LocalIntel: Generating Organizational Threat Intelligence from Global and Local Cyber Knowledge

Shaswata Mitra[1][0009−0002−9722−5312], Subash Neupane[1][0000−0001−9260−3914], Trisha Chakraborty[1][0009−0002−8531−0667], Sudip Mittal[1][0000−0001−9151−8347], Aritran Piplai[2][0000−0002−6437−1324], Manas Gaur[3][0000−0002−5411−2230], and Shahram Rahimi[1][0000−0003−2779−0076]

[1] Mississippi State University, MS, USA
{sm3843, sn922, tc2006}@msstate.edu, {mittal, rahimi}@cse.msstate.edu
[2] The University of Texas at El Paso, TX, USA
apiplai@utep.edu
[3] University of Maryland Baltimore County, MD, USA
manas@umbc.edu

Abstract. Security Operations Center (SoC) analysts gather threat reports from openly accessible *global threat repositories* and tailor the information to their organization's needs, such as developing threat intelligence and security policies. They also depend on organizational internal repositories, which act as private *local knowledge database*. These local knowledge databases store credible cyber intelligence, critical operational and infrastructure details. SoCs undertake a *manual labor-intensive* task of utilizing these global threat repositories and local knowledge databases to create both organization-specific threat intelligence and mitigation policies. Recently, Large Language Models (LLMs) have shown the capability to process diverse knowledge sources efficiently. We leverage this ability to automate this organization-specific threat intelligence generation. We present LOCALINTEL, a novel automated threat intelligence contextualization framework that retrieves zero-day vulnerability reports from the global threat repositories and uses its local knowledge database to determine implications and mitigation strategies to alert and assist the SoC analyst. LOCALINTEL comprises two key phases: *knowledge retrieval* and *contextualization*. Quantitative and qualitative assessment has shown effectiveness in generating up to 93% accurate organizational threat intelligence with 64% inter-rater agreement.

Keywords: Cybersecurity, Cyber Threat Intelligence (CTI), Knowledge Contextualization, Generative AI, Large Language Model (LLM)

1 Introduction

In 2023, there were 2,365 cyberattacks, with 29,065[4] reported Common Vulnerabilities and Exposures (CVE)[5]. Cyber analysts in the Security Operations

---

[4] URLs: bit.ly/3zccFKK and bit.ly/4g8bdKk

Center (SoC) retrieve malware samples from the internet. These samples are executed in sandboxes for behavior analysis. This analysis leads to developing defensive strategies to detect and prevent cyber-attacks that use such malware. The findings are shared publicly as *generic cyber threat intelligence (CTI)* in *global threat repositories* like CVE, National Vulnerability Database (NVD)[5], Common Weakness Enumeration (CWE)[5] or as third-party threat reports. Security analysts of an organization manually contextualize this generic knowledge to that organization's unique operating conditions by considering factors like network, hardware & software specifics and business needs to protect from such cyber-attacks. Security measures such as policies and protocols are then deployed depending on this contextualized information to maintain secured operations. Organizations maintain this operating information and contextualized threat intelligence documented in their *local knowledge database.*

However, expeditiously developing appropriate contextualized reports is a critical challenge before deploying security policies. Manual generation not only consumes high costs but can also be erroneous and require plenty of time due to the volume and criticality of unstructured information. On the other hand, organizations must immediately integrate policies for any novel threat to safeguard operations. Failure of *timely and correct* contextualized CTI generation for policy updation can incur heavy losses. Consider a couple of scenarios where either knowledge's availability is insufficient. **Scenario 1**: An organization's internal rules detect an unknown process attempting to communicate with an external server. The Endpoint Detection and Response (EDR) team flags and blocks the process. However, without global CTI, they are unaware that this process is part of a larger ransomware campaign. Without this knowledge, the EDR team's response is inadequate, as it fails to recognize additional Indicators of Compromises (IoCs). A secondary payload may go undetected, encrypting the organization's data. **Scenario 2**: During a routine penetration test, the software and corresponding versions used by the company are identified. Based on CVE/CWE data, the testers flag many software versions due to reported vulnerabilities. However, update log reveals that the flagged software has already been patched, making the alerts unnecessary. Modern IDEs or cybersecurity tools such as Nessus[6] or Nexpose[7] can instantly notify the SoC analyst regarding the zero-day vulnerability. However, these solutions cannot suggest accurate counteractions as they cannot assume organizational status since local knowledge resides within the organizational scope. Furthermore, organizations resist granting access to this local knowledge to a third-party vendor. This situation presents a challenge for SoC analysts as they are dealing with two sets of unstructured information. They may require more time to fully understand the context to develop the right policy before the vulnerability gets exploited in an active attack.

To address this problem, we developed LOCALINTEL. Our motivation stems from the idea that an *on-premise* system capable of *automatically* generating

---

[5] CVE: cve.mitre.org | CWE: cwe.mitre.org | NVD: nvd.nist.gov

[6] Nessus: tenable.com/products/nessus
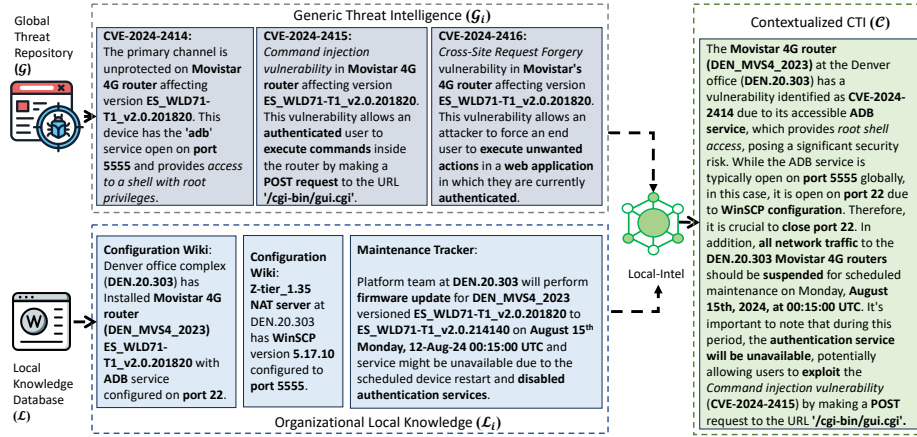
[7] Nexpose: rapid7.com/products/nexpose

