AGIR: Automating Cyber Threat Intelligence Reporting with Natural Language Generation

Perrina, Filippo ; Marchiori, Francesco; Conti, Mauro; Verde, Nino Vincenzo

DOI
10.1109/BigData59044.2023.10386116

Publication date
2023

Document Version
Final published version

Published in
Proceedings of the 2023 IEEE International Conference on Big Data (BigData)

AGIR: Automating Cyber Threat Intelligence Reporting with Natural Language Generation

Filippo Perrina*, Francesco Marchiori*, Mauro Conti*†, Nino Vincenzo Verde‡
*Department of Mathematics
University of Padova, Padua, Italy
filippo.perrina@studenti.unipd.it, francesco.marchiori@math.unipd.it, mauro.conti@unipd.it
†Faculty of Electrical Engineering, Mathematics and Computer Science
Delft University of Technology, Delft, Netherlands
‡Leonardo Cyber&Security
Leonardo S.p.A., Rome, Italy
nino.verde@leonardo.com

2023 IEEE International Conference on Big Data (BigData) | 979-8-3503-2445-7/23/$31.00 ©2023 IEEE | DOI: 10.1109/BigData59044.2023.10386116

***Abstract*—Cyber Threat Intelligence (CTI) reporting is pivotal in contemporary risk management strategies. As the volume of CTI reports continues to surge, the demand for automated tools to streamline report generation becomes increasingly apparent. While Natural Language Processing techniques have shown potential in handling text data, they often struggle to address the complexity of diverse data sources and their intricate interrelationships. Moreover, established paradigms like STIX have emerged as de facto standards within the CTI community, emphasizing the formal categorization of entities and relations to facilitate consistent data sharing.**

In this paper, we introduce AGIR (Automatic Generation of Intelligence Reports), a transformative Natural Language Generation tool specifically designed to address the pressing challenges in the realm of CTI reporting. AGIR's primary objective is to empower security analysts by automating the labor-intensive task of generating comprehensive intelligence reports from formal representations of entity graphs. AGIR utilizes a two-stage pipeline by combining the advantages of template-based approaches and the capabilities of Large Language Models such as ChatGPT. We evaluate AGIR's report generation capabilities both quantitatively and qualitatively. The generated reports accurately convey information expressed through formal language, achieving a high recall value (0.99) without introducing hallucination. Furthermore, we compare the fluency and utility of the reports with state-of-the-art approaches, showing how AGIR achieves higher scores in terms of Syntactic Log-Odds Ratio (SLOR) and through questionnaires. By using our tool, we estimate that the report writing time is reduced by more than 40%, therefore streamlining the CTI production of any organization and contributing to the automation of several CTI tasks.

***Index Terms*—Cyber Threat Intelligence, Natural Language Generation, Threat Reports, STIX**

I. Introduction

The evolving cyber threat landscape has witnessed a dramatic surge in the frequency and sophistication of attacks in recent years. From traditional phishing emails to the stealthy and advanced operations of highly sophisticated cybercriminal groups known as Advanced Persistent Threats (APTs), the digital realm has become a battleground for organizations seeking to safeguard their data and infrastructure [1]. To counter these evolving threats effectively, organizations have begun implementing the discipline of Cyber Threat Intelligence (CTI) to address them proactively. CTI involves systematically collecting, analyzing, and disseminating data from diverse sources, including network logs, social media, and dark web forums, to identify, comprehend, and mitigate cyber threats. By providing actionable insights into the Tactics, Techniques, and Procedures (TTPs) employed by cybercriminals, as well as the associated Indicators Of Compromise (IOCs), CTI can enhance an organization's situational awareness and reinforce its ability to detect and respond to attacks swiftly, ultimately reducing overall risk exposure [2].

An integral part of CTI is the production of comprehensive security reports. Indeed, while several standards such as Structured Threat Information Expression (STIX) are employed to facilitate the sharing of structured data, natural language remains the most common and easily understandable format to collect and disseminate intelligence [3]. These reports are a repository of detailed information about cyber threats, encompassing TTPs, exploited vulnerabilities, and IOCs. Furthermore, they are vital in sharing CTI knowledge internally within organizations and externally with law enforcement agencies and other cybersecurity entities. Unfortunately, the manual creation of these reports can be an exceptionally time-consuming and resource-intensive task. Indeed, security analysts must aggregate and analyze extensive datasets before synthesizing their findings into clear and concise reports. Additionally, the accurate reconstruction of a specific incident or a threat might need the collaboration of several analysts and the congregation of multiple intelligence sources, further aggravating the complexity of the task [4]. To tackle this challenge, Natural Language Generation (NLG) techniques have emerged as a promising solution for automating the report generation process [5]. Natural Language Generation models are already deployed in many instances to facilitate the production of textual data, such as financial summaries [6], user tailoring and profiling in healthcare [7], and chatbots [8]. Thus, NLG tools can save security analysts substantial time

and resources by automating structured data conversion into well-crafted written content. However, despite the evident advantages NLG offers, applying such tools within the realm of cybersecurity, specifically for generating security reports, remains partially unexplored.

Contribution. This paper aims to address this gap in the cybersecurity literature by introducing **AGIR**, an NLG tool designed to automate the creation of cybersecurity reports from structured data. AGIR leverages STIX graphs, constituted by threat entities and their relationships, to parse them with a template-based approach and subsequently leverage a Large Language Model (LLM) to improve the fluency and utility of the generated report. While AGIR currently supports four distinct report types, its pipeline is designed with extensibility in mind, allowing for the incorporation of additional report formats as needed. The quantitative evaluation of our approach shows that the information in the structured data is conveyed in the generated reports with almost perfect recall values (0.99) without introducing any hallucination in the process. Furthermore, we qualitatively evaluate the outputs of our tool through the Syntactic Log-Odds Ratio (SLOR) metric and questionnaires answered by experienced cyber threat analysts. The results show that AGIR produces fluent reports that outclass state-of-the-art models while maintaining a high level of utility and reducing report writing time by 42.6%

Our contributions can be summarized as follows.

