From Text to Insight: Encoding Cyber Attack Patterns in Threat Intelligence Reports Using Knowledge Graphs and LLMs

Patrick Zounon[1], Yufei Han[1], Michel Hurfin[1], and Frédéric Majorczyk[2]

[1]INRIA, Univ. Rennes, IRISA, {firstname.lastname}@inria.fr
[2]DGA-MI, Univ. Rennes, IRISA, frederic.majorczyk@def.gouv.fr

***Abstract*—Cyber Threat Intelligence (CTI) reports provide valuable insights into cyberattacks, detailing target systems, attack methods, and vulnerabilities. However, their volume and unstructured format hinder security professionals' ability to efficiently extract, summarize, and predict attack patterns. In this paper, we propose automating the transformation of CTI reports into a graph-based Cyber Security Knowledge Graph (CSKG) using Large Language Models (LLMs) for security entity recognition. Specifically, we outline our initial efforts to define essential cyber security entities and relationships, which serve as the foundational elements of cyber attack descriptions. These efforts mark a significant step toward our broader objective of encoding cyber security knowledge in a graph-structured format. Additionally, we discuss future directions, including the development of a comprehensive cyber attack knowledge graph that integrates multiple CTI reports, and explore its potential applications in inferring, reasoning, and prioritizing attack behaviors.**

***Index Terms*—Cyber Threat Intelligence (CTI), Named Entity Recognition (NER), Knowledge Representation, Large Language Models (LLMs)**

I. Introduction

Cyber Threat Intelligence (CTI) aims to share information about cyber attacks, including the attack technologies and the vulnerabilities exploited by attackers. Nevertheless, the main bottleneck of sharing security knowledge with CTI reports is that they lack a unified structure to model the attack behaviors observed during description of the threat action. They are usually produced by technical audiences from different cybersecurity vendors or government agencies, which leverage their own analysis tools, deployed sensor networks and collected incident response data. Additionally, open-source communities, including ethical hackers and independent researchers, publish findings on blogs, forums, and GitHub repositories. As a result, the generated CTI reports use unstructured formats, highly diverse technical terminologies and granularity levels of cyber attack analysis to describe the attack behavior patterns of the investigated attack scenarios. Extracting knowledge contained in CTI reports thus becomes label-intensive. As reported in SANS CTI Survey 2023 [1], over half of the security analysis teams spend more than 40% of their overheads in identifying entities related to cyber attacks and shaping their own structured CTI requirements. Besides, it is challenging to attribute and retrieve similar attack kill chain patterns contained in the unstructured reports, hence preventing the use of CTI reports to enhance reasoning and inference of future security incidents.

While the Structured Threat Information Expression (STIX) format [2] is gaining traction for standardizing cyber intelligence, it struggles with capturing nuanced details like attacker motives and has a steep learning curve. It hinders analysts to create, interpret, and use STIX objects effectively without specialized training. This complexity can lead to inconsistent implementations and misinterpretations. Therefore, an automated framework is needed to extract indicators of compromise (IOCs) and attack patterns from unstructured CTI reports, enabling advanced analytics like attack categorization, reasoning, and prediction.

Pioneering efforts to standardize knowledge representation in CTI reports [3], [4] highlight Knowledge Graphs (KGs) as a promising approach for encoding the unstructured text-based descriptions of cyber attacks. Cyber security knowledge graphs (CSKGs) represent threat intelligence using a graph-based data model, where each node corresponds to an entity associated with a cyber attack. For example, these entities can include malware families deployed in attacks, exploited vulnerabilities, targeted hardware configurations, and technical indicators of compromise such as URLs or file hashes. The edges of the graph capture the relations between these entities. Knowledge graphs facilitate efficient retrieval and analysis by enabling analysts to quickly identify and understand the intricate relations among attack components. They support inference and clustering of similar attack behaviors, enhancing the ability to recognize attack behavioral patterns and predict potential threats. By aggregating and integrating information from diverse sources, CSKGs offer significant benefits [3]: they provide analysts with actionable cyber threat intelligence, enable high-level cyber situational awareness, and assist in uncovering new insights. Additionally, KGs enhance visualization of networks, data flows, and attack paths while clarifying data correlations, making them indispensable tools for describing and understanding complex attack processes.