Fig. 1: Overview of our LOCALINTEL framework with an example use case.

relevant and *accurate* organization-specific threat intelligence, which includes *threat implications* and *counteractions*, by assimilating global and local knowledge, would empower SoC analysts to quickly understand the effects of new cyber threats on their infrastructure, thereby saving valuable time from the manual effort. Hence, SoC analysts can develop, modify, or update their cyber defense strategies in real-time, mitigating the risk of early cyber-attacks. Considering the diverse organizational infrastructure, we have designed our LOCALINTEL framework to be modular, meaning the framework is customizable based on the use case. To the best of our knowledge, this is the first research that contextualizes global threat intelligence adapted for an organization-specific context. Our work makes the following contributions:

- We demonstrate the feasibility of producing accurate and relevant organization-specific CTI from generic threat intelligence and its operational knowledge.
- We built a knowledge-contextualization framework that generates real-time organizational CTI from publicly available and organizational knowledge.
- We construct a prototype repository of local organizational knowledge and an evaluation dataset to assess the generation of contextualized CTI.
- Through our evaluation dataset, we illustrate LOCALINTEL's ability to generate precise organizational CTI using qualitative and quantitative metrics.

In Section 2, we discuss the problem statement and theoretical foundations. Section 3 provides a detailed description of our LOCALINTEL framework. The experiment and evaluation are presented in Section 4. Moving forward, Section 5 explores the related works. Concluding remarks are in Section 6.

Table 1: Description of Notation.

Notation	Description
$\{\mathcal{G}_i \mid \mathcal{G}_i \in \mathcal{G}\}$	Global Threat Repository
$\{\mathcal{L}_i \mid \mathcal{L}_i \in \mathcal{L}\}$	Local Organizational Database
$\mathcal{Q}$	Query to fetch global ($\mathcal{G}_i$) & local ($\mathcal{G}$) knowledge
$\mathcal{C}$	Contextualized Completion

2   Research Objective & Theoretical Foundations

Global threat repository ($\mathcal{G} = \{\mathcal{G}_1, \mathcal{G}_2, ..., \mathcal{G}_n\}$) is a publicly available set of online
CTI reports ($\mathcal{G}_i$). Local knowledge database ($\mathcal{L} = \{\mathcal{L}_1, \mathcal{L}_2, ..., \mathcal{L}_n\}$) consists of
policies and procedures of an organization's operating environments ($\mathcal{L}_i$), such
as business requirements, trusted cyber intelligence, allowed system software list
and version details, cyber knowledge about the organization, asset location and
configurations, DMZ configurations, and maintenance reports.

Problem Statement

For a given set of vulnerability $\mathcal{G}_i$ in $\mathcal{G}$ and corresponding relevant organiza-
tional knowledge $\mathcal{L}_i$ in $\mathcal{L}$. The task is to generate Completion ($\mathcal{C}$), which is the
contextualized knowledge of $\mathcal{G}_i$ w.r.t. $\mathcal{L}_i$. $\mathcal{C}$ can be considered as contextualiza-
tion function $f(\cdot)$, that translates $\mathcal{G}_i$ to organizational context using $\mathcal{L}_i$.

$$f(\mathcal{G}_i, \mathcal{L}_i) = \mathcal{C}_i \forall \mathcal{G}_i \cap \mathcal{L}_i \neq \phi \tag{1}$$

For instance, in Figure 1, where $\mathcal{G}_i$ is a set containing information stating
vulnerability (v) through process (p). Alternatively, $\mathcal{L}_i$ contains information re-
garding the organization using process (p) for its operations and other relevant
information. Hence, in the process of generating contextualized threat intelli-
gence $\mathcal{C}$ considering v and p, $\mathcal{G}_i$ is being translated through $\mathcal{L}_i$, when $\mathcal{G}_i \cap \mathcal{L}_i \neq \phi$.

3   LOCALINTEL Framework

In this section, we explain our LOCALINTEL framework. We first explain our
solution and each module with its functionality in detail (refer to Figure 3). Fi-
nally, we discuss the system implementation and module interactions to generate
the final contextualized threat intelligence $\mathcal{C}$.

3.1   Solution Approach

LOCALINTEL consists of two core phases: knowledge retrieval *(Retrieval Phase)*
and generation *(Generation Phase)*. In the *retrieval phase*, knowledge from *global
($\mathcal{G}$)* and *local ($\mathcal{L}$)* sources are retrieved, and in the *generation phase*, a final
contextualized threat intelligence $\mathcal{C}$ based on the retrieved knowledge $\mathcal{G}_i \cup \mathcal{L}_i$ is
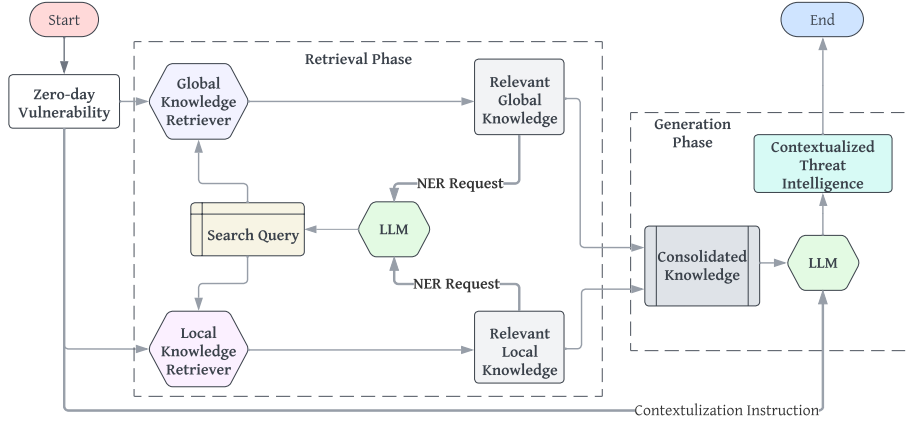generated. Refer to Figure 2 and Algorithm 1 for framework overview.