- We introduce **AGIR**, a Natural Language Generation (NLG) tool designed to automate the creation of cybersecurity reports from structured data.
- We propose a pipeline design aimed at enhancing the scalability of our tool and enabling the inclusion of a wider range of supported report types.
- We assess AGIR's performance through a combined quantitative and qualitative approach, utilizing several metrics and conducting surveys with expert threat analysts.
- We make several samples of the generated reports available at https://github.com/Mhackiori/AGIR.

Organization. The paper is organized as follows. In Section II, we introduce key CTI and NLG concepts, setting the stage for our discussion. In Section III, we explore the use of Natural Language Processing (NLP) and NLG techniques in the cybersecurity domain. Section IV gives an overview of the system model and the possible implementations of AGIR. Section V delves into the technical specification of AGIR, offering insights into its pipeline. In Section VI, we evaluate AGIR's quantitative and qualitative performance through human evaluation and SLOR metric. Finally, Section VII concludes this work.

II. Background

This section gives a more thorough background on the techniques and notions we use in the methodology. In particular, we focus on Cyber Threat Intelligence and the type of data that this discipline deals with (Section II-A) and the core concepts of Natural Language Generation (Section II-B).

A. Cyber Threat Intelligence

NIST, the National Institute of Standards and Technology, defines Cyber Threat Intelligence as "any information that can help an organization to identify, assess, monitor, and respond to cyber-threats" [9]. This intelligence empowers quicker, data-driven security decisions, shifting from a reactive to a proactive stance against threat actors. Threat intelligence sources encompass open source data, social media, device logs, internet traffic, and deep and dark web information. In today's cybersecurity landscape, it is pivotal in enabling organizations to proactively identify and mitigate potential threats, making them more resilient against cyberattacks.

Threat intelligence can be divided into three main categories: *strategic intelligence* (intended for non-technical audiences and offering high-level insights into threats and vulnerabilities), *tactical intelligence* (intended for technically prolific teams and providing immediate threat indicators), and *operational intelligence* (intended for professionals and offering in-depth insight into TTPs) [10]. Other sources instead divide intelligence into four categories, with the addition of *technical intelligence*, i.e., information that focuses on forensic intelligence and technical descriptions [11]. Additionally, each piece of intelligence undergoes a defined life cycle, from planning and direction to dissemination and integration [12].

Each of the intelligence types can be shared among organizations in various formats. While natural language reports are the most common medium when dealing with elaborate threats, structured data standards have been created to facilitate machine-readability and ease of dissemination. The most common standard in CTI is STIX (Structured Threat Information Expression), which includes several entities and relationships that allow for a graph representation of the intelligence [3]. Effectively, with STIX, intelligence is shared in JSON files that can be represented as a connected graph of nodes and edges, in which each node represents an entity and each edge represents a relationship between entities. A graphical overview of one of these graphs is shown in Figure 1. These JSON files constitute part of AGIR's input.

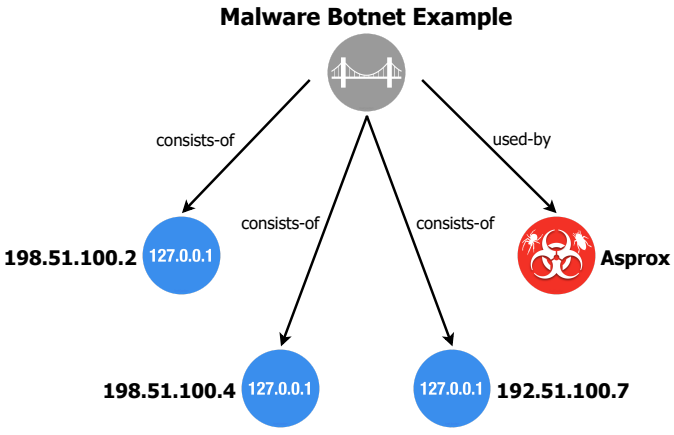

Fig. 1. Graphical example of a STIX graph. Icons used in this paper are from the 'stix2-graphics' repository by Bret Jordan [13].

B. Natural Language Generation

Natural Language Generation (NLG) is a subset of Natural Language Processing (NLP) centered on creating computer systems capable of generating human-like language output, such as documents and reports [5]. NLG can be categorized into text-to-text and data-to-text applications. Text-to-text processes utilize existing texts to generate coherent new text, while data-to-text systems transform non-linguistic structured data, like tables or graphs, into natural language text. Our focus is on data-to-text generation, as it aligns with the contribution of this paper.

Data-to-text generation approaches fall into three categories.

- **Rule-based** – These methods follow a three-stage pipeline, starting with content selection and text structuring in the Document Planner [14]. The Microplanner combines sentence aggregation, lexicalization, and referring expression generation [15]. Finally, the Linguistic Realizer generates grammatically correct sentences [16].
- **Template-based** – Unlike rule-based approaches, template-based methods directly map non-linguistic input to linguistic surface structure. Text generation occurs through string manipulation, where users create programs with string patterns containing empty slots to be filled with relevant information.
- **Neural-based** – Neural approaches are data-driven and require no manual feature engineering. They develop end-to-end models where neural networks learn to generate high-quality text descriptions directly from input data, bypassing explicit modeling of intermediate stages.

In the early stages of NLG, rule-based systems were favored for embodying linguistic insights [17]. This perception shifted with the developing of more sophisticated template-based systems like D2D, which could adapt output based on context and perform complex syntactic operations [18]. Template-based systems offer ease of development, control, and speed but may sacrifice fluency, maintainability, and flexibility compared to rule-based systems. In modern NLG, neural-based approaches have taken precedence, thanks to their superior generalization and output variation capabilities. However, neural-based systems might lack control over the generation process, potentially resulting in inaccurate text generation. They also require extensive training data, limiting their feasibility when large datasets are unavailable. Hybrid approaches, which combine two or more methods, aim to leverage the strengths of each approach. For instance, Kale and Ratsogi implemented a two-stage pipeline combining a template-based approach for a baseline response and a pre-trained language model (T5) for rewriting the response into coherent, natural-sounding text [19]. This hybrid method blends the control and ease of development from templates with the fluency and diversity offered by neural approaches.

III. Related Works

In this section, we overview related works on applying Natural Language Processing and Natural Language Generation in the cybersecurity domain, respectively, in Section III-A and Section III-B.