In this paper, we present our PhD thesis dedicated to the construction of CSKG to encode the knowledge of cyber attack kill chains embedded within unstructured CTI reports. We further explore two key directions for leveraging the extracted

KG to enhance security analysis. The first one aims to facilitate the triaging of newly disclosed cyber attacks in CTI reports. The second explores predicting cyber attack behaviors using the structured knowledge representation offered by the CSKG.

II. Cyber Attack Knowledge Graph Construction

The task of encoding security knowledge contained in CTI reports into a graph-structured representation of attack process has recently gained significant research attention, yielding commendable results. To summarize and compare the methods employed in previous research works, we categorize the process into the following 4 main successive steps:

• **Parsing of CTI reports (Step 1).** CTI reports available in the market vary significantly in the level of detail provided in their attack descriptions. Some reports offer comprehensive insights into the entire attack process, explicitly detailing the attacker's actions, while others present only a high-level overview of the threat scenario. Consequently, the size of these documents can differ substantially. The content types within the reports also vary, encompassing text, images, and graphics. As a result, the analysis can either consider the entire document or focus on specific sections. For textual content, the analysis can be performed at different levels of granularity—such as sentences, paragraphs, or sections—depending on the requirements. The chosen segmentation approach should maintain the document's semantic structure while adhering to the capacity constraints of the analysis tools used to build the knowledge graph, such as the context window size of Large Language Models (LLMs). Furthermore, since CTI reports are often published in formats like PDF or HTML, it is also necessary to extract the text from these reports into a more accessible format, such as raw text or Markdown, to enable direct processing by programming languages.

• **Extracting security entities (Step 2)**. A graph is an ideal structure for organizing multiple attack descriptions by defining a set of security entities and relationships. In the textual contents of CTI reports, entities related to the described attack process, such as attack technologies, are represented as nodes. In addition, the relations between these entities are identified from the texts as the edges between the nodes. Specifically, this process includes two Natural Language Processing (NLP) tasks, *Named Entity Recognition* (NER) and *Relation Extraction* (RE). The created entities (nodes) should encompass comprehensively key factors in the attack process, including the attack actors, victims, timestamps, geolocations, exploited vulnerabilities, malware used, hardware/software configurations, and indicators of compromise. The definition of the entities provides explanations to the attack process with two different granularity levels. A high-level explanation would be, for example, one that relates different attackers groups to some tactics, techniques and procedures (TTP) referenced in the MITRE ATT&CK matrix [5]. The other explanation level is to take into account malware deployed by attackers, IP addresses/URLs connected by compromised devices, exploitation tools used by attackers, files dropped during the attack and so on. Furthermore, [6], [7] adopts a different entity definition structuring any sentence in CTI reports as a triple {subject-action-object}. Subjects and objects become the nodes of the graph. Nodes are therefore not typed according to their cyber security categories but depend on the writing style used by the report's author.

• **Extracting relations between entities (Step 3)**. Edges in a CSKG indicate the relationships between the security entities extracted from the attack descriptions (nodes identified in step 2 such as attackers, targeted systems, exploited vulnerabilities and deployed attack strategies). The edges are crucial for mining attack behavioral patterns from the constructed KG. For example, if we consider the sentence 'The trader downloaded the malware KPL on the website klas.bu' then a relationship exists between the nodes "Trader" and "Malware KPL" and, if a directed edge is created, it could be labeled "download". This information can be stored in the graph by simply adding an attribute to each edge. In ATTACKG [4], however, the representation of relationships is less precise. In their KG, an edge between two entities is just an oriented arrow representing a dependency between these entities, without any textual description of the dependency itself.

