5G Specifications Formal Verification with Over the Air Validation: Prompting is All You Need

Tom Wray
Stevens Institute of Technology, Hoboken, NJ
thomas.wray@L3harris.com

Ying Wang
Stevens Institute of Technology,Hoboken, NJ
ywang6@stevens.edu

***Abstract*—The critical role of 5G and other complex systems in infrastructure necessitates rigorous protocol verification and system validation to ensure security and reliability. This paper explores the application of applying Large Language Model enabled auto Formal Verification with Real-world Prompting on Large Language Models (LLMs) for 5G and NextG protocols, addressing ambiguities and security concerns in network infrastructure protocol and specification design. By leveraging generative transformer-based LLMs, we present a formal approach to prompt engineering that validates complex specifications and implements formal verification techniques to detect and eliminate hallucinations. Our approach is agnostic to specific LLMs, with performance comparisons across currently popular models. We thoroughly examine the human processes involved to identify entry points where Prompt Engineering can reduce process overhead. We have developed a novel framework for iterative prompting and self-monitoring to aid in formal verification using 5G reasoner, enabling closed-loop automatic 5G protocol verification. Focusing on the RRC layer of 5G release 17, specifically sections 5.3.3.3, 5.3.3.4, and 5.3.5.3, we examined the liveness properties and detected a total of seven vulnerabilities, including variations of Null Cipher, Denial of Service (DoS), Lullaby, and Incarceration attacks. Further, we established a general testing framework that spans conception, virtualization, and over-the-air testing, providing a holistic approach to security assessment. This comprehensive framework underscores the importance of robust protocol verification and system validation in the deployment of critical infrastructure technologies.**

***Index Terms*—5G, Artificial Intelligence, Prompt Engineering, Large Language Models, Security**

I. Introduction

The 3rd Generation Partnership Project has led the development of telecommunication standards for over twenty years. Initially focused on producing Technical Specifications and Technical Reports for 3G-based systems, 3GPP's scope has since expanded to include specifications for 4G, LTE, and 5G. The combination of technical acceleration, utilization of telecommunications, and time to market has made these specifications quick to release and slow to review. This often results in a gap between the initial interception of the protocol and its actual implementation and testing. Existing research has exploited this gap to carry out numerous attacks on end users.

With the increasing reliance on next-generation telecommunications for mission-critical technologies, the impact of exploited security protocols has become significantly more profound. For example, in the context of autonomous vehicles, a Denial of Service (DoS) attack could have catastrophic consequences. Such an attack could disrupt the vehicle's communication systems, potentially leading to fatal outcomes for individuals interacting with the technology [1]. The heightened integration of advanced telecommunications in critical applications indicates the urgent need to address security vulnerabilities and ensure robust protection against potential threats.

Due to the rapid time-to-market, many specifications are in a constant state of review and updates, leaving opportunities for unresolved exploits. Formal verification techniques, such as model checking, theorem proving, and symbolic execution, are essential for ensuring the reliability and security of 5G protocols. Tools like SPIN [2], UPPAAL [3], and 5GReasoner [4] address protocol vulnerabilities. However, the complexity and scalability of 5G protocols pose significant challenges.

5GReasoner, using Finite State Machines (FSMs) and combining Model Checker (MCheck) with Cryptographic Protocol Verifiers (CPVerif), effectively checks authenticity, availability, integrity, secrecy, and replay protection [4]. LTEInspector, a predecessor, focused on a single protocol layer [5]. Tools like NuXmv and Kind 2 are used for safety and liveness properties. ProVerif and Tamarin have also been effective in verifying complex security properties in protocols.

Recent research efforts have further enhanced vulnerability detection for 5G protocols. For instance, in [6] the authors proposed a Listen-and-Learn framework to identify vulnerabilities in the 5G Radio Resource Control (RRC) protocol stack. They also developed a multi-fuzzing approach that adapts prior knowledge to discover unintended 5G vulnerabilities [7] [8], focusing on runtime profiling for 5G vulnerabilities. Dauphinais et al. in [9] [10] introduced an Automated Vulnerability Testing and Detection framework for 5G systems, employing a digital twin methodology. Additionally, in [11], formal-guided fuzz testing are explored to enhance security assurance from specification to implementation in 5G and beyond networks.