Fig. 2: LOCALINTEL data-flow diagram. The zero-day vulnerability triggers the system to retrieve information from global $\mathcal{G}_i$ and local $\mathcal{L}_i$ knowledge sources and then contextualizes the results, producing a final output Completion $\mathcal{C}$.

Algorithm 1: LOCALINTEL Pseudo-code

Input: Generic Threat Intelligence ($\mathcal{G}_i$)
Output: Contextualized Threat Intelligence ($\mathcal{C}$)
RETRIEVAL PHASE:
$\mathcal{L}_i \leftarrow execute\_local\_search(\mathcal{G}_i, \mathcal{L})$
while $\mathcal{L}_i \cap \overline{\mathcal{G}_i} \neq \phi$ **do**
$\quad$ $\mathcal{Q} \leftarrow get\_search\_query(\mathcal{G}_i \cup \mathcal{L}_i)$
$\quad$ **forall** $\alpha \in \mathcal{Q}$ **do**
$\quad\quad$ $\mathcal{G}_i \leftarrow execute\_global\_search(\alpha, \mathcal{G})$
$\quad$ $\mathcal{L}_i \leftarrow execute\_local\_search(\mathcal{G}_i, \mathcal{L})$
GENERATION PHASE:
$\mathcal{C} \leftarrow generate\_completion(\mathcal{G}_i \cup \mathcal{L}_i)$
return $\mathcal{C}$

– In the Retrieval Phase, the system retrieves generic CTI $\mathcal{G}_i$ from $\mathcal{G}$ and relevant local knowledge $\mathcal{L}_i$ from $\mathcal{L}$ based on the relevancy. The system performs Named Entity Recognition (NER) to identify search keywords/queries $\mathcal{Q}$ over on the acquired knowledge ($\mathcal{G}_i \cup \mathcal{L}_i$). Then, it executes the search for all search queries in $\mathcal{Q}$ in the global knowledge repository $\mathcal{G}$ and local knowledge database $\mathcal{L}$ to fetch relevant threat reports and associated details. This phase continues until no additional knowledge is required ($\mathcal{L} \cap \overline{\mathcal{G}_i} \neq \phi$) to generate final contextualized threat intelligence.

– Finally, in the Generation Phase, the system generates contextualized threat intelligence $\mathcal{C}$ for the zero-day vulnerability generic threat intelligence based on the retrieved global knowledge and local knowledge ($\mathcal{G}_i \cup \mathcal{L}_i$).

3.2   LOCALINTEL System Modules

To implement LOCALINTEL, we first discuss system modules, which are *Global Threat Repository ($\mathcal{G}$), Local Knowledge Database ($\mathcal{L}$), Agent, Tools, LLM*, with zero-day threat report input $\mathcal{G}_i$ and contextualized completion $\mathcal{C}$) output.

Global Threat Repository ($\mathcal{G}$) refers to publicly available cybersecurity threat intelligence (CTI), such as threat reports from CVE, NVD, CWE, security blogs and bulletins, social media updates, and third-party reports. These repositories contain well-documented reports on cybersecurity threats, such as malware, vulnerability, cyber attacks, and many more. The primary purpose of these repositories is to facilitate information sharing among cybersecurity professionals regarding the latest developments. However, the global knowledge $\mathcal{G}_i$ is generic and may not directly apply to an organization's needs as organizations tend to customize their infrastructure depending upon the business. Moreover, the knowledge obtained from unverifiable sources is only directly usable by an organization after thorough analysis. LOCALINTEL is expected to be connected to these threat repositories for automated zero-day vulnerability report retrieval.

Local Knowledge Database ($\mathcal{L}$) refers to an organizations' operational information repository. Due to the generic characteristics, $\mathcal{G}$ contains a wide range of CTI, but they must be supplanted with organization-specific information to be useable. Hence, local knowledge databases or wikis are private knowledge repositories containing critical information related to organizational operations and trusted threat intelligence, such as specifics regarding the environment, operating systems, infrastructure, software, third-party systems, and processes. Confluence[8], Notion[9], are a few instances of such wiki platforms. The primary goal of these wikis is to facilitate structured development and knowledge sharing among the working professionals in an organization. Due to the unstructured nature of this information, we assume wiki platforms to be our local knowledge database. However, more structured sources like knowledge graphs can replace them with similar searching functionalities.

Agent is the main controller in our LOCALINTEL framework. It controls the overall flow, from receiving the input vulnerability report trigger to returning the final contextualized completion $\mathcal{C}$. Specifically, the Agent's function is to determine and regulate the sequence of actions among two phases for generating the output. The Agent actions are primarily of three types: *Query generation*, *Query execution*, and *Completion generation*. To achieve this, the Agent interacts with the other two modules: *Tool* and *LLM*, detailed following.

- *Query generation* refers to generating search queries for information retrieval from either $\mathcal{G}$ or $\mathcal{L}$. The Agent generates a search query to retrieve all the

---

[8] Confluence: atlassian.com/software
[9] Notion: notion.so/product/wikis

relevant information from pre-acquired knowledge ($\mathcal{G}_i \cup \mathcal{L}_i$). The Agent performs contextual embedding and keyword identification through named entity recognition (NER) to generate search queries $\mathcal{Q}$.

