Uncovering Reliable Indicators: Improving IoC Extraction from Threat Reports

Evangelos Froudakis*, Athanasios Avgetidis*, Sean Tyler Frankum†,
Roberto Perdisci†, Manos Antonakakis*, Angelos Keromytis*
*Georgia Institute of Technology
{efroudakis3, avgetidis, manos, angelos}@gatech.edu
†University of Georgia
{Sean.Frankum, perdisci}@uga.edu

***Abstract*—Indicators of Compromise (IoCs) are critical for threat detection and response, marking malicious activity across networks and systems. Yet, the effectiveness of automated IoC extraction systems is fundamentally limited by one key issue: the lack of high-quality ground truth. Current extraction tools rely either on manually extracted ground truth, which is labor-intensive and costly, or on automated ground truth creation methods that include non-malicious artifacts, leading to inflated false positive (FP) rates and unreliable threat intelligence. In this work, we analyze the shortcomings of existing ground truth creation strategies and address them by introducing the first hybrid human-in-the-loop pipeline for IoC extraction, which combines a large language model–based classifier (LANCE) with expert analyst validation. Our system improves precision through explainable, context-aware labeling and reduces analysts' work factor by 43% compared to manual annotation, as demonstrated in our evaluation with six analysts. Using this approach, we produce PRISM, a high-quality, publicly available benchmark of 1,791 labeled IoCs from 50 real-world threat reports. PRISM supports both fair evaluation and training of IoC extraction methods and enables reproducible research grounded in expert-validated indicators.**

1. Introduction

Detecting and analyzing malicious activity is essential in modern cybersecurity defense. Indicators of Compromise (IoCs), such as IP addresses, domain names, URLs, and file hashes, which signal malicious activity, are among the most important parts of threat intelligence [1], [2], [3]. Analysts rely on IoCs for both immediate defense, such as creating block lists and firewall rules, and for activities such as threat hunting. Accurate and timely identification of IoCs can therefore significantly improve an organization's ability to detect, respond to, and mitigate cyber-threats.

One of the most prominent sources of IoCs is threat reports (TRs), which are published by cybersecurity companies, research institutions, and government agencies, and contain critical information about attacks, campaigns, and incidents [4], [2], [5], [1]. Threat reports vary widely in structure and detail due to the lack of standardized formatting. This inconsistency makes accurate IoC extraction challenging, as indicators may be embedded in diverse formats and contexts, yet their precise identification is critical for effective threat response.

Previous work has focused on three general approaches for extracting IoCs from threat reports: manual labeling, automated extraction, and automated retrieval from threat intelligence sharing platforms. As noted repeatedly [6], [1], [7], [8], manual labeling is the most accurate approach to extract IoCs from unstructured threat reports. However, this requires significant domain knowledge and can be very time-consuming, making it expensive and difficult to scale [1], [6], [7]. Automated IoC extraction tools, including recent work that leverage large language models (LLMs) [9], [10], [11], offer potential efficiency but often lack the necessary accuracy due to the complexity of natural language in technical cybersecurity reports, the variability in IoC presentation, and the difficulty in distinguishing true IoCs from similar-looking but benign artifacts (e.g., indicators such as benign domain names, IP addresses, and URLs that are not related to the threats described in the reports). Finally, while there exist efforts to distribute IoCs to the community in an easy-to-parse structured format, for instance via threat exchanges (e.g., AlienVault [12]), threat report databases (e.g., orkl.eu), and online security scanners (e.g., VirusTotal [13], [14], [15], [16]), we find that popular threat report IoC exchanges like AlienVault, often automatically extract IoCs from unstructured TRs with low precision (<80%), recall (<80%) and/or coverage. As a result, these approaches either incur high manual work factor costs or yield unreliable intelligence, underscoring the need for a new solution that is both accurate and efficient.

To address these challenges, we propose the first IoC extraction approach that combines automated extraction supported by explainable AI with *human-in-the-loop* verification. Our central hypothesis is that large language models (LLMs), when *guided by properly engineered prompts and paired with analyst feedback*, can match or surpass the accuracy of manual-only labeling while significantly reducing the time cost of human effort. In our evaluation, we observe a 43% reduction in parsing time compared to manual-only workflows. Importantly, our pipeline design and prompt engineering strategies are model-agnostic and generalize across most state-of-the-art LLMs, enabling flexible deployment in diverse environments. By integrating human analysts in the IoC extraction process, we can leverage the speed of automation while ensuring high precision through human judgment. Similar "human-in-the-loop" (HITL) approaches have been successfully demonstrated in other domains where

high-accuracy data labeling and expert oversight are critical, such as medical applications and some computer vision and natural language processing tasks [17], [18], [19], [20]. To the best of our knowledge, we are the first to adapt the HITL approach for the efficient and accurate labeling of IoCs from unstructured reports. To this end, we develop a HITL system that provides: (1) visual assistance (via highlighted text) that allows analysts to quickly identify indicators of interest in unstructured (potentially long) threat reports; and (2) color-coded IoC labels assigned by an LLM, each accompanied by an explanation of the contextual cues used to justify the label – designed following principles of effective data visualization [21], [22], [23].

Figure 1 shows an overview of our system. It consists of LANCE and a human-in-the-loop (HITL) component where analysts, supported by LANCE and by a custom web-based user interface, assign labels to IoCs found in threat reports. Thanks to our system's custom user interface and explainable LLM-based IoC labels, analysts can quickly confirm or correct the label recommendations throughout the report, thus boosting accuracy and efficiency. To evaluate our approach, we applied it to 50 real-world threat reports from reputable sources and observed that junior analysts made fewer labeling errors when assisted by our system. We then used the system to construct PRISM, a high-quality, manually validated dataset of 1,774 labeled IoCs, including domains, URLs, IPs, and file hashes. PRISM supports both training and evaluation of IoC extraction tools and serves as the first openly available benchmark for this task.

In summary, the main contributions of this work are:

- **Human-in-the-Loop IoC Labeling System:** The first human-in-the-loop system for IoC extraction and labeling from threat reports, which combines an LLM-based automated and explainable IoC labeling process with manual verification to minimize labeling costs and maximize accuracy.
- **LANCE:** The LLM–based labeling component that provides explainable classifications of indicators from unstructured threat reports. This explainability allows for faster human validation, reducing annotation time by 43% compared to manual labeling.
- **Review of Existing IoC Extraction/Labeling Systems:** We review and attempt to reproduce previous work on IoC extraction and labeling, including automated IoC extraction systems and threat exchanges. Our results show that existing available systems (specifically, systems whose code is available and reusable, or can be accessed via an API) either lack precision (i.e., high false positives) and/or recall (i.e., high false negatives).
- **PRISM:** The first open, fully manually validated, and well-documented benchmark dataset of IoCs from threat reports, consisting of 1,791 indicators extracted from 50 real-world threat reports. Our dataset, called PRISM, is designed to support both future training/testing of IoC extraction/labeling models and as a common evaluation dataset for comparing IoC extraction systems among each other.

We will make the HITL system, LANCE, and PRISM available to the research community upon publication.

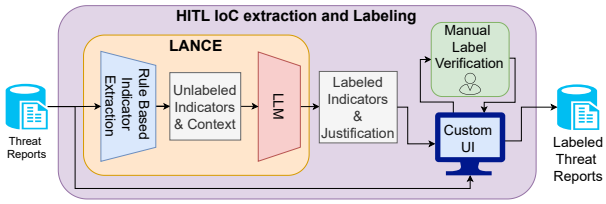

Figure 1: Overview of the human in the loop (HITL) IoC extraction pipeline. It combines automated extraction and LLM-based labeling (LANCE) with manual annotation, thus promoting efficient, high-confidence IoC labeling.

2. Background and Related Work

In this section, we analyze the challenges that affect automated IoC extraction and the implications of IoC misclassifications. We then present the IoC ground truth generation methods used in recent works.

2.1. IoC Extraction Challenges and Previous Work

Extracting Indicators of Compromise (IoCs) from unstructured threat reports is far from straightforward. While IoCs like IP addresses, domains, URLs, and file hashes are vital for threat detection and response, they are often buried in reports that lack consistent formatting. These indicators may appear inline, in tables, or within images, and their relevant context may be scattered throughout the document [8]. This fragmented presentation makes it difficult for automated tools to reliably detect and interpret IoCs.

A further complication lies in differentiating true IoCs from benign artifacts that share similar structure [24], [5]. Regular expression–based extractors, though widely used, frequently overreport by flagging non-malicious entities such as popular domains embedded within malicious URLs [1]. Because maliciousness is context-dependent, identifying IoCs accurately requires understanding context that is not always located near the indicator itself, making it difficult for NLP methods to draw accurate associations.

To address these challenges, a range of extraction methods have been proposed, spanning rule-based approaches [1], [25], NLP [8], [26], [27], machine learning (ML) [28], [29], [30], [31], deep learning (DL) [7], [32], [24], [2], [6], [5], [33], including LLMs [9], [10], [11], and graph-based techniques [34], [35]. Yet each approach has its drawbacks. Rule-based systems lack contextual awareness. NLP models often struggle with inconsistent grammar, distant dependencies, and report structure [29]. ML and DL techniques require large, accurately labeled datasets, which are scarce [8], [6], [2]. LLM-based approaches are prone to hallucinations and are limited by their context window [9], [10], [11], [36], [37].

The most reliable and commonly used method remains manual labeling [1], [25], [8], [27], [29], [28], [31], [5], [7], [2], [32], [33], [6], [35], [34]. Unfortunately, despite its higher accuracy compared to automated extraction, it is expensive, time-consuming, and does not scale to meet the growing volume of threat intelligence [1]. Similarly, Cyber Threat Intelligence (CTI) platforms, which aim to streamline

indicator sharing, often provide IoCs without essential contextual metadata, such as attack stage or malware behavior, that is necessary for assessing relevance [26].

Our HITL system is different from previous work in that it addresses the above challenges by providing LLM-generated, context-aware IoC labels to human analysts who can quickly validate or correct ambiguous labels with full visibility into the broader document structure.

2.2. IoC Ground Truth Generation Methods

Our HITL approach integrates analyst review into the pipeline to improve IoC labeling fidelity and enable the generation of high-quality ground-truth benchmark datasets.

Incorrect ground truth (GT) data, with mislabeled benign/malicious indicators, harms both training and evaluation of automated systems [38], [39]. Models trained on noisy labels risk learning incorrect patterns, overfitting, or failing to generalize. For example, labeling "github.com" as malicious teaches the model to overreport benign indicators, while missing actual threats reduces future detection capability. Ambiguity in what constitutes maliciousness, often context-dependent, further complicates matters.

In practice, misclassifications carry real-world consequences [40], [41], [42]. False positives increase alert fatigue and waste analyst effort, while false negatives let threats bypass detection. These mistakes degrade the efficacy of threat detection systems and damage trust in automation. Addressing them requires both better extraction methods and access to clean, well-labeled datasets that reflect real operational challenges.