Existing Large Language Models (LLMs) have shown promise in converting informal natural language system designs and protocols into formal descriptions, as demonstrated by Cosler et al. [12]. These models can generate formal specifications from unstructured natural language, addressing complexity and scalability challenges in 5G verification. In [13], the authors further demonstrated that LLMs can automate the modeling process and enhance the efficiency of formal

verification through advanced attention mechanisms. Similarly, Peng et al. [14] developed a deep learning framework to evaluate fuzz testing performance for NextG vulnerability detection. Other notable contributions include RAFT, a real-time framework for root cause analysis in vulnerability detection [15]peng2023smile. [16] also introduced formal analysis techniques for 5G Authentication and Key Agreement (AKA) protocols using dependency graphs for assumption propagation and verification.

However, the use of sampling methods like top-k and nucleus sampling introduces non-deterministic outputs and variability, complicating iterative formal verification and reducing precision [17]. Despite advancements such as leveraging regular expressions in deep learning frameworks [18], achieving complete accuracy remains challenging. More deterministic and supervised approaches are needed to improve reliability and effectiveness in formal verification tasks.

This paper aims to utilize prompt-based LLMs to interpret complex text accurately, integrating them with formal verification to thoroughly examine potential vulnerabilities in telecommunication specifications and protocols. To illustrate this clearly and within the scope of this paper, the 5G NR documentation examined is the 3GPP RRC protocol specification [19]. The integrated formal verification tool used is 5GReasoner. However, this approach could be applied to a broader range of formal verification systems and expanded to large-scale 5G or next-generation specifications, as well as other types of telecommunication protocols.

The contributions of this paper are summarized as follows:

- Leveraging LLMs to enable prompt-based automatic identification of properties in complex 5G protocols, which are then used by 5GReasoner for formal verification and vulnerability detection.
- Development of a novel framework for iterative prompting and self-monitoring, enhancing the formal verification process of 5G protocols, specifically in the RRC layer of 5G release 17.
- Identification and demonstration of seven vulnerabilities, including Null Cipher, DoS, Lullaby, and Incarceration attacks, were validated and thoroughly analyzed for impact and risk using over-the-air experimental platforms
- Establishment of a comprehensive testing framework from virtualization to over-the-air testing, ensuring robust protocol verification and system validation.

II. System Design

The goal established was to support the formal verification and validation (V&V) of complex telecommunication standards. Two possible approaches were identified: a prompt engineering approach and a software engineering approach. The software engineering approach required a multi-system approach combining LLMs with iterative feedback building from previous works [20]. This approach produced exciting results, but this paper focuses on a Prompt-based approach. We first used industry opinion to outline the steps a human needs to take to fully realize the V&V of a telecommunication specification. After outlining the human interaction, we needed entry points to reduce the time needed to formally validate specifications.

A. Prompt-Based Inspector Aided Formal VErification (PAVE)

An extrapolated view of the proposed Prompt-Based Inspector Aided Formal VErification (PAVE) is shown in Fig.1. Our framework is structured into four primary components, each clearly delineated to ensure a robust separation of subsystems. Building upon existing research, this framework progresses from conformance tests and specifications to virtual and real-world validation comprehensively.

The initial stage involves gathering requisite documentation, which demands significant human effort due to challenges such as the context window's limitations and the large language model's susceptibility to focus deviations based on prompt injection locations. Careful curation of documentation is essential. For example, focusing on the `RRC_setup_complete` command might render the inclusion of an erroneous conformance test due to the model's response variability. The chosen entry point for LLMs is deciphering complex specifications to facilitate protocols that can be inspected for vulnerabilities.