• **Graph construction and merging (Step 4)**. The ultimate objective is to build an unified KG that integrates information extracted from various CTI reports. When analyzing reports on similar topics, the extracted data often references common entities. At first glance, merging nodes and edges extracted from different CTI reports appears straightforward. However, the same entities are frequently described using varying terminologies across different reports. Merging these entities, despite varied terminologies, is essential to maintaining the uniqueness of each entity within the knowledge graph. For instance, if nodes are labeled as 'DarkMe malware', 'DarkMe RAT', and 'DarkMe RAT malware', they must be consolidated into a single node. This merging process occurs at two levels: First, merging is performed *with subgraphs derived from the same CTI report.* Indeed, a CTI report is divided into segments of texts. Entities and relationships are extracted from each chunk to generate a subgraph per chunk. Nodes and edges from these subgraphs have to be merged. Additionally, merging is also carried out *across graphs generated from different reports* to ultimately produce an unified CSKG covering attack process descriptions across multiple CTI reports. Various algorithms and tools have been proposed to perform this merging process.

III. Related Work and Our Approach

We focus on the state-of-the-art approaches using Machine Learning techniques to build cyber attack knowledge graphs, ATTACKG [4], ATTACKG+ [6], CTIKG [7] and LLM-TIKG [8]. We summarize their contributions in the 4 steps of knowledge graph construction, as defined in the previous section. For each step, we provide insights into our own strategy.

• **Step 1**. The works [4], [6]–[8] use the same technique at this step. They extract the text from the reports using the mainstream OCR (Optical Character Recognition) Python libraries as *pdfplumber*, *html2text*, *ioc-parser* and *corefee* for

ATTACKG, *pdfplumber* for ATTACKG+. The OCR tools used in CTIKG and LLM-TIKG are not specified. The only difference is that ATTACKG use regexes to identify IoC and replace them in the text with a commonly used word according to their type to avoid misunderstanding from NLP tools they used. They also take note of the location of the IoC in the text before making the change. In addition, LLM-TIKG is specially tailored for extracting the descriptions of API attacks. It first analyses the first page of each report and proceeds to the full text analysis only if the text on the first page is related to Advanced Persistent Threat (APT) attacks.

Our approach (1): Inspired by these works, we plan to use the PyMuPDF library [9] to extract the text from the reports in the Markdown format in order to differentiate between titles (that will be ignored) and contents. The contents will be divided into text segments to fit the context window of LLMs. We will prompt LLMs to extract entities and relations from each text segment. Furthermore, we plan to extract the images and graphics contained in these documents. LLMs will be used to explain them according to the context of the report. We will filter the extracted data to keep only those strongly related to the described attack scenario. Thus, we aim to achieve more comprehensive coverage of all the relevant descriptions contained in a CTI report, compared to the previous methods.

• **Step 2**. Previous works present two main approaches for entity extraction. ATTACKG and LLM-TIKG define a list of entities related to cyber attacks as nodes in the constructed CSKG. In contrast, ATTACKG+ and CTIKG extract the subjects and objects of each sentence from the textual content of CTI reports to serve as candidate nodes in the CSKG.

The set of security entities defined in ATTACKG and LLM-TIKG is constituted of: Malware, Threat type, Attacker, Technique, Tool, Vulnerability, IP address, Domain, URL, File, Hash, Executable and Registry. ATTACKG scans the text sentence by sentence and then use the pre-trained pipeline *en_core_web_sm* [10] of *Spacy* library and regulars expressions to identify the matched strings in the texts to the set of cyber entities types. LLM-TIKG improves the entity recognition accuracy by conducting fine-tuning of a Qwen [11] LLM model, instead of using a rule-based named entity recognition algorithm. In LLM-TIKG, the Qwen model is tuned with 1762 manually tagged entities. The goal of the fine-tuning step is to make LLMs better recognize the pre-defined security entities in the text contents of CTI reports. Thanks to the flexibility of LLM in recognizing entities given semantic contexts, LLM-TIKG reports superior accuracy in identifying attack related entities than ATTACKG. Given the limitation of the input context window size in the LLM model, LLM-TIKG is constrained to take one short text segment each time. Similarly, CTIKG and ATTACKG+ divides the text contents by paragraphs. Each paragraph has a size less than to the context window limitation of the LLMs such as Llama and GPTs used in their works.