To understand how IoC ground truth has been generated so far by the research community and extract useful lessons, we conducted an extensive survey of peer-reviewed papers focused on extracting IoCs from unstructured threat reports. We selected papers that use diverse techniques for extraction and IoC extraction is one of their main contributions. Table 1 summarizes each method's technique, GT strategy, dataset size, availability of the code and dataset, and usability of any available code. For all the tools for which the code and/or the dataset were not available, we contacted the authors twice in an attempt to obtain them. Our analysis reveals a lack of transparency in GT construction and limited availability of datasets and code, hindering reproducibility and fair comparison.

Based on our extensive literature search, we identified six common GT creation strategies: manual labeling, VirusTotal (VT)-based validation, RegEx-based matching, curated lists, unsupervised methods, and forum-based sources.

Manual Labeling: The vast majority of prior works rely on manual GT creation, with over half using it exclusively. Manual efforts ensure precision but are time-consuming and unscalable. Tools across all categories, including rule-based systems [1], [25], NLP-based systems [8], [27], machine learning (ML) systems [29], [28], [31], deep learning (DL) models [5], [7], [2], [32], [33], [6], and graph-based approaches [35], [34], use manual labels to train or validate their systems. Despite the high precision, this approach lacks scalability.

VirusTotal-Based Validation: More than one third of the works, mainly ML systems [28], [29], [30], [31] but also DL models [24] and graph-based implementations [35], use VirusTotal (VT) to validate extracted IoCs. While VT offers broad coverage, threshold choice is inconsistent across studies, and VT can introduce false positives or negatives [14], [15], [16].

RegEx-Based Matching: Widely used in both extraction and GT creation, regular expressions (RegEx) provide high recall but poor precision due to limited contextual awareness (e.g., IoC Searcher [1]). Most systems use RegEx in conjunction with other methods to improve accuracy [26], [30], [7], [24], [2], [6].

Threat Exchanges and Other Information Sharing Platforms: Some systems enrich their datasets with indicators gathered from forums or open-source threat sharing platforms (e.g., AlienVault) [7], [2], [6], [35]. However, these forums are often noisy and unreliable, requiring extensive filtering and manual verification.

Curated Lists: Static IoC lists, while easy to use, are rarely sufficient alone. They are typically combined with RegEx or manual validation to boost coverage. Tools such as Twiti [29] and CTI View [6] use curated lists for labeling or cross-checking, though the lists' limited update frequency can hinder relevance for emerging threats.

Unsupervised Methods: Techniques like those used in TIMiner [26] leverage context-based heuristics and keyword co-occurrence to identify IoCs. While scalable, these methods are prone to context-related mislabeling.

We evaluate the most prominent ground truth creation methods and combinations of methods in section 5.1.1.

2.3. Investigating Errors of Existing Ground Truth Creation Systems

To understand the errors that common ground truth creation methods make and their sources, we review two representative systems that are often used to collect IoCs found in unstructured threat reports, IoC Searcher [1] and AlienVault [7], [12]. We evaluate the tools on 10 threat reports published in 2024, sourced from AlienVault's database, and parsed manually by two analysts, resulting in 517 confirmed IoCs. IoC Searcher is a state-of-the-art rule-based tool with public code that has been compared to other similar tools by previous work [1], [30], and this evaluation provides a robust baseline for assessing both tools.

IoC Searcher produced 126 false positives. The most common error was "Not Malicious" (52 instances), where benign entities like domains (e.g., "msn.com", "github.com"), IPs, (e.g., "1.1.1.1", "4.2.2.4"), URLs, and hashes were misclassified as IoCs solely based on their format, without considering the surrounding context indicating they were benign. A second major error type involving semantic misclassification ("File Path"), was benign paths flagged as domains/URLs due to superficial string patterns (e.g., "do.zip", "asp.net"). Other types of errors occurred by "TLD" misclassified as IoC domains, indicators pointing to the "Reporting Company" extracted as IoCs, substrings of an actual IoC, and indicators with placeholders or reduced parts in them (e.g., "www.[reducted].com", "microsoft-update-com.github.io/XXX/update.html?id=[GUID"). These results reflect the limits in contextual understanding and formatting variability. Despite these issues, IoC Searcher

Technique	IoC extraction Tool	Ground Truth (GT) Creation Methods						GT Size		Availability		Code Usability
		Manual	VT	RegEx	TI Platforms	Lists	Unsupervised	Malicious	Not Malicious	Code	Dataset	
Rule Based	IoC Searcher [1]	●						29	77	✓	✓	●
	STIXnet [25]	●								✓		●
NLP	ChainSmith [8]	●						6,264				
	TIMiner [26]			✓			✓					
	Xiao [27]	–						435				
Machine Learning	IoCMiner [28]	○	✓					45	297	✓	✓	●
	Gharibshah [31]	–	✓					14,268				
	Twiti [29]	○	✓			✓		50,653		✓		
	IoC Stalker [30]		✓	✓				63,903	439,586	✓		◑
Deep Learning & LLM	Long et al. [5]	–						69,032				
	iACE [7]	●		✓	✓			1,500	3,000			
	AITI [32]	–						1,782	1,782			
	AspIOC [24]		✓	✓				50,235	50,185	✓		
	STIOCS [2]	–		✓	✓			5,922		✓		
	Zhou et al. [33]	–						69,032				
	CTI View [6]	–		✓	✓	✓		17,364				
Graph-based DL	HinCTI [35]	◑	✓		✓			11,340				
	HINTI [34]	–						30,000		✓		

TABLE 1: Summary of IoC extraction tools and their ground truth (GT) creation methods, grouped by extraction technique. The *Manual* column denotes the extent of manual labeling, with ● indicating extensive and clearly documented manually labeling, ◑ partial manual labeling with documentation, ○ minimal manual labeling with documentation, and "–" a claim of manual labeling but without sufficient explanation. Check marks (✓) in the *GT Creation Methods* columns indicate that the paper employed the corresponding method. We report malicious and non-malicious samples' GT sizes where available. ✓ in *Availability* indicates publicly available code and/or dataset. A ● in the *Usability* column marks that we successfully used the tool, while a ◑ indicates that the tool was usable but out of scope (e.g., works on input types that do not include threat reports).

showed high recall, missing only 9 indicators. Most errors stemmed from newlines splitting indicators, incorrect type detection, or partial matches ("Found Longer Variant").

Most of the 64 false positives in AlienVault's dataset were indicators that did not appear in the reports, often due to inferred relationships like alternate file hashes (e.g., SHA1→SHA256). Even though these indicators might be useful, they do not appear in the report and are difficult to validate. In terms of false negatives, the dataset missed 67 IoCs. Among these, 60 were entirely missing, 6 were split across lines, and 1 was only partially matched. These results highlight that even widely used platforms like AlienVault suffer from coverage gaps and labeling inconsistencies.

Overall, this analysis illustrates the key limitations of automated IoC extraction tools: poor context handling, formatting brittleness, and inconsistent labeling. These challenges motivate our hybrid human-in-the-loop (HITL) design, which combines automated context-aware suggestions with validation from analysts, thus avoiding these pitfalls and producing a more precise and high-quality dataset.

3. IoC Labeling Methodology

In this section, we present our methodology for labeling IoCs extracted from unstructured threat reports. We begin by formally defining the problem, the goals of our methodology, and the constraints that guide our human-in-the-loop design. We then describe LANCE, our LLM-based system for generating explainable IoC labels, detailing its architecture. Next, we introduce the custom user interface developed to support analyst interaction, enabling efficient validation and correction of LLM-generated labels. Finally, we explain the full dataset creation process.

3.1. Problem Definition, Goals, and Constraints

In Section 2.2 we examined the ground truth creation methodology of 18 prominent IoC extraction papers, analyzing the transparency of their methods, the availability, and usability of their code and datasets, as summarized in Table 1. We observed significant variance in how ground truth was created. Manual labeling was often poorly documented, making it unclear what role human input played in the dataset creation process. Furthermore, only one paper made both its code and dataset publicly available, with one more sharing them upon request. These limitations have been echoed in recent studies [43], [1], [2], [8], [4].

The lack of transparency in ground truth construction, combined with the unavailability of the underlying datasets, severely impedes reproducibility, fair tool evaluation, and comparison. This not only slows progress in the field but also diminishes trust in automated IoC extraction systems due to unclear validation paths and opaque data pipelines.

To overcome these limitations, we introduce a novel IoC extraction and labeling methodology based on a Human-in-the-Loop (HITL) framework, which we use to construct the first openly available and thoroughly documented benchmark dataset for IoC extraction from threat reports, called **PRISM**. Our HITL pipeline integrates large language model (LLM)–generated labels with analyst oversight, enabling both high-precision labeling and significant reductions in human effort. Rather than treating human labeling and automation as mutually exclusive approaches, we combine them into a cohesive workflow in which explainable LLM predictions guide human decisions, and human validation resolves ambiguity.

Our methodology aims to achieve these main objectives:

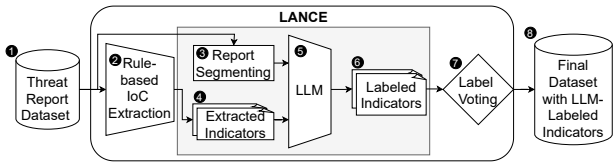

Figure 2: Overview of the LANCE pipeline. Indicators are extracted using regular expressions, labeled by an LLM using contextual report segments, and finalized through a voting mechanism to resolve overlapping predictions.

- **High Accuracy:** The dataset must minimize both false positives and false negatives to ensure fidelity.
- **Manual Work Factor Effectiveness:** The labeling workflow should reduce total analyst hours without compromising label quality.

3.2. LANCE

To streamline the IoC extraction and labeling process, so we can minimize the time the analysts need to spend on the task, we developed LANCE: LLM-Assisted Notation and Classification Engine, a tool that automatically extracts and labels Indicators of Compromise (IoCs) from unstructured threat reports. An overview of LANCE can be seen in Figure 2. The extraction phase utilizes a rule-based approach using regular expressions, while the labeling is performed by a Large Language Model (LLM). Explainability is central to our HITL design. By requiring the LLM to justify each classification decision, we shift the downstream analyst's role from being a manual extractor to that of an informed validator.

3.2.1. Rule-based Extraction. Regular expression-based tools are highly effective in achieving high recall for IoC extraction but struggle with precision as discussed in Section 2.3, as they cannot differentiate between malicious and non-malicious indicators. Given the above, we employed regular expressions for the extraction step to ensure broad coverage of potential IoCs.

While RegEx-based tools like IoC Searcher maximize recall, they lack contextual understanding. In our pipeline, this extraction phase is intentionally broad: it ensures full coverage so that no potential IoC is missed, with human and LLM refinement steps later correcting for over-inclusion. This supports a recall-first strategy where precision is restored through layered validation.