The state-of-the-art process of manually converting 3GPP documentation into verifiable guarantees and protocols using the SMV model checker encounters significant scalability issues. The advent of PAVE enables the generation of verifiable outputs at the virtualization layer. Constructing the desired output entails techniques detailed in the prompt construction section, leveraging iterative prompting and meta-language. Our presented approach yields outputs amenable to analysis via the NuXmv model checker, revealing potential vulnerabilities.

Transitioning from virtualization to the real-world over-the-air testbed is a crucial phase in our framework. Real-world validation offers deeper insights into system resilience and unveils additional vulnerabilities within the same attack vector family. Characterization of attacks in the real world involves injecting a man-in-the-middle attack at the RRC layer. This allows a malicious actor to tamper with command messages exchanged between the User Equipment (UE) and the Serving Radio System (SRS) Radio Access Network (RAN). We connected the output of the formal verification to our existing experimental fuzzing platform, 'HyFuzz' [10]. The integration of theoretical formal verification with over-the-air (OTA) validation provides a thorough inspection of attacks and accounts for emergent system behaviors.

Prompt Based Formal Properties Identifications: Large Language models are multi-use tools capable of great feats, but require complex architecture to reach greater heights. Despite significant progress, LLMs exhibit several limitations in technical applications including; bias, limited contextual understanding, and hallucinations. Although LLMs possess a broad range of capabilities, the core functionality lies in the ability to "guess the next word" given a prompt. This limits the ability to perform logical operations, but with iterative feedback, the LLM can reach 100% accuracy in smaller scoped

Figure/system Diag_1.png

Fig. 1. Mobilized System Design of Prompt-Based Inspector Aided Formal Verification (PAVE), facilitated with four connected components: 1. Specification Preparation, 2. Prompt-Based Inspector, 3. Formal Verification, and 4. Over-the-Air Validation. Prompt-Based Inspector replaces human conversion of specifications to formal properties.

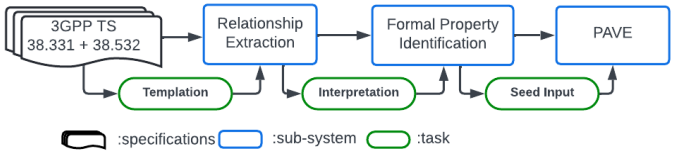

Fig. 2. Process in generating seed examples for PAVE. High level properties are extracted from TS38.532 and TS38.331. Formal Properties and guarantees are then generated to be used as examples for a multi-shot prompt inside PAVE.

tasks. [20] Due to this inaccuracy factor, the output must be closely monitored and iterated as needed.

The main entry point discovered for the LLM was to analysis test specifications and conformance tests, identify security related ones, and output them in a manner that could be read into the 5GReasoner. High-level conformance tests of interest were mapped to liveness properties, in an NuXmv acceptable format for model checking, then used as seeds for the LLM.

Access to the particular model GPT-3 was not possible, GPT-3.5 was utilized to recreate the same sequence output. This showed that a zero-shot prompt is possible for logical sequencing of smaller tasks using GPT-3.5. This highlights the positive utility that LLMs are growing towards in each large model update.

Complex tasks such as code generation have been examined, further showing that LLMs are naturally non-deterministic. [17] This leads to the current need to have expert opinion to ensure accuracy of the PAVE purposed verification method.

Formal Verification: Further honing in on a particular specification for reference, in [4], the 5GReasoner validates the ”Replay Protection” protocol. It was then extrapolated into a relevant conformance test. That conformance test was then manually interpreted into a formal property. This current methodology is very labor intensive and requires strong telecommunications knowledge to generate proper coverage for implementation and testing of specifications.

In this section, we describe how Prompt-based Formal Properties Identification output can replace the manual process required by the state of art of the formal verification. The complexity of telecommunication specifications spans several generations of backward-compatible and sun-setting technologies, making the generation of test events for specification validation a challenging task that requires deep technical understanding and historical knowledge. This manual process is time-consuming, error-prone, and often reliant on expert opinion.