– *Query execution* refers to executing search query $\mathcal{Q}$ in global threat repository $\mathcal{G}$ and local knowledge database $\mathcal{L}$ to retrieve relevant knowledge. Due to the different characteristics of $\mathcal{G}$ and $\mathcal{L}$, the retrieval process can either be a keyword search through online API calls or a semantic similarity search.

– *Text generation* can be considered an LLM inference scenario, where the Agent passes an input text to generate desired output text using LLM. Task-specific input prompts are pre-designed in the Agent.

Tool are functions that help the Agent execute some third-party actions. The actions can be diverse in type, for instance, making an online API call, performing a database search, executing custom scripts, invoking other software, and many more. However, for the scope of our research, tool functionality is limited to query generation using LLM, query execution through API calls and vector database search, and contextualized generation using LLM functionalities only. Therefore, in our LOCALINTEL framework, tools are responsible for executing online searches through API calls or vector database searches and parsing the results while bridging the Agent's access to different framework modules.

Large Language Model (LLM) acts as the brain of our LOCALINTEL framework to process diverse information and generate contextualized CTI. Besides contextualized threat intelligence generation, it acts as the parser that processes retrieved knowledge to generate queries for structured information retrieval. Depending on the task, the Agent invokes LLM with instructions and information.

Input: Zero-day threat report ($\mathcal{G}_i$) refers to publicly available CTI reports regarding any discovered vulnerability or malware. We assume that LOCALINTEL is connected with the global CTI repositories ($\mathcal{G}$) with active triggers to receive any newly disclosed threat reports for instant processing.

Output: Contextualized completion ($\mathcal{C}$) is the real-time generated threat intelligence specifically tailored for an organization depending on its unique operating condition. The objective of $\mathcal{C}$ is to assist SoC by providing mitigating strategies or relevant information on the specific zero-day threat ($\mathcal{G}_i$). We assume the local knowledge database ($\mathcal{L}$) contains all required organizational knowledge.

3.3   LOCALINTEL Implementation & Module Interactions

Previously, we have described each module in the architecture. Here, we explain the implementation phases with intermediate module interactions (refer to Figure 3). LOCALINTEL initiates when a vulnerability report is received. The report can be pushed manually or via automated zero-day triggers.
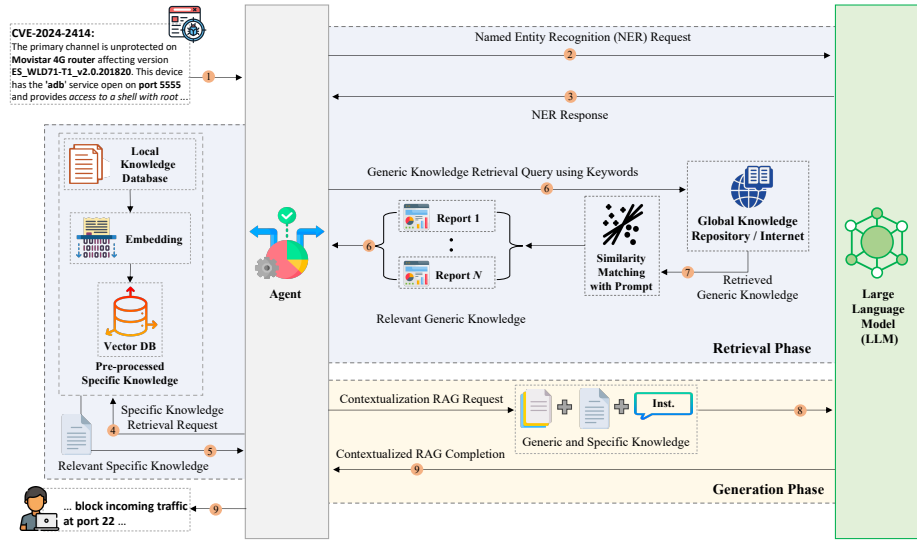
Fig. 3: LOCALINTEL framework module interaction in the two phases: knowledge retrieval, and contextualized threat intelligence generation. Processes are numbered in ascending execution order following the data-flow diagram (Fig. 2).

Knowledge Retrieval (Phase 1): This is the first phase of our framework where the *Agent* retrieves generic threat intelligence ($\mathcal{G}_i$). It generates search queries ($\mathcal{Q}$) for relevant knowledge retrieval. The initial local knowledge search (refer to Algorithm 1) plays a crucial role in identifying whether the threat intelligence is relevant to the organization. If there is no overlap ($\mathcal{G}_i \cap \mathcal{L} = \phi$), then the Agent discards input $\mathcal{G}_i$, as there are no connections; hence, it cannot be contextualized. Upon overlaps discovered, it iteratively generates $\mathcal{Q}$ and executes knowledge retrieval from both global ($\mathcal{G}$) and local ($\mathcal{L}$) sources until all required knowledge needed to be considered for contextualization is retrieved. For the scope of our experiment, we implemented global knowledge retrieval from the Internet through keyword search via API endpoints of global threat repositories such as NIST, CVE, and ensemble [2] vector similarity search for local knowledge retrieval from the organizational wikis. This simplified approach efficiently fetches corresponding relevant knowledge from both sources. For instance, for the following threat intelligence, the execution is as follows:

Invoked Generic Threat Intelligence ($\mathcal{G}_i$)

CVE-2024-2414: The primary channel is unprotected on Movistar 4G router affecting E version S_WLD71-T1_v2.0.201820. This device has the 'adb' service open on port 5555 and provides access to a shell with root privileges.

Upon receiving $\mathcal{G}_i$ above, the Agent generates query embedding to perform ensemble retrieval in $\mathcal{L}$. After executing $\mathcal{Q}$ in the vector-indexed $\mathcal{L}$, the Agent identifies "Movistar 4G" to be the affecting device with following knowledge:

Phase 1: NER Query Generation ($\mathcal{Q}$)

Agent Instruction:
You are a named entity recognition (NER) tool. Given the following classes, perform NER for the provided Input text.
Classes: software, device, library, functionality, attack_vector, vulnerability ...
Input Threat Intelligence:
CVE-2024-2414: The primary channel is unprotected on Movistar 4G router affecting E version S_WLD71-T1_v2.0.201820. This device has the 'adb' service open on port 5555 and provides access to a shell with root privileges.
Output Keywords:
{"device": "Movistar 4G", "attack_vector": "port 5555", "functionality": "adb" }

The semantic search is performed through vector embedding generation and execution of similarity matching algorithms (cosine, euclidean, dot-product).