For this purpose, we adopted IoC Searcher, a regular expression-based tool proposed by Caballero et al.[1], and integrated it into our pipeline (Figure 2 part 2). As discussed in Section 2.3, IoC Searcher demonstrates high recall across various IoC types, making it well-suited for the initial extraction phase [1], [30]. We used the tool without modifications, as it has been evaluated in prior research and outperformed other regular expression-based IoC extraction tools [1].

3.2.2. LLM-based Indicator Labeling. For the labeling phase, we are employing a conversational LLM with zero-shot learning. Each report is processed iteratively for each IoC type, with the LLM classifying extracted indicators as either *IoC* or *nonIoC* based on their surrounding context.

To handle long reports, we implemented a rolling window approach with a segment size of approximately 8000 characters (Figure 2 part 3), ensuring splits occurred at the nearest whitespace to preserve readability. This approach has been shown to improve the performance of LLM [44], [45]. To mitigate content loss at segment boundaries, we introduced a 50% overlap between consecutive segments. This implementation ensured that no critical context was lost in segmentation [46].

For each segment, we provide the LLM with the segment text along with a list of all extracted indicators (of the target type) from that segment (Figure 2 parts 3 and 4). We then prompt it to classify each indicator as IoC or nonIoC and provide a justification for its label (Figure 2 part 5). Including justifications enhances the labeling process's interpretability, offering insights into why a particular indicator was classified as malicious or benign. This level of explainability is crucial for understanding the decision-making process and enables better auditing of the output.

Since some indicators appeared in multiple segments, we implemented a voting mechanism to consolidate classifications across the entire report (Figure 2 part 7). The voting thresholds and the ratio of *IoC* to *nonIoC* labels required for a final classification were treated as hyperparameters. We empirically fine-tune the voting threshold for each type (see Appendix A).

Regarding the prompts given to the LLM, after extensive iteration and error analysis, we finalized a robust structure that improves both precision and model reliability across IoC types. The final prompts consist of the following components:

- Specify the role of the LLM.
- Clear definition of what the input is.
- Clear definition of the classification task.
- Explicit definitions for "IoC" vs "nonIoC".
- Descriptions of common false positives/negatives.
- Clear definition of the justification task.
- Explicit instructions to be careful and thorough.
- Specification for the expected output format.

These design elements significantly reduced error rates, particularly in complex cases, including domains and URLs, where indicators often resemble benign artifacts or contain embedded nonIoC substrings. A detailed analysis of the prompt engineering phase and evaluation of the prompts can be found in Appendix A.

3.3. Custom User Interface

Once LANCE-labeled IoCs are extracted, they are shown to the analyst in context within the related threat report document, alongside LLM-generated label explanations. This structure reduces human effort and allows experts to focus only on the most ambiguous cases with full visibility into the model's reasoning.

To this end, we developed a custom user interface (UI) to assist analysts in the label verification/correction task. In the UI, the analysts are able to load a threat report for processing. All indicators (IPs, URLs, domains, and hashes)

extracted by LANCE are highlighted with a color-coded scheme, making them easily visible to the analysts [21], [22], [23]. The colors we use are either green, for non-IoC indicators, or red for indicators labeled as IoC. A view of the User Interface can be seen in Appendix E, Figure 12.

The user interface provides the analyst with the LANCE-generated label and justification. The analyst has the option to either accept or override the generated label. The interface also enables analysts to view justifications by hovering over indicators as well as adding additional comments for future reference. After all labels are finalized by the analyst, they can be exported in a structured format along with all justifications and additional comments optionally provided by the analyst.

The user interface is not merely a visualization tool. It operationalizes the human-in-the-loop paradigm by making model predictions transparent, actionable, and easy to confirm or override. This design reduces annotation effort and supports our goal of a scalable expert-guided validation.

4. PRISM Dataset Creation

We now describe how we use the HITL system described earlier (Figure 1) to create our PRISM dataset. The labeling pipeline we use is depicted in Figure 3. We start by describing our ground truth generation methodology with human vetting, and then discuss our concrete application of this methodology to the PRISM dataset. We describe PRISM first here, and later (in Section 5) use the manually verified IoC labels obtained during this dataset creation process to evaluate LANCE's accuracy by measuring its level of agreement with expert threat analysts.

4.1. Methodology Overview

For the dataset creation, given a set of threat reports, we first divide them into two equal subsets, ensuring a balance in average report length, indicator density, and distribution of indicator types. The two sets are used in two separate manual labeling pipelines: the Baseline Annotation Pass (BAP) and the Guided Annotation Pass (GAP). A detailed analysis of each is given in this section.

All reports are processed through LANCE to extract and label indicators. For the creation of PRISM, we assigned each report to junior analysts who parsed the report independently to facilitate comparison and identify discrepancies.

To further increase the quality of the PRISM dataset as well as to evaluate the effects LANCE had on the junior analysts, we also employed a senior analyst. The senior analyst annotated only the reports that had indicators that the junior analysts disagreed on. This way, we minimized the time the senior analyst needed to spend on labeling indicators and used their expertise only on the difficult-to-label indicators.

4.1.1. Baseline Annotation Pass (BAP). For this part, we gave the first set of reports to the junior analysts, with the goal of labeling the indicators in the reports as "IoC" or "nonIoC". The reports were given to the analysts through the user interface described in Section 3.3, with the crucial difference that the analysts could not see the LANCE-generated labels and justifications. In each report, they were able to see all the indicators highlighted in the report with a neutral color (blue). A view of the User Interface used for BAP can be seen in Appendix E in Figure 13. The analysts were able to assign and change the label of each indicator in an easy and quick way.

The junior analysts were instructed not to use any external sources that would help them choose the label of an indicator, and to rely on the context in which the indicator appears in the report. The types of indicators that were targeted were explained to the analysts so they could avoid confusion with indicator-like strings that the regular expression part of LANCE might have extracted. In the cases of an indicator appearing more than once in a report, the label assigned by the analyst to that indicator appeared in all instances of the indicator.

A short video tutorial of the User Interface was provided, as well as written user instructions and task details. Finally, after labeling all the indicators of a report, the labels were stored in a JSON file.

The junior analyst received the following instructions:

- Label all extracted indicators (domain, IP, URL, HASH) in each of the assigned reports as **IoC** or **nonIoC** using the web interface.
- An **IoC** is an indicator that suggests compromise or malicious activity based on the context provided in the report.
- Complete labeling for each report in **one sitting**, but you may take breaks between reports.
- **Do not** use the internet or external resources. Instead, base your decisions only on the report content.

After the end of the junior analysts' task for BAP, we used the labels from the junior analysts to find *disputed* indicators. We define a *disputed* indicator as an indicator for which there was no total agreement between the labels of the analysts and the LANCE-generated label. For example, if LANCE and a junior analyst agree that an indicator should be labeled as an "IoC" but another junior analyst labeled that indicator as "nonIoC", then this is a *disputed* indicator.

The reports that included at least one disputed indicator were tagged as disputed reports and were given to a senior analyst to determine the correct label of the indicators. These reports were given to the senior analyst through the same UI and with the same instructions as it was given to the junior analysts.

The senior analyst was tasked to label all indicators in the disputed reports, not only the disputed indicators. This way, we could see if there were other indicators that were labeled falsely by both the junior analysts and the LLM.

4.1.2. Guided Annotation Pass (GAP). For the second part of the ground truth creation setup, we used the second set of reports. The setup for GAP was similar to that of BAP with the exception that the analysts did have access to the labels that were assigned to the indicators by LANCE, as well as the justifications that LANCE provided for this choice. This allowed the junior analysts to be "advised" by LANCE while labeling each indicator.

The junior analysts did not have access to any source of information other than the report and LANCE-generated labels and justifications.

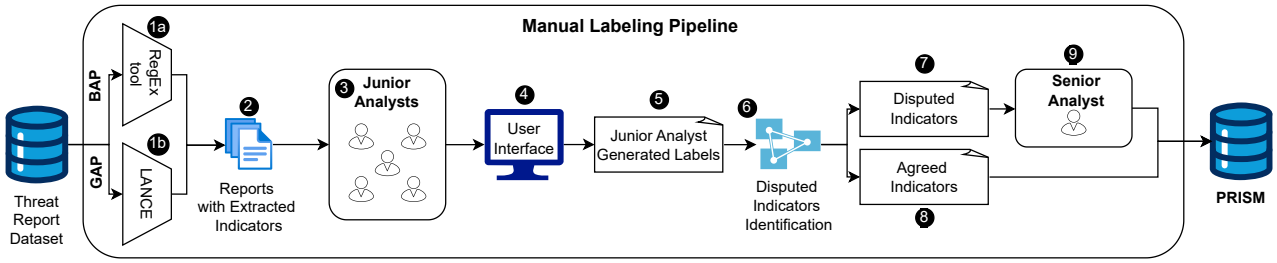

Figure 3: Structure of the manual annotation process used to create the PRISM dataset. The process includes two phases: the Baseline Annotation Pass (BAP), where analysts label indicators without assistance, and the Guided Annotation Pass (GAP), where analysts review indicators pre-labeled by LANCE along with justifications. In both phases, labels from junior analysts are compared, and any disagreements are resolved by a senior analyst. The final consensus labels are incorporated into the PRISM dataset.

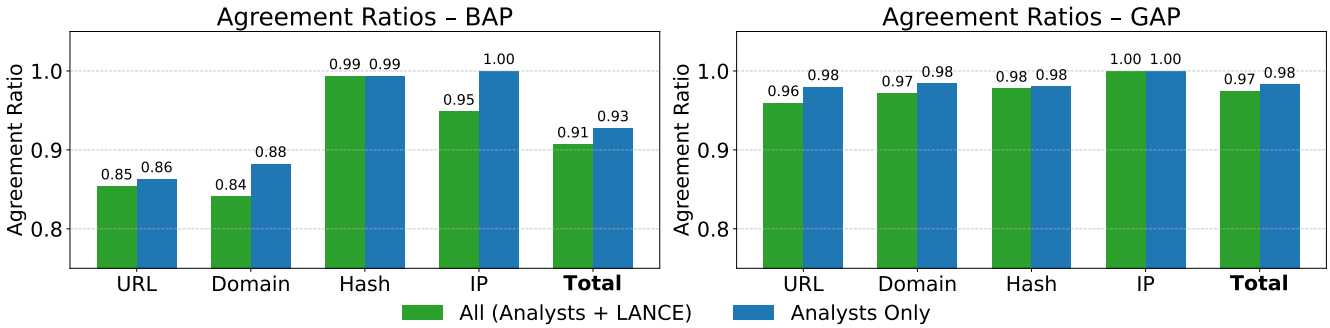

Figure 4: Agreement ratios across IoC types during the manual annotation process. The left plot shows results from BAP (Baseline Annotation Pass) and the right plot from GAP (Guided Annotation Pass). Each bar pair reflects agreement between the junior analysts (Analysts Only) and including LANCE (All).