A. NLP for CTI

While the use of Natural Language Generation in Cyber Threat Intelligence applications has not been thoroughly explored in the literature, Natural Language Processing has been extensively researched for Information Extraction (IE) purposes. Indeed, automatically processing CTI reports to retrieve entities and relationships can be pivotal for an organization seeking to gather intelligence with minimal time delays and manpower. Effectively, these systems drive researchers towards fully automating CTI reporting, making cyber defense practices more accessible to all organizations. One of these systems is STIXnet, which deploys several IE models to extract all STIX entities and relationships from natural language reports [20]. By combining IE systems such as STIXnet with AGIR, it will be possible to automatically process a multitude of reports and parse their intelligence in one single report, tailored according to the user's needs.

B. NLG for CTI

To the best of our knowledge, only one NLG approach is applied to CTI in a setting similar to ours, while only a few instances are applied to the cybersecurity domain. One of these examples involves Das and Varma's work, who developed a Recurrent Neural Networks (RNN) system to generate text for advanced email masquerading [21]. Other instances instead include implementing NLG models for cybersecurity education and training purposes [22]. The usage of NLG models for CTI report generation has also been explored by Ranade et al. for poisoning attack purposes [23]. In their paper, the authors use GPT-2 to generate fake CTI text to poison the dataset of Cybersecurity Knowledge Graphs. However, while this paper also treats the problem from an attacker's perspective, it also utilizes textual inputs. It thus differs from our application, which uses structured data as input. The research work that deals with a setting similar to ours is the one by S. Polzunov and J. Abraham, where they introduced Narrator, a tool capable of creating intelligence reports from the JSON representation of STIX graphs [24]. Narrator employs a rule-based approach to generate four report types, which can be edited and exported in PDF format. AGIR expands upon the foundation laid by Narrator, enhancing overall performance. In its first pipeline step, AGIR adopts a template-based approach inspired by the D2D system. The second step incorporates a technique akin to that employed by Kale and Ratsogi, utilizing ChatGPT to rewrite reports in a more human-like manner.

IV. System Model

As an automatic CTI report generation system, AGIR leverages the STIX graphs of entities and relationships to fulfill its objective. Thus, an underlying Knowledge Base (KB) can be deployed to store the STIX data to maximize its capabilities. The usage of a KB yields several advantages.

- **Intelligence Categorization** – Entities and relations can be stored and categorized using the STIX guidelines.
- **Incident Recostruction** – Several intelligence sources can be parsed together to generate a single report containing the most information on a specific threat or incident.
- **Threat Evolution** – Intelligence collected over a period of time can be aggregated to construct a timeline, providing insights into the evolution of a specific threat.

The Knowledge Base can be an instance of existing CTI platforms that use STIX as the underlying model or can be built from scratch. Several sources can be employed to collect the intelligence stored in the KB. One example is the MITRE ATT&CK framework, which publicly stores and updates intelligence on threat groups, TTPs, mitigations, and software [25]. Through the usage of APIs, it is also possible to collect the intelligence automatically in a structured format, which can then be converted to STIX. Reports and CTI bulletins can also be used as a starting point for populating the KB. Indeed, as anticipated in Section III-A, IE systems can be deployed for this purpose, making CTI reporting almost fully automatic.

The storage of entities, relations, and the reports from which the intelligence has been collected allows for a more flexible implementation of AGIR. Suppose the generation of a report on a specific infrastructure is requested. The system can automatically query the Knowledge Base with all the intelligence related to that specific entity, reporting attributes, dates, and other useful information. Users can dynamically select the amount of intelligence they want to include by using the service's Graphical User Interface (GUI), expanding nodes on the graphs as shown in Figure 2.

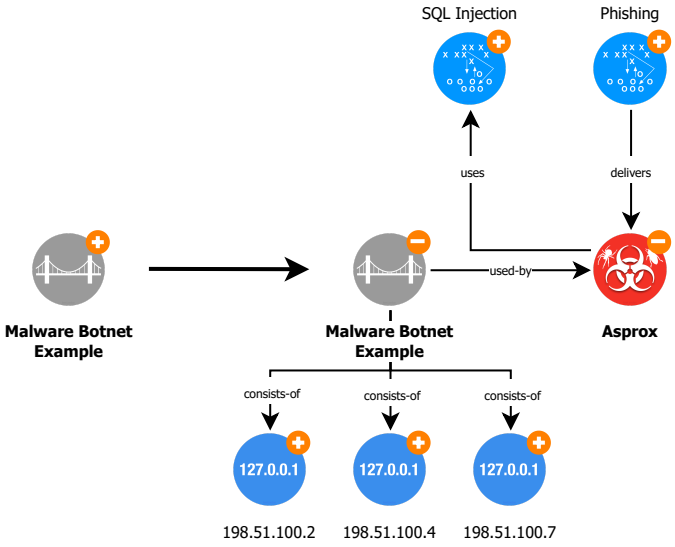

Fig. 2. Proof-of-concept of AGIR's GUI.

In this example, we use the same dummy data in Figure 1. We first consider the "Malware Botnet Example" infrastructure as a singular entity, which will constitute the main subject of the report. By expanding the node, all the information about IPv4 addresses and malware previously shown in Figure 1 appear. We can further expand those nodes, as it is possible to see for the "Asprox" entity, which will introduce two attack patterns in the graph. The selected STIX graph can then be fed as input to AGIR, creating a CTI report accordingly. Figure 3 shows an overview of the overall system model.

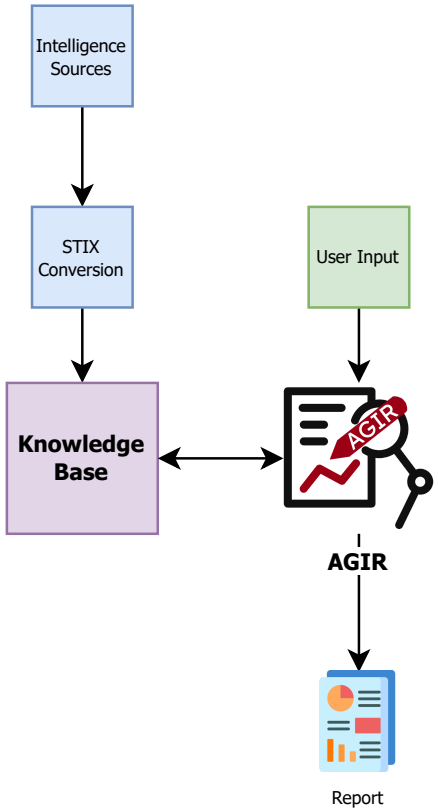

Fig. 3. Overview of the system model.

