A Knowledge Base Question Answering System for Cyber Threat Knowledge Acquisition

Zhengjie Ji
KTH Royal Institute of Technology
zj@kth.se

Edward Choi
University of California, Berkeley
edwardc1028@berkeley.edu

Peng Gao
Virginia Tech
penggao@vt.edu

2022 IEEE 38th International Conference on Data Engineering (ICDE) | 978-1-6654-0883-7/22/$31.00 ©2022 IEEE | DOI: 10.1109/ICDE53745.2022.00287

***Abstract*—Open-source cyber threat intelligence (OSCTI) provides a form of evidence-based knowledge about cyber threats, enabling businesses to gain visibility into the fast-evolving threat landscape. Despite the pressing need for high-fidelity threat knowledge, existing cyber threat knowledge acquisition systems have primarily focused on providing low-level, isolated indicators. These systems have ignored the rich higher-level threat knowledge entities and their relationships presented in OSCTI reports, and do not provide a flexible and intuitive way for threat analysts to acquire the desired knowledge. To bridge the gap, we propose THREATQA, a system that facilitates cyber threat knowledge acquisition via knowledge base question answering. Particularly, THREATQA uses a combination of AI-based techniques to (1) automatically harvest comprehensive knowledge about trending threats from massive OSCTI reports from various sources and construct a large threat knowledge base, and (2) intelligently respond to an input natural language threat knowledge acquisition question by fetching the answer from the threat knowledge base via question answering.**

I. INTRODUCTION

Sophisticated cyber attacks have plagued many high-profile businesses [1]. To gain visibility into the fast-evolving threat landscape, open-source cyber threat intelligence (OSCTI) [2] has emerged as an important source of threat knowledge and has received growing attention. OSCTI provides a form of evidence-based knowledge about threats in a number of reports in various forms (e.g., threat reports, security articles, security news [3], [4]). Despite the pressing need for high-quality threat knowledge, existing threat knowledge acquisition systems [5]–[7] have primarily focused on providing simple Indicators of Compromise (IOCs, e.g., signatures of artifacts, malicious file names, IP addresses) extracted from OSCTI reports [8]. Though able to provide low-level, isolated indicators, these systems have ignored (i) higher-level entities (e.g., adversary tactics, techniques, and procedures (TTPs) [9]) that are tied to the attacker's goals and thus are much harder to change, and (ii) semantic relationships between indicators that are critical to uncovering the complete, multi-step threat scenario. As the volume of OSCTI sources increases day-by-day, it becomes increasingly challenging for enterprise threat analysts to manually maneuver through and correlate the myriad of sources (e.g., tons of threat reports) to gain useful knowledge. Towards this end, there is a pressing need for a new system that can harvest and supply high-fidelity threat knowledge in an intelligent, efficient, and principled way.

To leverage the rich threat knowledge provided by OSCTI, we envision that the system maintains a *threat knowledge base*, which automatically harvests and manages high-fidelity threat knowledge from massive OSCTI reports. Furthermore, to release threat analysts from the burden of manual and tedious knowledge acquisition, the system needs to provide knowledge to the user in a flexible and intuitive manner. Database management systems provide a way to search through the data via issuing queries (e.g., SQL, Cypher). However, threat analysts who are not familiar with the query syntaxes might experience a steep learning curve, and the manual query construction process is labor-intensive and error-prone. To enable flexible and intuitive knowledge acquisition, we envision an *intelligent threat knowledge acquisition process through question answering (QA)*: the user asks natural-language questions on desired attributes of threats, and the system returns the answer fetched from the underlying threat knowledge base.

There are two major challenges for building such a system. First, to comprehensively model the threats in the threat knowledge base, the system needs to be able to extract knowledge from massive OSCTI reports and cover a wide range of threat-related entities and relations. However, OSCTI reports come in diverse formats; some reports contain structured fields such as lists and tables, and some reports primarily consist of unstructured natural-language texts. Second, accurately extracting threat knowledge from unstructured OSCTI texts is non-trivial, due to the presence of massive nuances particular to the security context (e.g., special characters like dots and underscores in IOCs). Second, to train AI-based QA models, a large dataset that contains a diverse set of question-answer pairs is needed. However, there is no existing QA dataset available for the cyber threat knowledge acquisition domain.