In the Prompt-based Formal Properties Identification module, we test a protocol for the interaction between the User Equipment (UE) and the Next Generation Radio Access Network (NG-RAN) as proof of concept. We start by examining the Radio Resource Control (RRC) protocol specification [19]. Typically, all 3GPP documentation includes a References section that is essential for fully understanding the protocols outlined. The RRC protocol specification references over 75 documents, with 65 being 3GPP publications. Manually cross-referencing these substantial and frequently updated documents to generate comprehensive test coverage is a daunting task.

To automate the extraction and cross-referencing of relevant information, we exam the `RRC_connection_resume` specifications, we parse and identify necessary connections and dependencies with the referencing over 10 other documents and self-references, as shown in Fig. 1. This reduces the complexity involved in verifying each function of the RRC protocol.

Using the Flesch Reading Ease formula to assess readability, documents like the 3GPP TS 38.331, with a low score of 6.3, are highly complex and require advanced levels of interpretation, typically at a ”college graduate” level. By breaking down these documents into more manageable parts, PAVE generate accurate conformance tests captured in Conformance Specifications [21]. These tests, which often lacked coverage in some attacks discovered by the 5GReasoner [4], can now be more thoroughly covered.

OTA System Validation: The detected vulnerabilities are reproduced in the real-world OTA testbed. We explore the experiments in our existing HyFuzz platform, a hybrid system consisting of two major components: a ZMQ virtual relay model and an OTA physical relay model. These two components run on top of the srsRAN project [22], an open-source 3GPP and O-RAN Alliance compliant RAN solution. Combining these two provides a robust vulnerability identification and analysis framework. ZMQ acts as a virtualization relay on top of the

physical (PHY) OTA layer. The uplink layer is redirected through a man-in-the-middle (MITM) relay node, allowing for parallel computing, inspection, and manipulation of MAC Protocol Data Units (PDUs) to identify vulnerabilities.

Through integrating LLMs into the formal verification process, PAVE not only automates the extraction and interpretation of complex specifications but also enhances the generation of comprehensive and accurate test cases. This reduces the reliance on manual effort, minimizes errors, and improves overall efficiency in validating telecommunication protocols.

III. Implementation

A. Data Extraction and Prompt Constructions

Some cursory knowledge of 3GPP Specifications and telecommunications is required to prompt the LLM to a desired output accurately. All specifications were extracted from Word documentation in text format and then concatenated to facilitate search speed. Once a desired specification is selected to be verified, all needed references are gathered to be later used as a component of a multi-shot prompt. As illustrated in Fig.2, the complexity necessitates manual extraction to accommodate the current context window limitations of GPT and similar transformer-based models.

Prompt engineering is a new and upcoming field from the introduction of Generative Artificial Intelligence. Templating is used to combat the non-deterministic nature of LLMs. Proper templates will provide consistency in guiding the user on what types of data to enter and in what forms, joining the separate questions into a single complex prompt that effectively communicates with the LLM.

The first aspect of the prompt we defined was role. The role is defined as the function of the LLM to assume for the duration of the interaction. The role also defines the LLM's tone in response to the prompt. For instance, defining the role as "Act as a parent explaining a concept to a 5-year-old child" versus "Act as a Security Researcher searching for vulnerabilities" has vastly different implications for the LLM. Using the Security Researcher role allows the LLM to include additional information about its output useful for comprehension and explanation. Without utilizing this role, our output was left to be decoded by the user. This valuable insight is depicted in table:variants in which we request GPT-3.5 to generate two variants in the protocols for the NAS layer in a NuXmv-acceptable syntax.

The goal attribute defines the purpose and expected outcome of the interaction with the LLM. It gives clear intent to ensure the output supports the specified goal. This allows the response to include information that would potentially be erroneous. This differs from the format attribute, which hosts information on tailoring the output. For instance, if you wanted to output a bulleted list or limit the amount of words in a response, you would utilize this attribute.