V. Methodology

This section overviews the methodology that constitutes AGIR and the modules that allow for generating CTI reports. The template-base module and the neural-based are presented, respectively, in Section V-A and Section V-B. An overview of AGIR's pipeline is shown in Figure 4.

A. Template Based Module

The template-based module is AGIR's first module and, thus, inherits its input, which is constituted as follows.

1) **STIX Graph** – A JSON file representing the graph from which to generate the report.
2) **Report Type** – A parameter given by the user indicating the type of report that the system should generate.

In the initial stage of the template-based module execution, we analyze the report type parameter and designate the template accordingly. Then, depending on the selected template, a subset of entities and relationships is selected from the JSON representation of the previously mentioned graph. The size of this subset is determined by the number of intelligence

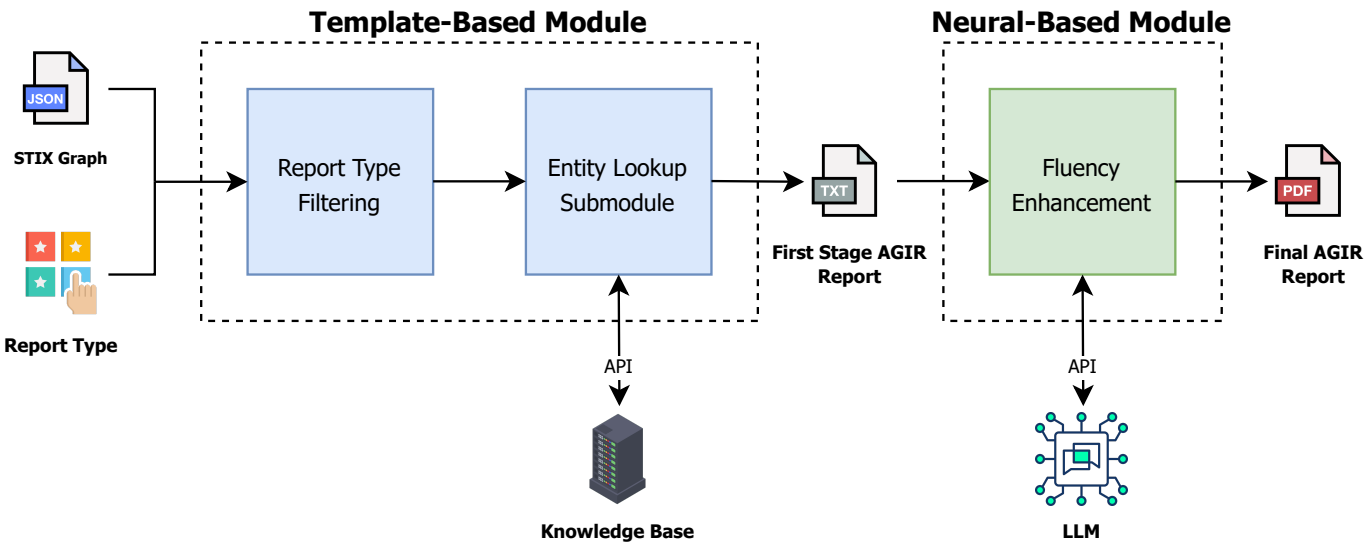

Fig. 4. Overview of AGIR's pipeline.

contained in the JSON file and the type of intelligence required by the template. This is done because of the specific type of intelligence that each report template contains. Thus, by first considering the template selection, we reduce the number of API calls we need to perform to retrieve the entities from the Knowledge Base. Indeed, our system supports four different report types.

1) **Overview Report** – Provides a summary of information from the STIX graph. Analysts can use this template to understand a specific event portrayed in the JSON file.
2) **Subject Report** – This template focuses on specific entities such as threat actors or intrusion sets. It also includes relationships, IOCs, and MITRE sections for all other selected entities. This type of intelligence helps understand the landscape where the subject is inserted.
3) **Timeline Report** – Provides a timeline overview of the entities related to the STIX graph. This template sorts the events chronologically and reports them according to their sequencing.
4) **Vulnerability Report** – Deals with vulnerabilities related to a specific entity and thus constitutes the most specific template out of the four. Each vulnerability also includes a table showing specific properties such as CVSS (Common Vulnerability Scoring System) score, mitigations, and vulnerable configurations.

After the template selection, we extract each entity in the STIX graph's type from the Knowledge Base, which will be retrieved in conjunction with its unique identifier. The usage of IDs allows for efficiently storing reports, relationships, and events associated with a specific entity. Thus, each time an entity is called from the Knowledge Base, we can retrieve its history and previous relationships with other entities. To do so, for each selected entity from the STIX graph, we initialize a dictionary with the following six keys: *overview*, *relationships*, *stats*, *useful resources*, *IOCs*, and *TTPs*. Based on the entity ID, we query the KB through several API calls to populate the dictionary accordingly. This means performing content selection based on a predefined set of rules (e.g., properties specific to the entity are inserted in the overview section, and information about related IOC is inserted in the IOC section). Once all the information is available, the module goes through each report template section and fills the gaps with the appropriate piece of intelligence.

B. Neural Based Module

While the output of the template-based module can already be considered a report, being generated from a set of rules implies that it is naturally mechanic and not fluent. Thus, to improve on this aspect, we use the neural-based module. To do so, we use ChatGPT, a Large Language Model that, in recent months, has attracted an incredible amount of attention due to its efficiency in Natural Language Generation [26]. Through its APIs, we prompt the model for a more fluent version of the report, highlighting the need to keep the text's information unchanged. Once the generation is complete, the final result is given as the output of the overall AGIR pipeline.

Two main challenges arise from using ChatGPT as a "fine-tuning" model for the report.

