `EmailEvents` or `EmailPostDeliveryEvents`, to find the suspicious activity.
Action 2: execute[SELECT * FROM SigninLogs WHERE IPAddress = '170.54.121.63' AND Timestamp = '2024-07-17 11:01:27';]
Observation 2: ProgrammingError: Unknown column 'Timestamp' in 'where clause'

➡ Error

Thought 3: It seems that the column name for the timestamp is not `Timestamp`. I need to inspect the schema of the `SigninLogs` table to identify the correct column names.
Action 3: execute[DESCRIBE SigninLogs;]
Observation 3: [('TenantId', 'text', 'YES', '', None, ''), ... ('IPAddress', 'text', 'YES', ...

➡ Get Schema

Thought 4: The correct column name for the timestamp is `TimeGenerated`. I will use this column to filter the sign-in logs.
Action 4: execute[SELECT * FROM SigninLogs WHERE IPAddress = '170.54.121.63' AND TimeGenerated = '2024-07-17 11:01:27';]
Observation 4: []

Thought 5: I will broaden the search by looking for any sign-ins from the IP address `170.54.121.63` around the date `2024-07-17`.
Action 5: execute[SELECT * FROM SigninLogs WHERE IPAddress = '170.54.121.63' AND TimeGenerated LIKE '2024-07-17%';]
Observation 5: ('e34d562e-ef12-4c4e-9bc0-7c6ae357c01, ... 'e036dee7-fa23-42ef-a019-468c8d3401f1', ...

➡ **User ID:** e34d562e-...c015

Step 6: Try to access "EmailEvents" but got Error

Step 7: Get EmailEvents Schema

Thought 8: The `EmailEvents` table has a column named `RecipientObjectId` which likely corresponds to the user ID. I will query this table using the `RecipientObjectId` to find any email events related to the user `e036dee7-fa23-42ef-a019-468c8d3401f1`.
Action 8: execute[SELECT * FROM EmailEvents WHERE RecipientObjectId = 'e036dee7-fa23-42ef-a019-468c8d3401f1';]
Observation 8: [('e34d562e-ef12-4c4e-9bc0-7c6ae357c015', 'Azure AD', '2024-07-17 10:59:48.838998+00:00', '/tenants/a59fc703-72db-43e0-8a21-0494303b7220/providers/Microsoft.aadiam', 'Sign-in activity', '1.0', 'SignInLogs', '0', '', '', '0', '3b3e34f2-9ee3-4e34-856b-2fd71f795389', ...

Step 9: Get EmailPostDeliveryEvents Schema

Thought 10: I will query this table to find any suspicious actions related to the user `e036dee7-fa23-42ef-a019-468c8d3401f1`.
Action 10 : execute[SELECT * FROM EmailPostDeliveryEvents WHERE RecipientEmailAddress = 'raphaelt@vnevado.alpineskihouse.co';]
Observation 10 : [('e34d562e-ef12-4c4e-9bc0-7c6ae357c015', ...'raphaelt@vnevado.alpineskihouse.co', 2dbfc9f0-951f-4dd2-692b-08dca64b9909-13604419176185138755'

➡ **NetworkMessageId:** 2dbfc9...

Thought 11: The `NetworkMessageId` for this event is `2dbfc9f0-951f-4dd2-692b-08dca64b9909`. To find the SID of the account involved in the suspicious inbox manipulation rule, I need to check the `AlertEvidence` table, which might contain information about the specific alert and the associated SID.
Action 11: execute[SELECT * FROM AlertEvidence WHERE NetworkMessageId = '2dbfc9f0-951f-4dd2-692b-08dca64b9909';]
Observation 11: [('e34d562e-ef12-4c4e-9bc0-7c6ae357c015', ... 'fab778cb93-a82e-3819-fc00-08dca64e349a',

➡ **AlertID:** fab778cb...

Thought 12: To find the SID of the account involved in the suspicious inbox manipulation rule, I need to check the Alert table. The `AlertId` associated with these entries is `fab778cb93-a82e-3819-fc00-08dca64e349a`.
Action 12: execute[SELECT * FROM Alert WHERE AlertId = 'fab778cb93-a82e-3819-fc00-08dca64e349a';]
Observation 12: []

Thought 13: I will check the `AlertInfo` table to see if it contains information about the alert.
Action 13: execute[SELECT * FROM AlertInfo WHERE AlertId = 'fab778cb93-a82e-3819-fc00-08dca64e349a';]
Observation 13: [('e34d562e-ef12-4c4e-9bc0-7c6ae357c015', '2024-07-17 10:50:25.276879+00:00', '2024-07-17 10:50:25.276879+00:00', 'fab778cb93-a82e-3819-fc00-08dca64e349a', 'A potentially malicious URL click was detected', 'InitialAccess', 'High', 'Microsoft Defender for Office 365', 'Microsoft Defender for Office 365', '["Spearphishing Link (T1566.002)"]', '', 'AlertInfo')]

Step 14: Get AlertEvidence Schema

Thought 15: I will query the `AlertEvidence` table to retrieve the `AccountSid` for the `AlertId` `fab778cb93-a82e-3819-fc00-08dca64e349a`.
Action 15: execute[SELECT AccountSid FROM AlertEvidence WHERE AlertId = 'fab778cb93-a82e-3819-fc00-08dca64e349a';]
Observation 15: [('',), ('S-1-5-21-1874151667-3554330288-105586563-1715',), ..

➡ **SID: S-1-5-21-1...**

Thought 16: The `AccountSid` column contains the SID `S-1-5-21-1874151667-3554330288-105586563-1715` for the account involved in the suspicious inbox manipulation rule. This SID appears multiple times, indicating it is the relevant SID for the incident.
Action 16: submit[`S-1-5-21-1874151667-3554330288-105586563-1715`]

Figure 15: The full example of agent (with GPT-4o) solving a question.

Question:
An automated investigation was initiated manually by u141(u141@ash.alpineskihouse.co) on the host ‘vnevado-win10e‘. This investigation aimed to identify and review threat artifacts for potential remediation. As part of this security incident, a suspicious process executed a PowerShell command. Can you identify the account name that was associated with this activity?

Thought: I will start by listing all the tables in the database to understand the structure.
Action: execute[SHOW TABLES]
Observation:

```
[('AADManagedIdentitySignInLogs',), ('AADNonInteractiveUserSignInLogs',), ('AADProvisioningLogs',),
('AADRiskyUsers',), ('AADServicePrincipalSignInLogs',), ('AADUserRiskEvents',), ('AlertEvidence',),
('AlertInfo',), ('AuditLogs',), ('AzureMetrics',), ('CloudAppEvents',), ('DeviceEvents',),
('DeviceFileCertificateInfo',), ('DeviceFileEvents',), ('DeviceImageLoadEvents',), ('DeviceInfo',),
('DeviceLogonEvents',), ('DeviceNetworkEvents',), ('DeviceNetworkInfo',), ('DeviceProcessEvents',),
('DeviceRegistryEvents',), ('EmailAttachmentInfo',), ('EmailEvents',), ('EmailUrlInfo',),
('Heartbeat',), ('IdentityDirectoryEvents',), ('IdentityLogonEvents',), ('IdentityQueryEvents',),
('MicrosoftAzureBastionAuditLogs',), ('MicrosoftGraphActivityLogs',), ('SecurityAlert',),
('SecurityIncident',), ('SigninLogs',), ('ThreatIntelligenceIndicator',), ('UrlClickEvents',),
('Usage',)]
```

Thought: The ‘SecurityIncident‘ table exists, which likely contains information about security incidents. I will query this table to find the incident initiated by ‘u141‘ on the host ‘vnevado-win10e‘.
Action: execute[SELECT * FROM SecurityIncident WHERE initiated_by = 'u141' AND host = 'vnevado-win10e']
Observation:
ProgrammingError: Unknown column 'initiated_by' in 'where clause'

Thought: It seems the ‘SecurityIncident‘ table does not have a column named ‘initiated_by‘. I need to inspect the structure of the ‘SecurityIncident‘ table to understand its columns and find the relevant information.
Action: execute[DESCRIBE SecurityIncident]
Observation:

```
[('TenantId', 'text', 'YES', '', None, ''), ('TimeGenerated', 'text', 'YES', '', None, ''), ...]
```

Thought: The ‘Owner‘ column may store the user who initiated the incident. I will check for incidents where ‘Owner = 'u141'‘ and the description or title mentions ‘vnevado-win10e‘.
Action: execute[SELECT * FROM SecurityIncident WHERE Owner = 'u141' AND (Description LIKE 'Observation:

```
[]
```

Thought: The incident details might be in ‘AlertInfo‘, which stores alerts related to security incidents. I will check its structure.
Action: execute[DESCRIBE AlertInfo]
Observation:

```
[('TenantId', 'text', 'YES', '', None, ''), ..., ]
```

Thought: To find the specific alert related to the PowerShell command execution, I will search for alerts with titles or categories related to PowerShell on the host ‘vnevado-win10e‘.
Action: execute[SELECT * FROM AlertInfo WHERE Title LIKE 'Observation:

```
[(..., 'dafa552c38-a2c9-4c21-a7a8-94e0ae671894_1', 'Suspicious PowerShell download or encoded command
execution', ...), (...,'da11fca4fc-2be9-4723-b9aa-7e66393c30c6_1', ...), ...]
```

Thought: I have found several alerts related to suspicious PowerShell activities. The next step is to identify the specific alert that occurred on the host ‘vnevado-win10e‘. I will need to look into the ‘AlertEvidence‘ table to find evidence related to these alerts and check if any of them are associated with the host ‘vnevado-win10e‘.
Action: execute[SELECT * FROM AlertEvidence WHERE AlertId IN ('dafa552c38-a2c9-4c21-a7a8-94e0ae671894_1', 'da11fca4fc-2be9-4723-b9aa-7e66393c30c6_1', 'dadcc0cbd1-785d-4bbc-8217-22d99635321b_1', 'da6d201663-dfda-4823-8952-1a50128ba9bc_1', 'da3395ee5a-7be3-49e7-8b10-adcfd307354a_1') AND HostName = 'vnevado-win10e']
Observation:
ProgrammingError: Unknown column 'HostName' in 'where clause'

Thought: It seems the ‘AlertEvidence‘ table does not have a column named ‘HostName‘. I need to inspect the structure of the ‘AlertEvidence‘ table to understand its columns and find the relevant information.
Action: execute[DESCRIBE AlertEvidence]
Observation:

```
[('TenantId', 'text', 'YES', '', None, ''), ...]
```

Thought: I will check ‘AlertEvidence‘ for any evidence linked to ‘vnevado-win10e‘.
Action: execute[SELECT * FROM AlertEvidence WHERE AlertId IN ('dafa552c38-a2c9-4c21-a7a8-94e0ae671894_1', ...) AND DeviceName = 'vnevado-win10e']
Observation:

```
[]
```

Thought: Since there are no results from the ‘AlertEvidence‘ table for the given criteria, it is possible that the evidence related to the host ‘vnevado-win10e‘ is not directly linked in the way I queried. I will query the ‘AlertEvidence‘ table again, but this time I will look for any evidence related to the alerts that might contain account information, such as ‘AccountName‘.
Action: execute[SELECT AlertId, AccountName FROM AlertEvidence WHERE AlertId IN ('dafa552c38-a2c9-4c21-a7a8-94e0ae671894_1', 'da11fca4fc-2be9-4723-b9aa-7e66393c30c6_1', 'dadcc0cbd1-785d-4bbc-8217-22d99635321b_1', 'da6d201663-dfda-4823-8952-1a50128ba9bc_1', 'da3395ee5a-7be3-49e7-8b10-adcfd307354a_1')]
Observation:

```
[('dafa552c38-a2c9-4c21-a7a8-94e0ae671894_1', ''), ('dafa552c38-a2c9-4c21-a7a8-94e0ae671894_1',
'samuelf'), ('dadcc0cbd1-785d-4bbc-8217-22d99635321b_1', ''), ('da6d201663-dfda-4823-8952-1a50128ba9bc_1',
'samuelf'), ...]
```

Thought: The account ‘samuelf‘ is associated with the suspicious PowerShell execution.
Action: submit[The account name associated with the suspicious PowerShell command execution is 'samuelf'.]

Figure 16: **ReAct Example.** For react prompt, we use the base prompt + 3 examples. Here we show one of the examples used.

Your goal is to ask a security question from the given data from a security analyst’s perspective. You are given the start alert and end alert, and corresponding entities. The two alerts are connected by a alert-entity path. The start and end alert might be the same. You will use the start alert as the context, and ask a question about the entities in the end alert.

The JSON must have the following fields:
- ”question”: the question about the end alert. The question should be carefully crafted so that:
1. The question should be natural and relevant to the context, and it should be clear and have a deterministic answer.
2. But it should not leak the answer. If the start and end alert are the same, you should be more careful since the given entities may have overlapping information.
3. The question should be specific of the answer you are looking for, and the answer should match the question.
- ”answer”: the answer to the question. You may be given one or more entities from the end alert, select the most meaningful entity and make sure it is not leaked in the context or question.
- ”context”: the context from the start alert. you should combine the alert and the entities given in a consistent sentence. You can simplify the context a bit if it is too long. Make sure the answer is not leaked in the context. If the start alert or the related entities contains the answer, you should remove it from the context.