The junior analyst received the following instructions in this part:

- Review the assigned reports where indicators (domain, IP, URL, HASH) have already been labeled by the system, along with justifications.
- An **IoC** is an indicator that suggests compromise or malicious activity based on the context provided in the report.
- Identify and correct any incorrect labels, providing a short comment for corrections where necessary.
- Complete labeling for each report in **one sitting**, but you may take breaks between reports.

The main motivation for this part of the setup was to evaluate the influence the LANCE labels and justifications had on the analysts. Given that the LLM might have more information about certain indicators, we expected it to be able to "persuade" the analysts that its label was correct without changing their minds when its approach is incorrect.

After annotation, disputed indicators were identified, and the reports that contained at least one disputed indicator were assigned to the senior analyst for final validation.

4.2. PRISM Dataset Details

To construct **PRISM**, our benchmark dataset of labeled IoCs, we first curated a diverse set of 500 threat reports from ORKL [47], an online repository containing over 13,000 reports from 11 reputable sources, including MITRE, APTnotes, CyberMonitor, and Malpedia [48], [49], [50], [51]. These reports span a 20-month period from April 2023 to November 2024 and vary in length, origin, and IoC density.

To ensure relevance and quality, we filtered for reports that also appeared in AlienVault [7], a widely used threat intelligence platform. Specifically, we selected reports published by AlienVault's official user account, which aggregates high-quality reports from vendors such as Kaspersky, Palo Alto Networks, and Microsoft. From this filtered corpus, we identified 50 reports that (i) appeared on both ORKL and AlienVault, (ii) were fewer than 30 pages in length, and (iii) contained more than 5 unique indicators. Although this threshold provides only an upper bound on the number of true IoCs, it serves as an effective proxy for report richness.

These 50 reports were used to construct the ground truth dataset following the annotation methodology described in Section 4.1. To annotate the reports, we recruited five junior analysts-PhD students specializing in cybersecurity, and one senior analyst with seven years of experience in threat hunting.

The 50 reports were divided into two equal subsets of 25 for the two annotation phases: the *Baseline Annotation Pass (BAP)* and the *Guided Annotation Pass (GAP)*. The sets were balanced in terms of average report length, indicator density, and distribution of IoC types. Table 2 shows the total number

of non-unique indicator instances per type for each set.

For the LANCE implementation, we used ChatGPT 4o as the underlying LLM [52]. Each report was independently labeled by two junior analysts. We evenly distributed the workloads in terms of reports, indicator counts, and inter-annotator overlap across the analyst pairs.

Indicator Type	BAP	GAP	Total
IP	177	112	289
Domain	694	729	1423
URL	426	445	871
HASH	962	758	1720
Total	2259	2044	4303

TABLE 2: Analysis of non-unique indicator instances in reports for BAP and GAP, including total sums and per IoC type.

Disagreements between annotators, or between annotators and the LANCE-generated labels, were flagged as *disputed*. We identified 15 reports from BAP that had at least one *disputed* indicator and 9 from GAP. These reports were escalated to the senior analyst, who re-labeled all indicators in those reports to ensure consistency and completeness. Figure 4 shows the agreement ratios between annotators and with LANCE.

Indicator Type	BAP		GAP		Total (IoC + nonIoC)
	IoC	nonIoC	IoC	nonIoC	
IP	97	0	58	3	158
Domain	176	113	105	143	537
URL	156	63	100	47	366
HASH	309	1	400	3	713
Total	738	177	663	196	1774

TABLE 3: Analysis of unique IoC and nonIoC indicators for BAP and GAP, including total sums and per IoC type.

This methodology resulted in 1,774 high-confidence, expert-reviewed labels. The combination of unaided annotation (BAP) and LANCE-assisted annotation (GAP), along with senior analyst arbitration, ensures PRISM is both rigorous and reproducible, offering a high-quality ground truth for fair evaluation and training of IoC extraction systems. Table 3 presents the number of unique labeled indicators, both IoC and nonIoC, for each annotation phase and indicator type.

The Precision, Recall, and F1 scores that the junior analysts achieved compared to the final ground truth can be seen in Figure 5.

5. IoC Labeling Evaluation

In this section, we evaluate LANCE by comparing it with other prominent automated ground truth creation methods. We also evaluate its downstream utility and its ability to generalize across different state-of-the-art LLMs. Finally, we evaluate the proposed HITL pipeline by comparing its impact on junior analysts, specifically in terms of how it influenced their precision, recall, and labeling speed.

5.1. LANCE Evaluation

5.1.1. Comparison with Existing Automated IoC Labeling Systems. We use PRISM, our dataset of IoCs that are manually labeled by multiple threat analysts (see Section 4), to evaluate the most prominent existing automated ground truth (GT) creation methods discussed in Section 2.2, and compare them with LANCE. Specifically, we assess four strategies based on their extraction and labeling performance: (1) IoC Searcher with whitelist-based filtering [1], (2) AlienVault [7], and (3,4) VirusTotal-based labeling with two thresholds [24], [35], [31], [30], [29]. None of the mentioned GT creation methods requires training.

Evaluation Setup: For the RegEx + Whitelist method, we implemented the GT creation pipeline from the GoodFATR paper [1], which supplements rule-based extraction by IoC Searcher with whitelist and frequency-based filtering to reduce false positives. The AlienVault method involved retrieving all IoCs (including inactive ones) linked to each report from the AlienVault platform [7]. All indicators not in the platform were considered unlabeled and treated like non-IoCs. For the VirusTotal-based methods, we followed prior work [30], [29], [24], [31] by extracting indicators with IoC Searcher and labeling them based on VirusTotal lookups. We used two thresholds: *Threshold 1* labels an indicator as IoC if at least one vendor flags it as malicious, and as nonIoC if all vendors deem it benign. *Threshold 5* raises the bar, requiring five or more malicious detections to label an indicator as IoC, while indicators with fewer than five malicious detections are labeled nonIoC. Indicators not included in the database are left unlabeled and are considered as nonIoC

Coverage Analysis: We assume a total of 1,789 candidate indicators, extracted using IoC Searcher, a state-of-the-art rule-based tool [1], [30], [24], [31]. Figure 6 shows the ratio of unlabeled indicators across methods. The RegEx + Whitelist method labels all extracted indicators but still suffers from poor precision due to insufficient context awareness. The AlienVault method yields the most unlabeled indicators (37%). When evaluating the ground truth provided by AlienVault, we also observe that the sum of Labeled and Unlabeled indicators exceeds the total of 1,789. This can be explained by the analysis in Section 2.3. The VirusTotal-based methods leave over 280 indicators unlabeled, suggesting that 16% of the indicators are absent from the VirusTotal database. In contrast, LANCE labeled over 99% of all extracted indicators, with the few unlabeled cases attributed to malformed LLM outputs.

Takeaway: LANCE provides more comprehensive extraction and labeling coverage than other prominent automated ground truth creation methods.

Performance Analysis: Figure 6 summarizes overall precision, recall, and F1 score for each method. We further break down performance across the four IoC types (IP, hash, URL, domain) in Appendix D (Table 7).

The RegEx + Whitelist method achieves almost perfect recall but low precision, particularly for domains and URLs, making it unsuitable for high-fidelity ground truth creation due to excessive false positives. AlienVault shows the lowest recall for URLs (25%) and significant gaps in domain coverage (73% recall), likely due to incomplete extraction. Its precision on hashes is also low (69%) because many labeled hashes do not appear in the report text.

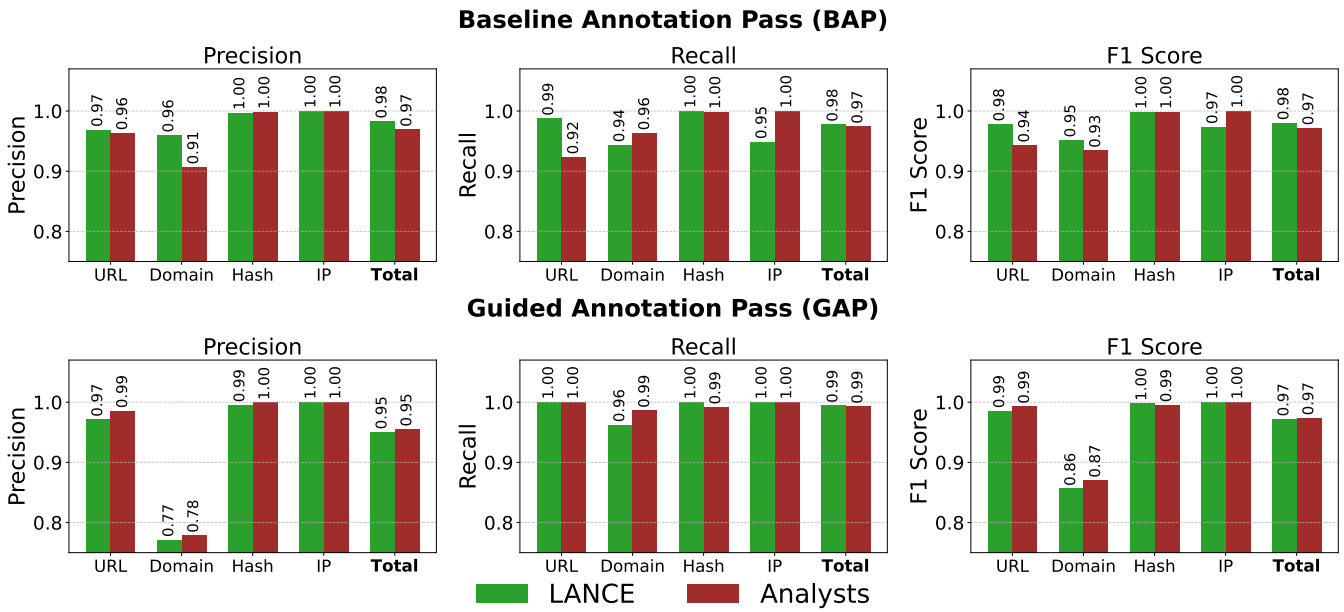

Figure 5: Comparison of LANCE and analyst performance across precision, recall, and F1 score during both annotation phases. The top row shows results from BAP (manual labeling without LANCE assistance), while the bottom row shows GAP (LANCE-assisted annotation).

VirusTotal with threshold 1 performs reasonably well, achieving an average F1 score of 86%, with consistent scores across indicator types. Threshold 5 has increased precision; however, it suffers from low recall due to strict filtering criteria, reducing its reliability for comprehensive dataset creation.

LANCE outperforms all other methods, consistently achieving over 90% F1 score across all types and 97.6% overall. Its lowest score, 86.8% recall on domains, is due to intentional disagreements between LANCE's labels and the senior analyst in cases involving compromised legitimate websites. Notably, LANCE's justifications showed awareness of this nuance, reinforcing the importance of contextual labeling aligned with downstream use cases. Appendix C reports some concrete examples of such scenarios. These discrepancies between LANCE and the senior analyst's labels highlight an important challenge: the definition of what constitutes an IoC can, in some cases, be ambiguous as it may depend on factors such as the time of observation, the current threat landscape, and the specific downstream task where the IoC labels will be used (e.g., detection, attribution, or sharing)[30], [53], [54]. This further justifies the need for manual verification with a human-in-the-loop approach.