To bridge the gap, we build THREATQA, an AI-based system for cyber threat knowledge acquisition via question answering. THREATQA takes as as input a natural-language question on attributes of certain threats, and returns the answer by fetching the corresponding knowledge from the underlying threat knowledge base. To harvest and store threat knowledge, THREATQA was built upon our prior efforts on automated construction of a threat knowledge graph from massive OSCTI reports [10], which covers a wide range of threat-related entities and relations. On top of the threat knowledge base, THREATQA adopts a *three-stage, AI-based pipeline* for knowledge base question answering (KBQA). In

2375-026X/22/$31.00 ©2022 IEEE
DOI 10.1109/ICDE53745.2022.00287

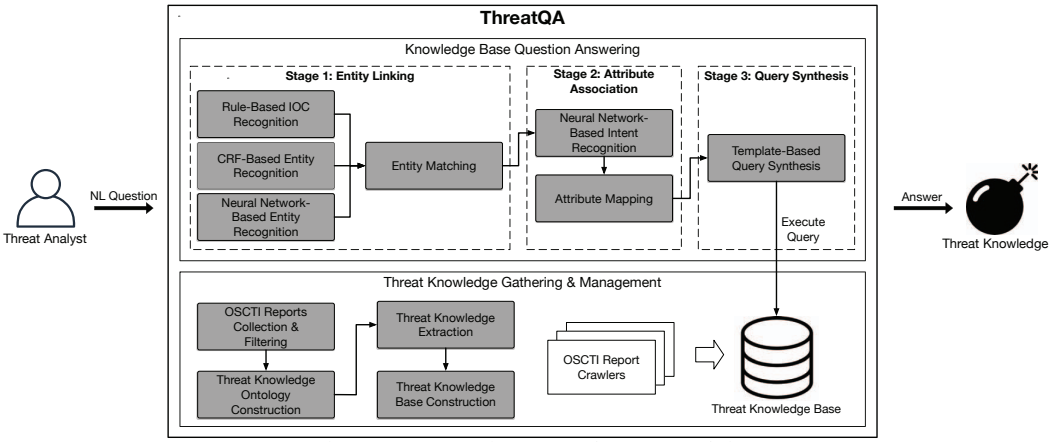

Fig. 1: The architecture of THREATQA

Stage 1, THREATQA performs entity linking by recognizing entity mention in the input question and linking it to the entity in the threat knowledge base. In Stage 2, THREATQA performs attribute association by first recognizing the intent of the question and then mapping the intent to the asked entity attribute/relation in the threat knowledge base. In Stage 3, THREATQA performs template-based query synthesis to synthesize a database query using the recognized entity information and the mapped attribute information. The synthesized query is then executed over the threat knowledge base to retrieve the answer. In addition, THREATQA provides a web UI for constructing questions and getting answers.

To the best of our knowledge, THREATQA is *the first system that facilitates cyber threat knowledge acquisition via KBQA*.

Demo video: https://youtu.be/lNAZYgjm7xE

II. THREATQA ARCHITECTURE

Figure 1 shows the system architecture. THREATQA collects OSCTI reports from various sources, uses AI-based techniques to extract high-fidelity threat knowledge as entities and relations based on a threat knowledge ontology, and constructs a threat knowledge base containing the entity-relation triplets. On top of the threat knowledge base, the QA pipeline consists of three stages: (1) recognizing the entity mention in the input question and linking it to the entity in the threat knowledge base; (2) recognizing the intent of the question and mapping it to the asked attribute/relation of the linked entity in the threat knowledge base; (3) synthesizing a database query by filling in a query template and executing the synthesized query over the threat knowledge base to retrieve the answer.

A. Threat Knowledge Base Construction

THREATQA leverages our prior work, SECURITYKG, for automated construction of a threat knowledge base (graph) [10]. Briefly speaking, SECURITYKG maintains robust and multi-threaded crawlers for 40+ major security websites (covering threat encyclopedias [3], security blogs [4], etc.). In addition, SECURITYKG maintains a set of (i) checkers to screen out OSCTI reports that are irrelevant to cyber threats, (ii) source-dependent parsers to handle the diverse formats of OSCTI reports and parse structured fields, and (iii) source-independent extractors to extract entities and relations from unstructured texts. Specifically, SECURITYKG uses regular expression rules to extract IOCs and Conditional Random Field (CRF) [11] model to extract other types of entities, and a dependency parsing-based NLP pipeline proposed in our other work [12] to extract relations between entities.