Examples:
##############
Start Alert:
Time: 8/14/2024, 10:34:41.578 PM
Name: Ntdsutil collecting Active Directory information
Description: Attackers might be using Ntdsutil to gather information for persistence or to move laterally in a network or organization. Ntdsutil is a command line tool that provides management facilities for Active Directory Domain Services (AD DS) and Active Directory Lightweight Directory Services (AD LDS). It was launched to maintain the database of AD DS.
Entities from this alert:
Type: process, Field: ExtractedFileName, Value: ‘powershell.exe‘
Type: host, Field: HostName, Value: ‘vnevado-dc‘

End Alert:
Time: 8/14/2024, 10:34:41.578 PM
Name: Ntdsutil collecting Active Directory information
Description: Attackers might be using Ntdsutil to gather information for persistence or to move laterally in a network or organization. Ntdsutil is a command line tool that provides management facilities for Active Directory Domain Services (AD DS) and Active Directory Lightweight Directory Services (AD LDS). It was launched to maintain the database of AD DS.
Entities from this alert: Type: process, Field: ProcessId__CreatedTimeUtc__CommandLine, Value: ‘2556__2024-08-01t12:37:29.6522416z__”powershell.exe” -encodedcommand iabuahqazabz...‘
##############
Your response:

”context”: ”A file ‘powershell.exe‘ was launched on host ‘vnevado-dc‘, which might be an indicator of an attacker using Ntdsutil to gather information for persistence or to move laterally in a network or organization. Note: Ntdsutil is a command line tool that provides management facilities for Active Directory Domain Services (AD DS) and Active Directory Lightweight Directory Services (AD LDS). It was launched to maintain the database of AD DS.”,
”question”: ”When was the last time the file ‘powershell.exe‘ was launched on host ‘vnevado-dc‘, and what was the process ID?”,
”answer”: ”Time: 2024-08-01t12:37:29.6522416, Process Id: 2556”

##############
##############
Start Alert:
Time: 8/14/2024, 10:34:41.429 PM
Name: Suspicious credential dump from NTDS.dit
Description: Attackers dump NTDS.dit in order to obtain user’s credentials which are stored in the domain controller.
Entities from this alert:
Type: process, Field: ProcessId__CreatedTimeUtc__CommandLine, Value: ‘6748__2024-08-01t12:37:30.2769191z__”ntdsutil.exe” ”ac i ntds” ifm ”create full c:
temp” q q‘
Type: process, Field: ExtractedFileName, Value: ‘ntdsutil.exe‘

End Alert:
Time: 8/14/2024, 10:37:13.064 PM
Name: Suspicious Azure Resource Management activities by a risky user
Description: Suspicious cloud Azure Resource Management (ARM) activities were performed by a user account that signed in to a risky session. This alert was triggered based on a Microsoft Defender for Cloud alert related to ARM and Microsoft Entra ID Protection risk scores. Entities from this alert:
Type: account, Field: Email, Value: ‘Megan Bower@vnevado.alpineskihouse.co‘
##############
Your response:

”context”: ”A file ‘ntdsutil.exe‘ was launched with this command line: ‘ntdsutil.exe ac i ntds ifm create full c:
temp q q‘. The Process ID was 6748. This process might be an indicator of an attacker dumping NTDS.dit in order to obtain user’s credentials which are stored in the domain controller.”,
”question: ”Related to this alert, there is also a suspicious Azure Resource Management (ARM) activities, which is likely from the same user. Can you get the email of the user who performed the suspicious ARM activities?”,
”answer”: ”Megan Bower@vnevado.alpineskihouse.co”,

##############
(...one more example)
##############

Figure 17: Question Generation Prompt.

```
Given an alert-entity path, please generate a solution path, where the question asks about the end entity.
In each step of the solution path, please make sure you include the entity field and value.

Your response should be in JSON format, containing field ”solution” which is a list of strings.

Examples:
##########
Solution path:
Time: 8/14/2024, 10:34:41.578 PM
Name: Ntdsutil collecting Active Directory information
Description: Attackers might be using Ntdsutil to gather information for persistence or to move laterally in a network or organization. Ntdsutil is a command line tool that provides management facilities for Active Directory Domain Services (AD DS) and Active Directory Lightweight Directory Services (AD LDS). It was launched to maintain the database of AD DS.
Entities from this alert:
Type: process, Field: ProcessId__CreatedTimeUtc__CommandLine, Value: ‘6748__2024-08-01t12:37:30.2769191z__”ntdsutil.exe” ”ac i ntds” ifm ”create full c:
temp” q q‘
##########
Your response:
{
”solution”: [
”The attacker launched ntdsutil with the command line ‘ntdsutil.exe ac i ntds ifm create full c:
temp q q‘ at ‘2024-08-01t12:37:30.2769191z‘, with Process ID ‘6748‘.”
]
}
##########
##########
Solution path:
Time: 8/14/2024, 10:34:41.578 PM
Name: Ntdsutil collecting Active Directory information
Description: Attackers might be using Ntdsutil to gather information for persistence or to move laterally in a network or organization. Ntdsutil is a command line tool that provides management facilities for Active Directory Domain Services (AD DS) and Active Directory Lightweight Directory Services (AD LDS). It was launched to maintain the database of AD DS.
Entities from this alert:
Type: host, Field: HostName, Value: ‘vnevado-dc‘

Time: 8/14/2024, 10:37:13.045 PM
Name: Azure Resource Manager operation from suspicious proxy IP address
Description: Microsoft Defender for Resource Manager detected a resource management operation from an IP address that is associated with proxy services, such as TOR. While this behavior can be legitimate, it’s often seen in malicious activities, when threat actors try to hide their source IP.
Entities from this alert:
Type: ip, Field: Address, Value: ‘185.220.101.1‘

Time: 8/14/2024, 10:37:13.064 PM
Name: Suspicious Azure Resource Management activities by a risky user
Description: Suspicious cloud Azure Resource Management (ARM) activities were performed by a user account that signed in to a risky session. This alert was triggered based on a Microsoft Defender for Cloud alert related to ARM and Microsoft Entra ID Protection risk scores.
Entities from this alert:
Type: account, Field: AadUserId, Value: ‘6c16dea3-5326-461e-a48e-38b527df3a70‘
##########
Your response:
{
”solution”: [
”There is a collection of active directory information with ntdsutil.exe on host ‘vnevado-dc‘.”,
”There is a suspicious Azure Resource Manager operation from a proxy IP address ‘185.220.101.1‘.”,
”There is a suspicious Azure Resource Management activities by a risky user with AadUserId ‘6c16dea3-5326-461e-a48e-38b527df3a70‘.”
]
}
##########
Solution path:
Time: 8/14/2024, 10:37:13.011 PM
Name: Email messages containing malicious URL removed after delivery
Description: Emails with malicious URL that were delivered and later removed -V1.0.0.3
Entities from this alert:
Type: account, Field: Name, Value: ‘Megan Bower‘

Time: 8/14/2024, 10:37:12.993 PM
Name: A potentially malicious URL click was detected
Description: We have detected that one of your users has recently clicked on a link that was found to be malicious. -V1.0.0.5
Entities from this alert:
Type: account, Field: Sid, Value: ‘S-1-5-21-1840151660-3534030288-105586563-1127‘
##########
Your response:

”solution”: [
”The email account ‘Megan Bower‘ received an email with a malicious URL.”,
”The user with SID ‘S-1-5-21-1840151660-3534030288-105586563-1127‘ clicked on the malicious URL.”
]

##########
```

Figure 18: Solution Generation Prompt.

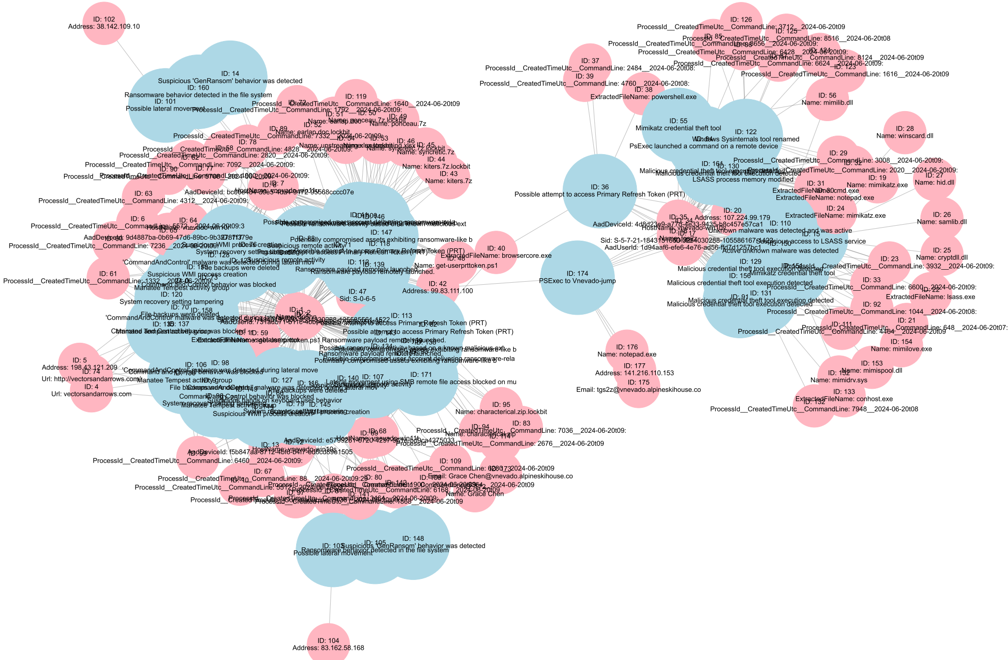

ID: 102
Address: 38.142.109.10
ID: 14
Suspicious 'GenRansom' behavior was detected
ID: 160
Ransomware behavior detected in the file system
ID: 101
Possible lateral movement
ID: 119
ProcessId__CreatedTimeUtc__CommandLine: 1640__2024-06-20t09
ID: 126
ProcessId__CreatedTimeUtc__CommandLine: 3712__2024-06-20t09
ID: 37
ID: 39
ProcessId__CreatedTimeUtc__CommandLine: 2484__2024-06-20t08:
ProcessId__CreatedTimeUtc__CommandLine: 4760__2024-06-20t08:
ID: 38
ExtractedFileName: powershell.exe
ID: 56
Name: mimilib.dll
ID: 55
Mimikatz credential theft tool
ID: 122
PsExec launched a command on a remote device
ID: 28
Name: winscard.dll
ID: 36
Possible attempt to access Primary Refresh Token (PRT)
ID: 29
ID: 27
Name: hid.dll
ID: 26
Name: samlib.dll
ID: 24
Name: mimikatz.exe
ID: 25
Name: cryptdll.dll
ID: 23
ID: 40
ExtractedFileName: browsercore.exe
ID: 174
PSExec to Vnevado-jump
ID: 42
Address: 99.83.111.100
ID: 47
Sid: S-0-6-5
ID: 113
Possible attempt to access Primary Refresh Token (PRT)
ID: 129
Malicious credential theft tool execution detected
ID: 131
ID: 33
ID: 92
ID: 21
ID: 154
Name: mimilove.exe
ID: 153
Name: mimispool.dll
ID: 152
Name: mimidrv.sys
ID: 133
ExtractedFileName: conhost.exe
ID: 176
Name: notepad.exe
ID: 177
Address: 141.216.110.153
ID: 175
Email: tgs2z@vnevado.alpineskihouse.co
ID: 6
ID: 61
ID: 5
Address: 198.43.121.209
ID: 4
Url: vectorsandarrows.com
ID: 106
ID: 98
ID: 127
ID: 171
Lateral movement using SMB remote file access blocked on mu
ID: 95
Name: characterical.zip.lockbit
ID: 109
Name: Grace Chen
ID: 67
ID: 105
ID: 148
Suspicious 'GenRansom' behavior was detected
Ransomware behavior detected in the file system
Possible lateral movement
ID: 104
Address: 83.162.58.168

Figure 19: Graph of Incident 5. The bigger blue nodes represent alerts, and the smaller red nodes represent entities. (Only for illustrative purposes. Details can be hard to see.)

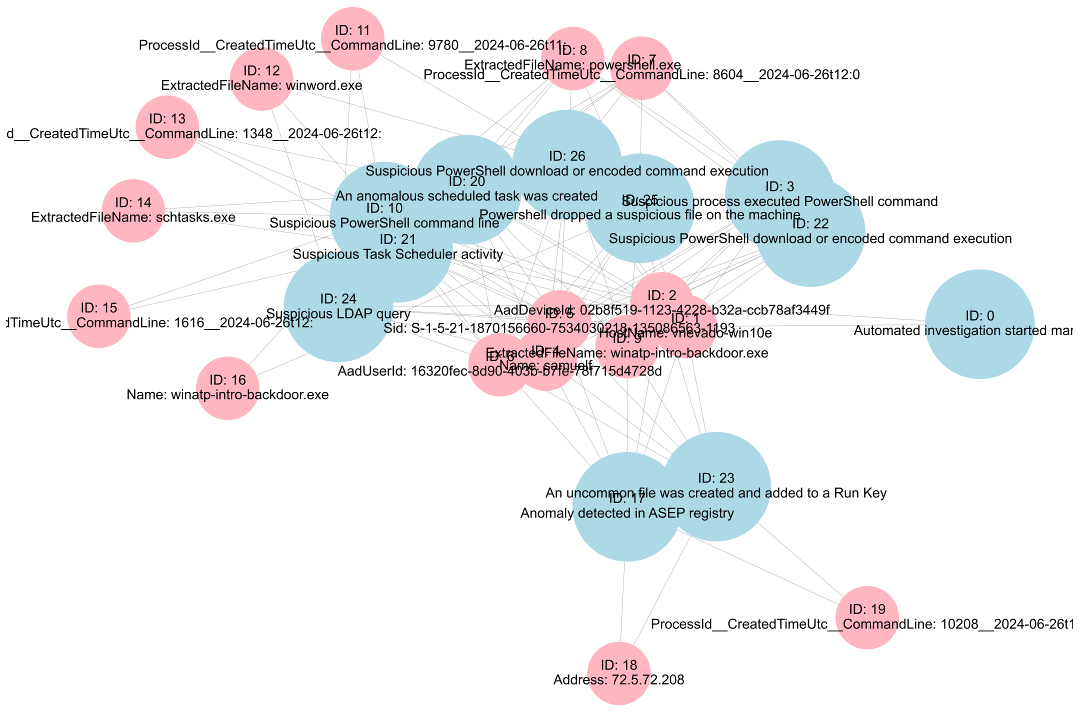

ID: 11
ProcessId__CreatedTimeUtc__CommandLine: 9780__2024-06-26t11:
ID: 8
ExtractedFileName: powershell.exe
ID: 7
ProcessId__CreatedTimeUtc__CommandLine: 8604__2024-06-26t12:0
ID: 12
ExtractedFileName: winword.exe
ID: 13
d__CreatedTimeUtc__CommandLine: 1348__2024-06-26t12:
ID: 26
Suspicious PowerShell download or encoded command execution
ID: 20
An anomalous scheduled task was created
ID: 3
Suspicious process executed PowerShell command
ID: 25
Powershell dropped a suspicious file on the machine
ID: 14
ExtractedFileName: schtasks.exe
ID: 10
Suspicious PowerShell command line
ID: 22
Suspicious PowerShell download or encoded command execution
ID: 21
Suspicious Task Scheduler activity
ID: 15
dTimeUtc__CommandLine: 1616__2024-06-26t12:
ID: 24
Suspicious LDAP query
ID: 2
AadDeviceId: 02b8f519-1123-4228-b32a-ccb78af3449f
ID: 5
ID: 1
Sid: S-1-5-21-1870156660-7534030218-135086563-1193
HostName: vnevado-win10e
ID: 9
ID: 0
Automated investigation started mar
ID: 4
ExtractedFileName: winatp-intro-backdoor.exe
ID: 6
Name: samuelf
AadUserId: 16320fec-8d90-403b-b7fe-78f715d4728d
ID: 16
Name: winatp-intro-backdoor.exe
ID: 23
An uncommon file was created and added to a Run Key
ID: 17
Anomaly detected in ASEP registry
ID: 19
ProcessId__CreatedTimeUtc__CommandLine: 10208__2024-06-26t1
ID: 18
Address: 72.5.72.208

Figure 20: Graph of Incident 34.