Takeaway: LANCE outperforms all other prominent automated ground truth creation methods in total F1 score, maintaining consistently high performance across all IoC types. This makes it the most suitable approach for creating ground truth in high-stakes security contexts.

5.1.2. LANCE Comparison With Naive LLM Prompts. We evaluate how the LANCE pipeline impacts labeling performance by comparing ChatGPT-4o [52] with and without it. For this experiment, we input to ChatGPT the 50 reports included in PRISM and naively prompt it to extract all IoCs from the text. We run four such experiments, one for each type of indicator. Finally, we compare the extraction and labeling results to LANCE. As mentioned in Section 3, we implement zero-shot learning, so there is no training needed.

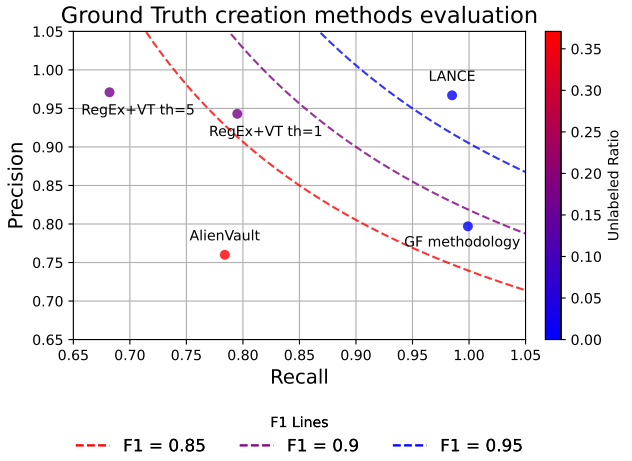

Figure 6: Performance evaluation of prominent Automated ground truth creation methods

As seen in Figure 7, LANCE performs significantly better than the baseline naive ChatGPT in all metrics. Specifically, the baseline approach achieves a 0.669 F1 score in total, with its F1 scores in URLs and Domains being 0.343 and 0.517, respectively. It is evident that the naive

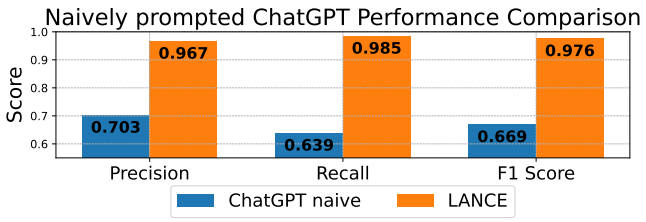

Figure 7: Performance comparison of LANCE and baseline, naively prompted ChatGPT on PRISM.

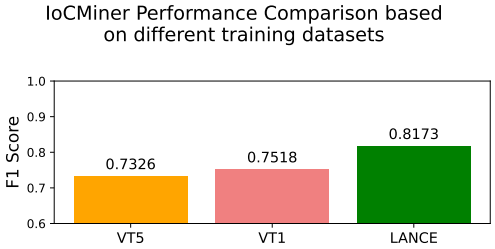

Figure 8: Comparison of IoCMiner's [28] accuracy on PRISM, based on differently created ground truths. The ground truths for this experiment were labeled by VirusTotal with threshold=5 (VT5), VirusTotal with threshold=1 (VT1), and LANCE. We see that the model trained with the LANCE-generated ground truth outperforms the other two.

approach suffers both in extraction and labeling capabilities, which has also been shown in previous work [55]. This demonstrates the necessity of the LANCE approach.

These results show that LANCE is critical in the pipeline. Rather than relying on naive prompting, LANCE strategically orchestrates LLM capabilities via prompt design and context-aware segmentation and achieves high precision on complex threat data. Without this pipeline, the LLM underperforms by 30% in F1 score (as shown in Figure 7).

Takeaway: Without structured guidance, LLMs like ChatGPT 4o underperform in IoC extraction and labeling. LANCE significantly boosts both precision and recall, underscoring the importance of a tailored pipeline.

5.1.3. Downstream Utility Evaluation. To evaluate the usefulness of the IoC labels automatically generated by LANCE, we use it to train and then measure the effect it had on the performance of IoCMiner [28], a Machine Learning-based IoC classification tool. IoCMiner was developed for labeling from tweets. Given that our dataset consists of threat reports, we adapt our data by extracting English sentences that include indicators. We only evaluate its performance on IPs, URLs, and hashes since it was designed to only extract these types.

Out of the 500 reports from ORKL mentioned in Section 4.2 we use 450 for training and the other 50, which are included in PRISM, for testing. To extract English sentences from the reports, we first split the text of the report on periods and then use IoC Searcher [1] to see if an indicator of the targeted type is included in each sentence. For the 450 reports of the training set, we generate labels for the indicator in each sentence using the two highest performing ground truth generation methods (VirusTotal with threshold=1 and threshold=5) as indicated in Figure 6, and LANCE, automatically generating three different training sets (VT1, VT5, and LANCE). We train IoCMiner [28] once for each different training set using the available code and without altering the hyperparameters, and evaluate the three separately trained versions of the model on PRISM. The result of the evaluation can be seen in Figure 8.

We observe that the model trained on the dataset labeled by LANCE achieves a superior F1 score in PRISM than the models trained on datasets labeled by the other two methods. This is consistent with previous evaluations of LANCE, as in Section 5.1.1 we see that LANCE has superior labeling capabilities than Virsus Total with thresholds 1 and 5 and better ground truth enables better-performing models [38], [39]. More specifically, we see that the LANCE-trained model outperforms the VT1-trained model by over 6% and the VT5-trained model by over 8% in terms of F1 score.

Takeaway: Datasets created by the LLM-based pipeline (LANCE) promote better and more robust training than other automated ground truth creation methods.

5.1.4. Generalization Across LLMs. To test the generalizability of our pipeline, we evaluate it across several state-of-the-art LLMs. This not only tests the adaptability of LANCE but also our broader hypothesis: that well-engineered systems can remain effective across model families by abstracting the labeling logic into prompts and input structure.

We tested LANCE on four additional prominent LLMs: LLama 3.3 70b [56], Gemma 3 27b[57], Gemini 2.0 Flash [58], and Nvidia Llama 3.1 Nemotron 70b [59]. We test those LLMs on PRISM ground truth with the same prompts that we developed in section 3.2. For our experiments on GPT and Gemini, we used the provided APIs, while we used the open-source pre-trained models LLama, Nemotron, and Gemma that are available, with quantization, on 2 A40 GPUs with 46068 MB of memory each. To ensure a fair evaluation, we only tested the LLMs on the 25 reports of BAP (where the GPT-based LANCE labels were not available to analysts during ground truth creation), eliminating potential bias.

We evaluated the four LLMs on the BAP-generated part of the PRISM ground truth dataset. The results, summarized in Figure 9, show that Gemma and Gemini perform comparably to GPT, achieving total F1 scores of 0.98 and 0.92, respectively. Llama and Nvidia Nemotron demonstrated moderate performance, which could likely be improved with targeted prompt engineering. A more detailed analysis of this comparison is presented in Appendix B.

In terms of coverage, Gemini and Nvidia Nemotron failed to label only 19 and 22 indicators, representing 2.1% and 2.4% of the dataset, highlighting their strong labeling capability relative to other leading approaches (see Section 5.1.1). Llama and Gemma missed 84 (9.2%)

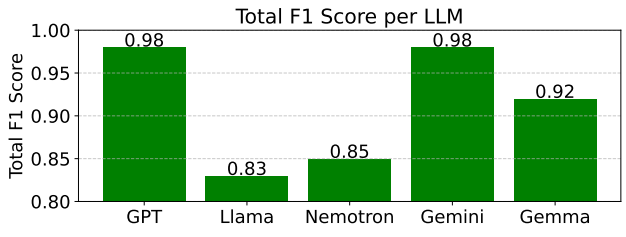

Figure 9: Comparison of the F1 Score of the LANCE implementation using GPT, Llama, Nvidia Nemotron, Gemini, and Gemma on IoC Classification.

and 102 (11.1%) indicators, respectively, indicating lower, though still substantial, labeling coverage.

In terms of processing time per report, GPT and Gemini required approximately 2–3 minutes, while Llama, Nemotron, and Gemma took around 20 minutes due to more limited computational resources. Precise timing comparisons are challenging, as external factors such as network latency and server response times can significantly influence the measurements and are not indicative of the models' inherent performance.

5.2. HITL Pipeline Evaluation

5.2.1. LANCE's Effect on Junior Analysts' Labels. To evaluate the LANCE-generated and the junior analysts-generated labels, we compare the Precision, Recall, and F1 scores they achieved on the 25 reports from the Baseline Annotation Pass (BAP) part of the ground truth generation methodology. We evaluate the results based on the final ground truth labels of those 25 reports. In Figure 5, we see that LANCE performed on par with the junior analysts. In the more difficult indicator types, namely domains and URLs we see an increase of 1.9% and 3.7% respectively in F1 score. We also observed an increase of 1.4% in total Precision, with the same metric increasing by 5.3% in domains. This means that LANCE produced fewer false positive domains than the junior analysts. Finally, we can see that LANCE achieved 6.9% higher recall in URLs, producing fewer false negative URLs than the junior analysts.

Takeaway: LANCE-generated labels exhibit similar levels of precision, recall, and F1 score to those produced by junior analysts.

In Figure 5, we also see the performance the junior analysts and LANCE achieved on the reports from the Guided Annotation Pass (GAP) part of the ground truth creation. In this part, the junior analysts had access to the LANCE-generated labels and justifications during the labeling of the indicators. With the assistance and the additional information provided by LANCE, the junior analysts performed better in total F1 score than LANCE and, by extension, better than without LANCE assistance.

Takeaway: Junior analysts assisted by LANCE-generated labels and justifications achieve higher precision, recall, and F1 scores when labeling IoCs from unstructured threat reports. This demonstrates the practical benefit of LLM-assisted annotation workflows for cyber-threat analysts.

5.2.2. Timing Evaluation. In addition to the Precision, Recall, and F1 evaluation of the analysts (junior and senior), we also evaluated the time variation due to the provided LANCE-generated labels and justification in the User Interface. We do that by automatically measuring the time each junior analyst spent on each report in BAP and GAP of the ground truth creation process.

From the results of those measurements, we see a 43% drop in the average and median time the analysts spent on a report. These findings validate a key goal of our HITL approach: significantly reducing analyst workload without compromising accuracy. The 43% reduction in annotation time, when paired with consistently high precision and recall, demonstrates that LLM-augmented interfaces can streamline expert workflows by offloading routine labeling while still allowing humans to oversee critical edge cases.

