Poster: Automated Security Property Extraction from Protocol Specifications

Hassan Asghar[†], Myeong-Ha Hwang[‡], Jeonghyun Joo[‡], Heewoon Kang[‡], YooJin Kwon[‡],
Kazi Samin Mubasshir[§], Imtiaz Karim[¶], Elisa Bertino[§], Hyunwoo Lee[†]
[†] KENTECH [‡] KEPRI [§] Purdue University [¶] The University of Texas at Dallas
hassanasghar@kentech.ac.kr, mh.hwang@kepco.co.kr, jh.joo590@kepco.co.kr, heewoon.kang03@kepco.co.kr
kwon43@kepco.co.kr, kmubassh@purdue.edu, imtiaz.karim@utdallas.edu, bertino@purdue.edu, hwlee@kentech.ac.kr

2025 IEEE 33rd International Conference on Network Protocols (ICNP) | 979-8-3315-0376-5/25/$31.00 ©2025 IEEE | DOI: 10.1109/ICNP65844.2025.11192438

***Abstract*—Formal verification of network protocols is a security-by-design approach, but extracting security properties from long, complex specifications remains manual, time-consuming, labor-intensive, and error-prone. We propose SPARTA, a framework that automatically extracts useful security properties from network specifications leveraging large language models (LLMs) and a three-step retrieval-augmented generation (RAG) pipeline.**

***Index Terms*—security properties extraction, large language model, RAG, standard specifications**

I. Introduction & Motivation

Formal verification has been widely adopted as a security-by-design approach to ensure the correctness and robustness of network protocols before they are deployed. By enabling the early detection of subtle flaws and inconsistencies, it helps minimize security vulnerabilities in real-world implementations [1]. A critical step in the verification process is the extraction of security properties from protocol specifications, which are often written in informal and ambiguous natural language. The effectiveness of formal verification highly depends on how reliably, completely, and accurately these properties are identified. In general, this task has been manually conducted by human experts, making it time-consuming and labor-intensive [2]. This motivates automation of property extraction from natural-language specifications.

Automating this step is hard due to three recurring reasons; 1) Scalability: requirements that span hundreds of pages with scattered security details. 2) Heterogeneity / terminology gap—the same concept appears under different names or acronyms (e.g., "MAC" as message authentication code vs. media access control), so keyword search and naive prompts miss passages; and 3) Implicitness: many guarantees are not stated verbatim but must be inferred across sentences and sections. One promising direction for addressing this challenge is to use off-the-shelf LLMs. However, due to the complexity and technical density of specifications, LLMs often struggle to capture subtle or implicit security requirements. They can hallucinate unsupported claims, overlook critical security guarantees, or simplify nuanced properties [3]. Previous approaches have been primarily applied to domains such as smart contracts [4]; therefore, their effectiveness in the context of protocol specifications remains uncertain, limiting their adaptability.

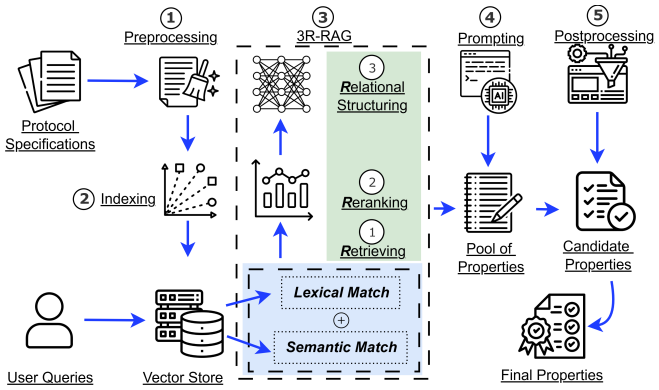

Fig. 1: SPARTA workflow illustrating the modular pipeline from specification to final security properties.