The threat knowledge base was constructed from 140K+ OSCTI reports and covers a wide range of entities, including (1) IOCs (e.g., file name, file path, IP, URL, domain, registry, hashes), (2) software products (e.g., `Microsoft Outlook`), (3) security-related tools (e.g., `Mimikatz`), (4) malware (e.g., `WannaCry`), (5) vulnerabilities (e.g., CVEs), (6) threat actors (e.g., `CozyDuke`), (7) tactics and techniques (e.g., `spearphishing link`), and (8) report-specific information (e.g., title, author). To persist the knowledge, the system currently uses Neo4j database for its storage, with nodes being entities and edges being relations. In total, the knowledge base contains 339,601 nodes and 170,8702 edges.

B. Question Generation for QA Dataset Construction

Natural-language questions can generally be divided into the following categories: factual questions, yes-no questions, contrastive questions, inferential questions, and opinion questions [13]. Among them, factual questions are the basis of other questions. Answers to many questions of other forms, such as yes-no questions and multiple choices, can be derived from answers to factual questions. Thus, THREATQA currently mainly focuses on factual questions. Specifically, we adopt a *template-based* approach to construct a large dataset of question-answer pairs. This approach allows us to construct a large dataset quickly while maintaining high quality, and improve the dataset by revising the templates.

First, we define a set of question intents based on the information (entities, attributes, relations) in the threat knowledge base. The question intents encode the relationships between the asked entity attributes/relations in the questions and the answers, and serve as the basis for constructing seed question templates later. The question intents also indicate the sentence structures of the questions, which will help us perform attribute

association in the QA pipeline later. After defining question intents, we construct seed question templates for all entity categories (e.g., malware) and entity attribute types (e.g., risk level of malware) in the threat knowledge base. Each seed question template corresponds to one question intent, but each question intent may correspond to multiple seed question templates that query the same information. Each seed question template contains at least one entity placeholder. Multi-hop question templates contain two or more entity placeholders.

Once the seed question templates are constructed, more question templates can be generated to increase variability in the QA dataset. Specifically, we use Parrot [14], a transformer-based utterance augmentation framework to paraphrase the seed question templates. The framework takes a seed question template as input and outputs several paraphrased questions with the same meaning. The generated question templates are semantically equivalent to and have the same question intent as the original seed question template. With the generated question templates, we can generate complete questions by traversing the entities in the threat knowledge base and replacing the placeholders in the question templates with the entity names. The corresponding attribute values are the answers. Our dataset in total contains 1M samples of 139 diferent question templates. Each sample contains the following information: (1) question; (2) location and type of the entities; (3) question intent. The train/validation/test split ratio is 8:1:1.

C. Entity Linking

This stage identifies the entities in the question and matches them with the entities in the threat knowledge base.

First, THREATQA leverages rule-based patterns to extract IOCs in the question. Then, THREATQA leverages a Conditional Random Field (CRF) [11] model and a neural network model (RoBERTa [15]) to identify the category information and the location of other types of entities in the question. The CRF model is designed for recognizing threat actors, malware, tactics and techniques, etc., which was trained using the MITRE ATT&CK [9] information and proposed in our prior work [10]. The neural network model is used to identify other types of entities (e.g., software products), and was trained on our QA dataset for the named entity recognition task. The hybrid entity recognition method allows THREATQA to have high entity recognition coverage and accuracy.

After identifying the entities in the question and their category information, for each entity-in-question, we traverse the entities of the corresponding category (for pruning the search) in the threat knowledge base and calculate their similarity (we currently use Jaccard similarity, but other similarity metrics are applicable). The entity-in-question is matched to the entity-in-knowledge-base with the highest similarity. A similarity value less than 0.5 indicates that the entity-in-question does not appear in the knowledge base, and the match will be rejected.

D. Attribute Association

In the attribute association stage, THREATQA identifies the intent of the input question and maps the intent to the entity attribute type in the threat knowledge base. For intent recognition, we model it as a multi-class classification problem and trained a neural network-based intent classifier using our QA dataset. The mapped attribute type represents the subgraph structure of the threat knowledge base and is used to retrieve the answer (i.e., attribute value). Specifically, for each entity attribute type, we design a Cypher query template, which encodes the path between the entity in the threat knowledge base and the target answer. Following are an example question template, its corresponding question intent, and the corresponding Cypher query template:

Question template:
What is the risk level of malware {MALWARE_NAME}?