6. Conclusion

We addressed the challenge of extracting high-quality Indicators of Compromise (IoCs) from unstructured reports by introducing the first hybrid labeling system that combines explainable LLMs with human-in-the-loop (HITL) validation. Unlike naive LLM prompting or existing automated systems like VirusTotal and AlienVault, LANCE achieves high precision and recall across all IoC types while providing context-aware justifications for the analysts. Our evaluation shows that while junior analysts alone perform well, those assisted by LANCE complete tasks 43% faster with improved accuracy, particularly on ambiguous types of indicators like domains and URLs. This human-augmented workflow enabled the creation of PRISM, the largest publicly available expert-validated IoC dataset from real-world reports. Our results show that a carefully crafted LLM-based system guided by human oversight addresses critical challenges in IoC extraction, and our dataset lays the foundation for more trustworthy and efficient methods in the future.

References

[1] J. Caballero, G. Gomez, S. Matic, G. Sánchez, S. Sebastián, and A. Villacañas, "The rise of goodfatr: A novel accuracy comparison methodology for indicator extraction tools," *Future Generation Computer Systems*, vol. 144, pp. 74–89, 2023. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0167739X23000535

[2] B. Tang, X. Li, J. Wang, W. Ge, Z. Yu, and T. Lin, "Stiocs: Active learning-based semi-supervised training framework for ioc extraction," *Comput. Electr. Eng.*, vol. 112, no. C, Feb. 2024. [Online]. Available: https://doi.org/10.1016/j.compeleceng.2023.108981

[3] M. van der Horst, R. Kho, O. Gadyatskaya, M. Mollema, M. Van Eeten, and Y. Zhauniarovich, "High stakes, low certainty: Evaluating the efficacy of high-level indicators of compromise in ransomware attribution."

[4] W. Tounsi and H. Rais, "A survey on technical threat intelligence in the age of sophisticated cyber attacks," *Computers & Security*, vol. 72, pp. 212–233, 2018. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0167404817301839

[5] Z. Long, L. Tan, S. Zhou, C. He, and X. Liu, "Collecting indicators of compromise from unstructured text of cybersecurity articles using neural-based sequence labelling," in *2019 international joint conference on neural networks (IJCNN)*. IEEE, 2019, pp. 1–8.

[6] Y. Zhou, Y. Tang, M. Yi, C. Xi, and H. Lu, "Cti view: Apt threat intelligence analysis system," *Security and Communication Networks*, vol. 2022, 01 2022.

[7] 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 *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, ser. CCS '16. New York, NY, USA: Association for Computing Machinery, 2016, p. 755–766. [Online]. Available: https://doi.org/10.1145/2976749.2978315

[8] Z. Zhu and T. Dumitras, "Chainsmith: Automatically learning the semantics of malicious campaigns by mining threat intelligence reports," in *2018 IEEE European Symposium on Security and Privacy (EuroS&P)*, 2018, pp. 458–472.

[9] Y. Schwartz, L. Ben-Shimol, D. Mimran, Y. Elovici, and A. Shabtai, "Llmcloudhunter: Harnessing llms for automated extraction of detection rules from cloud-based cti," in *Proceedings of the ACM on Web Conference 2025*, 2025, pp. 1922–1941.

[10] J. Zhang, H. Bu, H. Wen, Y. Liu, H. Fei, R. Xi, L. Li, Y. Yang, H. Zhu, and D. Meng, "When llms meet cybersecurity: A systematic literature review," *Cybersecurity*, vol. 8, no. 1, pp. 1–41, 2025.

[11] J. Liu and J. Zhan, "Constructing knowledge graph from cyber threat intelligence using large language model," pp. 516–521, 2023.

[12] AlienVault, "Alienvault: Unified security management and threat intelligence," 2025, accessed: 2025-01-24. [Online]. Available: https://otx.alienvault.com/

[13] VirusTotal, "VirusTotal - Free Online Virus, Malware and URL Scanner," https://www.virustotal.com/, 2025, accessed: 2025-03-27.

[14] S. Zhu, J. Shi, L. Yang, B. Qin, Z. Zhang, L. Song, and G. Wang, "Measuring and modeling the label dynamics of online Anti-Malware engines," in *29th USENIX Security Symposium (USENIX Security 20)*. USENIX Association, Aug. 2020, pp. 2361–2378. [Online]. Available: https://www.usenix.org/conference/usenixsecurity20/presentation/zhu

[15] A. Salem, "Towards accurate labeling of android apps for reliable malware detection," *CoRR*, vol. abs/2007.00464, 2020. [Online]. Available: https://arxiv.org/abs/2007.00464

[16] E. Choo, M. Nabeel, D. Kim, R. De Silva, T. Yu, and I. Khalil, "A large scale study and classification of virustotal reports on phishing and malware urls," *ACM SIGMETRICS Performance Evaluation Review*, vol. 52, no. 1, pp. 55–56, 2024.

[17] J. Irvin, P. Rajpurkar, M. Ko, Y. Yu, S. Ciurea-Ilcus, C. Chute, H. Marklund, B. Haghgoo, R. Ball, K. Shpanskaya *et al.*, "Chexpert: A large chest radiograph dataset with uncertainty labels and expert comparison," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 33, no. 01, 2019, pp. 590–597.

[18] A. Holzinger, "Interactive machine learning for health informatics: when do we need the human-in-the-loop?" *Brain informatics*, vol. 3, no. 2, pp. 119–131, 2016.

[19] B. Settles, "Closing the loop: Fast, interactive semi-supervised annotation with queries on features and instances," in *Proceedings of the 2011 conference on empirical methods in natural language processing*, 2011, pp. 1467–1478.

[20] H. Su, J. Deng, and L. Fei-Fei, "Crowdsourcing annotations for visual object detection." *HCOMP@ AAAI*, vol. 1, 2012.

[21] C. Ware, *Information visualization: perception for design*. Morgan Kaufmann, 2019.

[22] C. Healey and J. Enns, "Attention and visual memory in visualization and computer graphics," *IEEE transactions on visualization and computer graphics*, vol. 18, no. 7, pp. 1170–1188, 2011.

[23] R. Michalski and J. Grobelny, "The role of colour preattentive processing in human–computer interaction task efficiency: A preliminary study," *International Journal of Industrial Ergonomics*, vol. 38, no. 3-4, pp. 321–332, 2008.

[24] S. Wang, B. Lang, N. Xiao, and Y. Chen, "Aspioc: Aspect-enhanced deep neural network for actionable indicator of compromise recognition," in *International Conference on Information Security*. Springer, 2022, pp. 411–421.

[25] 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*, 2023, pp. 1–11.

[26] J. Zhao, Q. Yan, J. Li, M. Shao, Z. He, and B. Li, "Timiner: Automatically extracting and analyzing categorized cyber threat intelligence from social data," *Computers & Security*, vol. 95, p. 101867, 2020. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0167404820301395

[27] Z. Xiao, "Towards a two-phase unsupervised system for cybersecurity concepts extraction," in *2017 13th International Conference on Natural Computation, Fuzzy Systems and Knowledge Discovery (ICNC-FSKD)*, 2017, pp. 2161–2168.

[28] A. Niakanlahiji, L. Safarnejad, R. Harper, and B.-T. Chu, "Iocminer: Automatic extraction of indicators of compromise from twitter," in *2019 IEEE International Conference on Big Data (Big Data)*, 2019, pp. 4747–4754.

[29] H. Shin, W. Shim, S. Kim, S. Lee, Y. G. Kang, and Y. H. Hwang, "#twiti: Social listening for threat intelligence," in *Proceedings of the Web Conference 2021*, ser. WWW '21. New York, NY, USA: Association for Computing Machinery, 2021, p. 92–104. [Online]. Available: https://doi.org/10.1145/3442381.3449797

[30] M. Mischinger, S. Pastrana, and G. Suarez-Tangil, "Ioc stalker: Early detection of indicators of compromise," in *2024 Annual Computer Security Applications Conference (ACSAC)*, 2024, pp. i–xvii.

[31] J. Gharibshah and M. Faloutsos, "Extracting actionable information from security forums," 05 2019, pp. 27–32.

[32] S. Xun, X. Li, and Y. Gao, "Aiti: An automatic identification model of threat intelligence based on convolutional neural network," in *Proceedings of the 2020 the 4th International Conference on Innovation in Artificial Intelligence*, ser. ICIAI '20. New York, NY, USA: Association for Computing Machinery, 2020, p. 20–24. [Online]. Available: https://doi.org/10.1145/3390557.3394305

[33] S. Zhou, Z. Long, L. Tan, and H. Guo, "Automatic identification of indicators of compromise using neural-based sequence labelling," *CoRR*, vol. abs/1810.10156, 2018. [Online]. Available: http://arxiv.org/abs/1810.10156

[34] J. Zhao, Q. Yan, X. Liu, B. Li, and G. Zuo, "Cyber threat intelligence modeling based on heterogeneous graph convolutional network," in *23rd International Symposium on Research in Attacks, Intrusions and Defenses (RAID 2020)*. San Sebastian: USENIX Association, Oct. 2020, pp. 241–256. [Online]. Available: https://www.usenix.org/conference/raid2020/presentation/zhao

[35] Y. Gao, X. Li, H. Peng, B. Fang, and P. S. Yu, "Hincti: A cyber threat intelligence modeling and identification system based on heterogeneous information network," *IEEE Transactions on Knowledge and Data Engineering*, vol. 34, no. 2, pp. 708–722, 2022.

[36] L. Huang, W. Yu, W. Ma, W. Zhong, Z. Feng, H. Wang, Q. Chen, W. Peng, X. Feng, B. Qin *et al.*, "A survey on hallucination in large language models: Principles, taxonomy, challenges, and open questions," *ACM Transactions on Information Systems*, vol. 43, no. 2, pp. 1–55, 2025.

[37] R. Patil and V. Gudivada, "A review of current trends, techniques, and challenges in large language models (llms)," *Applied Sciences*, vol. 14, no. 5, p. 2074, 2024.

[38] C. Agnew, C. Eising, P. Denny, A. Scanlan, P. Van De Ven, and E. M. Grua, "Quantifying the effects of ground truth annotation quality on object detection and instance segmentation performance," *IEEE Access*, vol. 11, pp. 25 174–25 188, 2023.

[39] L. Budach, M. Feuerpfeil, N. Ihde, A. Nathansen, N. Noack, H. Patzlaff, F. Naumann, and H. Harmouch, "The effects of data quality on machine learning performance," *arXiv preprint arXiv:2207.14529*, 2022.

[40] B. A. Alahmadi, L. Axon, and I. Martinovic, "99% false positives: A qualitative study of SOC analysts' perspectives on security alarms," in *31st USENIX Security Symposium (USENIX Security 22)*. Boston, MA: USENIX Association, Aug. 2022, pp. 2783–2800. [Online]. Available: https://www.usenix.org/conference/usenixsecurity22/presentation/alahmadi

[41] L. Layman and W. Roden, "A controlled experiment on the impact of intrusion detection false alarm rate on analyst performance," in *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, vol. 67, no. 1. SAGE Publications Sage CA: Los Angeles, CA, 2023, pp. 220–225.