Motivation and Goals. We seek a method that (**G1**) scales to large protocol specifications, (**G2**) is robust to vocabulary variation while keeping outputs traceable to the source, and (**G3**) surfaces implicit constraints with clear, reviewable reasoning rather than opaque heuristics. Guided by these goals, we develop SPARTA, which couples targeted retrieval with structure-aware reasoning to turn informal clauses into concise, source-grounded security properties suitable for downstream formal use.

II. System Overview

SPARTA is a framework that automates the extraction of security properties from protocol specifications by combining: (i) Retrieval, (ii) Reranking, and (iii) Relational structuring with an LLM-guided prompting.

A. High-Level Overview

SPARTA automates the extraction of security properties from network protocol specifications using an LLM guided by a 3R-RAG workflow. SPARTA is motivated by three challenges observed in standards. ❶ To address scalability, SPARTA performs preprocessing to segment the text and indexing to build both lexical and semantic indices, focusing retrieval on the most relevant regions. ❷ To cope with heterogeneity and the terminology gap, SPARTA uses 3R-RAG (Retrieving) with hybrid lexical+semantic search followed by 3R-RAG (Reranking) to validate terminology, capture paraphrases, and assemble a compact, diverse evidence set. ❸ To

Relational Structuring Prompt (CoT Enabled)

Cybersecurity Analyst: You are a skilled cybersecurity analyst extracting security properties from the "<SPECIFICATION_NAME>" specification. Follow these structured steps:

Step1: Identify Security-Related Entities

- Protocols, algorithms, methods (e.g., HTTPS, TLS, cipher suites)
- Devices, roles, components (e.g., client, server, end device, resource)
- Certificates, tokens, keys
- Concepts such as authentication, authorization, confidentiality, access control.

Step 2: Identify Relationships

- Determine interactions and dependencies among the identified entities as stated or strongly implied in the context.

Step 3: Evaluate Each Statement

- Decide if the relationship or clause indicates a security requirement.
- Focus on normative terms such as **MUST**, **SHALL**, **SHOULD**, **RECOMMENDED**.

Fig. 2: Prompt used for relational structuring (IEEE 2030.5).

surface implicit properties SPARTA applies 3R-RAG (Relational Structuring) with chain-of-thought (CoT) prompting that makes intermediate logic explicit. These steps feed a structured prompting stage that asks the LLM to distill well-grounded properties and a postprocessing stage that deduplicates and validates them against the source. The SPARTA pipeline is depicted in Figure 1.

B. Pipeline & Prompts

① **Preprocessing**. The specification is divided into chunks, each of which serves as a potential knowledge passage during retrieval. ② **Index Construction**. For each word, two indices are constructed. One is lexical indices which are used to search exact-matching words while the other is semantic indices which are used to search semantically similar words. ③ **3R-RAG**. This procedure is divided into three steps: (i) *Retrieving*. It is a technique that enhances LLM prompting by supplying relevant context from external sources, typically through keyword-based search [5]. (ii) *Reranking*. It consolidates semantically redundant results into a single representative item while maintaining diversity by promoting content that differs semantically [6]. (iii) *Relational structuring*. Chain-of-thought prompting is a strategy in which an LLM explicitly outlines intermediate reasoning steps before providing a final answer [7], [8]. ③ **Prompting**. SPARTA leverages the language model to interpret the selected specification excerpts and extract security properties. Prompt for relational structuring is shown in Figure 2. ④ **Postprocessing**. Extracted properties undergo semantic deduplication to remove redundant statements and validation against the original specification texts. Security properties generation prompt is shown in Figure 3.

III. Preliminary Results

Dataset. We report a preliminary evaluation of the IEEE 2030.5 protocol specification only. We constructed an expert-curated ground-truth list by reviewing the standard and extracting normative security statements (authentication, confidentiality, integrity, availability, and access control).

Baselines & Metrics. We compared our framework against two baselines: a naive keyword-based extraction baseline that

Security Properties Generation Prompt

Extract Security Properties List **only those lines** that imply or define security properties. *Avoid paraphrasing*. **Reference Example:**
Context:

> "If a client PUTs or POSTs a resource to a server containing attributes or elements that instead are to be populated by the server (e.g., href), the server SHALL return an HTTP 400 error..."