The context window is defined as the amount of tokens or text the LLM can input and utilize to form an output. Initially, the context window length for GPT-3 was 2048 tokens; this has evolved to 16,385 tokens for the latest GPT-3.5 turbo and 128k tokens for GPT-4 [23]. In the case of the Prompt-based 5GReasoner, the Context represents additional information that should be referenced when generating an output. This is one of the prompt template's most important attributes because it is one of the most powerful but limiting areas. This attribute will be used to input the specification documentation with any additional required references to realize the end goal.

TABLE I
Variants for Adversary Actions and UE NAS State Transitions

Variant	Description
Restricting adversary actions	`LTLSPEC G((! (ini_adv_act_BU = adv_BU_rrc_release | ini_adv_act_BU = adv_BU_rrc_release_suspend | ini_adv_act_BU = adv_BU_paging_imti | ini_adv_act_BU = adv_BU_paging_tmsi) -> (X(ue_rrc_state = ue_rrc_idle)))`
Allowing adversary actions restricting UE NAS state transitions	`LTLSPEC G(((ue_rrc_state = ue_rrc_connected & bs_rrc_security_context) -> (X((ue_rrc_state = ue_rrc_connected) | X(ue_nas_req_establishment))))`

The last key component is the actual order of the information throughout the prompt as it is input into the context window. [24] We found it important when working with GPT-3.5 turbo to put the most important aspects of the prompt in the front and not to confuse the prompt with incomplete or poorly labeled example data. In the case of requesting verifiable protocols in a NuXmv acceptable format, it is necessary to provide the current possible states of the model. Otherwise, the LLM will hallucinate its own undefined variable states when trying to generate the protocol. We also saw this to be the case when providing incomplete source code to build from, but we were able to get verifiable protocols that were accepted by NuXmv through a maximum of three iterations. A mapping of high-level conformance tests to NuXmv protocols [4] serves as an excellent seed for the LLM to utilize as an example in a multi-shot prompt.

B. Auto-modeling of Formal Verification and Real-world Experimental Fuzzing Platform

In our previous work [10], a platform for experiment fuzzing, 'HyFuzz' was developed. HyFuzz is an over-the-air platform allowing deep fuzzing and thorough inspection of fuzzed commands down to hex number formatting. [10] Fuzzing is the act of sending random data inputs into a complex system to uncover security vulnerabilities and emergent behavior. These vulnerabilities can potentially be utilized by malicious actors to perform nefarious actions over the air (OTA). This platform will allow us to validate vulnerabilities discovered through virtualization or modeling and analyze them in real-time.

The OTA layer is especially important because it allows a realistic testing environment to evaluate real-world performance. This Layer configures srsRAN as the UE to control a USRP B210 device, while an Amarisoft Call Box acts as the gNB and core network. A proxy layer allows for attack

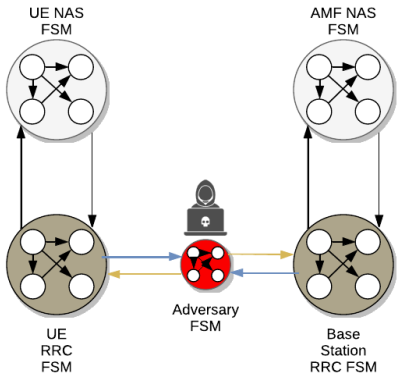

Fig. 3. Virtualization layer depicting the Finite State Machines utilized for protocol verification on the virtualization layer. [4] The RRC Layer contains FSMs for both the UE and base station, while the NAS layer contains the FSMs for the UE and AMF. The Adversary FSM acts as a MitM able to inject states to the UE and Base station FSMs.