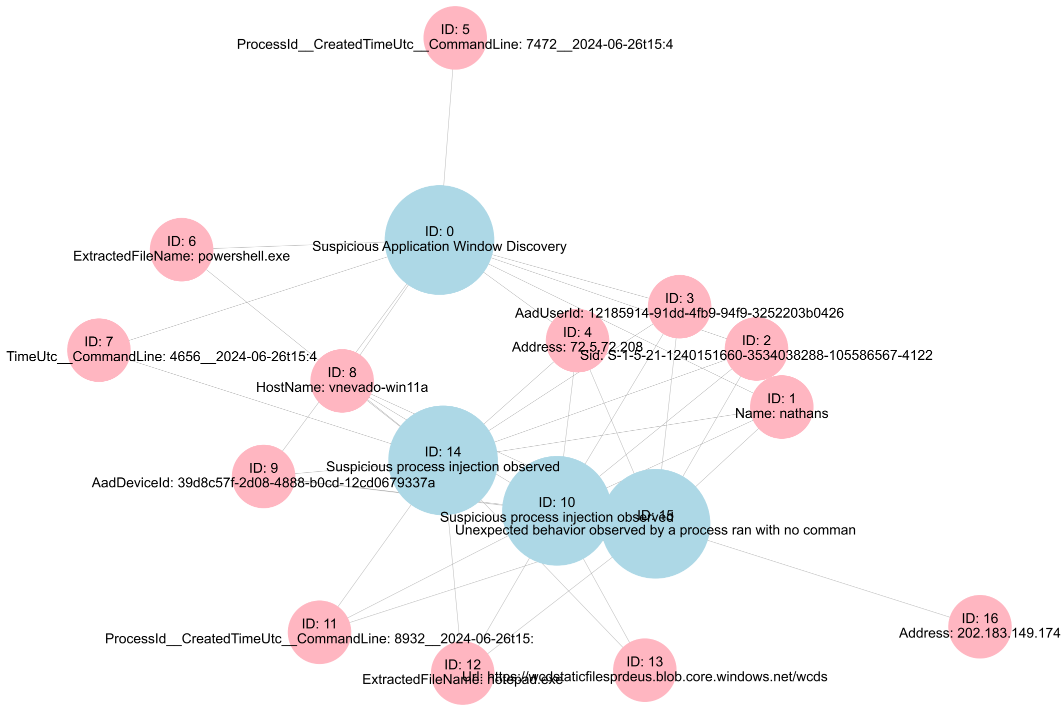

ID: 5
ProcessId__CreatedTimeUtc__CommandLine: 7472__2024-06-26t15:4
ID: 0
Suspicious Application Window Discovery
ID: 6
ExtractedFileName: powershell.exe
ID: 3
AadUserId: 12185914-91dd-4fb9-94f9-3252203b0426
ID: 7
TimeUtc__CommandLine: 4656__2024-06-26t15:4
ID: 4
Address: 72.5.72.208
ID: 2
Sid: S-1-5-21-1240151660-3534038288-105586567-4122
ID: 8
HostName: vnevado-win11a
ID: 1
Name: nathans
ID: 14
Suspicious process injection observed
ID: 9
AadDeviceId: 39d8c57f-2d08-4888-b0cd-12cd0679337a
ID: 10
Suspicious process injection observed
ID: 15
Unexpected behavior observed by a process ran with no comman
ID: 11
ProcessId__CreatedTimeUtc__CommandLine: 8932__2024-06-26t15:
ID: 16
Address: 202.183.149.174
ID: 12
ExtractedFileName: notepad.exe
ID: 13
Url: https://wcdstaticfilesprdeus.blob.core.windows.net/wcds

Figure 21: Graph of Incident 38.

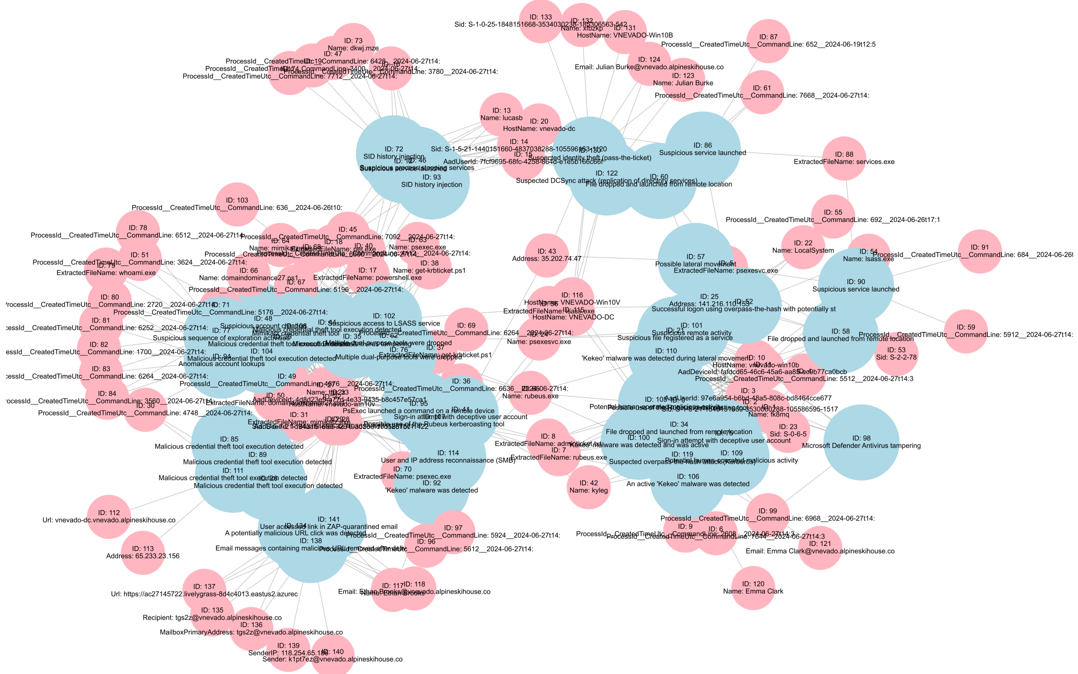

ID: 72
SID history injection
Suspicious service launched
ID: 93
SID history injection
ID: 86
Suspicious service launched
ID: 122
Suspected DCSync attack (replication of directory services)
ID: 60
File dropped and launched from remote location
ID: 57
Possible lateral movement
ID: 25
Successful logon using overpass-the-hash with potentially st
ID: 90
Suspicious service launched
ID: 58
File dropped and launched from remote location
ID: 101
Suspicious remote activity
ID: 110
'Kekeo' malware was detected during lateral movement
ID: 102
Suspicious access to LSASS service
ID: 104
Malicious credential theft tool execution detected
ID: 85
Malicious credential theft tool execution detected
ID: 89
Malicious credential theft tool execution detected
ID: 111
Malicious credential theft tool execution detected
ID: 114
User and IP address reconnaissance (SMB)
ID: 92
'Kekeo' malware was detected
ID: 100
Sign-in attempt with deceptive user account
ID: 106
An active 'Kekeo' malware was detected
ID: 98
Microsoft Defender Antivirus tampering
ID: 141
A potentially malicious URL click was detected
ID: 138
Email messages containing malicious URL removed after delivery
ID: 133
Sid: S-1-0-25-1848151668-3534030230-10558563-542
ID: 131
HostName: VNEVADO-Win10B
ID: 124
Email: Julian Burke@vnevado.alpineskihouse.co
ID: 123
Name: Julian Burke
ID: 87
ProcessId__CreatedTimeUtc__CommandLine: 652__2024-06-19t12:5
ID: 61
ProcessId__CreatedTimeUtc__CommandLine: 7668__2024-06-27t14:
ID: 88
ExtractedFileName: services.exe
ID: 13
Name: lucasb
ID: 20
HostName: vnevado-dc
ID: 73
Name: dkwj.mze
ID: 103
ProcessId__CreatedTimeUtc__CommandLine: 636__2024-06-26t10:
ID: 78
ProcessId__CreatedTimeUtc__CommandLine: 6512__2024-06-27t14:
ID: 51
ProcessId__CreatedTimeUtc__CommandLine: 3624__2024-06-27t14:
ExtractedFileName: whoami.exe
ID: 66
Name: domaindominance27.ps1
ID: 17
ExtractedFileName: powershell.exe
ID: 38
Name: get-krbticket.ps1
ID: 43
Address: 35.202.74.47
ID: 116
HostName: VNEVADO-Win10V
HostName: VNEVADO-DC
ID: 69
Name: psexesvc.exe
ID: 55
ProcessId__CreatedTimeUtc__CommandLine: 692__2024-06-26t17:1
ID: 22
Name: LocalSystem
ID: 54
Name: lsass.exe
ID: 91
ProcessId__CreatedTimeUtc__CommandLine: 684__2024-06-26
ID: 59
ProcessId__CreatedTimeUtc__CommandLine: 5912__2024-06-27t14:
ID: 53
Sid: S-2-2-78
ID: 80
ProcessId__CreatedTimeUtc__CommandLine: 2720__2024-06-27t14:
ID: 81
ProcessId__CreatedTimeUtc__CommandLine: 6252__2024-06-27t14:
ID: 82
ProcessId__CreatedTimeUtc__CommandLine: 1700__2024-06-27t14:
ID: 83
ProcessId__CreatedTimeUtc__CommandLine: 6264__2024-06-27t14:
ID: 84
ProcessId__CreatedTimeUtc__CommandLine: 3560__2024-06-27t14:
ID: 30
ProcessId__CreatedTimeUtc__CommandLine: 4748__2024-06-27t14:
ID: 36
Name: rubeus.exe
ID: 8
ExtractedFileName: rubeus.exe
ID: 42
Name: kyleg
ID: 23
Sid: S-0-6-5
ID: 112
Url: vnevado-dc.vnevado.alpineskihouse.co
ID: 113
Address: 65.233.23.156
ID: 97
ProcessId__CreatedTimeUtc__CommandLine: 5924__2024-06-27t14:
ID: 96
ProcessId__CreatedTimeUtc__CommandLine: 5612__2024-06-27t14:
ID: 99
ProcessId__CreatedTimeUtc__CommandLine: 6968__2024-06-27t14:
ID: 121
Email: Emma Clark@vnevado.alpineskihouse.co
ID: 120
Name: Emma Clark
ID: 137
Url: https://ac27145722.livelygrass-8d4c4013.eastus2.azurec
ID: 135
Recipient: tgs2z@vnevado.alpineskihouse.co
ID: 136
MailboxPrimaryAddress: tgs2z@vnevado.alpineskihouse.co
ID: 139
SenderIP: 118.254.65.185
ID: 140
Sender: k1pt7ez@vnevado.alpineskihouse.co
ID: 117
ID: 118

Figure 22: Graph of Incident 39.

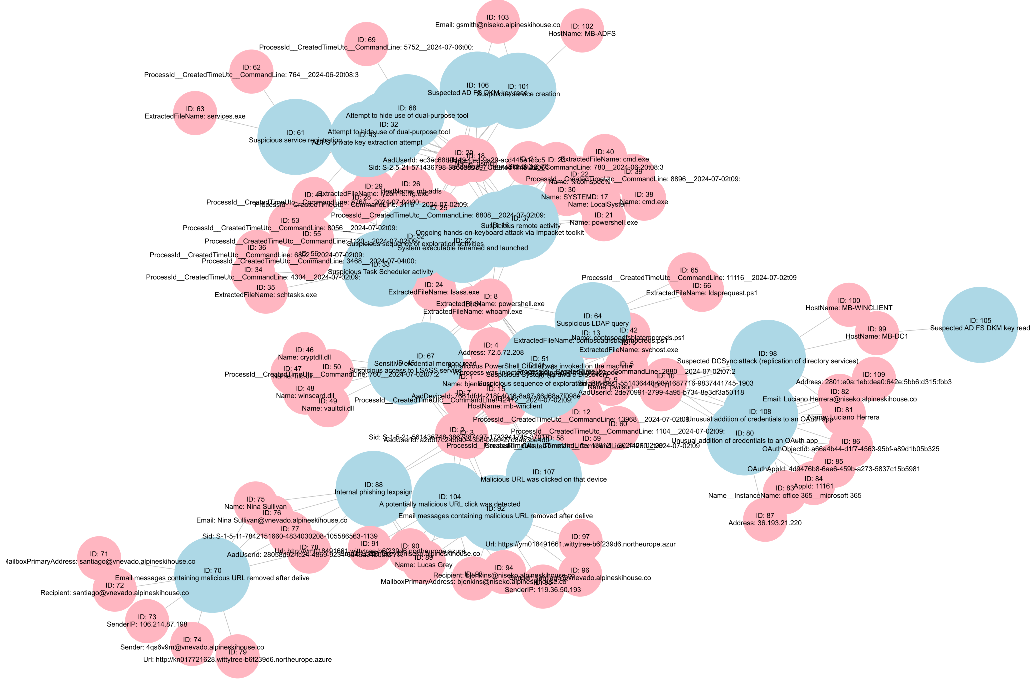

ID: 103
Email: gsmith@niseko.alpineskihouse.co
ID: 102
HostName: MB-ADFS
ID: 69
ProcessId__CreatedTimeUtc__CommandLine: 5752__2024-07-06t00:
ID: 62
ProcessId__CreatedTimeUtc__CommandLine: 764__2024-06-20t08:3
ID: 63
ExtractedFileName: services.exe
ID: 106
Suspected AD FS DKM key read
ID: 101
Suspicious service creation
ID: 68
Attempt to hide use of dual-purpose tool
ID: 61
Suspicious service registration
ID: 32
Attempt to hide use of dual-purpose tool
ADFS private key extraction attempt
ID: 40
ExtractedFileName: cmd.exe
ID: 38
Name: cmd.exe
ID: 21
Name: powershell.exe
Suspicious remote activity
Ongoing hands-on-keyboard attack via Impacket toolkit
Suspicious sequence of exploration activities
System executable renamed and launched
Suspicious Task Scheduler activity
ID: 24
ExtractedFileName: lsass.exe
ID: 34
ProcessId__CreatedTimeUtc__CommandLine: 4304__2024-07-02t09:
ID: 35
ExtractedFileName: schtasks.exe
ID: 65
ProcessId__CreatedTimeUtc__CommandLine: 11116__2024-07-02t09
ID: 66
ExtractedFileName: ldaprequest.ps1
ID: 64
Suspicious LDAP query
ID: 100
HostName: MB-WINCLIENT
ID: 99
HostName: MB-DC1
ID: 105
Suspected AD FS DKM key read
ID: 98
Suspected DCSync attack (replication of directory services)
ID: 109
Address: 2801:e0a:1eb:dea0:642e:5bb6:d315:fbb3
ID: 82
Email: Luciano Herrera@niseko.alpineskihouse.co
ID: 81
Name: Luciano Herrera
ID: 108
Unusual addition of credentials to an OAuth app
ID: 80
Unusual addition of credentials to an OAuth app
ID: 86
OAuthObjectId: a66a4b44-d1f7-4563-95bf-a89d1b05b325
ID: 85
OAuthAppId: 4d9476b8-6ae6-459b-a273-5837c15b5981
ID: 84
AppId: 11161
Name__InstanceName: office 365__microsoft 365
ID: 87
Address: 36.193.21.220
ID: 46
Name: cryptdll.dll
ID: 48
Name: winscard.dll
ID: 49
Name: vaultcli.dll
ID: 67
Sensitive credential memory read
Suspicious access to LSASS service
ID: 4
Address: 72.5.72.208
ID: 107
Malicious URL was clicked on that device
ID: 88
Internal phishing lexpaign
ID: 104
A potentially malicious URL click was detected
Email messages containing malicious URL removed after delive
ID: 75
Name: Nina Sullivan
ID: 76
Email: Nina Sullivan@vnevado.alpineskihouse.co
ID: 77
Sid: S-1-5-11-7842151660-4834030208-105586563-1139
ID: 90
Name: Lucas Grey
ID: 97
Url: https://ym018491661.wittytree-b6f239d6.northeurope.azur
ID: 94
MailboxPrimaryAddress: bjenkins@niseko.alpineskihouse.co
ID: 96
SenderIP: 119.36.50.193
ID: 71
MailboxPrimaryAddress: santiago@vnevado.alpineskihouse.co
ID: 70
Email messages containing malicious URL removed after delive
ID: 72
Recipient: santiago@vnevado.alpineskihouse.co
ID: 73
SenderIP: 106.214.87.198
ID: 74
Sender: 4qs6v9m@vnevado.alpineskihouse.co
ID: 79
Url: http://kn017721828.wittytree-b6f239d6.northeurope.azure

Figure 23: Graph of Incident 55.