[42] Adobe Systems Incorporated, "Security update available for Adobe Commerce — APSB24-40," https://helpx.adobe.com/security/products/magento/apsb24-40.html, June 2024, accessed: 2025-05-29.

[43] D. Olszewski, A. Lu, C. Stillman, K. Warren, C. Kitroser, A. Pascual, D. Ukirde, K. Butler, and P. Traynor, ""get in researchers; we're measuring reproducibility": A reproducibility study of machine learning papers in tier 1 security conferences," in *Proceedings of the 2023 ACM SIGSAC Conference on Computer and Communications Security*, ser. CCS '23. New York, NY, USA: Association for Computing Machinery, 2023, p. 3433–3459. [Online]. Available: https://doi.org/10.1145/3576915.3623130

[44] B. Li, Y. Wang, T. Meng, K.-W. Chang, and N. Peng, "Control large language models via divide and conquer," 2024. [Online]. Available: https://arxiv.org/abs/2410.04628

[45] Z. Zhou, C. Li, X. Chen, S. Wang, Y. Chao, Z. Li, H. Wang, R. An, Q. Shi, Z. Tan, X. Han, X. Shi, Z. Liu, and M. Sun, "Llm×mapreduce: Simplified long-sequence processing using large language models," 2024. [Online]. Available: https://arxiv.org/abs/2410.09342

[46] J. Zhao, Z. Ji, Y. Feng, P. Qi, S. Niu, B. Tang, F. Xiong, and Z. Li, "Meta-chunking: Learning efficient text segmentation via logical perception," 2024. [Online]. Available: https://arxiv.org/abs/2410.12788

[47] ORKL, "Orkl: Open research knowledge library," https://orkl.eu/, accessed: 2025-03-11.

[48] MITRE Corporation, "Mitre att&ck® stix data," https://github.com/mitre-attack/attack-stix-data, accessed: 2025-03-11.

[49] APTnotes, "Aptnotes," https://github.com/aptnotes/data, accessed: 2025-03-11.

[50] CyberMonitor, "Apt & cybercriminal campaign collection," https://github.com/CyberMonitor/APT_CyberCriminal_Campagin_Collections, accessed: 2025-03-11.

[51] F. FKIE, "Malpedia: A collaborative effort to inventorize the malware landscape," https://malpedia.caad.fkie.fraunhofer.de/, accessed: 2025-03-11.

[52] OpenAI, J. Achiam, S. Adler, S. Agarwal *et al.*, "Gpt-4 technical report," 2024. [Online]. Available: https://arxiv.org/abs/2303.08774

[53] A. Villalón-Huerta, I. Ripoll-Ripoll, and H. Marco-Gisbert, "Key requirements for the detection and sharing of behavioral indicators of compromise," *Electronics*, vol. 11, no. 3, p. 416, 2022.

[54] A. Iklody, G. Wagener, A. Dulaunoy, S. Mokaddem, and C. Wagner, "Decaying indicators of compromise," *arXiv preprint arXiv:1803.11052*, 2018.

[55] E. Mezzi, F. Massacci, and K. Tuma, "Large language models are unreliable for cyber threat intelligence," *arXiv preprint arXiv:2503.23175*, 2025.

[56] A. Grattafiori, A. Dubey, A. Jauhri *et al.*, "The llama 3 herd of models," 2024. [Online]. Available: https://arxiv.org/abs/2407.21783

[57] Google, "Gemma 3 technical report," 2025. [Online]. Available: https://arxiv.org/abs/2503.19786

[58] ——, "Gemini 2.0 flash," 2024. [Online]. Available: https://cloud.google.com/vertex-ai/generative-ai/docs/gemini-v2

[59] NVIDIA, "Llama-3.1-nemotron-70b-instruct," 2024. [Online]. Available: https://huggingface.co/nvidia/Llama-3.1-Nemotron-70B-Instruct

[60] H. W. Chung, L. Hou, S. Longpre, B. Zoph, Y. Tay, W. Fedus, Y. Li, X. Wang, M. Dehghani, S. Brahma *et al.*, "Scaling instruction-finetuned language models," *Journal of Machine Learning Research*, vol. 25, no. 70, pp. 1–53, 2024.

[61] T. Kojima, S. S. Gu, M. Reid, Y. Matsuo, and Y. Iwasawa, "Large language models are zero-shot reasoners," *Advances in neural information processing systems*, vol. 35, pp. 22 199–22 213, 2022.

[62] T. Khot, H. Trivedi, M. Finlayson, Y. Fu, K. Richardson, P. Clark, and A. Sabharwal, "Decomposed prompting: A modular approach for solving complex tasks," 2023. [Online]. Available: https://arxiv.org/abs/2210.02406

[63] J. White, Q. Fu, S. Hays, M. Sandborn, C. Olea, H. Gilbert, A. Elnashar, J. Spencer-Smith, and D. C. Schmidt, "A prompt pattern catalog to enhance prompt engineering with chatgpt," *arXiv preprint arXiv:2302.11382*, 2023.

[64] L. Giray, "Prompt engineering with chatgpt: a guide for academic writers," *Annals of biomedical engineering*, vol. 51, no. 12, pp. 2629–2633, 2023.

[65] G. Marvin, N. Hellen, D. Jjingo, and J. Nakatumba-Nabende, "Prompt engineering in large language models," in *International conference on data intelligence and cognitive informatics*. Springer, 2023, pp. 387–402.

Appendix A. LANCE Prompts

This section presents the prompt engineering phase for LANCE as well as the evaluation of the final prompts. For both the prompt engineering phase and the evaluation of the prompts, we used ChatGPT 4o [52] as the underlying LLM.

A.1. Prompt Engineering and Input Structuring

To evaluate prompt performance, we manually extracted and labeled all indicators in 10 reports. These reports were selected from AlienVault with the turgent of ensuring diversity in structure, size, and IoC density. The indicator extraction and labeling process was iterative, as some indicators were missed or misclassified in the initial extraction iteration.

We conducted 27 structured prompt tests to refine our approach. Initially, we tested prompts that processed the entire report while simultaneously targeting all IoC types. However, this resulted in poor performance, including high confusion and inconsistency in model responses. This behavior has been well documented in the literature [60], [61], [62].To address this, we modified the prompt structure to classify each IoC type separately, leading to more reliable and consistent results.

Each iteration involved reviewing model outputs for false positives and false negatives. Based on observed errors, we adjusted prompts for each IoC type to address recurring misclassifications.

Due to the nature of the different types of indicators, as well as the reasons for appearing in a report, some types were more difficult to prompt than others. Specifically, domain labeling was the most difficult task to prompt-engineer for. In our experiments, we came across several domain-like strings that were not actual domains. This was to be expected if taking into consideration the analysis in

Prompt Technique	Description	Example
Role specification	Specify the role of the LLM.	You are a cybersecurity analyst.
Input definition	Clear definition of what the input is.	I will give you part of a cybersecurity report along with some URLs found in that section.
Task definition	Clear definition of the classification task.	Your Task is to label each of the given URLs as either “IoC” or “nonIoC” based on the context that the URL is given in the report.
Definition of Terms	Explicit definitions for “IoC” vs “nonIoC”.	Label as “IoC” the URLs that are Indicators of Compromise (IoCs) related to malicious activities, such as phishing, malware, or other cyber threats, and as “nonIoC” if they are not referenced in the context of IoCs.
Reference to common mistakes	Descriptions of common false positives/negatives.	If a given URL is not complete (eg. is split by a new line or has a placeholder), do not label it as an IoC. Do not return URLs that were not in the given list, but make sure that you return all the URLs in the given list. I want you to make sure that the number of URLs in the “Extracted URLs” list that I give you is the same as the number of URLs you return.
Secondary task definition	Clear definition of the justification task.	I also want you to justify your choice of the label for each URL.
Thoroughness Reminder	Explicit instructions to be careful and thorough.	Go through the report part twice to make sure your labeling and justifications are correct. Make sure that for each URL you followed all the given instructions.
Output format request	Specification for the expected output format.	Return all the URLs, their label, and the justification separated by “,”, one per line, with no additional text. The list might be empty, in that case, return an empty output.

TABLE 4: LANCE prompt structure and example. The example prompt in the corresponding column is the final prompt used for LANCE for the labeling of URLs. The same techniques and structures were used as needed for the rest of the IoC Types.

Section 2.3. In addition, the appearance of nonIoC domains in IoC URLs was very common, making it necessary for the prompt to cover these instances of domains.

The second most difficult type was URLs. The difficulty of labeling URLs arose from the fact that in several cases, there were placeholders in the extracted URLs, as they were included in the threat report as examples. Even though the obvious cases could easily be tackled with rule-based implementations, there were several cases where the example/placeholder part of the URL was not obvious or representable by a specific rule. Thus, we needed to find a way to describe the problem to the model and request it to label as “nonIoC” these URLs.

The prompts for IPs and file hashes were easily engineered in early iterations of the process.

During our tests, there were some occasions when we found that the prompt engineering was not enough to tackle the problem at hand, so we implemented other techniques, such as the rolling window input. We chose a size of 8000 characters, as this is close to the maximum number of tokens that state-of-the-art LLMs can output. Even though a token can include more than one character, we wanted to make sure that in cases where all 8000 characters were a list of important strings (e.g., Hashes), the model would be able to output them properly with their label. To make sure that no information would be lost due to the splitting of the report, we implemented an overlap of 50% in each consecutive report segment [46], [63], [64], [65].

The techniques analyzed in Section 3.2.2 as well as the corresponding parts from the final prompt used for URLs, can be seen in Table 4. By constructing our prompts this way, we address most of the pitfalls the LLM can fall into and steer it away from common inaccuracies.

We used the same set of reports to fine-tune the voting threshold for each indicator type. We initially set the voting threshold at 50% and adjusted it iteratively to optimize performance, ensuring consistency in classification on the prompt training set.

A.2. Prompt Evaluation

For the evaluation of the automated IoC extraction pipeline, we collected the 10 most recent reports added to the AlienVault database from their original user profile. We then manually extracted and labeled the indicators that existed in the text of the reports to create our test dataset.

We evaluated the Precision, Recall, and F1 scores of LANCE and compared them with IoC Searcher. The evaluation can be seen in Figure 10. The Recall of both LANCE and IoC Searcher was 100% in all types of IoCs and in total. LANCE consistently achieves higher precision and F1, particularly on domains and URLs. The values above each bar indicate the exact performance for each indicator type and overall.

Appendix B. LANCE Generalizability

In Table 5 we see a detailed evaluation of LANCE using 5 state of the art LLMs for the LLM component: ChatGPT 4o [52], LLama 3.3 70b [56], Nvidia Llama 3.1 Nemotron 70b [59], Gemini 2.0 Flash [58], and Gemma 3 27b [57]. The table shows the precision, recall, and F1 score the pipeline achieved with each model per indicator type and in total. The total F1 score for each LLM is also shown in Figure 9.