Our approach (2): First, we propose a set of entities describing the attack process with different granularity levels. At the high level, we propose to include attack actors, target host, the geolocation and timestamp of the attack, the professions/domains of the victim, the exploited vulnerabilities (including the CVE/CWE indexes) and the mentioned TTP strategies. At a more fine-grained level, we plan to include file paths, registry key modification, urls, hashes of dropped and/or executed files, created/injected processes and threads, code injected/executed in the attack, deployed malware families and IP addresses of C2 servers. We plan to use Mistral and Llama LLMs to extract these entities from both text contents and images/graphics in the reports. Currently, LLMs are the most effective tools for this task. Modern models such as GPT-4, Llama [12], and Mistral Large [13] offer context windows of up to 128,000 tokens (words or word sets) and are equipped with approximately 70 billion parameters, enabling them to comprehend the intricacies of complex documents. In 2024, researchers demonstrated the effectiveness of LLMs for the NER task on CTI reports, publishing a benchmark [14].

• **Step 3**. In [4], [6]–[8], the edges represent the relationships between the security entities extracted in Step 2. Action verbs are used : *use*, *deploy*, *download*, *modify*, *store* and more.

Our approach (3): We will perform relation extraction with LLMs similarly to ATTACKG+, LLM-TIKG, and CTIKG.

• **Step 4**. ATTACKG [4] pioneered the construction of attack graphs that summarize all attack techniques mentioned in Cyber Threat Intelligence (CTI) reports. Their approach involved creating individual graphs for each sentence in a report and subsequently merging these graphs. Nodes of the same type were consolidated based on their semantic meaning, as determined using the Spacy Python library. In the second phase, the focus shifted to constructing a technique graph derived from the MITRE ATT&CK matrix [5], which provides detailed intelligence at the technique level. Ultimately, the two graphs—representing technical and practical intelligence—were merged into a comprehensive graph. However, ATTACKG did not propose a mechanism for merging edges. Similarly, LLM-TIKG splits the text into chunks and generate a graph for each chunk. They merged nodes of the same type based on semantic meaning gave by a Llama model. They employed the HDBScan algorithm [15] to cluster edges and attach them to corresponding clusters. ATTACKG+ and CTIKG treated the subjects and objects of sentences as graph nodes and the relations between them as edges. These approaches used a Llama model to derive node definitions, enabling the merging of nodes with identical meanings, and a RoBERTa [16] model to generate node embeddings, allowing the merging of nodes with similar embeddings. Note that ATTACKG+ maps each triple extracted from CTI reports to a corresponding label in the MITRE ATT&CK matrix, thereby enriching the contextual description of attack processes.

Our approach (4): For the CSKG construction, we plan to define merging rules for entities (nodes) and relations (edges). Between *Step 1* and *Step 2*, we will map each report in our dataset to some specific tactics in the MITRE ATT&CK matrix to specify these merging rules. In *Step 4*, we will generate individual graph for each report and merge them. To facilitate this merging process, we will employ two new models: the first