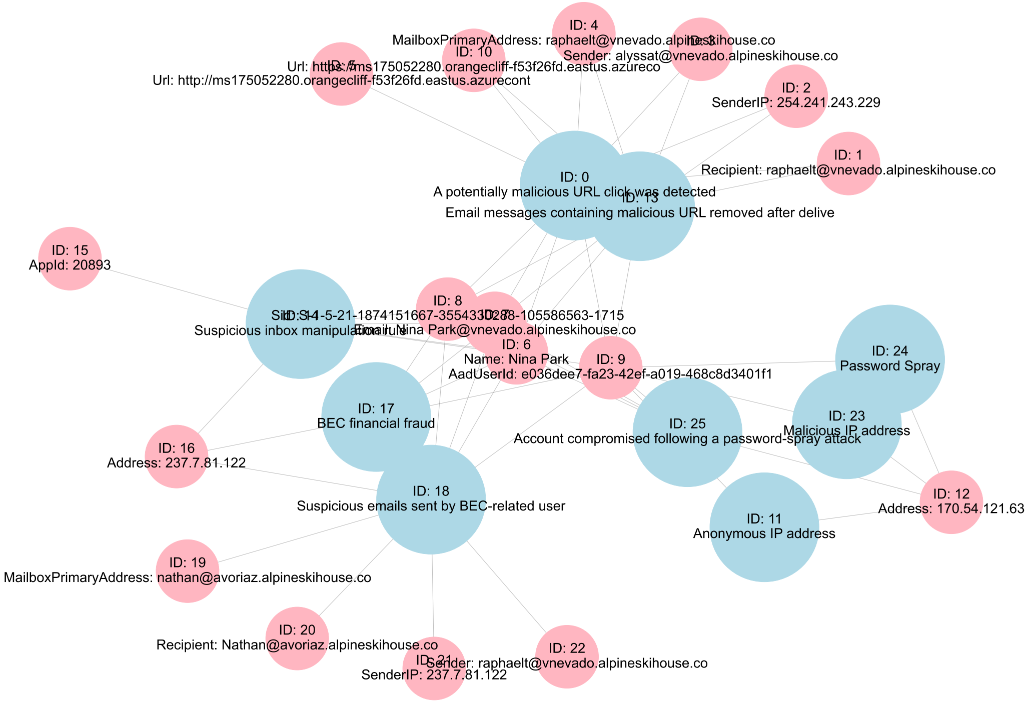

ID: 4
MailboxPrimaryAddress: raphaelt@vnevado.alpineskihouse.co
ID: 3
Sender: alyssat@vnevado.alpineskihouse.co
ID: 10
Url: https://ms175052280.orangecliff-f53f26fd.eastus.azureco
ID: 5
Url: http://ms175052280.orangecliff-f53f26fd.eastus.azurecont
ID: 2
SenderIP: 254.241.243.229
ID: 1
Recipient: raphaelt@vnevado.alpineskihouse.co
ID: 0
A potentially malicious URL click was detected
ID: 13
Email messages containing malicious URL removed after delive
ID: 15
AppId: 20893
ID: 14
Suspicious inbox manipulation rule
ID: 8
Sid: S-1-5-21-1874151667-3554330288-105586563-1715
ID: 7
Email: Nina Park@vnevado.alpineskihouse.co
ID: 6
Name: Nina Park
ID: 9
AadUserId: e036dee7-fa23-42ef-a019-468c8d3401f1
ID: 24
Password Spray
ID: 17
BEC financial fraud
ID: 23
Malicious IP address
ID: 25
Account compromised following a password-spray attack
ID: 16
Address: 237.7.81.122
ID: 18
Suspicious emails sent by BEC-related user
ID: 12
Address: 170.54.121.63
ID: 11
Anonymous IP address
ID: 19
MailboxPrimaryAddress: nathan@avoriaz.alpineskihouse.co
ID: 20
Recipient: Nathan@avoriaz.alpineskihouse.co
ID: 22
Sender: raphaelt@vnevado.alpineskihouse.co
ID: 21
SenderIP: 237.7.81.122

Figure 24: Graph of Incident 134.

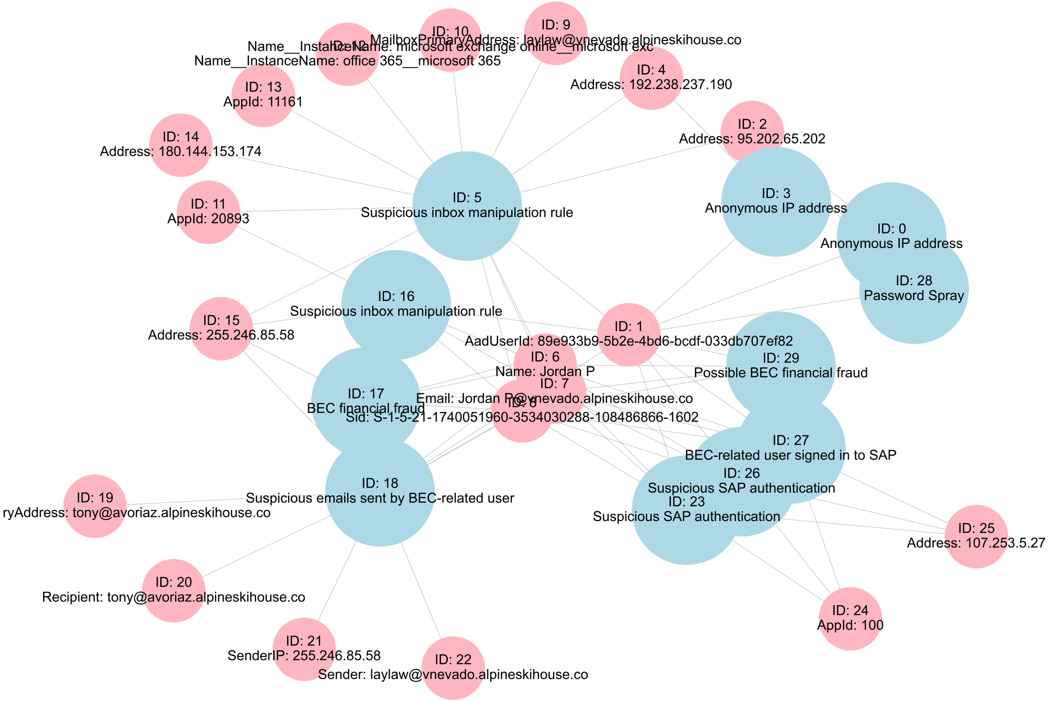

ID: 9
ID: 10
MailboxPrimaryAddress: laylaw@vnevado.alpineskihouse.co
Name__InstanceName: microsoft exchange online__microsoft exc
ID: 12
Name__InstanceName: office 365__microsoft 365
ID: 4
Address: 192.238.237.190
ID: 13
AppId: 11161
ID: 14
Address: 180.144.153.174
ID: 2
Address: 95.202.65.202
ID: 5
Suspicious inbox manipulation rule
ID: 3
Anonymous IP address
ID: 11
AppId: 20893
ID: 0
Anonymous IP address
ID: 16
Suspicious inbox manipulation rule
ID: 28
Password Spray
ID: 15
Address: 255.246.85.58
ID: 1
AadUserId: 89e933b9-5b2e-4bd6-bcdf-033db707ef82
ID: 6
Name: Jordan P
ID: 29
Possible BEC financial fraud
ID: 7
Email: Jordan P@vnevado.alpineskihouse.co
ID: 17
BEC financial fraud
ID: 8
Sid: S-1-5-21-1740051960-3534030288-108486866-1602
ID: 27
BEC-related user signed in to SAP
ID: 26
Suspicious SAP authentication
ID: 18
Suspicious emails sent by BEC-related user
ID: 19
ryAddress: tony@avoriaz.alpineskihouse.co
ID: 23
Suspicious SAP authentication
ID: 25
Address: 107.253.5.27
ID: 20
Recipient: tony@avoriaz.alpineskihouse.co
ID: 24
AppId: 100
ID: 21
SenderIP: 255.246.85.58
ID: 22
Sender: laylaw@vnevado.alpineskihouse.co

Figure 25: Graph of Incident 166.

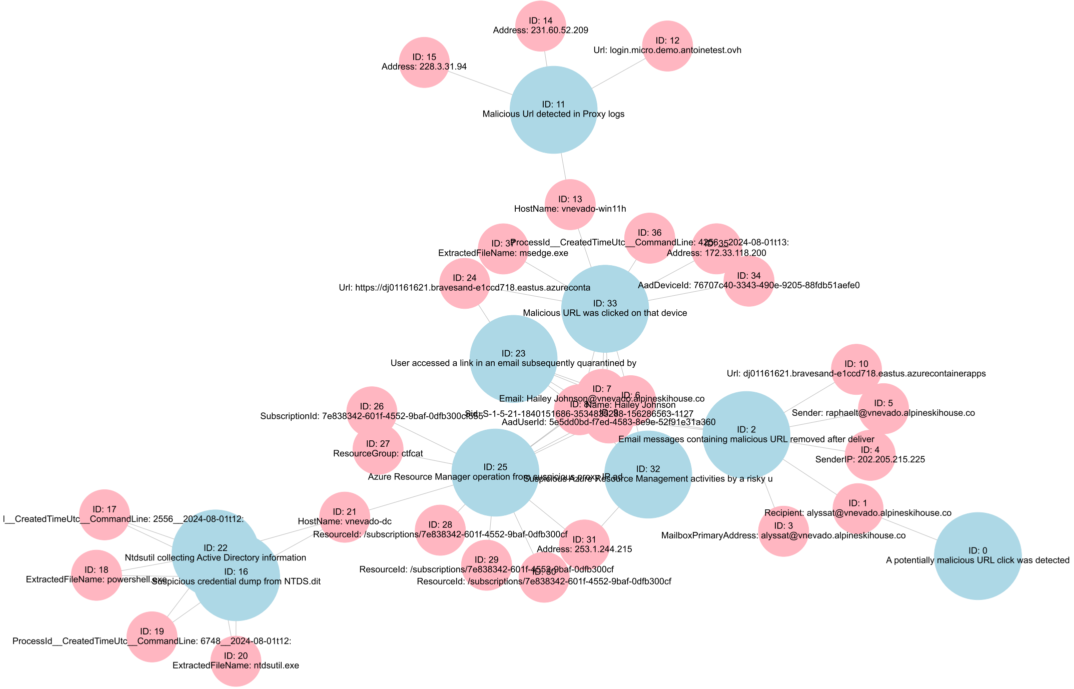
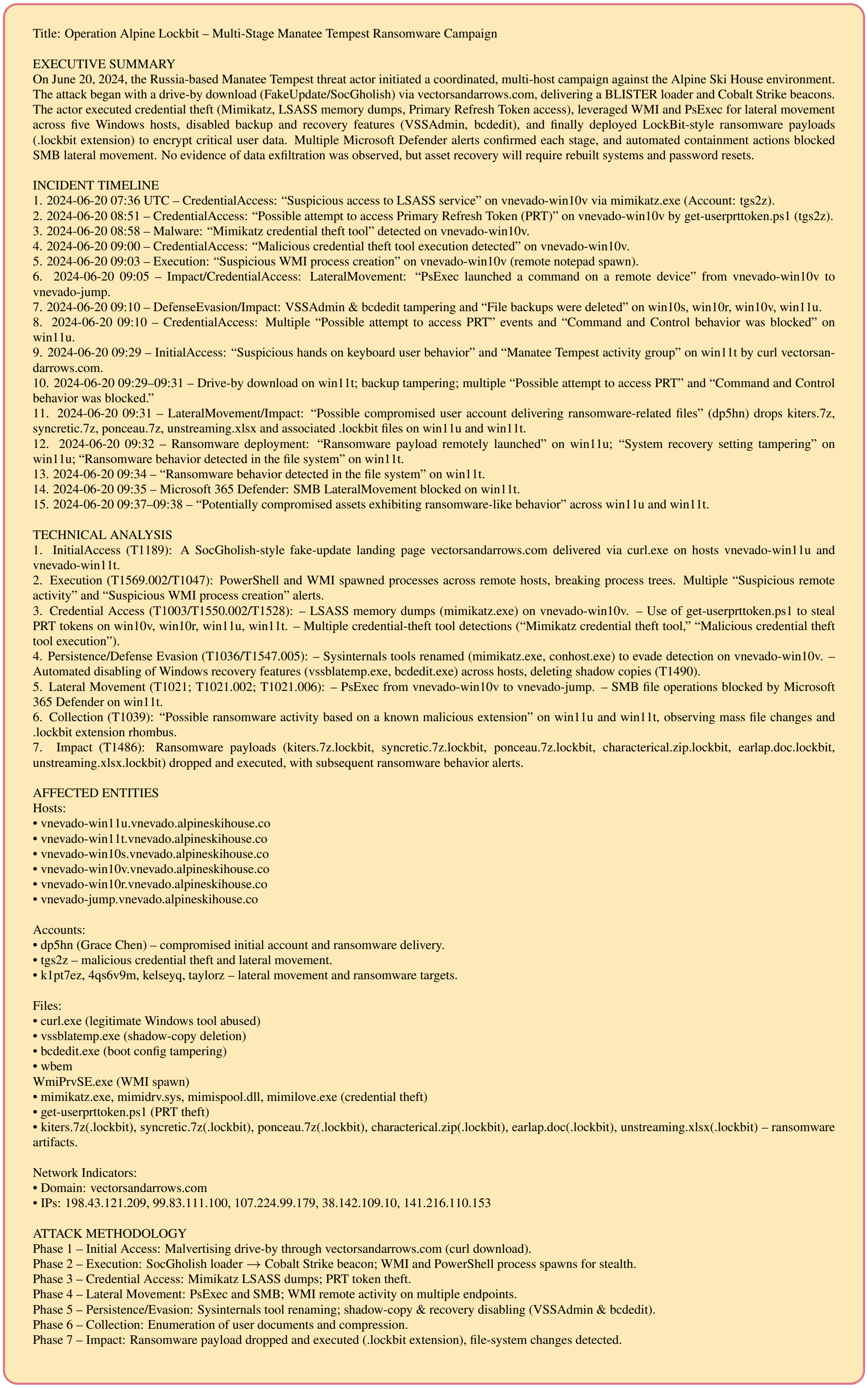