Phase 1: Retrieved Local Knowledge ($\mathcal{L}_i$) using $\mathcal{Q}_{\mathcal{L}}$

Configuration Wiki::
– Denver office complex (DEN.20.303) has Installed **Movistar 4G router** (DEN_MVS4_2023) **ES_WLD71-T1_v2.0.201820** with ADB service configured on port 22.
– Z-tier_1.35 NAT server at DEN.20.303 has WinSCP version 5.17.10 configured to **port 5555**.

After searching $\mathcal{L}$, the Agent performs similar query generation $\mathcal{Q}$ and execution iteratively in $\mathcal{G}$ and $\mathcal{L}$ for additional context retrieval. For this example, the additional retrieved knowledge $\mathcal{G}_i$ and $\mathcal{L}_i$ from Phase 1 is below:

Phase 1: Additional Global and Local Knowledge ($\mathcal{G}_i \cup \mathcal{L}_i$)

Global Knowledge:
– **CVE-2024-2415**: Command injection vulnerability in **Movistar 4G router** affecting version **ES_WLD71-T1_v2.0.201820**. This vulnerability allows an authenticated user to execute commands inside the router by making a POST request to the URL **'/cgi-bin/gui.cgi'**.
– **CVE-2024-2416**: Cross-Site Request Forgery vulnerability in Movistar's 4G router affecting version **ES_WLD71-T1_v2.0.201820**. This vulnerability allows an attacker to force an end user to execute unwanted actions in a web application in which they are currently authenticated.
Local Knowledge:
– **Maintenance Tracker**: Platform team at **DEN.20.303** will perform firmware update for **DEN_MVS4_2023** versioned **ES_WLD71-T1_v2.0.201820** to ES_WLD71-T1_v2.0.214140 on August 15th Monday, 12-Aug-24 00:15:00 UTC and service might be unavailable due to the scheduled device restart and disabled authentication services.

Consolidated knowledge $(\mathcal{G}_i \cup \mathcal{L}_i)$ is passed for query generation. For this case, *DEN.20.303* and *DEN_MVS4_2023* used to identify related information.

Contextualized Generation (Phase 2): In this phase, upon complete retrieval of $\mathcal{G}_i \cup \mathcal{L}_i$, the Agent invokes LLM for final contextualization.

Phase 2: Contextualized Organizational Threat Intelligence ($\mathcal{C}$)

Agent Instruction:

You are an honest network security analyst. Given public threat intelligence reports fetched from trusted cybersecurity sources and organizational infrastructure and operations details. Generate a cyber threat intelligence report with all details, including the impact and mitigation strategies. Do not include any information that is not provided as additional knowledge.

Retrieved Global Knowledge ($\mathcal{G}_i$):

– **CVE-2024-2414**: The primary channel is unprotected on Movistar 4G router affecting E version S_WLD71-T1_v2.0.201820. This device has the 'adb' service open on port 5555 and provides access to a shell with root privileges.

– **CVE-2024-2415**: Command injection vulnerability in Movistar 4G router affecting version ES_WLD71-T1_v2.0.201820. This vulnerability allows an authenticated user to execute commands inside the router by making a POST request to the URL '/cgi-bin/gui.cgi'.

– **CVE-2024-2416**: Cross-Site Request Forgery vulnerability in Movistar's 4G router affecting version ES_WLD71-T1_v2.0.201820. This vulnerability allows an attacker to force an end user to execute unwanted actions in a web application in which they are currently authenticated.

Retrieved Local Knowledge ($\mathcal{L}_i$):

– **Configuration Wiki**: Denver office complex (DEN.20.303) has Installed Movistar 4G router (DEN_MVS4_2023) ES_WLD71-T1_v2.0.201820 with ADB service configured on port 22

– **Configuration Wiki**: Z-tier_1.35 NAT server at DEN.20.303 has WinSCP version 5.17.10 configured to port 5555.

– **Maintenance Tracker**: Platform team at DEN.20.303 will perform firmware update for DEN_MVS4_2023 versioned ES_WLD71-T1_v2.0.201820 to ES_WLD71-T1_v2.0.214140 on August 15th Monday, 12-Aug-24 00:15:00 UTC and service might be unavailable due to the scheduled device restart and disabled authentication services.

Contextualized Completion:

The **Movistar 4G router (DEN_MVS4_2023)** at the **Denver office (DEN.20.303)** has a vulnerability identified as **CVE-2024-2414** due to its accessible ADB service, which provides root shell access, posing a significant security risk. **While the ADB service is typically open on port 5555 globally, in this case, it is open on port 22 due to WinSCP configuration**. Therefore, it is crucial to **close port 22**. In addition, all **network traffic to the DEN.20.303 Movistar 4G routers should be suspended** for scheduled maintenance on **Monday, August 15th, 2024, at 00:15:00 UTC**. It's important to note that during this period, the **authentication service will be unavailable, potentially allowing users to exploit the Command injection vulnerability (CVE-2024-2415)** by making a POST request to the URL '/cgi-bin/gui.cgi'.