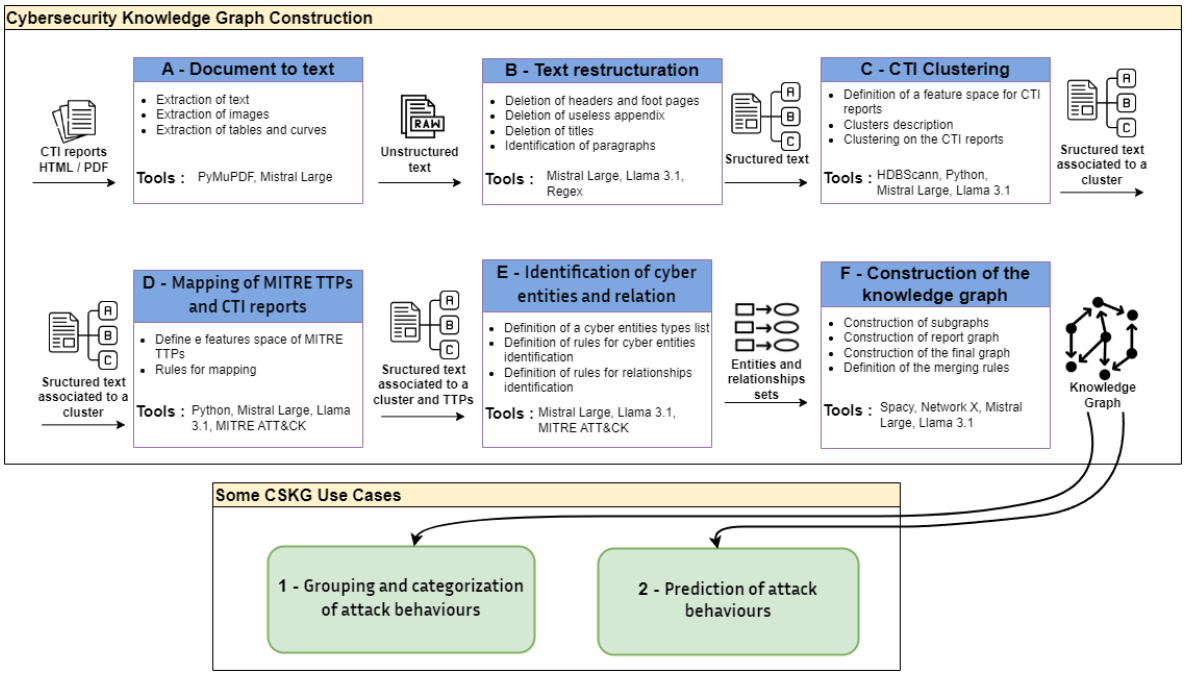

Fig. 1. Pipeline of Knowledge Graph construction and some use cases.

one will create a definition for each node based of its name and the names of its five nearest neighbors, while the second will assess the similarity of these definitions and merge nodes if appropriate. Each node in the final graph will be tagged with one or more tactics from the MITRE ATT&CK matrix, enabling us to categorize nodes frequently involved in specific attack processes.

IV. Our proposed workflow for KG construction

Figure 1 describes our pipeline (composed of 6 phases) for a LLM-driven CKSG construction. Phases A and B align with the parsing step (Step 1). Phase E corresponds to Step 2 and 3. Phase F represents the CSKG construction step (Step 4). The tools planned for use, along with the new actions to be undertaken, are listed in this figure. We enhance the workflow with 2 additional phases between Step 1 and Step 2: CTI clustering (Phase C) and MITRE TTP mapping (Phase D). CTI clustering will be used to group the cyber attacks descriptions in the reports. Reports regarding similar semantic themes, e.g. attack goals and exploited vulnerabilities, are grouped together to leverage variations in the attack description. MITRE tactics mapping matches the attack description texts in the reports to the TTP strategies listed in MITRE ATT&CK knowledge base, enriching the security entities in CSKG. For experiments, we use a GitHub repository [17] containing around 1500 CTI reports from 2006 to 2024 in PDF format. With this workflow, the graph will be designed to enable two future use cases:

Grouping and categorizing attack behaviors. Today, the MITRE ATT&CK matrix allows for the description of attack behaviors in a structured way according to the attack process. Our graph will enable us to differentiate threat actions according to the attack process, but also according to computing processes and social/cultural factors such as nationality and location. For instance, we will be able to identify the locations most targeted by phishing campaigns, or the tools most frequently used by a given group of attackers.

Graph-enhanced attack behavioral prediction. Certain types of nodes in the CSKG can serve as indicators to monitor a computer architecture. If there are already similarities between concrete nodes in our graph and observed elements in the architecture, potential threat actions can be predicted. For example, if a cybersecurity operator identifies a suspicious file with the same hash as another file listed in an attack campaign in our CSKG, we are able to predict the attack that the system is undergoing, the group of attackers responsible and the possible next threat actions in the attack.