ID: 14
Address: 231.60.52.209
ID: 12
Url: login.micro.demo.antoinetest.ovh
ID: 15
Address: 228.3.31.94
ID: 11
Malicious Url detected in Proxy logs
ID: 13
HostName: vnevado-win11h
ID: 36
ProcessId__CreatedTimeUtc__CommandLine: 4256__2024-08-01t13:
ID: 37
ExtractedFileName: msedge.exe
ID: 35
Address: 172.33.118.200
ID: 24
Url: https://dj01161621.bravesand-e1ccd718.eastus.azureconta
ID: 34
AadDeviceId: 76707c40-3343-490e-9205-88fdb51aefe0
ID: 33
Malicious URL was clicked on that device
ID: 23
User accessed a link in an email subsequently quarantined by
ID: 10
Url: dj01161621.bravesand-e1ccd718.eastus.azurecontainerapps
ID: 7
Email: Hailey Johnson@vnevado.alpineskihouse.co
ID: 6
Name: Hailey Johnson
ID: 8
Sid: S-1-5-21-1840151686-3534835288-156286563-1127
ID: 9
AadUserId: 5e5dd0bd-f7ed-4583-8e9e-52f91e31a360
ID: 5
Sender: raphaelt@vnevado.alpineskihouse.co
ID: 26
SubscriptionId: 7e838342-601f-4552-9baf-0dfb300cf555
ID: 2
Email messages containing malicious URL removed after deliver
ID: 4
SenderIP: 202.205.215.225
ID: 27
ResourceGroup: ctfcat
ID: 25
Azure Resource Manager operation from suspicious proxy IP ad
ID: 32
Suspicious Azure Resource Management activities by a risky u
ID: 1
Recipient: alyssat@vnevado.alpineskihouse.co
ID: 17
l__CreatedTimeUtc__CommandLine: 2556__2024-08-01t12:
ID: 21
HostName: vnevado-dc
ID: 28
ResourceId: /subscriptions/7e838342-601f-4552-9baf-0dfb300cf
ID: 31
Address: 253.1.244.215
ID: 3
MailboxPrimaryAddress: alyssat@vnevado.alpineskihouse.co
ID: 0
A potentially malicious URL click was detected
ID: 22
Ntdsutil collecting Active Directory information
ID: 18
ExtractedFileName: powershell.exe
ID: 16
Suspicious credential dump from NTDS.dit
ID: 29
ResourceId: /subscriptions/7e838342-601f-4552-9baf-0dfb300cf
ID: 30
ResourceId: /subscriptions/7e838342-601f-4552-9baf-0dfb300cf
ID: 19
ProcessId__CreatedTimeUtc__CommandLine: 6748__2024-08-01t12:
ID: 20
ExtractedFileName: ntdsutil.exe

Figure 26: Graph of Incident 322.



Title: Operation Alpine Lockbit – Multi-Stage Manatee Tempest Ransomware Campaign

EXECUTIVE SUMMARY

On June 20, 2024, the Russia-based Manatee Tempest threat actor initiated a coordinated, multi-host campaign against the Alpine Ski House environment. The attack began with a drive-by download (FakeUpdate/SocGholish) via vectorsandarrows.com, delivering a BLISTER loader and Cobalt Strike beacons. The actor executed credential theft (Mimikatz, LSASS memory dumps, Primary Refresh Token access), leveraged WMI and PsExec for lateral movement across five Windows hosts, disabled backup and recovery features (VSSAdmin, bcdedit), and finally deployed LockBit-style ransomware payloads (.lockbit extension) to encrypt critical user data. Multiple Microsoft Defender alerts confirmed each stage, and automated containment actions blocked SMB lateral movement. No evidence of data exfiltration was observed, but asset recovery will require rebuilt systems and password resets.

INCIDENT TIMELINE

1. 2024-06-20 07:36 UTC – CredentialAccess: “Suspicious access to LSASS service” on vnevado-win10v via mimikatz.exe (Account: tgs2z).
2. 2024-06-20 08:51 – CredentialAccess: “Possible attempt to access Primary Refresh Token (PRT)” on vnevado-win10v by get-userprttoken.ps1 (tgs2z).
3. 2024-06-20 08:58 – Malware: “Mimikatz credential theft tool” detected on vnevado-win10v.
4. 2024-06-20 09:00 – CredentialAccess: “Malicious credential theft tool execution detected” on vnevado-win10v.
5. 2024-06-20 09:03 – Execution: “Suspicious WMI process creation” on vnevado-win10v (remote notepad spawn).
6. 2024-06-20 09:05 – Impact/CredentialAccess: LateralMovement: “PsExec launched a command on a remote device” from vnevado-win10v to vnevado-jump.
7. 2024-06-20 09:10 – DefenseEvasion/Impact: VSSAdmin & bcdedit tampering and “File backups were deleted” on win10s, win10r, win10v, win11u.
8. 2024-06-20 09:10 – CredentialAccess: Multiple “Possible attempt to access PRT” events and “Command and Control behavior was blocked” on win11u.
9. 2024-06-20 09:29 – InitialAccess: “Suspicious hands on keyboard user behavior” and “Manatee Tempest activity group” on win11t by curl vectorsandarrows.com.
10. 2024-06-20 09:29–09:31 – Drive-by download on win11t; backup tampering; multiple “Possible attempt to access PRT” and “Command and Control behavior was blocked.”
11. 2024-06-20 09:31 – LateralMovement/Impact: “Possible compromised user account delivering ransomware-related files” (dp5hn) drops kiters.7z, syncretic.7z, ponceau.7z, unstreaming.xlsx and associated .lockbit files on win11u and win11t.
12. 2024-06-20 09:32 – Ransomware deployment: “Ransomware payload remotely launched” on win11u; “System recovery setting tampering” on win11u; “Ransomware behavior detected in the file system” on win11t.
13. 2024-06-20 09:34 – “Ransomware behavior detected in the file system” on win11t.
14. 2024-06-20 09:35 – Microsoft 365 Defender: SMB LateralMovement blocked on win11t.
15. 2024-06-20 09:37–09:38 – “Potentially compromised assets exhibiting ransomware-like behavior” across win11u and win11t.

TECHNICAL ANALYSIS

1. InitialAccess (T1189): A SocGholish-style fake-update landing page vectorsandarrows.com delivered via curl.exe on hosts vnevado-win11u and vnevado-win11t.
2. Execution (T1569.002/T1047): PowerShell and WMI spawned processes across remote hosts, breaking process trees. Multiple “Suspicious remote activity” and “Suspicious WMI process creation” alerts.
3. Credential Access (T1003/T1550.002/T1528): – LSASS memory dumps (mimikatz.exe) on vnevado-win10v. – Use of get-userprttoken.ps1 to steal PRT tokens on win10v, win10r, win11u, win11t. – Multiple credential-theft tool detections (“Mimikatz credential theft tool,” “Malicious credential theft tool execution”).
4. Persistence/Defense Evasion (T1036/T1547.005): – Sysinternals tools renamed (mimikatz.exe, conhost.exe) to evade detection on vnevado-win10v. – Automated disabling of Windows recovery features (vssblatemp.exe, bcdedit.exe) across hosts, deleting shadow copies (T1490).
5. Lateral Movement (T1021; T1021.002; T1021.006): – PsExec from vnevado-win10v to vnevado-jump. – SMB file operations blocked by Microsoft 365 Defender on win11t.
6. Collection (T1039): “Possible ransomware activity based on a known malicious extension” on win11u and win11t, observing mass file changes and .lockbit extension rhombus.
7. Impact (T1486): Ransomware payloads (kiters.7z.lockbit, syncretic.7z.lockbit, ponceau.7z.lockbit, characterical.zip.lockbit, earlap.doc.lockbit, unstreaming.xlsx.lockbit) dropped and executed, with subsequent ransomware behavior alerts.

AFFECTED ENTITIES

Hosts:

- vnevado-win11u.vnevado.alpineskihouse.co
- vnevado-win11t.vnevado.alpineskihouse.co
- vnevado-win10s.vnevado.alpineskihouse.co
- vnevado-win10v.vnevado.alpineskihouse.co
- vnevado-win10r.vnevado.alpineskihouse.co
- vnevado-jump.vnevado.alpineskihouse.co

Accounts:

- dp5hn (Grace Chen) – compromised initial account and ransomware delivery.
- tgs2z – malicious credential theft and lateral movement.
- k1pt7ez, 4qs6v9m, kelseyq, taylorz – lateral movement and ransomware targets.

Files:

- curl.exe (legitimate Windows tool abused)
- vssblatemp.exe (shadow-copy deletion)
- bcdedit.exe (boot config tampering)
- wbem

WmiPrvSE.exe (WMI spawn)

- mimikatz.exe, mimidrv.sys, mimispool.dll, mimilove.exe (credential theft)
- get-userprttoken.ps1 (PRT theft)
- kiters.7z(.lockbit), syncretic.7z(.lockbit), ponceau.7z(.lockbit), characterical.zip(.lockbit), earlap.doc(.lockbit), unstreaming.xlsx(.lockbit) – ransomware artifacts.

Network Indicators:

- Domain: vectorsandarrows.com
- IPs: 198.43.121.209, 99.83.111.100, 107.224.99.179, 38.142.109.10, 141.216.110.153

ATTACK METHODOLOGY

Phase 1 – Initial Access: Malvertising drive-by through vectorsandarrows.com (curl download).
Phase 2 – Execution: SocGholish loader → Cobalt Strike beacon; WMI and PowerShell process spawns for stealth.
Phase 3 – Credential Access: Mimikatz LSASS dumps; PRT token theft.
Phase 4 – Lateral Movement: PsExec and SMB; WMI remote activity on multiple endpoints.
Phase 5 – Persistence/Evasion: Sysinternals tool renaming; shadow-copy & recovery disabling (VSSAdmin & bcdedit).
Phase 6 – Collection: Enumeration of user documents and compression.
Phase 7 – Impact: Ransomware payload dropped and executed (.lockbit extension), file-system changes detected.

Figure 27: Incident 5 Report.

INDICATORS OF COMPROMISE
Malicious Domains / URLs:

- vectorsandarrows.com

Malicious IPs:

- 198.43.121.209
- 99.83.111.100
- 107.224.99.179
- 38.142.109.10
- 141.216.110.153

Malicious Files & Hashes (SHA256):

- BLISTER loader (curl.exe misuse): 2bbad800bc5058cad5631dbffd39fb8a293616479250c47b38dc8e8eb61dc3da
- vssblatemp.exe: 8c1fabcc2196e4d096b7d155837c5f699ad7f55edbf84571e4f8e03500b7a8b0
- mimikatz.exe: 61c0810a23580cf492a6ba4f7654566108331e7a4134c968c2d6a05261b2d8a1
- mimidrv.sys: 4ff7578df7293e50c9bdd48657a6ba0c60e1f6d06a2dd334f605af34fe6f75a5
- mimispool.dll: 05842de51ede327c0f55df963f6de4e32ab88f43a73b9e0e1d827bc70199eff0
- kiters.7z.lockbit, syncretic.7z.lockbit, ponceau.7z.lockbit, characterical.zip.lockbit, earlap.doc.lockbit, unstreaming.xlsx.lockbit

SEVERITY ASSESSMENT
Overall Impact: High

- Multiple confirmed credential thefts and lateral movements.
- Automated disabling of backup and recovery features.
- Deployment and execution of ransomware payloads across key user data.
- Loss of data integrity and potential, though unconfirmed, data encryption.
- Significant operational disruption requiring system rebuilds and password resets.

KEY LABELS & KEYWORDS Manatee Tempest, LockBit, SocGholish, BLISTER loader, Cobalt Strike, Mimikatz, LSASS dump, PRT theft, VSSAdmin, bcdedit, WMI, PsExec, Shadow copy deletion, Ransomware payload, .lockbit extension, Lateral Movement, Credential Access, Impact.

Figure 28: Incident 5 Report (Continued.)

Title

Macro-Enabled Document Dropper with PowerShell Backdoor Deployment and Dual Persistence Mechanisms

EXECUTIVE SUMMARY

On June 26, 2024, the user “samuelf” on Windows 10 host vnevado-win10e opened a weaponized Word document (RS4_WinATP-Intro-Invoice.docm). A malicious macro triggered PowerShell execution in memory (T1059.001), which decoded and dropped a backdoor executable (WinATP-Intro-Backdoor.exe) to the user’s Desktop. The attacker established persistence via:

- A RunOnce registry key entry (T1547.001) pointing to the dropped backdoor.
- A one-time Scheduled Task named “Yrei” (T1053.005) set to run the backdoor at a scheduled time.

Shortly thereafter, the backdoor initiated LDAP reconnaissance against the domain controller (T1018, T1087.x), obtaining directory information. Microsoft Defender ATP generated a sequence of alerts spanning “Execution,” “Persistence,” and “Discovery” tactics. Automated and manual investigations deemed some artifacts benign after remediation, but confirmed the attacker’s multi-stage activity.

INCIDENT TIMELINE

2024-06-26 11:57:19 UTC
- winword.exe (PID 9780) launched via user opening RS4_WinATP-Intro-Invoice.docm.
- PowerShell invoked by WINWORD.EXE with execution policy bypass to run embedded Base64 decoder script.

2024-06-26 12:00:39 UTC
- Suspicious PowerShell command line detected (Alert #3, T1059.001).
- PowerShell dropped WinATP-Intro-Backdoor.exe to Desktop and executed it (Alert #25).

2024-06-26 12:00:40 UTC
- schtasks.exe created a one-off scheduled task “Yrei” to run the backdoor (Alert #20).
- schtasks.exe ran the “Yrei” task immediately, launching the backdoor (Alert #21).

2024-06-26 13:17:18 UTC
- reg.exe added a RunOnce registry value under HKCU
RunOnce
Yrei to launch the backdoor on next logon (Alert #17 & #23).

2024-06-26 11:57:25 UTC
- Backdoor (or payload script) executed an LDAP query against the domain controller to enumerate users and groups (Alert #24).

2024-07-04 22:35 UTC
- Automated and user-initiated investigations in MDATP triaged and resolved alerts.

TECHNICAL ANALYSIS

1. Delivery & Initial Execution (T1204.002 → T1059.001)
- Node 11: WINWORD.EXE opened the malicious .docm.
- Nodes 7 & 8: PowerShell launched with “-Exec Bypass -Command” to assemble Base64 chunks and write WinATP-Intro-Backdoor.exe.
- Alerts:
– #3 “Suspicious process executed PowerShell command”
– #10 “Suspicious PowerShell command line”
– #26 “Suspicious PowerShell download or encoded command execution”

2. Payload Drop & Execution (T1059.001)
- Node 16: WinATP-Intro-Backdoor.exe created on Desktop.
- Alert #25: “PowerShell dropped a suspicious file on the machine.”