After completing both phases, we can observe that through an initial local search, the agent identified the "Movistar 4G router" as the device of interest ($\mathcal{G}_i \cap \mathcal{L}$) with additional relevant knowledge. Then, it iteratively retrieved additional threat intelligence (CVE-2024-2415 and CVE-2024-2416) for the device from $\mathcal{G}$ (we considered NVD as $\mathcal{G}$ for the experiment) and $\mathcal{L}$ (considered an organizational wiki) to obtain additional local context. The concatenation of knowledge prior to retrieval allowed the discovery of indirect relevant documents such as maintenance schedules. Without this knowledge, the mitigation strategy might become ineffective. Finally, by providing all relevant information ($\mathcal{G}_i \cup \mathcal{L}_i$) and task instruction, the Agent invokes LLM for real-time organization-specific threat intelligence generation. This relevant real-time update then equips SoC analysts with all relevant information without investing any time in manual investigation. The SoC analyst can then utilize this knowledge to take the necessary actions to safeguard the organization against imminent cyber threats.

4   Experiment and Evaluation

In this section, we discuss our experiments and the achieved evaluation results. For our evaluations, we performed experiments considering 58 publicly available threat intelligence scenarios to demonstrate the feasibility of the LocalIntel framework and assess contextualization relevancy. For the global threat repository ($\mathcal{G}$), we considered NVD-CVE data, and for the local knowledge database ($\mathcal{L}$), a curated organizational wiki (PII anonymized for confidentiality). However, as described in Section 3, LocalIntel[10] is modular, allowing flexibility to modify the modules depending on requirements and organization-specifics. For example, other generic threat intelligence sources can be integrated with $\mathcal{G}$, different local knowledge sources such as knowledge graphs can be incorporated, and other generative language models can be adopted for a more controlled generation. Following, we will delve into the evaluation dataset and experiment setup. Finally, we will describe our evaluation measures and findings with justifications.

4.1   Data Description and Experiment Setup

Our dataset includes (1) **58** trigger/zero-day generic threat intelligence reports ($\mathcal{G}_i$), (2) **5 organizational wikis** resembling an organizational local knowledge database source ($\mathcal{L}$), and (3) **58** subject matter expert (SME) generated (manually unbiased) ground truth ($\overline{\mathcal{C}}$). A trigger ($\mathcal{G}_i$) can be a report of any malware, vulnerability, attack vector, or security updates. For further automated relevant global knowledge retrieval, LocalIntel is connected with CVE API endpoints. The 58 trigger reports contain both positive and negative test cases. We gathered **5 organizational wikis** corresponding to 5 real-time applications and curated them (PII removed) suitable to the research. For each

---

[10] LocalIntel Repository: github.com/shaswata09/LocalIntel

positive test scenario, we ensured the corresponding knowledge was present in the local knowledge database. In addition to the organizational wiki, we also collected **326** confidential organizational trusted CTI reports to allow LOCAL-INTEL to retrieve more infrastructural context and threat implications. These reports offer detailed analyses and insights from security analysts studying various global cyber attacks within the organization. For negative test scenarios, there was no intersecting knowledge present in $\mathcal{L}$ i.e. $\mathcal{G}_i \cap \mathcal{L} = \phi$. In conducting our experiments, we tested with proprietary *GPT-3.5-turbo, and GPT-4o*[11], *and open-source meta-llama/Llama-2-7b-chat-hf, meta-llama/Meta-Llama-3.1-8B-Instruct, mistralai/Mistral-7B-Instruct-v0.2, nvidia/Mistral-NeMo-Minitron-8B-Base, Qwen/Qwen1.5-7B-Chat, AiMavenAi/AiMaven-Prometheus, senseable/WestLake-7B-v2, PetroGPT/WestSeverus-7B-DPO-v2* downloaded from *huggingface.co* as the LLM models. All models' temperatures were deliberately kept default, and instructions prompts were set the same for neutral comparison. The global knowledge was retrieved from NVD-CVE sources through search API. For local knowledge retrieval, we store the 5 organizational wikis and 326 threat reports in a vector database (Chroma [12]). We segmented and organized the data into smaller chunks to enhance processing efficiency. In our experimental setup, we opted for a *chunk size* of 1500 with a *chunk overlap* of 150. We used the *text-embedding-ada-002*[13] as our base model for embedding each chunk of data in Chroma DB and used Maximal Marginal Relevance (MMR) sorting for dense retrieval of relevant chunks. The experiment was performed over Intel i9-12900 with 24 GB GeForce RTX™ 3090Ti GPU and 128 GB of RAM.

Table 2: Evaluation results of our LOCALINTEL framework over following LLMs.

Model	Ragas (Sim.)	GEval (Cor.)	BertSc-F1
gpt-3.5-turbo	0.92	0.75	0.68
gpt-4o	0.91	0.75	0.66
qwen1.5-7b-chat	0.92	0.78	0.66
llama-3.1-8b-Instruct	0.85	0.46	0.53
westlake-7b-v2	0.92	0.69	0.65
llama2-7b-chat	0.91	0.69	0.65
mistral-7b-instruct-v2	0.90	0.67	0.63
prometheus-7b	0.93	0.71	0.66
westseverus-7b-dpo-v2	0.90	0.60	0.60
mistral-nemo-minitron-8b	0.84	0.56	0.55