Step-by-step Reasoning:

- **Entities:** client, server, resource, href, HTTP method, ACL...
- **Relationships:**
  - client submits → server verifies → may reject
  - method authorization → validated by ACL
- **Evaluation:** "SHALL" indicates a security requirement (validation, authorization logic)

Extracted Security Properties:

1) "If a client PUTs or POSTs a resource... SHALL return an HTTP 400 error."
2) "The HTTP method of an incoming request is checked..."
3) "Authorization is granted if Method, AuthType, and DeviceType are TRUE..."

Instruction: Now apply this process to the given context: **Context:** <CONTEXT_CHUNK> **Step-by-step Reasoning: Extracted Security Properties:**

Fig. 3: Prompt used for security property generation (IEEE 2030.5).

simulates a straightforward requirements mining methodology, and a vanilla RAG baseline. We use BERTScore to measure semantic similarity between generated security properties and our ground truth properties.

Findings (work-in-progress). On IEEE 2030.5 protocol specification, SPARTA produces properties that align with expert-curated ground truth and shows higher recall than the keyword and vanilla RAG baselines while maintaining competitive precision. Qualitatively, SPARTA captures specification-grounded authorization and validation requirements that the baselines often miss.

Future Work. We plan to extend SPARTA to extract implicit properties and to systematically align formal-model properties (used in prior verification work) with their informal textual counterparts in the specification. This includes targeted query expansion for cross-reference resolution, improved relational structuring for cross-clause reasoning, and broader evaluation to other network protocol specifications.

IV. Conclusions

SPARTA's hybrid Retrieval-Augmented Generation (RAG) pipeline addresses critical challenges associated with mining specifications: it retrieves relevant yet potentially distant contextual information, employs an adaptive query strategy to capture implicit requirements, and directs an LLM using a chain-of-thought prompt to produce clear and consolidated security properties.

Acknowledgment

This work was supported by project R23IA01, "Development of the Information Model Management and Certification System."

References

[1] K. Keerthi, I. Roy, A. Hazra, and C. Rebeiro, "Formal verification for security in iot devices," *Security and fault tolerance in internet of things*, pp. 179–200, 2018.

[2] X. Ma, L. Luo, and Q. Zeng, "From one thousand pages of specification to unveiling hidden bugs: Large language model assisted fuzzing of matter {IoT} devices," in *33rd USENIX Security Symposium (USENIX Security 24)*, 2024, pp. 4783–4800.
[3] H. Li, Z. Dong, S. Wang, H. Zhang, L. Shen, X. Peng, and D. She, "Extracting formal specifications from documents using llms for automated testing," 2025. [Online]. Available: https://arxiv.org/abs/2504.01294
[4] Y. Liu, Y. Xue, D. Wu, Y. Sun, Y. Li, M. Shi, and Y. Liu, "Propertygpt: Llm-driven formal verification of smart contracts through retrieval-augmented property generation," *arXiv preprint arXiv:2405.02580*, 2024.
[5] "GitHub - facebookresearch/faiss: A library for efficient similarity search and clustering of dense vectors. — github.com," https://github.com/facebookresearch/faiss, [Accessed 15-08-2025].
[6] Y. Mao, Y. Qu, Y. Xie, X. Ren, and J. Han, "Multi-document summarization with maximal marginal relevance-guided reinforcement learning," *arXiv preprint arXiv:2010.00117*, 2020.
[7] X. Wang, J. Wei, D. Schuurmans, Q. Le, E. Chi, S. Narang, A. Chowdhery, and D. Zhou, "Self-consistency improves chain of thought reasoning in language models," 2023. [Online]. Available: https://arxiv.org/abs/2203.11171
[8] J. Wei, X. Wang, D. Schuurmans, M. Bosma, B. Ichter, F. Xia, E. Chi, Q. Le, and D. Zhou, "Chain-of-thought prompting elicits reasoning in large language models," 2023. [Online]. Available: https://arxiv.org/abs/2201.11903