- **Cost** – While using the LLM through its graphical interface is free for all registered users, the payment of a fee is needed for each API call. At the time of the development of AGIR, the cost of generating a single report is, on average, 0.0024 US dollars. Since, in real-world scenarios, the number of reports generated each day is not exceptionally high, the price does not constitute an obstacle for companies and organizations.
- **Lack of Control** – As stated in Section II-B, neural-based NLG approaches present a lack of control over

the generated output. To still assess the contribution of this module on the overall fluency of the report, in Section VI-B, we qualitatively evaluate the outputs of both modules, proving the benefits of ChatGPT in the report generation.

VI. Evaluation

This section presents an experimental evaluation of our system. We aim to write a precise human-like report that supports analysts and reduces their time in the report-writing process. Given the importance of content and style in generating a report, we split the evaluation into two parts. First, we evaluate AGIR quantitatively, thus assessing the completeness of intelligence contained in the output (Section VI-A). In the second part, we evaluate the style of those reports in terms of fluency and utility (Section VI-B).

A. Quantitative Results

We assess AGIR's accuracy to determine whether the incorporation of ChatGPT has led to any instances of omission or hallucination in the generated text. It's worth noting that in this analysis, we do not evaluate the initial step of AGIR singularly (i.e., reports generated by the template-based module). This is because the module relies entirely on predefined rules, and unless there are implementation errors, they are not expected to introduce accuracy concerns.

For the quantitative evaluation, we will use True Positives (TP), False Positives (FP), and False Negatives (FN), which in this specific application are defined as follows.

- **True Positive** – Information present both in the final report and in the input JSON file.
- **False Positive** – Information present in the report but not in the input JSON file.
- **False Negative** – Information present in the input JSON file but is not found in the report.

Using these indicators, we use three metrics to evaluate AGIR accuracy: precision, recall, and F1-score. These metrics are defined as follows.

$$Precision = \frac{TP}{TP + FP}, \tag{1}$$

$$Recall = \frac{TP}{TP + FN}, \tag{2}$$

$$F1 = 2\frac{Precision \cdot Recall}{Precision + Recall}. \tag{3}$$

To evaluate our system, we use a sample of 12 STIX graphs. We execute AGIR and generate 12 reports from each of these graphs, split equally between each report type. Each JSON file underwent a manual process in which the intended report information was identified and verified for its presence within the generated report. Results of this evaluation are shown in Table I. As evident from the results, the utilization of ChatGPT consistently avoided introducing new information (i.e., hallucination events) into the report. However, in a very limited number of cases, the model did omit information from the JSON file. Nonetheless, having a recall and F1 score value close to 1 means that the generated reports almost always contain the same amount of information from the JSON file, highlighting the completeness of the reports.

TABLE I
AGIR's quantitative evaluation results.

Precision	Recall	F1 Score
1.000	0.993	0.996

B. Qualitative Results

We now perform a qualitative evaluation of the reports generated by AGIR. Thus, we now focus on fluency, correctness, and utility of the information expressed in natural language. To do so, we further divide the evaluation into two parts: a syntactic evaluation (Section VI-B1) and a linguistic evaluation (Section VI-B2).

1) Syntactic Evaluation: We introduce SLOR (Syntactic Log-Odds Ratio) [27] for the syntactic evaluation of the generated reports' qualitative properties. We opted for this metric because it is widely recognized as the de facto standard for assessing text fluency in referenceless evaluations, which aligns with our specific evaluation context. SLOR exhibits the strongest correlation with human sentence acceptability compared to various sentence probability-based scoring methods [28]. Furthermore, its effectiveness has been demonstrated in unsupervised text compression tasks. SLOR assigns a score to each sentence S by calculating its log probability using a specific Language Model (LM). This score is then normalized by the log probability of unigrams and the sentence's length.

$$SLOR\left(S\right) = \frac{1}{S}\left(\ln\left(p_M\left(S\right)\right)\right) - \ln\left(p_u\left(S\right)\right). \tag{4}$$

In Equation 4, $p_M\left(S\right)$ is the probability assigned to the sentence under the LM, while $p_u\left(S\right)$ is the unigram probability of the sentence S. These quantities are defined, respectively, as follows.

$$p_M\left(S\right) = p\left(\langle t_1, t_2, ..., t_{|S|}\rangle\right) = p\left(t_1\right)\prod_{i=2}^{|S|} p\left(t_i|t_1, ..., t_{i-1}\right), \tag{5}$$

$$p_u\left(S\right) = \prod_{t\in S} p\left(t\right). \tag{6}$$

The rationale behind subtracting unigram log probabilities is to mitigate the impact of a token's rarity when considered individually, as opposed to its rarity in a specific sentence position. Normalizing by sentence length is essential to ensure that shorter sentences are not favored over equally fluent longer ones. It is worth noting that the log probability of a sentence normalized by its length corresponds to the negative cross-entropy of that sentence, as per the employed language model during the evaluation. To calculate sentence probabilities, we utilize the pre-trained XLNet language model [29]. SLOR

scores are theoretically unbounded, with higher scores indicating better text fluency. The value range depends on many factors, such as the dataset, language model, and the nature of the text being evaluated. Thus, we are interested in comparing the results from our testbed with other state-of-the-art NLG models applied on CTI, such as Narrator. Furthermore, we also evaluate the fluency of the reports in different stages of the pipeline, thus assessing the contribution of each module. Therefore, the three models we evaluate are the following: Narrator, first step AGIR (i.e., the output of the template-based module), and final AGIR (i.e., the output of the neural-based module). The evaluation dataset is the same as the one used for the quantitative evaluation in Section VI-A, and thus comprises 3 STIX graphs for each report type supported by AGIR. Results are shown in Table II. As we can see, the SLOR score obtained by Narrator and first stage AGIR are very close to one another, while final AGIR significantly increases the score. These results confirm not only the contribution that the neural-based module has on the overall quality of the reports but also highlight the possible limitations of template-based approaches when considered independently.

TABLE II
AVERAGE SLOR SCORES AND STANDARD DEVIATION OF NARRATOR AND DIFFERENT STAGES OF AGIR'S PIPELINE.