Question intent:
Malware.risk_level

Cypher query template:
MATCH (var1:Malware) WHERE var1.name ="{MALWARE_NAME}" RETURN var1.risk_level

E. Query Synthesis

For an input question, once the contained entity and question intent are recognized, such information can then be used to fill in the corresponding Cypher query template to synthesize a query for execution. A Cypher query template may contain a single constraint in the WHERE clause, or a combination of constraints combined with logical operations. Such template-based approach ensures that the synthesized query is grammatically correct and thus has high reliability, which is particularly important for security-related purposes. By executing the query, THREATQA then obtains the final answer from the threat knowledge base.

F. Frontend UI Design

To facilitate the user (e.g., threat analysts) to interact with THREATQA, we built a web UI using React and hosted the system on a Flask server. After the user enters a question in the input box, a request is sent to the server, which runs the THREATQA pipeline. The server then sends the QA results back to the UI, which are then displayed to the user. The UI displays the details for intent classification, named entity recognition, entity linking, synthesized Cypher query, and final answer. For named entity recognition, the original question is displayed, with recognized entity mentions colored in red together with the entity types as subscript texts. For the synthesized query, the user has the option to modify it to query additional information from the threat knowledge base.

III. DEMONSTRATION OUTLINE

In our demonstration, we aim to show the complete usage scenario of THREATQA by inputting different questions through its web UI and checking the displayed results.

Question 1. "What is the type of Downloader.Slime?" Here we demonstrate the basic pipeline of the system. After the user enters the question, the UI displays the results of each QA processing stage. Specifically, the UI displays: (1) the

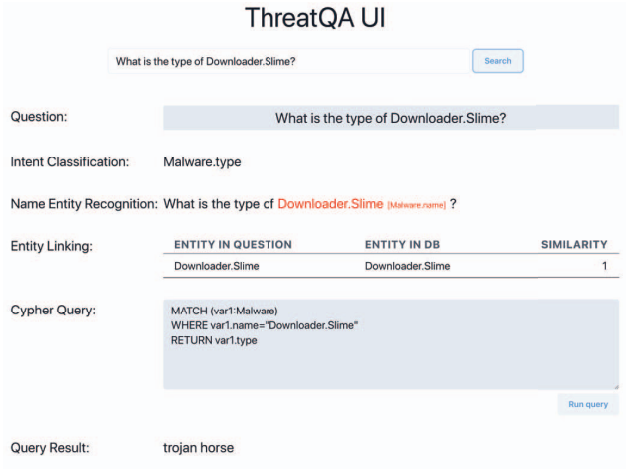

Fig. 2: The web UI of THREATQA

question's intent, with the recognized entities in the question and their categories colored in red; (2) the entity linking result and the similarity value for each candidate; (3) the synthesized Cypher query and the final answer after query execution.

Question 2. "What is the risk level of W32.Wullik.Bamm?" Details on the entity linking stage are highlighted. Jaccard similarity is calculated between the entity in the question and the entity in the knowledge base. The system accepts this match as the similarity score is greater than 0.5.

Question 3. "What is the variant of Infostealer.Lemir.F?" Through this question, we show how to edit the synthesized Cypher query. The original question asks about the variant of malware. By replacing the attribute type "variant" with other attribute types (e.g., "discovered_date") in the Cypher query, the user can obtain more information about the malware.

Question 4. "What is the detailed behavior of Backdoor. Lassrv?" Through this question, we show the QA pipeline of a question that asks about the specific behaviors of the malware. The returned answer consists of several sentences.

Question 5. "Which article reports the fancy bear and the lazarus group?" Through this question, we show the QA pipeline for a multi-hop question. This question asks about the relationship between two entities. Through entity recognition, the system recognizes the two threat actor entities in the question. The intent classification component shows that the question aims at finding the OSCTI reports that include both threat actor entities. After matching the entities in the question to the entities in the threat knowledge base, the system fills the two entities in the corresponding question template, synthesizes a question that contains a multi-hop search path for the MATCH clause, and returns the final answer.