---

[11] GPT Models: platform.openai.com/docs

[12] Chroma: trychroma.com
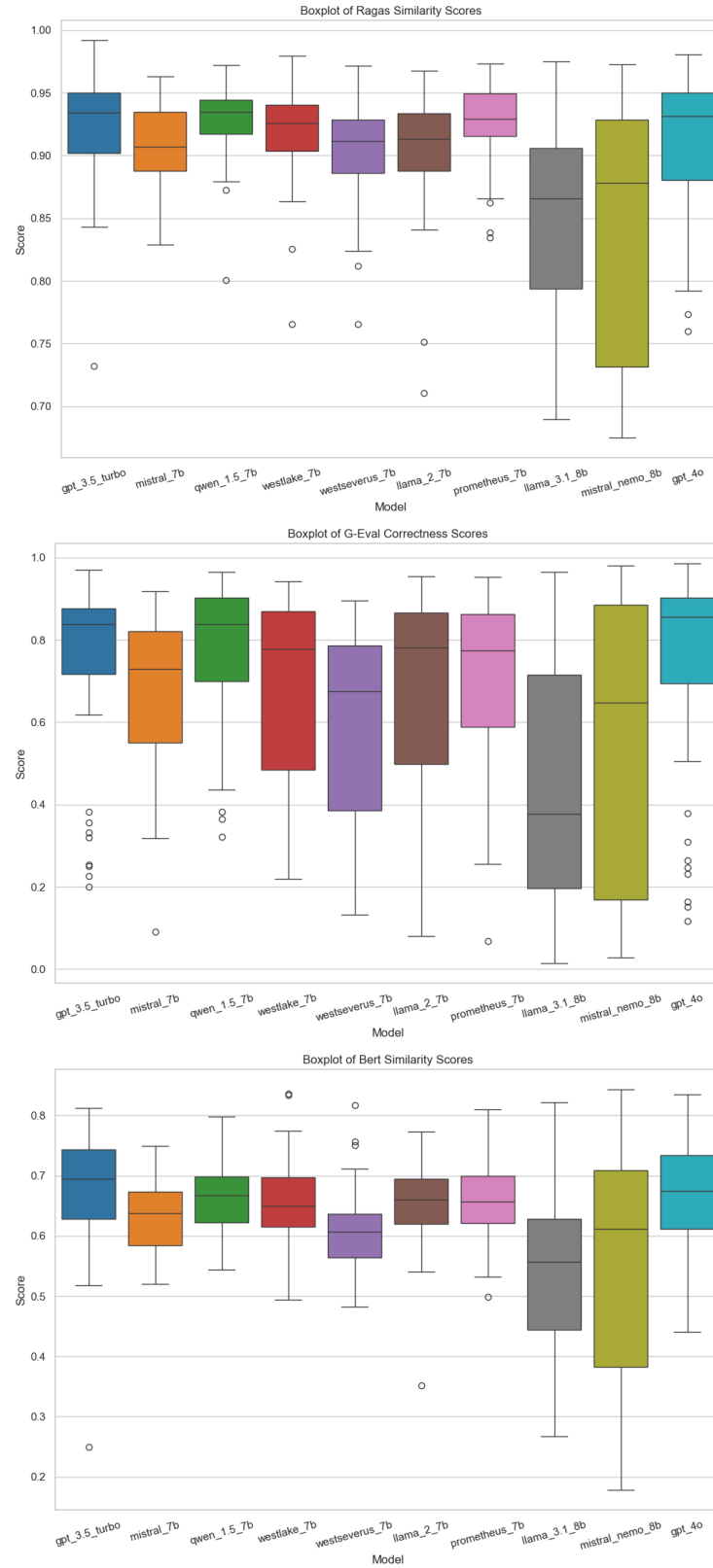
[13] OpenAI Embedding: platform.openai.com/docs

Fig. 4: Evaluation box-plot for Completion $\mathcal{C}$ with respect to ground truth.

4.2   Quantitative Evaluation

For the quantitative assessment of LOCALINTEL's performance in generating contextually relevant organizational threat intelligence $\mathcal{C}$, we utilize three frameworks: Retrieval Augmented Generation Assessment (RAGAs) [3], G-EVAL [6], and BertScore [15]. Using these frameworks, we evaluate two metrics, including *similarity* and *correctness*. Similarity measures the semantic similarity between ground truth and $\mathcal{C}$, while correctness measures answer correctness compared to ground truth as a combination of factuality and semantic similarity. Both metrics range from 0 to 1, with higher values indicating optimal $\mathcal{C}$. In our case, RAGAs and BertScore is used to evaluate similarity, whereas G-EVAL is used to evaluate correctness of $\mathcal{C}$. Results of our evaluation is presented in Table 2.

In our evaluation, the model *Qwen1.5-7B-Chat* performed the best, with the highest similarity score and the lowest standard deviation, as depicted in Fig 4. On the other hand, *Mistral-NeMo-Minitron-8B-Base* was the least-performing model. We found that 'qwen' was the most stable, which is essential in critical domains such as cybersecurity. Contrarily, 'mistral-nemo' showed lower accuracy and higher variance. This can be explained through Mistral's sliding attention mechanism that struggles to retail critical information over longer contexts. We also discovered that due to the task criticality, llama 3.1 avoided suggesting a solution, indicating its cautious generation. We observed a similar trend with the 'GPT 4o' model. Another critical point to note is that we used a generic instruction prompt for all models, and it is also worth mentioning that model-specific prompt engineering techniques may lead to even better results.

4.3   Qualitative Evaluation

To justify our quantitative findings (refer to Section 4.2), we qualitatively evaluate the performance of LOCALINTEL in generating contextually relevant organizational threat intelligence through human evaluation. Given the expensive nature of human evaluation, we engage a panel of **3** Subject Matter Experts (SMEs), including *one security analyst* and *two cybersecurity researchers*. We task these SMEs to evaluate the *correctness* of generated threat intelligence based on the 58 scenarios and ground truths explained in the preceding section. The SMEs were instructed to rate the correctness of the response on a scale of **1 to 5**, where 1 represents an incorrect response, and 5 indicates a correct response. We then compare the inter-rater agreement using Fleiss Kappa [7] measure. The result of this evaluation shows an agreement score of **0.6477** with a standard error of **0.0767**, indicating that the raters' evaluations are not random and are generally aligned, and they ***substantially agree*** on the *correctness* of the threat intelligence responses generated by LOCALINTEL. Moreover, *qualitative results* aligns closely with the *quantitative results*, justifying the evaluation.