V. Conclusion

We laid the groundwork for a robust approach to constructing KGs from CTI reports. The objective is to provide cybersecurity analysts with a powerful framework for understanding and mitigating cyber threats. By leveraging Large Language Models (LLMs), we have proposed an automated pipeline that aims to parse CTI reports, extract relevant nodes and edges, and construct coherent KGs. This pipeline is designed to handle the diversity in report structures and terminologies, ensuring that the extracted information is both accurate and meaningful. While this paper focuses on the initial stages of KG construction, the potential uses of these graphs in cybersecurity analysis are vast. Future work will explore these applications in greater detail, further demonstrating the value of structured knowledge in the cybersecurity domain.

REFERENCES

[1] R. Brown and K. Nickels. Sans 2023 cti survey: Keeping up with a changing threat landscape. https://www.sans.org/white-papers/2023-cti-survey-keeping-up-changing-threat-landscape, Last accessed on Feb. 07 2025.

[2] Cyber Threat Intelligence Technical Committee. Introduction to stix, 2024. https://oasis-open.github.io/cti-documentation/stix/intro.html, Last accessed on Feb. 07 2025.

[3] Leslie Siko. Cybersecurity knowledge graphs. In *Knowledge Information System*, volume 65, pages 3511–3531, 2023.

[4] Zhenyuan Li, Jun Zeng, Yan Chen, and Zhenkai Liang. Attackg: Constructing technique knowledge graph from cyber threat intelligence reports. In *Computer Security – ESORICS 2022: 27th European Symposium on Research in Computer Security, Copenhagen, Denmark, September 26–30, 2022, Proceedings, Part I*, page 589–609, Berlin, Heidelberg, 2022. Springer-Verlag.

[5] MITRE Corporation. Mitre att&ck framework, 2025. https://attack.mitre.org/, Last accessed on Feb. 07 2025.

[6] Yongheng Zhang, Tingwen Du, Yunshan Ma, Xiang Wang, Yi Xie, Guozheng Yang, Yuliang Lu, and Ee-Chien Chang. Attackg+:boosting attack knowledge graph construction with large language models, 2024.

[7] Liangyi Huang and Xusheng Xiao. CTIKG: LLM-powered knowledge graph construction from cyber threat intelligence. In *First Conference on Language Modeling*, 2024.

[8] Yuelin Hu, Futai Zou, Jiajia Han, Xin Sun, and Yilei Wang. Llm-tikg: Threat intelligence knowledge graph construction utilizing large language model. *Computers & Security*, 145:103999, 2024.

[9] Pymupdf. https://pymupdf.readthedocs.io/en/latest/, Last accessed on Feb. 07 2025.

[10] English pipeline optimized for cpu. https://github.com/explosion/spacy-models/releases/tag/en_core_web_sm-3.1.0, Last accessed on Feb. 07 2025.

[11] Ctiqwen. https://huggingface.co/revealcti/cti-qwen1.5-70b-awq, Last accessed on Feb. 07 2025.

[12] Llama 3.1. https://llama.com/docs/model-cards-and-prompt-formats/llama3_1/, Last accessed on Feb. 07 2025.

[13] Mistral models. https://mistral.ai/fr/technology/#models, Last accessed on Feb. 07 2025.

[14] Md Tanvirul Alam, Dipkamal Bhusal, Le Nguyen, and Nidhi Rastogi. Ctibench: A benchmark for evaluating llms in cyber threat intelligence, 2024. https://arxiv.org/abs/2406.07599, Last accessed on Feb. 07 2025.

[15] How hdbscan works. https://hdbscan.readthedocs.io/en/latest/how_hdbscan_works.html, Last accessed on Feb. 07 2025.

[16] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized bert pretraining approach, 2019.

[17] CyberMonitor. Apt & cybercriminals campaign collection. https://github.com/CyberMonitor/APT_CyberCriminal_Campagin_Collections/tree/master, Last accessed on Feb. 07 2025.