Type	Precision					Recall					F1-Score				
	GPT	Llama	Nemotron	Gemini	Gemma	GPT	Llama	Nemotron	Gemini	Gemma	GPT	Llama	Nemotron	Gemini	Gemma
URL	0.97	0.71	0.80	0.97	0.90	0.99	0.35	0.52	0.98	0.84	0.98	0.47	0.63	0.97	0.87
Domain	0.96	0.83	0.76	0.94	0.75	0.94	0.78	0.78	1.00	0.77	0.95	0.80	0.77	0.97	0.76
HASH	1.00	1.00	1.00	1.00	1.00	1.00	0.86	1.00	0.99	0.99	1.00	0.93	1.00	0.99	0.99
IP	1.00	1.00	1.00	1.00	0.99	0.95	0.99	0.68	0.98	0.95	0.97	0.99	0.81	0.99	0.97
Total	**0.98**	**0.92**	**0.90**	**0.98**	**0.93**	**0.98**	**0.76**	**0.80**	**0.99**	**0.91**	**0.98**	**0.83**	**0.85**	**0.98**	**0.92**

TABLE 5: Comparison of the LANCE implementation using GPT, Llama, Nvidia Nemotron, Gemini, and Gemma on IoC Classification.

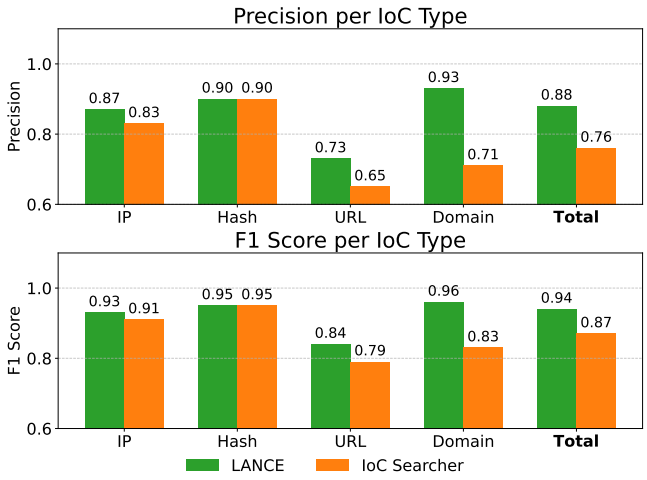

Figure 10: Comparison of precision and F1 score of LANCE and IoC Searcher across multiple IoC types.

Appendix C. Ambiguous IoCs' Labels

In Figure 11 we show a segment of a report which is part of PRISM dataset. Several URLs of legitimate websites were compromised and used as command and control (C2) points for the campaign. According to our senior analyst, the domains (e.g., "abert-online.de") are not IoCs (as they are a legitimate website), whereas the complete URLs are. LANCE labeled these domains as "IoC" and justified the label(see Table 6) by referencing their "*involvement in the malicious activities*." Notably, LANCE acknowledged the domains' legitimate nature but still marked them as IoCs due to their role in the attack. The justifications for the remaining domains in Figure 11 were similar, and all were consistently labeled as IoCs. In this case, both interpretations – labeling as IoC or non-IoC – can be considered correct, depending on the downstream use of the extracted indicators.

Legitimate compromised websites

JackalControl C2
hxxp://abert-online[.]de/meeting/plugins[.]php
hxxp://acehigh[.]host/robotx[.]php
hxxp://assistance[.]uz/admin/plugins[.]php
hxxp://cnom[.]sante[.]gov[.]ml/components/com_avreloaded/views/popup/tmpl/header[.]php

Figure 11: Segment of a report included in PRISM

LANCE justification
The domain abert-online.de is classified as an IoC because it is identified as a legitimate website that has been compromised and used as a command and control (C2) server for JackalControl, indicating its involvement in malicious activities.

TABLE 6: The justification LANCE gave for the "IoC" label it assigned to domain "abert-online.de" which appear in the report segment shown in Figure 11.

Appendix D. Ground truth creation methods evaluation details

In Table 7 we see the performance of the five ground truth creation methods discussed in Section 5.1.1 on coverage as well as Precision, Recall, and F1 score on each type of IoCs (IPs, hashes, URLs, and domains).

Appendix E. User Interface

Figures 13 and 12 show a view of the custom user interface developed for our system. In the proposed user interface (Figure 12), the analyst can see all indicators highlighted by a red or green rectangle. In the cases that the LANCE-generated label is *IoC*, the rectangle is red, while if the label is *nonIoC* it is green [21], [22], [23]. This way, the analyst can quickly understand the generated label. By hovering over the indicator or by selecting the indicator, they can see the justification that LANCE provided for the selected label. The label of each indicator can be changed by the analyst either by selecting the indicator and clicking the designated button or by double-clicking the indicator. The label change propagates to all instances of this indicator throughout the report.

For the baseline annotation pass (BAP), the user interface was altered to not show the designated colors unless the label was assigned by the analyst. The analyst could also not see the label, justification, or any other LANCE-generated information about any indicator. For this pass, we also implemented a counter that showed the remaining unlabeled indicators.

Method		Labeled Indicators	Unlabeled Indicators	IP			Hash			URL			Domain		
				Precision	Recall	F1	Precision	Recall	F1	Precision	Recall	F1	Precision	Recall	F1
RegEx + Whitelists [1]		1789	0	0.987	**1.000**	**0.994**	0.994	**1.000**	0.997	0.720	**0.992**	0.835	0.523	**1.000**	0.687
AlienVault [7]		1464	653	0.936	0.948	0.942	0.686	0.965	0.802	0.855	0.256	0.394	0.928	0.737	0.821
RegEx+VT	th=1 [29], [30]	1502	287	0.993	0.929	0.960	**0.998**	0.674	0.805	0.965	0.863	0.911	0.826	0.961	0.888
[24], [35], [31]	th=5 [30]	1502	287	**1.000**	0.503	0.670	**0.998**	0.626	0.769	**0.995**	0.785	0.878	**0.891**	0.730	0.802
LANCE		1774	15	**1.000**	0.968	0.984	0.996	**1.000**	**0.998**	0.970	**0.992**	**0.981**	0.878	0.950	**0.913**

TABLE 7: Performance comparison of four prominent ground truth creation methods from Table 1, across four IoC types (IP, hash, URL, domain). The RegEx + Whitelists methodology [1] exhibits near-perfect recall but low precision due to excessive false positives. VirusTotal-based methods vary depending on the chosen threshold, but they have a high number of unlabeled indicators. The same can be said for AlienVault, which has the highest number of unlabeled indicators. LANCE achieves the highest F1 score across all indicator types while providing excellent coverage.

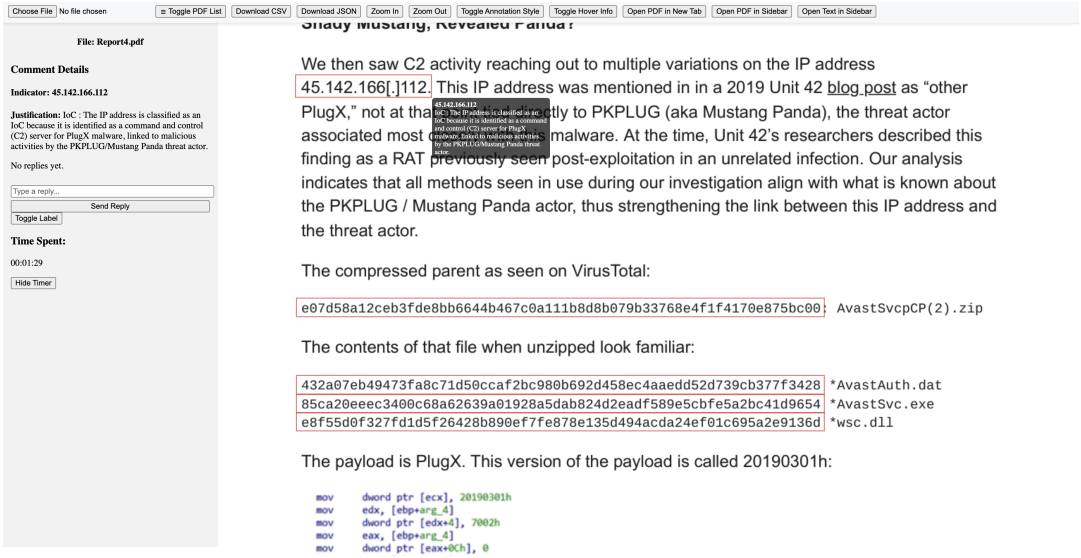

Figure 12: This is a view of the user interface that the analysts used for the GAP of the ground truth creation. Contrary to the User Interface for BAP, the indicators are extracted and labeled by LANCE. The indicators labeled as IoC can be seen in a red rectangle, while those labeled as nonIoC can be seen in a green rectangle. A justification is provided for all indicators.

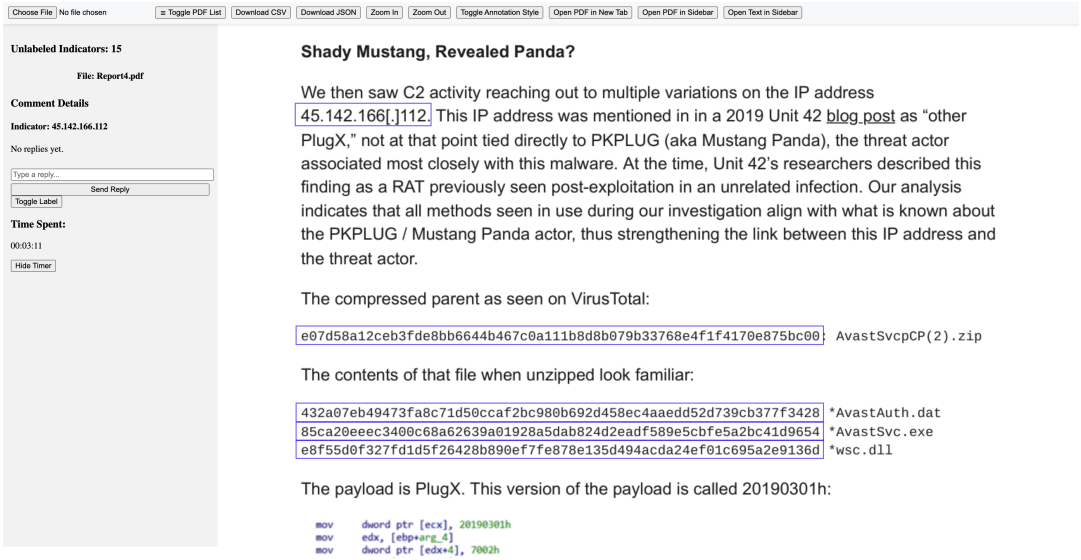

Figure 13: This is a view of the user interface, the analysts used for the BAP of the ground truth creation. The extracted Indicators can be seen marked in a blue rectangle, which changes color according to the label assigned by the analyst.