3. Persistence Mechanisms
a. Scheduled Task (T1053.005)
- Node 13 (PID 1348): schtasks.exe /create /SC ONCE /TN Yrei /TR “. . . Backdoor.exe”
- Alert #20: “An anomalous scheduled task was created.”
- Alert #21: “Suspicious Task Scheduler activity” when the task ran.

b. Registry RunOnce (T1547.001)
- Node 19 (PID 10208): reg.exe ADD HKCU
RunOnce /v Yrei /d • Alert #17: “Anomaly detected in ASEP registry”
- Alert #23: “An uncommon file was created and added to a Run Key”

4. Discovery & Reconnaissance (T1018, T1069, T1087, T1558.003)
- Node 24: LDAP query via backdoor or script to enumerate directory info.
- Alert #24: “Suspicious LDAP query.”

5. Investigation & Triage
- Node 0: Automated investigation started and marked benign for some artifacts.
- Alerts resolved in MDATP console; manual follow-up recommended (patch, AV, forensic).

Figure 29: Incident 34 Report

```
AFFECTED ENTITIES
——————————
Hosts:
• vnevado-win10e.vnevado.alpineskihouse.co (MdatpDeviceId cbb9f. . . , Windows 22H2)

Users/Accounts:
• samuelf (UPN: samuelf@vnevado.alpineskihouse.co, SID S-1-5-21. . . -1193)

Processes:
• WINWORD.EXE (PID 9780)
• powershell.exe (PIDs 8604 & 200)
• schtasks.exe (PIDs 1348 & 1616)
• reg.exe (PID 10208)
• cmd.exe (PID 2264)

Files:
• RS4_WinATP-Intro-Invoice.docm (delivery document)
• WinATP-Intro-Backdoor.exe (backdoor) – SHA1: 5e1c8874b29de480a0513516fb542cad2b049cc3; SHA256: 929cf5c2a2ce25d82699fc1bfe578bbe8abedce0e477a40980016ee32c2c7cbe
• YreianBackdoor.ps1 (indicated in LDAP-stage parent PS command)

Registry Keys/Values:
• HKCU
RunOnce
Yrei → “
Network Indicators:
• LDAP target domain controller (no external C2 observed)
• External IP seen (72.5.72.208) associated with Run key anomaly

ATTACK METHODOLOGY
——————————-
Tactics & Techniques (MITRE):
Execution (T1059.001) – PowerShell
Persistence (T1053.005) – Scheduled Task
Persistence (T1547.001) – Registry Run Keys
Discovery (T1018, T1069, T1087.x) – LDAP, account enumeration
Privilege Escalation (T1112) – Potential credential theft via registry
Defense Evasion – Encoded commands in PowerShell

INDICATORS OF COMPROMISE
————————————-
File Hashes:
• WinATP-Intro-Backdoor.exe – SHA256 929cf5. . . 7cbe
• PowerShell.exe on device – SHA256 9785001b0dcf755eddb8af294a373c0b87b2498660f724e76c4d53f9c217c7a3

Registry:
• HKCU
RunOnce
Yrei

Scheduled Task:
• TaskName: Yrei – triggers C:
Users
samuelf
Desktop
WinATP-Intro-Backdoor.exe

Network:
• 72.5.72.208 (external IP in registry anomaly)

Process Artifacts:
• Command lines showing Base64 assembly and file write operations

SEVERITY ASSESSMENT
——————————-
Impact: High – execution of a persistent backdoor, credential and directory enumeration.
Confidentiality & Integrity: Attacker maintained foothold and could exfiltrate or modify data.
Availability: No direct impact observed, but persistence allows future disruptive actions.

Important Labels & Keywords
——————————————
T1059.001, T1053.005, T1547.001, T1018, T1087.002, T1087.003, T1558.003, WinATP-Intro-Backdoor.exe, RS4_WinATP-Intro-Invoice.docm, Yrei, RunOnce, schtasks, LDAP reconnaissance, Base64 PowerShell injector.
```

Figure 30: Incident 34 Report (Continued.)

Title

Multi-Stage Fileless Attack: PowerShell Execution, Process Injection, and Covert C2 over Azure Blob and External IP

EXECUTIVE SUMMARY

On June 26, 2024, the AlpineSkiHouse host vnevado-win11a was compromised by a low-and-slow, fileless attack. The attacker used PowerShell to fetch and execute code from an Azure blob URL, performed reconnaissance (Application Window Discovery), injected malicious code into Notepad.exe to evade defenses, and established covert command-and-control (C2) communications with an external IP. No new files were written to disk; all steps leveraged living-off-the-land binaries and in-memory execution.

INCIDENT TIMELINE

2024-06-26 15:49:15 UTC

- WindowsTerminal (wt.exe) spawns PowerShell (PID 7472) with an encoded command referencing an Azure blob URL (WinATP-Intro-Fileless.txt).

2024-06-26 15:49:16 UTC

- Alert ”Suspicious Application Window Discovery” (Low) triggered during reconnaissance (T1010) on host vnevado-win11a.

2024-06-26 15:49:41 UTC

- Second PowerShell instance (PID 4656) launched with execution-policy bypass and the same encoded C2 retrieval command.

2024-06-26 15:49:42 UTC

- Alert ”Suspicious process injection observed” (Medium) marks the moment PowerShell (PID 4656) injects code into a target process (T1055.001).

2024-06-26 15:49:51 UTC

- Alert ”Suspicious process injection observed” (Medium) again flags injection of code into Notepad.exe (PID 8932). Notepad launches with no arguments, exhibiting anomalous behavior.

2024-06-26 15:49:52 UTC

- Alert ”Unexpected behavior by a process ran with no command line arguments” (Medium) records Notepad connecting out to IP 202.183.149.174 (T1218.011).

TECHNICAL ANALYSIS

1. Reconnaissance (T1010,T1518)

- PowerShell enumeration commands gathered system and application window data.
- Microsoft Defender ATP generated a low-severity ”Suspicious Application Window Discovery” alert at 15:49:16.

2. Payload Retrieval & Execution (T1059.001)

- Encoded PowerShell fetched content from https://wcdstaticfilesprdeus.blob.core.windows.net/.../WinATP-Intro-Fileless.txt.
- No files dropped; execution happened in memory under ”bypass” policy.

3. Process Injection (T1055, sub-techniques .001–.005)

- The in-memory payload injected into Notepad.exe (PID 8932), hiding malicious code inside a trusted process.
- Two separate Defender ATP alerts (”Suspicious process injection observed”, Medium) fired covering injection start and end times.

4. Masquerading & Unexpected Behavior (T1036,T1218.011)

- Notepad.exe, normally benign, exhibited network behavior without CLI args.
- It reached out to external IP 202.183.149.174, triggering the ”Unexpected behavior” alert.

AFFECTED ENTITIES

Hosts

- vnevado-win11a.vnevado.alpineskihouse.co (DeviceId: 39d8c57f-2d08-4888-b0cd-12cd0679337a, RiskScore: Medium)

Accounts

- nathans (Luis Martinez; Sid: S-1-5-21-1240151660-3534038288-105586567-4122; UPN: nathans@vnevado.alpineskihouse.co)

Processes

- WindowsTerminal.exe (PID 9484 → 7472)
- PowerShell.exe (PID 7472, 4656)
- Notepad.exe (PID 8932)

External IPs & URLs

- 72.5.72.208 (public IP seen on host’s last external IP)
- 202.183.149.174 (C2 callback)
- https://wcdstaticfilesprdeus.blob.core.windows.net/.../WinATP-Intro-Fileless.txt

ATTACK METHODOLOGY

- Initial reconnaissance via built-in tools (MITRE T1010,T1518)
- Fileless payload retrieval and execution through PowerShell (T1059.001)
- In-memory code injection into Notepad for defense evasion and persistence (T1055.*)
- Masquerading / unexpected behavior detection in Notepad (T1036)
- Network-based C2 communication over HTTPS and direct IP (T1218.011)

INDICATORS OF COMPROMISE (IOCs)

File Hashes (PowerShell.exe – legitimate but abused)

- SHA256: 3247bcfd60f6dd25f34cb74b5889ab10ef1b3ec72b4d4b3d95b5b25b534560b8

Notepad.exe Hash (legitimate)

- SHA256: 93379a2f43ac1189f29533c4de3f594bb04db76461f1d704620b9cc7cc718eeb

URLs & Domains

- wcdstaticfilesprdeus.blob.core.windows.net – WinATP-Intro-Fileless.txt

External IPs

- 202.183.149.174 (C2)
- 72.5.72.208 (host external IP)

SEVERITY ASSESSMENT

Overall Impact: High

- Multi-stage, in-memory compromise without disk artifacts
- Successful defense evasion via living-off-the-land binaries and process injection
- Unauthenticated remote C2 established
- Potential pivot and credential theft risk

LABELS / KEYWORDS

#Fileless #Discovery #Execution #DefenseEvasion #ProcessInjection #PowerShell #Notepad #CobaltStrike #AzureBlob #C2 #T1059 #T1055 #T1010 #T1036 #T1218

Figure 31: Incident 38 Report

Title
Operation Alpine Storm

EXECUTIVE SUMMARY
Between June 27, 2024 14:30 UTC and June 27, 2024 14:35 UTC, a human-operated intrusion chain hit AlpineSkiHouse's environment. Starting with a malicious URL click, the adversary executed a PowerShell "DomainDominance27" script to drop dual-use tools (Mimikatz, Rubeus, PsExec), disabled real-time antivirus, harvested credentials via Mimikatz and Rubeus (kerberoasting), created a backup domain account (BDAdmin), injected SID history into that account for elevated privileges, performed DCSync against the domain controller, and used overpass-the-hash and pass-the-ticket techniques to move laterally. Core systems compromised: vnevado-Win10V (user tgs2z/Ethan Brooks), vnevado-Win10B (user fk8mq/Emma Clark), and vnevado-DC.

INCIDENT TIMELINE
2024-06-27 14:31:27 UTC
• "Orchestrator.ps1" kicks off on vnevado-Win10V as user tgs2z.
2024-06-27 14:32:08 UTC
• DomainDominance27.ps1 executed by tgs2z → drops PsExec, Mimikatz, Rubeus.
2024-06-27 14:32:12 → 14:32:25 UTC
• Recon (whoami, net user/group/domain queries) by tgs2z (Discovery T1087).
2024-06-27 14:32:21 UTC
• Antivirus alert: Kekeo malware detected (informational).
2024-06-27 14:32:35 UTC
• Mimikatz "sekurlsa::logonpasswords" (Credential Access T1003).
2024-06-27 14:32:37 UTC
• Mimikatz Pass-the-Hash ("sekurlsa::pth") targeting accounts kyleg & fk8mq (T1550.003).
2024-06-27 14:32:46 → 14:32:52 UTC
• Mimikatz "sekurlsa::pth /run:Get-KRBTicket.ps1" and Rubeus ticket theft (kerberoasting T1558.003).
• Disable Windows Defender real-time monitoring (Defense Evasion T1562.001).
2024-06-27 14:33:14 → 14:33:27 UTC
• PsExec from vnevado-Win10V to vnevado-Win10B to run Rubeus and dump service tickets (Lateral Movement T1021.002).
2024-06-27 14:33:32 → 14:33:38 UTC
• Mimikatz "kerberos::ptt" and DCSync via "lsadump::dcsync" (Credential Access T1003.006).
2024-06-27 14:33:43 UTC
• New domain user BDAdmin created on vnevado-Win10V (Persistence T1136.002).
2024-06-27 14:34:10 UTC
• PowerShell adds SIDHistory for BDAdmin in NTDS and restarts service (Privilege Escalation T1134.005).
• Suspicious service launched on vnevado-DC (Execution T1569.002).
2024-06-27 14:34:38 → 14:34:44 UTC
• Additional SID history injection and suspicious service injection on DC.
2024-06-27 14:35:00 UTC
• Root cause analysis and remediation completed (alerts resolved).
2024-06-27 17:04:00 UTC
• tgs2z clicks malicious URL in quarantined email (Initial Access T1566.002).

TECHNICAL ANALYSIS
1. Initial Access (Phishing T1566.002):
– vnevado-Win10V user tgs2z clicks URL "ac27145722.livelygrass-8d4c4013..." delivered via email.
2. Execution & Tool Deployment:
– DomainDominance27.ps1 drops PsExec, Mimikatz, Rubeus. Tools hashed as:
• PsExec.exe SHA256: 57492d33b7c0755bb411b22d2dfdfdf088cbbfcd010e30dd8d425d5fe66adff4
• Mimikatz.exe SHA256: 912018ab3c6b16b39ee84f17745ff0c80a33cee241013ec35d0281e40c0658d9
• Rubeus.exe SHA256: a1fddd460edd35ed449d32cc43bc15675c48a314a6fa5fb158e3bc4fea460be1
3. Defense Evasion & Persistence:
– Disabled Defender real-time monitoring via Set-MpPreference.
– Created BDAdmin account and injected SIDHistory (S-1-5-32-544) to escalate privileges.
4. Credential Access:
– Mimikatz "sekurlsa::logonpasswords" dumps plaintext creds from LSASS.
– Kerberoasting via Rubeus "dump /service:xfbzkp /user:lucasb" to steal service tickets.
– Overpass-the-hash: forging TGT from NTLM hashes.
5. Lateral Movement (T1021.002):
– PsExec remote executions to vnevado-Win10B and vnevado-DC to run Rubeus/Mimikatz.
6. Domain Persistence & Control:
– SID history injection and NTDS restarts on vnevado-DC.
– DCSync replication from DC to exfiltrate all account hashes (T1003.006).
7. Pass-the-Ticket (T1550.003):
– Pass TGTs to authenticate as Julian Burke on alternate hosts.
AFFECTED ENTITIES
Hosts:
• vnevado-Win10V (MachineID 7cc55a46...) – initial foothold, tgs2z "DomainDominance".
• vnevado-Win10B (MachineID 5c626a5b...) – lateral target for Rubeus.
• vnevado-DC (MachineID 43a4c3f27...) – domain controller impacted by DCSync & SID injection.
Accounts:
• tgs2z / Ethan Brooks (S-5-7-21-...1422) – initial operator account.
• fk8mq / Emma Clark (S-1-5-21-...1517) – service account & ticket target.
• lucasb / Julian Burke (S-1-5-21-...1120) – IT director, ticket impersonation.
• BDAdmin – attacker-created backup domain admin.
Network & URLs:
• IP 118.254.65.186 – phishing link source.
• IP 141.216.110.153 – RDP lateral drop.
• IP 35.202.74.47 – Win10B management.
• URL https://ac27145722.livelygrass-8d4c4013.eastus2.azurecontainerapps.io/

Figure 32: Incident 39 Report

ATTACK METHODOLOGY (MITRE ATT&CK)
- Initial Access: T1566.002 Phishing Link
- Execution: T1059.001 cmd, T1059.003 PowerShell
- Defense Evasion: T1562 (Disable AV), T1134.005 SID History Injection
- Persistence: T1136.002 New Account
- Privilege Escalation: T1134.005, T1550.002 Pass-the-Hash
- Credential Access: T1003.* (LSASS dump, DCSync), T1558.003 Kerberoasting
- Lateral Movement: T1021.002 PsExec, T1105 Remote File Copy
- Collection: T1550.003 Pass-the-Ticket
- Discovery: T1087, T1049 SMB sessions
- Impact: T1489 Service Stop (NTDS), T1543.003 Service Registry

INDICATORS OF COMPROMISE
File hashes:
- Mimikatz.exe SHA256: 912018ab3c6b16b39ee84f17745ff0c80a33cee241013ec35d0281e40c0658d9
- Rubeus.exe SHA256: a1fddd460edd35ed449d32cc43bc15675c48a314a6fa5fb158e3bc4fea460be1
- PsExec.exe SHA256: 57492d33b7c0755bb411b22d2dfdfdf088cbbfcd010e30dd8d425d5fe66adff4
- DomainDominance27.ps1 SHA256: b284932e65dd50b731f2c6dc266ab4fe46287581498ac4dc50f13922b58d8c72

Malicious URL:
- https://ac27145722.livelygrass-8d4c4013.eastus2.azurecontainerapps.io/

Phishing IP:
- 118.254.65.186

SEVERITY ASSESSMENT
Overall Impact: Critical
- Complete domain compromise via DCSync and account migration
- Backdoor domain admin created (BDAdmin)
- Credentials for all high-value accounts dumped
- Persistent access and full lateral control across environment

IMPORTANT LABELS AND KEYWORDS
AlpineSkiHouse, human-operated, DomainDominance27, Kerberoasting, DCSync, SIDHistory, overpass-the-hash, pass-the-ticket, Mimikatz, Rubeus, PsExec, Phishing, Zero-hour Auto Purge (ZAP), Domain Controller.