Model	SLOR
Narrator	2.13±0.90
First Step AGIR	2.16±1.07
Final AGIR	**2.75±0.72**

2) Linguistic Evaluation: By using SLOR, we can assess the fluency of each report with formally defined metrics, thus giving an objective perspective on AGIR's efficacy. However, given our system's most important feature, we are also interested in the human perspective: the reduction of processing times for CTI reporting. For this reason, we conduct another type of qualitative evaluation based on the opinions shared by expert threat analysts. All questioned analysts are employers of Leonardo S.p.A., an Italian multinational company that collaborated in this research. Opinions were collected in the form of surveys, which involved rating the quality of the reports based on the following three dimensions.

- **Fluency** – Whether the text is easy to read and understand.
- **Correctness** – Whether the content of the text is true and derivable from the input data.
- **Utility** – Whether the text helps the user to write the final report faster.

The questionnaire collected opinions on all of these aspects from 38 analysts. Fluency and correctness have been measured using a Likert scale from 1 (not good) to 5 (very good) [30]. Instead, the utility dimension is evaluated by asking each analyst how long it will take to write a final report starting from the system's output. The utility value is then compared with a baseline score of 127.3 minutes, which was obtained by asking all analysts how much time, on average, it takes for them to write a report. To prevent the outcomes from being biased by a single questionnaire instance, we have devised three questionnaires with identical structures and distributed the analysts evenly into three groups. Furthermore, the presented reports are unmarked, so analysts cannot know from which model they are generated. The questionnaires consist of four sections, each corresponding to a report type and containing one report from each system, resulting in 12 reports for each questionnaire. Each report is assessed based on the three dimensions mentioned earlier. In Table III, we show the obtained results from the survey. As we can see, we also notice that first step AGIR still outclasss Narrator on every metric. Moreover, final AGIR reports outperform the other models used for comparison in all three dimensions. Fluency is the metric most impacted by the neural-based module, confirming our hypothesis and thus further asserting its contribution to the report generation process. An increase in correctness from first step AGIR to final AGIR stands out from this evaluation, dispelling previous uncertainty about possible omissions or hallucinations that could have appeared with the usage of LLMs. As for the model's utility, analysts believe that AGIR can reduce report production times by 42.6% with respect to the baseline, resulting in a total time reduction of 54 minutes. It should be noted that the results shown here are averaged on all the report types. A more detailed analysis can be found in Appendix A.

TABLE III
QUESTIONNAIRE RESULTS GROUPED BY DIMENSION.

Model	Fluency	Correctness	Utility
Narrator	2.98	3.00	97.5 min
First Step AGIR	3.48	3.77	79.6 min
Final AGIR	**4.13**	**3.90**	**74.3 min**

VII. CONCLUSIONS

Cyber Threat Intelligence is an important topic for all companies and organizations. With the use of CTI, defenses against threat actors can be built proactively and more efficiently, thus increasing the security of each asset. However, the sharing of CTI data is still anchored to natural language, which further delays the integration of the intelligence and the application of the defense mechanisms. Furthermore, the dissemination of the information is extremely time-consuming, forcing analysts to write several reports each day to distribute their data.

Contribution. In this paper, we presented **AGIR**, a system for the Automatic Generation of Intelligence Reports. AGIR takes in input the JSON representation of a STIX graph and uses it to generate a report containing all entities and relationships contained. AGIR uses two different approaches for Natural Language Generation: a template-based module used for building a baseline report and a neural-based module used to increase its fluency. Using this pipeline, we allow for the generation of four different types of reports (overview, subject, timeline, and vulnerability). We experimentally evaluated AGIR both quantitatively and qualitatively. We showed that

reports almost perfectly include the entities and relationships contained in the STIX graph given in input (recall value of 0.99) without introducing any hallucination (precision value of 1). Also, we assessed through Syntactic Log-Odds Ratio scores and questionnaires that our reports are more fluent with respect to other state-of-the-art models and that the usage of AGIR can reduce production times by 42.6%.

Future Works. For future research endeavors, we aim to validate the accuracy of the extrinsic evaluation results by involving more domain experts who can assess AGIR in the context of crafting actual security reports. Presently, AGIR relies on ChatGPT, which may raise concerns related to cost and privacy. An intriguing avenue for further investigation would be to explore the development of an equivalent deep learning model that can be employed locally, thereby mitigating the aforementioned issues. Additionally, a valuable contribution could involve the creation of an extensive dataset containing the pertinent STIX properties for a range of entities. Such a dataset could serve as the foundation for training a language model to generate initial reports, thus addressing the maintainability challenge associated with template usage.

REFERENCES

[1] P. Chen, L. Desmet, and C. Huygens, "A study on advanced persistent threats," in *Communications and Multimedia Security: 15th IFIP TC 6/TC 11 International Conference, CMS 2014, Aveiro, Portugal, September 25-26, 2014. Proceedings 15*, pp. 63–72, Springer, 2014.

[2] T. D. Wagner, K. Mahbub, E. Palomar, and A. E. Abdallah, "Cyber threat intelligence sharing: Survey and research directions," *Computers & Security*, vol. 87, p. 101589, 2019.

[3] S. Barnum, "Standardizing cyber threat intelligence information with the structured threat information expression (stix)," *Mitre Corporation*, vol. 11, pp. 1–22, 2012.

[4] K. Oosthoek and C. Doerr, "Cyber threat intelligence: A product without a process?," *International Journal of Intelligence and CounterIntelligence*, vol. 34, no. 2, pp. 300–315, 2021.

[5] R. Dale, "Natural language generation: The commercial state of the art in 2020," *Natural Language Engineering*, vol. 26, no. 4, pp. 481–487, 2020.

[6] V. Plachouras, C. Smiley, H. Bretz, O. Taylor, J. L. Leidner, D. Song, and F. Schilder, "Interacting with financial data using natural language," in *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval*, pp. 1121–1124, 2016.

[7] S. Balloccu, S. Pauws, and E. Reiter, "A nlg framework for user tailoring and profiling in healthcare.," in *SmartPhil@ IUI*, pp. 13–32, 2020.

[8] E. Adamopoulou and L. Moussiades, "An overview of chatbot technology," in *IFIP international conference on artificial intelligence applications and innovations*, pp. 373–383, Springer, 2020.

[9] C. Johnson, L. Feldman, and G. Witte, "Cyber threat intelligence and information sharing," 2017-05-08 2017.