5   Related Works

In the last decade, within the realm of cybersecurity, NLP tasks over unstructured CTI text primarily encompass *Named Entity Recognition, text summariza-*

tion, and *analysis of semantic relationships between entities*[13], etc. Researchers have demonstrated numerous real-world applications using these techniques utilizing CTI gathered from diverse sources [9,11,12,8]. With the advancement of generative AI in this decade, the application horizon of CTI has proportionally expanded. Liu et al. [5] introduced a trigger-enhanced CTI (TriCTI) discovery system designed to identify actionable CTI automatically. They utilized a fine-tuned BERT with an intricate design to generate triggers, training the trigger vector based on sentence similarity. Similarly, in [1], the researchers employed a BERT classifier to map Tactics, Techniques, and Procedures (TTPs) to the MITRE ATT&CK framework. On the other hand, Niakanlahiji et al., [10], proposes an information retrieval system called SECCMiner utilizing various NLP techniques. With SECCMiner, unstructured APT reports can be analyzed, and critical security concepts (e.g., adversarial techniques) can be extracted. A question and answering model called LogQA that answers log-based questions in natural language form using base BERT model and large-scale unstructured log corpora is proposed by Huang et al. [4]. Recently, BERT has also been explored to generate contextualized embedding [14] in cybersecurity. Cybersecurity is a critical domain, and this specialized embedding enables language models to understand the context better. On top of the improvements mentioned, we attempt to integrate LLM to understand the problem context and generate real-time scope-specific threat intelligence while considering different factors. This work is the first attempt to generate complete CTI from diverse sources.

6   Conclusion

This paper introduced LOCALINTEL, a novel framework that generates contextualized CTI uniquely tailored for an organization depending on its operations. LOCALINTEL is a valuable tool for SoC analysts due to its unique ability to seamlessly contextualize generic global threat intelligence specific to local operations. The main benefit of this system is its ability to efficiently customize global threat intelligence for local contexts, reducing the need for manual efforts. This gives SoC analysts the necessary information to concentrate on essential tasks, such as developing defensive strategies. We employed qualitative and quantitative evaluations to evaluate LOCALINTEL's confidence in delivering accurate and relevant threat intelligence. The system exhibited remarkable proficiency in both evaluations, supported by human-generated ground truth responses. It achieved a remarkable RAGAs contextual similarity score of 92% and a correctness score of 78%, with a low standard deviation. This underscores the feasibility of automated CTI generation using LLMs and our LOCALINTEL's robust performance and ability to generate relevant CTI. In the future, we plan to perform further performance improvement measures, such as developing task-specific retrievers and connecting with cybersecurity knowledge graphs as our local knowledge database for broader evaluations. Additionally, we plan to fine-tune LLMs as part of our performance improvement measures.

References

1. Alves, P.M., Geraldo Filho, P., Gonçalves, V.P.: Leveraging bert's power to classify ttp from unstructured text. In: 2022 Workshop on Communication Networks and Power Systems (WCNPS). pp. 1–7. IEEE (2022)
2. Arabzadeh, N., Yan, X., Clarke, C.L.: Predicting efficiency/effectiveness trade-offs for dense vs. sparse retrieval strategy selection. In: Proceedings of the 30th ACM International Conference on Information & Knowledge Management. pp. 2862–2866 (2021)
3. Es, S., James, J., Espinosa-Anke, L., Schockaert, S.: Ragas: Automated evaluation of retrieval augmented generation. arXiv preprint arXiv:2309.15217 (2023)
4. Huang, S., Liu, Y., Fung, C., Qi, J., Yang, H., Luan, Z.: Logqa: Question answering in unstructured logs. arXiv preprint arXiv:2303.11715 (2023)
5. Liu, J., Yan, J., Jiang, J., He, Y., Wang, X., Jiang, Z., Yang, P., Li, N.: Tricti: an actionable cyber threat intelligence discovery system via trigger-enhanced neural network. Cybersecurity **5**(1), 8 (2022)
6. Liu, Y., Iter, D., Xu, Y., Wang, S., Xu, R., Zhu, C.: G-eval: Nlg evaluation using gpt-4 with better human alignment. arXiv preprint arXiv:2303.16634 (2023)
7. McHugh, M.L.: Interrater reliability: the kappa statistic. Biochemia medica **22**(3), 276–282 (2012)
8. Mitra, S., Piplai, A., Mittal, S., Joshi, A.: Combating fake cyber threat intelligence using provenance in cybersecurity knowledge graphs. In: 2021 IEEE International Conference on Big Data (Big Data). pp. 3316–3323. IEEE (2021)
9. Mittal, S., Das, P.K., Mulwad, V., Joshi, A., Finin, T.: Cybertwitter: Using twitter to generate alerts for cybersecurity threats and vulnerabilities. In: 2016 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM). pp. 860–867. IEEE (2016)
10. Niakanlahiji, A., Wei, J., Chu, B.T.: A natural language processing based trend analysis of advanced persistent threat techniques. In: 2018 IEEE International Conference on Big Data (Big Data). pp. 2995–3000. IEEE (2018)
11. Pingle, A., Piplai, A., Mittal, S., Joshi, A., Holt, J., Zak, R.: Relext: Relation extraction using deep learning approaches for cybersecurity knowledge graph improvement. In: Proceedings of the 2019 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining. pp. 879–886 (2019)
12. Piplai, A., Mittal, S., Joshi, A., Finin, T., Holt, J., Zak, R.: Creating cybersecurity knowledge graphs from malware after action reports. IEEE Access **8**, 211691–211703 (2020)
13. Rahman, M.R., Mahdavi-Hezaveh, R., Williams, L.: A literature review on mining cyberthreat intelligence from unstructured texts. In: 2020 International Conference on Data Mining Workshops (ICDMW). pp. 516–525. IEEE (2020)
14. Ranade, P., Piplai, A., Joshi, A., Finin, T.: Cybert: Contextualized embeddings for the cybersecurity domain. In: 2021 IEEE International Conference on Big Data (Big Data). pp. 3334–3342. IEEE (2021)
15. Zhang, T., Kishore, V., Wu, F., Weinberger, K.Q., Artzi, Y.: Bertscore: Evaluating text generation with bert. arXiv preprint arXiv:1904.09675 (2019)