Figure 33: Incident 39 Report (Continued.)

Title of the Multi-Stage Attack
“Phishing-Enabled ADFS Key Exfiltration and Lateral Movement Campaign”

EXECUTIVE SUMMARY
Between July 1 and July 10, 2024, an adversary executed a phishing-enabled, hands-on-keyboard campaign against AlpineSkiHouse’s Windows estate. Initial access was gained via a malicious URL in a spear-phishing email to internal mailboxes. Once a user clicked the URL, the attacker ran PowerShell scripts to harvest credentials (LSASS memory read), performed reconnaissance and hardware enumerations, then leveraged Impacket to move laterally from the initial endpoint (MB-WINCLIENT) to the ADFS server (MB-ADFS). On MB-ADFS they established persistence (scheduled tasks, service creation, renamed executables), injected code into trusted processes, stole ADFS private keys via LDAP queries, and executed a DCSync attack against the domain controller (MB-DC1) to replicate directory services. They also abused a compromised service account to add credentials to an OAuth application (SimulandApp) and launched an internal phishing campaign. The impact includes credential theft, unauthorized replication of AD data, exposure of ADFS signing keys, and persistent footholds.

INCIDENT TIMELINE
2024-07-01 21:49 UTC
• “Email messages containing malicious URL removed after delivery” (O365 ATP – InitialAccess). Spear-phishing emails with URL http://kn017721628.wittytree-b6f239d6.northeurope.azurecontainerapps.io delivered to user “Nina Sullivan” (santiago@vnevado. . .).

2024-07-02 09:45–09:48 UTC
• User “bjenkins” on MB-WINCLIENT clicked the malicious URL (“Potentially malicious URL click detected”).
• PowerShell launched in user context to download/run Midnight14 payload.
• “Suspicious system hardware discovery” and “Malicious PowerShell Cmdlet invoked” alerts triggered.
• LSASS memory accessed and dumped (“Suspicious access to LSASS service” & “Sensitive credential memory read”).
• “ContosoADFSblatempcreds.ps1” executed under pwilson’s context to harvest ADFS creds (Process injection alert).
• Adversary performed reconnaissance (“Suspicious sequence of exploration activities”).

2024-07-02 09:47–09:48 UTC
• Attacker executed Impacket toolkit on MB-WINCLIENT (“Ongoing hands-on-keyboard attack via Impacket”).
• Used WMI/SMB to reach MB-ADFS (“Suspicious remote activity”).
• Created scheduled task “Run-ExportADFSTokenSigninCert. . . ” via schtasks.exe on MB-ADFS (“Suspicious Task Scheduler activity”).
• Renamed system executable (iy2orr1e.rrg.exe – renamed PowerShell) and launched it to evade detection (“System executable renamed and launched”).
• Injected code into services.exe and svchost.exe (“Process was injected . . . malicious code”).
• Registered a malicious Windows service DDLOXJDSQSNGMUKKFUXQ (“Suspicious service registration” & Azure ATP “Suspicious service creation”).
• Performed LDAP queries against ADFS objects to extract private keys (“ADFS private key extraction attempt” & Azure ATP “Suspected AD FS DKM key read”).

2024-07-02 09:48 UTC
• DCSync replication request issued from MB-WINCLIENT to MB-DC1 (“Suspected DCSync attack”).
• Extracted NTDS.dit data and domain credentials.

2024-07-02 12:07–12:15 UTC
• pwilson added credentials of type Password to SimulandApp in Azure AD (MCAS “Unusual addition of credentials to an OAuth app”).

2024-07-02 15:01 UTC
• Compromised accounts sent internal phishing messages to other employees (“Internal phishing campaign”).

2024-07-03 and July 6–9
• Additional MCAS detections of dual-purpose tool executions under unexpected filenames and cleaning up proof-of-concept artifacts on MB-ADFS.

TECHNICAL ANALYSIS
1. Initial Access (T1566.002):
– Malicious URL delivered via O365 ATP, removed after delivery but clicked by bjenkins.
2. Execution (T1059): Powershell processes launched—Midnight14 payload in Downloads, ContosoADFSblatempcreds.ps1 to extract ADFS creds.
3. Discovery (T1082, T1016, T1087): Hardware enumeration and “whoami.exe” and network reconnaissance commands executed.
4. Credential Access (T1003, T1550): LSASS memory access and dump; DCSync requests to a domain controller.
5. Lateral Movement (T1021.002, T1105): Impacket toolkit used to pass WMI and SMB commands to MB-ADFS.
6. Persistence (T1053.005, T1543.003, T1098.001): Scheduled task creation, malicious service registration, OAuth app secret addition.
7. Defense Evasion (T1036.003): System executable renamed to “iy2orr1e.rrg.exe” to hide from default path checks.
8. Privilege Escalation (T1055, T1569.002): Code injection into trusted processes, service creation for elevated execution.
9. Credential Access – ADFS (T1087.002, T1528): LDAP queries to DKM and private key objects in ADFS, exfiltrating key material.
10. AD Replication (T1003.006): DCSync replication of directory services.
11. Phishing & Internal Recon (T1534): Compromised accounts sending phishing inside network.

Figure 34: Incident 55 Report

AFFECTED ENTITIES

Hosts

- MB-WINCLIENT (initial compromise, credential harvesting, lateral pivot)
- MB-ADFS (persistence, code injection, ADFS key extraction)
- MB-DC1 (directory replication target)

Accounts

- bjenkins (clicked link, ran PowerShell, LSASS access)
- pwilson (executed ContosoADFSblatempcreds, Impacket lateral, DCSync source)
- gsmith (lateral pivot user on MB-ADFS, service creation, key read)
- santiago@vnevado (initial phishing target; internal sender)
- Nina Sullivan, Lucas Grey (mailboxes used/compromised)

Files & Processes

- powershell.exe (encoded commands, script execution)
- ContosoADFSblatempcreds.ps1 (ADFS credential extractor)
- iy2orr1e.rrg.exe (renamed PowerShell payload)
- schtasks.exe, services.exe, svchost.exe (persistence)

Network Indicators

- 72.5.72.208 (phishing URL host)
- 106.214.87.198 / 119.36.50.193 (sender IPs)
- phishing URLs:

– http://kn017721628.wittytree-b6f239d6.northeurope.azurecontainerapps.io/

– http://ym018491661.wittytree-b6f239d6.northeurope.azurecontainerapps.io/

OAuth & Cloud

- SimulandApp (OAuth App ID 4d9476b8-6ae6-459b-a273-5837c15b5981)

ATTACK METHODOLOGY (MITRE ATT&CK)

Initial Access T1566.002

Execution T1059.001, T1059.003

Persistence T1053.005, T1543.003, T1098.001

Privilege Escalation T1055, T1569.002

Defense Evasion T1036.003

Credential Access T1003.001, T1003.006, T1550.002, T1087.002, T1528

Discovery T1007, T1016, T1018, T1033, T1049, T1069, T1087

Lateral Movement T1021.002, T1105

Collection & Exfil implicit via DCSync

Command & Control observed in impacket remote session

Impact unauthorized data access & ADFS key theft

INDICATORS OF COMPROMISE

Files/Hashes

- ContosoADFSblatempcreds.ps1 SHA256: ad6997e67a2625a8663cb9f84d2461048b0a973b5015ae4f4cba717745cab602
- iy2orr1e.rrg.exe SHA256: 75d6634a676fb0bea5bfd8d424e2bd4f685f3885853637ea143b2671a3bb76e9
- DKM key object reads of GUID 4cac49d3-29d3-407e-8c9b-b7ebb21541b2

Processes

- svchost.exe -k netsvcs -p -s Winmgmt (PID 2880)
- powershell.exe -EncodedCommand QwA6. . .

Network

- 72.5.72.208
- 106.214.87.198, 119.36.50.193
- phishing URLs above

Scheduled task name

- Run-ExportADFSTokenSigninCert.2024-07-02_09_48_23

SEVERITY ASSESSMENT

Overall Impact: High

- Multiple accounts compromised.
- Domain controller replication via DCSync.
- Exfiltration of ADFS private keys and DKM keys enables forging SAML/ADFS tokens.
- Persistent footholds on AD FS server and cloud application (OAuth secret).
- Internal phishing indicates attacker control of user identities.

LABELS & KEYWORDS

Phishing, PowerShell, Impacket, Lateral Movement, DCSync, ADFS Key Theft, ADFS Distributed Key Manager, LSASS Dump, Scheduled Task, Service Creation, Code Injection, OAuth App Secret, Internal Phishing, Cloud App Security.

Figure 35: Incident 55 Report (Continued.)

Title: Multi-Stage Phishing-Driven BEC and Account Takeover Attack

1. EXECUTIVE SUMMARY
On July 17–18, 2024, the user “raphaelt@vnevado.alpineskihouse.co” (Nina Park) fell victim to a phishing campaign that delivered a malicious URL. Following the click, an attacker leveraged that access to perform unauthorized sign-ins from anonymizing and malicious IP addresses, manipulated the user’s Outlook inbox rules to hide incoming mail, and executed a business email compromise (BEC) fraud attempt by sending spoofed emails to external recipients. Over the next 24 hours, the attacker escalated with a password-spray attack against Azure AD, resulting in a confirmed account takeover. This multi-stage chain encompassed Initial Access (T1566.002), Credential Access (T1110.003/T1110.001), Defense Evasion (T1564.008), Collection (T1114.002), and Reconnaissance/Suspicious Activity (T1586).

2. INCIDENT TIMELINE
2024-07-17 10:49:35 UTC
• Alert (13): “Email messages containing malicious URL removed after delivery”
– A phishing email from alyssat@vnevado.alpineskihouse.co (IP 254.241.243.229) with subject “Lee Don’t miss 1969-Con Event next month” delivered to raphaelt@… and quarantined post-delivery.

2024-07-17 10:50:25 UTC
• Alert (0): “A potentially malicious URL click was detected”
– User clicked http://ms175052280.orangecliff-f53f26fd.eastus.azurecontainerapps.io/ (T1566.002).

2024-07-17 10:56:50 UTC
• Alert (11): “Anonymous IP address”
– Sign-in to raphaelt@… from Tor/VPN IP 170.54.121.63 (Amsterdam) (Initial Access).
• Alert (23): “Malicious IP address”
– Corroborates sign-in from 170.54.121.63 flagged as malicious.
• Alert (24): “Password Spray”
– Credential spray detected from same IP targeting multiple accounts.

2024-07-17 11:04:19 UTC
• Phish URL cleanup processed (alert 13 final state).

2024-07-17 11:06:04 UTC
• Alert (14): “Suspicious inbox manipulation rule”
– Attacker created hidden/move/delete rule in Nina Park’s mailbox (Defense Evasion T1564.008).
• Alert (17): “BEC financial fraud”
– Attacker created hide-incoming-mail rule in Azure AD session (Collection T1114.002).

2024-07-17 11:06:54 UTC
• Alert (18): “Suspicious emails sent by BEC-related user”
– Spoofed email “Re: Lee Don’t miss 777-Con Event next month” sent from raphaelt@… to Nathan@avoriaz.alpineskihouse.co (IP 237.7.81.122) (T1586).

2024-07-18 13:48:02 UTC
• Duplicate “Malicious IP address” alert reconfirms prior sign-in risk.

2024-07-18 14:36:59 UTC
• Alert (24) processing completes for Password Spray detection.

2024-07-18 14:54:18 UTC
• Alert (25): “Account compromised following a password-spray attack”
– Confirmed unauthorized sign-in from unusual location/browser; Nina Park’s account fully compromised.

3. TECHNICAL ANALYSIS
Alert 13 & 0 (T1566.002)
– A phishing email from alyssat@… (IP 254.241.243.229) targeted raphaelt@… with a malicious link. Office 365 ATP quarantined subsequent deliveries, but the user clicked before removal, establishing initial foothold.

Alert 11, 23 & 24 (Initial Access & Credential Access)
– Shortly after the click, a cloud-logon from IP 170.54.121.63 (Amsterdam; anonymizer/Tor) succeeded, indicating the attacker either harvested credentials or session tokens. Azure AD Identity Protection flagged the IP as both anonymous and malicious and detected a password spray against multiple accounts.

Alerts 14 & 17 (Defense Evasion & Collection)
– Within 15 minutes of initial access, the attacker modified Nina Park’s Outlook inbox rules to hide or delete incoming messages, preventing detection and facilitating covert BEC operations.

Alert 18 (Suspicious Activity)
– Using the compromised mailbox, the attacker dispatched fraudulent invoices or event invitations to Nathan@avoriaz…, likely aiming to extort or redirect payments (BEC/T1586).

Alert 25 (Account Compromise Confirmation)
– A day later, a full account takeover is confirmed post-password-spray; sign-in patterns and browser attributes were anomalous.