Our demo video gives a complete walk through of these questions and features. To facilitate virtual demonstrations, the audience can remotely access THREATQA's web UI using a browser to input questions. Furthermore, a virtual machine with a functional instance of the system and the correct environment configured will be provided. The audience can download the virtual machine and run THREATQA locally.

IV. RELATED WORK

Besides threat knowledge acquisition systems based on OSCTI [5]–[7], research progress has been made to better analyze OSCTI reports, including extracting IOCs [8], understanding vulnerability reproducibility [16], and measuring threat intelligence quality [2]. THREATQA distinguishes from all these works in the sense that it targets facilitating flexible and intuitive cyber threat knowledge acquisition via knowledge base question answering. In future work, we plan to connect THREATQA to our system auditing-based threat protection systems [12], [17]–[19] to enable intelligent and knowledge-aware threat protection.

V. CONCLUSION

We have presented THREATQA, a novel system that facilitates cyber threat knowledge acquisition via KBQA.

Acknowledgement. This work was supported in part by Cisco and the Commonwealth Cyber Initiative (CCI), an investment in the advancement of cyber R&D, innovation, and workforce development.

REFERENCES

[1] "The 15 biggest data breaches of the 21st century," https://www.csoonline.com/article/2130877/the-biggest-data-breaches-of-the-21st-century.html.
[2] V. G. Li, M. Dunn, P. Pearce, D. McCoy, G. M. Voelker, and S. Savage, "Reading the tea leaves: A comparative analysis of threat intelligence," in *USENIX Security*, 2019.
[3] "Trend micro threat encyclopedia," https://www.trendmicro.com/vinfo/us/threat-encyclopedia/.
[4] "SecureList," https://securelist.com/.
[5] "ThreatMiner," https://www.threatminer.org/.
[6] "ThreatCrowd," https://www.threatcrowd.org/.
[7] "Alienvault otx," https://otx.alienvault.com/.
[8] X. Liao, K. Yuan, X. Wang, Z. Li, L. Xing, and R. Beyah, "Acing the ioc game: Toward automatic discovery and analysis of open-source cyber threat intelligence," in *CCS*, 2016.
[9] "Mitre att&ck," https://attack.mitre.org.
[10] P. Gao, X. Liu, E. Choi, B. Soman, C. Mishra, K. Farris, and D. Song, "A system for automated open-source threat intelligence gathering and management," in *SIGMOD*, 2021, demonstrations track.
[11] J. D. Lafferty, A. McCallum, and F. C. N. Pereira, "Conditional random fields: probabilistic models for segmenting and labeling sequence data," in *ICML*, 2001.
[12] P. Gao, F. Shao, X. Liu, X. Xiao, Z. Qin, F. Xu, P. Mittal, S. R. Kulkarni, and D. Song, "Enabling efficient cyber threat hunting with cyber threat intelligence," in *ICDE*, 2021.
[13] D. Jurafsky, *Speech & language processing*. Pearson Education India, 2000.
[14] "Parrot: Paraphrase generation for nlu." https://github.com/PrithivirajDamodaran/Parrot_Paraphraser.
[15] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov, "Roberta: A robustly optimized bert pretraining approach," *arXiv preprint arXiv:1907.11692*, 2019.
[16] D. Mu, A. Cuevas, L. Yang, H. Hu, X. Xing, B. Mao, and G. Wang, "Understanding the reproducibility of crowd-reported security vulnerabilities," in *USENIX Security*, 2018.
[17] P. Gao, X. Xiao, Z. Li, F. Xu, S. R. Kulkarni, and P. Mittal, "AIQL: Enabling efficient attack investigation from system monitoring data," in *USENIX ATC*, 2018.
[18] P. Gao, X. Xiao, D. Li, Z. Li, K. Jee, Z. Wu, C. H. Kim, S. R. Kulkarni, and P. Mittal, "SAQL: A stream-based query system for real-time abnormal system behavior detection," in *Proceedings of the 27th USENIX Conference on Security Symposium*, 2018, pp. 639–656.
[19] P. Fang, P. Gao, C. Liu, E. Ayday, K. Jee, T. Wang, Y. F. Ye, Z. Liu, and X. Xiao, "Back-propagating system dependency impact for attack investigation," in *USENIX Security*, 2022.