[10] Y. Keim and A. Mohapatra, "Cyber threat intelligence framework using advanced malware forensics," *International Journal of Information Technology*, pp. 1–10, 2019.

[11] D. Chismon and M. Ruks, "Threat intelligence: Collecting, analysing, evaluating," 2015.

[12] Z. Porkorny, "What are the phases of the threat intelligence lifecycle," *The Threat Intelligence Handbook*, 2018.

[13] B. Jordan, "GitHub - freetaxii/stix2-graphics: Graphics, icons, and diagrams to support STIX 2 — github.com." https://github.com/freetaxii/stix2-graphics. [Accessed 21-09-2023].

[14] D. D. McDonald, "Issues in the choice of a source for natural language generation," *Computational Linguistics*, vol. 19, no. 1, pp. 191–197, 1993.

[15] E. Reiter and R. Dale, "Building applied natural language generation systems," *Natural Language Engineering*, vol. 3, no. 1, pp. 57–87, 1997.

[16] A. Gatt and E. Krahmer, "Survey of the state of the art in natural language generation: Core tasks, applications and evaluation," *Journal of Artificial Intelligence Research*, vol. 61, pp. 65–170, 2018.

[17] E. Reiter, "Nlg vs. templates," *arXiv preprint cmp-lg/9504013*, 1995.

[18] M. Theune, E. Klabbers, J.-R. De Pijper, E. Krahmer, and J. Odijk, "From data to speech: a general approach," *Natural Language Engineering*, vol. 7, no. 1, pp. 47–86, 2001.

[19] M. Kale and A. Rastogi, "Template guided text generation for task-oriented dialogue," *arXiv preprint arXiv:2004.15006*, 2020.

[20] F. Marchiori, M. Conti, and N. V. Verde, "Stixnet: A novel and modular solution for extracting all stix objects in cti reports," in *Proceedings of the 18th International Conference on Availability, Reliability and Security*, ARES '23, (New York, NY, USA), Association for Computing Machinery, 2023.

[21] A. Das and R. Verma, "Automated email generation for targeted attacks using natural language," *arXiv preprint arXiv:1908.06893*, 2019.

[22] H. K. Skrodelis, A. Romanovs, N. Zenina, and H. Gorskis, "The latest in natural language generation: Trends, tools and applications in industry," in *2023 IEEE 10th Jubilee Workshop on Advances in Information, Electronic and Electrical Engineering (AIEEE)*, pp. 1–5, IEEE, 2023.

[23] P. Ranade, A. Piplai, S. Mittal, A. Joshi, and T. Finin, "Generating fake cyber threat intelligence using transformer-based models," in *2021 International Joint Conference on Neural Networks (IJCNN)*, pp. 1–9, IEEE, 2021.

[24] J. Abraham and S. Polzunov, "Narrator: Generating intelligence reports from structured data," *Forum of Incident Response and Security Teams (FIRST)*, 2020.

[25] B. E. Strom, A. Applebaum, D. P. Miller, K. C. Nickels, A. G. Pennington, and C. B. Thomas, "Mitre att&ck: Design and philosophy," in *Technical report*, The MITRE Corporation, 2018.

[26] M. Mijwil, M. Aljanabi, *et al.*, "Towards artificial intelligence-based cybersecurity: the practices and chatgpt generated ways to combat cybercrime," *Iraqi Journal For Computer Science and Mathematics*, vol. 4, no. 1, pp. 65–70, 2023.

[27] K. Kann, S. Rothe, and K. Filippova, "Sentence-level fluency evaluation: References help, but can be spared!," *arXiv preprint arXiv:1809.08731*, 2018.

[28] J. H. Lau, A. Clark, and S. Lappin, "Grammaticality, acceptability, and probability: A probabilistic view of linguistic knowledge," *Cognitive science*, vol. 41, no. 5, pp. 1202–1241, 2017.

[29] Z. Yang, Z. Dai, Y. Yang, J. Carbonell, R. R. Salakhutdinov, and Q. V. Le, "Xlnet: Generalized autoregressive pretraining for language understanding," *Advances in neural information processing systems*, vol. 32, 2019.

[30] R. Likert, "A technique for the measurement of attitudes.," *Archives of psychology*, 1932.

APPENDIX A
QUESTIONNAIRES

In this appendix section, we give more details on the questionnaires given to the expert analysts for the qualitative evaluation of AGIR. In Appendix A-A, we show a sample of the questionnaires and the reports provided to the analysts. Moreover, we divide the evaluation of each model on each report type and show their results in Appendix A-B.

A. Samples

The survey was conducted through a Google Forms module sent to Leonardo's employees. In particular, we focused on the Security Operation Center (SOC) staff members, given their experience in CTI and, in particular, in threat intelligence reporting. The questionnaires given to the analysts are structured as follows. First, to establish a baseline to evaluate AGIR's utility, we ask each expert how much time, on average, it takes for them to write a full report. Afterward, we do the following for each of the four supported report types.

1) We describe the aim of the report type and the focus of their entities and relationships. We also provide several use cases to contextualize the usage of those particular reports further.
2) We provide a graphical overview of the JSON input from which the reports have been generated. Some examples of those STIX graphs are shown in Figure 5, Figure 6, and Figure 7.
3) We provide three examples of generated reports. Those three samples are the report generated by Narrator, the report generated by first step AGIR, and the report generated by final step AGIR. To avoid biases, the samples are unmarked and randomized so that the analysts do not know which report has been generated by our system.
4) We ask for an evaluation of each report's accuracy, described as the presence of true intelligence derivable from the input.
5) We ask for an evaluation of each report's fluency, described as the quality of the text, its clarity, ease of reading, and how close it is to a human-generated report.
6) We ask for an evaluation of each report's utility, described as how long the analyst would take to write a full report starting from the presented one.