4. AFFECTED ENTITIES
– User Account: Nina Park (raphaelt@vnevado.alpineskihouse.co; AadUserId e036dee7-…)
– Mailbox: raphaelt@vnevado.alpineskihouse.co
– Phishing sender: alyssat@vnevado.alpineskihouse.co; IP 254.241.243.229
– Malicious URLs:
• http://ms175052280.orangecliff-f53f26fd.eastus.azurecontainerapps.io/
• https://ms175052280.orangecliff-f53f26fd.eastus.azurecontainerapps.io/
– Attacker IPs: 170.54.121.63 (anon/malicious), 237.7.81.122 (SMTP relay for BEC)
– External Target: Nathan@avoriaz.alpineskihouse.co
– Cloud App: Microsoft Exchange Online (AppId 20893)

Figure 36: Incident 134 Report

5. ATTACK METHODOLOGY (TTP)
– T1566.002 Phishing: Malicious link delivered via email.
– T1110.003/T1110.001 Credential Access: Password spray.
– T1564.008 Defense Evasion: Inbox rule manipulation.
– T1114.002 Collection: Hidden rule created to exfiltrate incoming email.
– T1586 Suspicious Activity: Outbound BEC emails.
– Persistence & Lateral Move: Retained mailbox control, risk session tokens.

6. INDICATORS OF COMPROMISE (IOCs)
– Phishing URL: ms175052280.orangecliff-f53f26fd.eastus.azurecontainerapps.io
– Sender IP: 254.241.243.229
– Attacker IPs: 170.54.121.63, 237.7.81.122
– Compromised account UPN: raphaelt@vnevado.alpineskihouse.co
– External recipient: nathan@avoriaz.alpineskihouse.co
– NetworkMessageIds: 2dbfc9f0-951f-4dd2-692b-08dca64b9909, 47e56987-44a0-45c4-d1b1-08dca6508dd1

7. SEVERITY ASSESSMENT
Overall Impact: High
– Initial phishing led to credential theft and unauthorized mailbox control.
– BEC attempts risk financial loss and reputational damage.
– Password spray expanded compromise to other accounts.
– Defense Evasion tactics concealed malicious activity.

8. KEY LABELS & KEYWORDS
Phishing, BEC, Business Email Compromise, Password Spray, Account Takeover, Inbox Rule Manipulation, Malicious URL, Anonymous IP, Malicious IP, Azure AD Identity Protection, Office 365 ATP.

Figure 37: Incident 134 Report (Continued.)

Title of the multi-stage attack
Business Email Compromise & Data Exfiltration via Inbox Rule Manipulation and SAP Access

1. EXECUTIVE SUMMARY
Over a 36-hour period beginning July 22, 2024, an attacker leveraged anonymous IP logons and a password-spray campaign to gain initial access to the corporate Azure AD account of “Jordan P” (laylaw@vnevado.alpineskihouse.co). Once inside, the actor deployed malicious inbox rules (T1564.008) to hide and move mail for exfiltration, conducted a business-email-compromise (BEC) campaign against an internal recipient (tony@avoriaz.alpineskihouse.co), and then authenticated to SAP (T1078) to harvest sensitive financial data. The incident spans techniques in Initial Access, Credential Access, Defense Evasion, Collection and Exfiltration, and culminates in potential financial fraud.

Figure 38: Incident 166 Report.

2. INCIDENT TIMELINE
2024-07-22 08:41:20 UTC
• Azure AD Identity Protection Alert (AnonymousLogin, Medium)
• Client IP 95.202.65.202 (Frankfurt, DE, Tor/anonymizer) signs in as AadUserId=89e933b9…ef82

2024-07-22 08:41:20 UTC (same event processed later)
• IPC Password Spray Alert (T1110.003, T1110.001, High) from IP 95.202.65.202

2024-07-22 09:07:43 UTC
• Second Anonymous IP sign-in (192.238.237.190, Hamburg, DE) for same user

2024-07-22 09:18–09:49 UTC
• MCAS "Suspicious inbox manipulation rule" (High)
– New MoveToFolder rule "ITCleanup" on laylaw@…
– IPs involved: 180.144.153.174, 95.202.65.202, 192.238.237.190, and attacker IP 255.246.85.58
• MTP / Defender365 "Suspicious inbox manipulation rule" (High, T1564.008)
• Azure AD "BEC financial fraud" (High)

2024-07-22 09:38:16 UTC
• MTP "Suspicious emails sent by BEC-related user" (High, T1586)
– laylaw@… → tony@avoriaz.alpineskihouse.co via IP 255.246.85.58

2024-07-22 09:46:21 UTC
• MTP "Suspicious SAP authentication" (Medium, T1078)
– laylaw@… signs in to SAP app "Lia" (AppId=100) from IP 107.253.5.27
• Repeat SAP alerts processed at 12:59, 13:19, 14:09 UTC

2024-07-23 14:23 UTC (retroactive processing)
• IPC Password Spray attack detection re-flagged against same request

2024-07-23 16:05–16:09 UTC
• MTP "Possible BEC financial fraud" (Medium, T1114.003)
– laylaw@… flagged as Compromised

3. TECHNICAL ANALYSIS
• Initial Access: Anonymous IP alerts (Nodes 0 & 3) and Password Spray (Node 28) indicate brute-force attempts from Tor-exit and anonymizing VPNs (95.202.65.202, 192.238.237.190).
• Defense Evasion & Persistence: MCAS and Defender365 detect creation of a stealth inbox rule "ITCleanup" (Nodes 5 & 16) to auto-move or hide legitimate emails in Jordan P's mailbox, with attacker IP 255.246.85.58 marked "Attacker."
• Collection & Exfiltration: Suspicious outbound email (Node 18) to tony@avoriaz.alpineskihouse.co carries potential fraudulent instructions or invoice requests (T1586). A mail-message entity (Nodes 20–22) tied to IP 255.246.85.58 confirms exfiltration channel.
• Lateral Movement & Data Harvesting: Multiple "Suspicious SAP authentication" alerts (Nodes 23, 26, 27) show same compromised account signing into enterprise SAP application "Lia" using stolen credentials, searching financial records (T1078).

4. AFFECTED ENTITIES
Accounts
• laylaw@vnevado.alpineskihouse.co (Jordan P, SID S-1-5-21-…-1602)
• tony@avoriaz.alpineskihouse.co (mailbox recipient)

IP Addresses
• 95.202.65.202 (Frankfurt, Tor exit)
• 192.238.237.190 (Hamburg)
• 180.144.153.174 (Contextual)
• 255.246.85.58 (Attacker source)
• 107.253.5.27 (SAP access)

Cloud Applications
• Microsoft Exchange Online (AppId 20893)
• Office 365 / Microsoft 365 (AppId 11161)
• SAP "Lia" (AppId 100)

Mailboxes & Messages
• Inbox rule "ITCleanup"
• mailMessage to tony@avoriaz.alpineskihouse.co

5. ATTACK METHODOLOGY (MITRE ATT&CK)
• Initial Access: T1110.003 Password Spray; AnonymousLogin (AZURE_AD_IDP)
• Defense Evasion: T1564.008 Inbox Rule Hiding
• Credential Access: Password Spray; Risky Sign-in Alerts
• Collection: T1114.002 Email Collection; T1586 Phishing
• Exfiltration: Native Mail Forwarding
• Lateral Movement: T1078 Valid Accounts (SAP sign-in)

6. INDICATORS OF COMPROMISE (IOCs)
• IPs: 95.202.65.202 / 192.238.237.190 / 255.246.85.58 / 107.253.5.27
• Compromised Account: AadUserId 89e933b9-5b2e-4bd6-bcdf-033db707ef82
• Inbox Rule Name: ITCleanup
• Malicious mail recipient: tony@avoriaz.alpineskihouse.co
• Cloud AppId: 100 (SAP Lia)

7. SEVERITY ASSESSMENT
Overall impact is High. The attacker achieved account takeover of a privileged user mailbox, established persistent mailbox rule-based exfiltration, orchestrated BEC fraud, and accessed sensitive financial systems. The combination of credential compromise, data manipulation, and potential fund diversion poses significant financial and reputational risk.

8. IMPORTANT LABELS & KEYWORDS
Business Email Compromise (BEC), Password Spray, Anonymous IP, Inbox Rule Manipulation, SAP Authentication, T1110.003, T1564.008, T1586, T1078, Initial Access, Defense Evasion, Collection, Exfiltration.

Figure 39: Incident 166 Report (Continued.)

Title
“Phishing-Driven Domain Credential Harvest and Cloud Evasion Attack”

EXECUTIVE SUMMARY
On August 1, 2024, an attacker deployed a multi-stage campaign against AlpineSkiHouse. A targeted phishing email containing a malicious URL was delivered to user “alyssat@vnevado.alpineskihouse.co” (Hailey Johnson). The user clicked the link, which initiated further payload retrieval and C2 communications from the host vnevado-win11h. The actor then performed suspicious Azure Resource Manager operations via a proxy/TOR-associated IP, abusing compromised credentials. Soon after, the adversary executed credential-dumping commands on the domain controller vnevado-dc, extracting NTDS.dit via ntdsutil. The attack blended traditional e-mail phishing, proxy evasion, and on-premises credential theft to achieve domain compromise.

INCIDENT TIMELINE
- 2024-08-01 11:26:07 UTC – Phishing mail with URL “dj01161621.bravesand-e1ccd718.eastus.azurecontainerapps.io” delivered to alyssat@…
- 12:26:22 UTC – “User accessed a link … quarantined by ZAP” alert (Node 23)
- 12:26:33 UTC – “Malicious URL was clicked on that device” on host vnevado-win11h (Node 33)
- 12:28:19–12:33:14 UTC – Proxy logs detect vnevado-win11h (231.60.52.209) connecting to login.micro.demo.antoinetest.ovh (Node 11)
- 12:32:45 UTC – Suspicious Azure Resource Management activities by a “risky” user (Hailey Johnson) flagged, involving proxy IP 253.1.244.215 (Node 32)
- 12:33:16–12:33:44 UTC – “Potentially malicious URL click detected” (Node 0)
- 12:34:30–12:36:30 UTC – “Emails containing malicious URL removed after delivery” (Node 2)
- 12:36:22 UTC – Azure Resource Manager operation from suspicious proxy IP (Node 25) targeting VM vnevado-dc
- 12:37:29 UTC – PowerShell invoked with encoded command dropping AD tools (Node 16 → PID 2556)
- 12:37:30 UTC – NTDS.dit dump via ntdsutil (“Suspicious credential dump” alert, Node 16)
- 12:37:30 UTC – ntdsutil collecting AD information for discovery & lateral movement (Node 22)
- 12:54:34–12:54:55 UTC – Follow-up ZAP and MDATP alerts reiterate link activity (Nodes 22, 23)

TECHNICAL ANALYSIS
Stage 1 – Initial Access (T1566.002)
- A phishing email (“Follow up – Security 101 content”) delivered to alyssat@… contained URL dj01161621.bravesand-e1ccd718.eastus.azurecontainerapps.io.
- Hailey Johnson clicked it via msedge.exe (PID 4256) on host vnevado-win11h (172.33.118.200), triggering Office 365 ATP and ZAP quarantine.

Stage 2 – Command & Control / Proxy Evasion
- vnevado-win11h (231.60.52.209) reached out to login.micro.demo.antoinetest.ovh, a TI-matched malicious domain.
- The attacker leveraged proxy/TOR IP 253.1.244.215 for Azure Resource Manager activities against subscription 7e838342-… (Resource Group ctfcat) and VM vnevado-dc.

Stage 3 – Credential Access (T1003, T1003.003)
- On domain controller vnevado-dc, PowerShell (PID 2556, SHA256=de96a6e6…ab32c) executed an encoded script to prepare an IFM snapshot.
- ntdsutil.exe (PID 6748, SHA256=0a302650…6e36b5e) ran “ac i ntds ifm create full c:
temp” to dump NTDS.dit.

Stage 4 – Discovery & Collection (T1018; T1069.002; T1087.002; T1482)
- ntdsutil usage flagged under Collection and Discovery categories—adversary gathering AD database and permissions for potential persistence or lateral movement.

Stage 5 – Cloud Evasion & Persistence (T1496)
- Risky user Hailey Johnson’s Azure activity raised Entra ID Protection alerts. The actor attempted ARM operations (VM run-command, listing NICs, schedules) to probe or alter cloud assets.

AFFECTED ENTITIES
Accounts
- Hailey Johnson (alyssat@vnevado.alpineskihouse.co; AAD 5e5dd0bd-…)

Hosts & Devices
- vnevado-win11h.vnevado.alpineskihouse.co (AadDeviceId 76707c40-…; IPs 231.60.52.209, 172.33.118.200)
- vnevado-dc.vnevado.alpineskihouse.co (Domain Controller; IP 65.233.23.156; MdatpDeviceId 43a4c3f27b4ff68-…)
- Azure resources in subscription 7e838342-…: VM vnevado-dc, NIC vnevado-dc-nic, DevTestLab schedule shutdown-computevm-vnevado-dc

Processes & Files
- msedge.exe (PID 4256)
- powershell.exe (PID 2556; SHA256=de96a6e6…)
- ntdsutil.exe (PID 6748; SHA256=0a302650…)

Network Indicators
- URLs:
– dj01161621.bravesand-e1ccd718.eastus.azurecontainerapps.io
– login.micro.demo.antoinetest.ovh
- IPs: 202.205.215.225; 228.3.31.94; 231.60.52.209; 253.1.244.215; 172.33.118.200

ATTACK METHODOLOGY
- Initial Access: Phishing via malicious URL (T1566.002)
- Execution: Browser & PowerShell encoded commands (T1059)
- Persistence: ARM operations, cloud resource probing
- Privilege Escalation: Credential dumping from NTDS.dit (T1003.003)
- Discovery: AD database enumeration & system/network inventory (T1018; T1069.002; T1087.002)
- Collection: Exfiltration of credentials & configuration data (T1482)
- Defense Evasion: Use of zero-hour auto purge (ZAP), TOR-associated proxy IP (T1496)

Figure 40: Incident 322 Report

INDICATORS OF COMPROMISE
• URLs:
– dj01161621.bravesand-e1ccd718.eastus.azurecontainerapps.io
– https://dj01161621.bravesand-e1ccd718.eastus.azurecontainerapps.io/
– login.micro.demo.antoinetest.ovh
• IP Addresses: 202.205.215.225; 231.60.52.209; 228.3.31.94; 253.1.244.215; 172.33.118.200
• File Hashes:
– powershell.exe (SHA256=de96a6e69944335375dc1ac238336066889d9ffc7d73628ef4fe1b1b160ab32c)
– ntdsutil.exe (SHA256=0a3026509dc46556021152242b9bb7956925d16953b05a2f548df717e5e36b5e)
• Accounts: Hailey Johnson (alyssat@. . .) – flagged as compromised
• Processes: PID 2556 (PowerShell); PID 6748 (ntdsutil.exe)

SEVERITY ASSESSMENT
Overall Impact: High
• Complete compromise of a user account and workstation
• Unauthorized credential dump of domain controller’s NTDS.dit threatens full AD domain takeover
• Use of proxy/TOR for cloud operations indicates intent to evade detection and abuse Azure resources
• Immediate risk of lateral movement, privilege escalation, and persistent foothold both on-prem and in cloud

LABELS & KEYWORDS
Phishing; ZAP; InitialAccess; T1566.002; Powershell; ntdsutil; CredentialAccess; T1003.003; Discovery; CloudEvasion; ARM; TOR; SuspiciousActivity; DomainController; AzureResourceManager; MITRE ATT&CK.

Figure 41: Incident 322 Report (Continued.)