We provide several samples of the generated reports in our publicly available repository.[1]

B. Results Divided by Report Type

The results of the survey divided into overview report, subject report, timeline report, and vulnerability report are shown, respectively, in Table IV, Table V, Table VI, and Table VII. As we can see, the report type can heavily influence the utility score, while fluency and correctness are more consistent across the different templates. In particular, with respect to Narrator, we can see a time reduction in report production of 27.8% for overview reports, 20.7% for subject

[1]https://github.com/Mhackiori/AGIR/tree/main/Reports

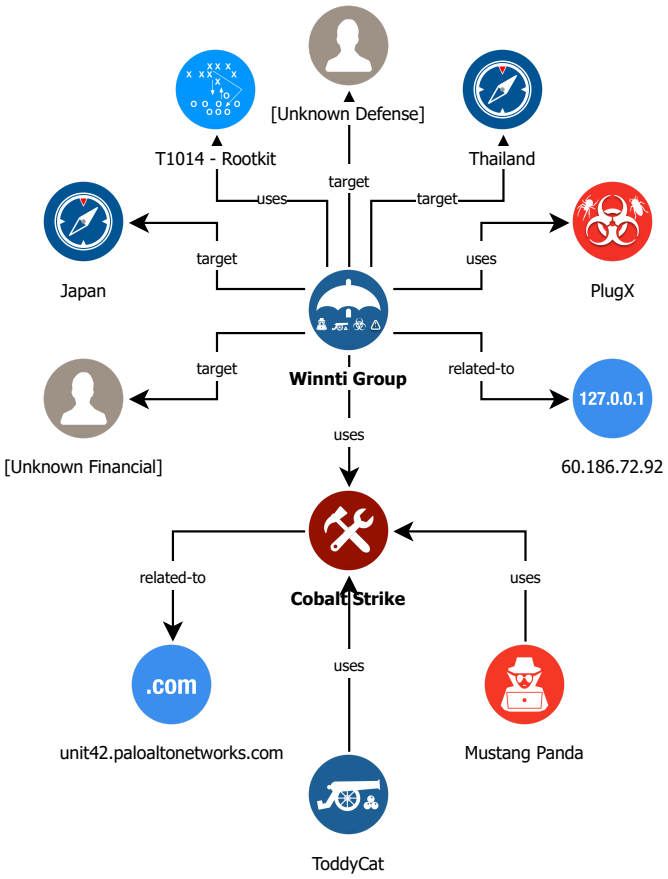

Fig. 5. JSON input example for Overview and Subject reports. We focus on the "Winnti Group" entity when dealing with the Subject report.

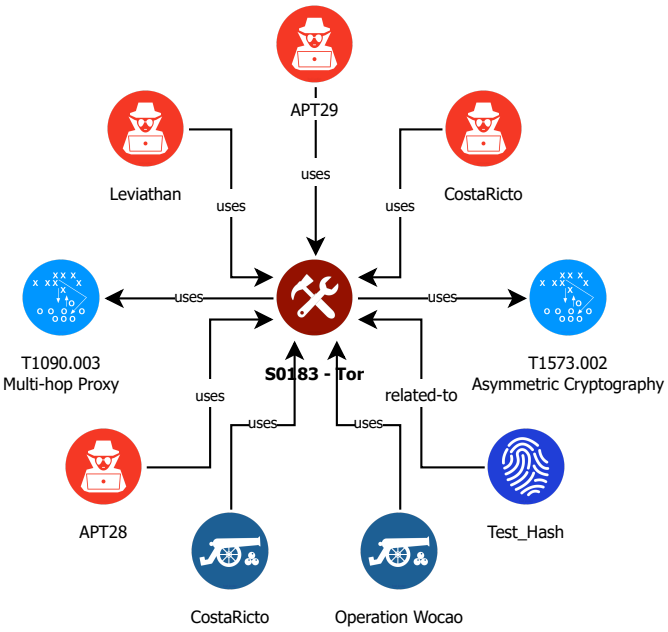

Fig. 6. JSON input example for Timeline reports.

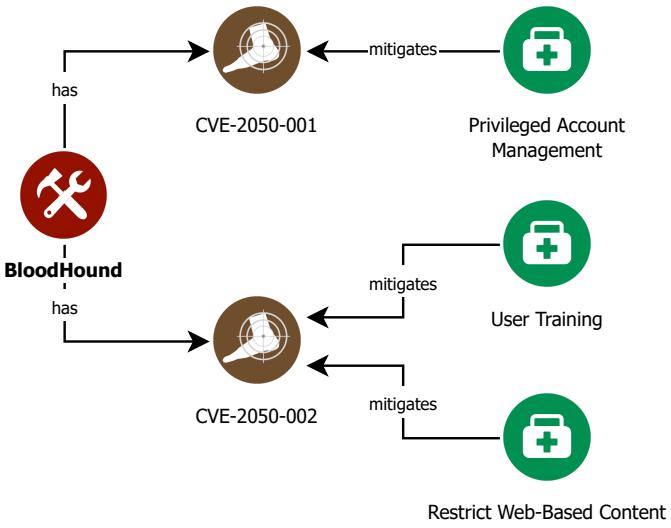

Fig. 7. JSON input example for Vulnerability reports.

reports, 6.2% in timeline reports, and 46.9% in vulnerability reports (on average 25.4%).

TABLE IV
QUESTIONNAIRE RESULTS FOR THE OVERVIEW REPORTS.

Model	Fluency	Correctness	Utility
Narrator	2.92	2.77	109.2 min
First Step AGIR	3.85	3.46	86.9 min
Final AGIR	**4.08**	**4.23**	**78.9 min**

TABLE V
QUESTIONNAIRE RESULTS FOR THE SUBJECT REPORTS.

Model	Fluency	Correctness	Utility
Narrator	2.85	2.92	91.2 min
First Step AGIR	3.92	3.54	68.1 min
Final AGIR	**4.00**	**4.15**	**72.3 min**

TABLE VI
QUESTIONNAIRE RESULTS FOR THE TIMELINE REPORTS.

Model	Fluency	Correctness	Utility
Narrator	3.38	3.15	111.5 min
First Step AGIR	3.46	3.38	114.6 min
Final AGIR	**3.61**	**3.92**	**104.6 min**

TABLE VII
QUESTIONNAIRE RESULTS FOR THE VULNERABILITY REPORTS.

Model	Fluency	Correctness	Utility
Narrator	2.84	3.07	78.1 min
First Step AGIR	3.84	3.53	48.8 min
Final AGIR	**3.92**	**4.23**	**41.5 min**