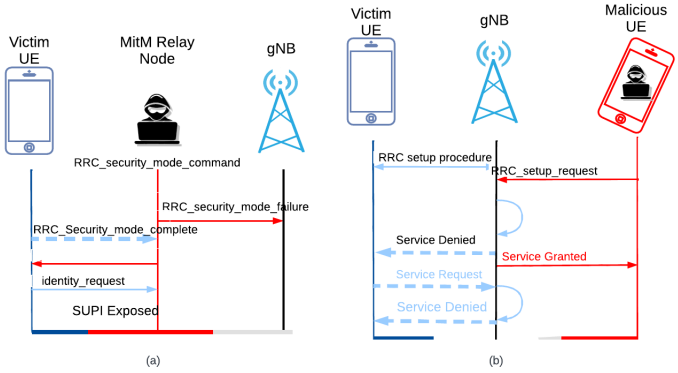

Fig. 4. (a) NULL Cipher Attack: An adversary intercepts the `RRC_security_mode_command` and responds with `RRC_security_mode_failure`, causing the UE and base station to use the Null Cipher. (b) Masquerade DoS Attack: An adversary impersonates a victim UE, sends an `RRC_setup_request` to the base station, deleting the current security context and gaining service.

modification without changing the UE state. The Amarisoft Call Box enables in-depth analysis of the communication log, further evaluating the system's resilience and potential impacts of discovered vulnerabilities. This framework layer has been further explored to generate a full digital twin for fuzz testing. [9] This framework was estimated to execute 1200 test cases per day successfully.

IV. Detected Vulnerabilities and Attack Models

Both the UE NAS/RRC channel and core network/base station channel are private. [4]. In this context, private means free of adversary influence. The channels between the UE RRC and Base Station RRC FSMs are modeled so that an adversary has Dolev-Yao-style abilities [25]. These abilities include drop, impersonation, injecting, and modifying packets. This interaction is depicted in Fig.3, showing the MitM adversary altering packets on the communication between the UE and base station. This interaction is based on the initial works of the 5GReasoner and expanded for clarity.

A. Null Cipher Attack Assumptions

The adversary knows the victim's C-RNTI and they can establish a Man in the Middle MitM relay. The Null Cipher attack aims to put the UE into a limited service mode state. "The 'NULL' integrity protection algorithm (nia0) is used only for SRBs and for the UE in limited service mode, see TS 33.501..." [19] This specification is troublesome if a malicious actor could force the UE into a limited service mode state. Once the Null Cipher and Null Integrity Protection are used at the RRC layer, the adversary can see and inject any control-plane messages. A fake base station may also force the UE to expose its SUPI in plaintext by sending an identity_request message.

RRC_Sec_Mode_Failure Vulnerability: A base station will initiate the RRC Security Mode procedure when it wants to initiate or refresh the RRC/PDCP layer security context. Lack of integrity protection allows an adversary to masquerade as the UE and send a `RRC_sec_mode_failure` message. Due to the following specification "continue using the configuration used prior to the reception of the `SecurityModeCommand` message, i.e. neither apply integrity protection nor ciphering." This means the UE and base station will continue to use the previous security context, utilizing the NULL Cipher. The null integrity protection shall only be used for control-plane messages and when the UE is in limited service mode. The adversary can then inject control-plane messages and the UE may expose its SUPI in plaintext when receiving an identity_request message as seen in Fig.4.a.

B. Masquerade Denial of Service Attack Assumptions

The adversary knows the victim's TMSI and can establish a fake UE to impersonate the victim. The primary goal of this attack is to put the UE in a DoS state. This attack is classified as a Masquerade DoS attack because the malicious UE impersonates the victim UE, forcing the gNB to deny service to the victim UE.

1) RRC_setup_request Vulnerability: The interaction during this attack is depicted in Fig.4.b. A victim UE sends an `RRC_connection_request` to the gNB. The base station then replies with an RRC connection setup to establish uplink and downlink information transfers. The base station then sends a `Security_Mode_Command` to setup the Security Context and waits for a `Security_Mode_Complete`. At this point, a malicious UE can send the base station an `RRC_Setup_ Request`, and the base station will delete the stored security context and then re-establish a connection with the malicious UE. This puts the victim UE in a state of Denied Service. This vulnerability is due to the lack of integrity protection on the `RRC_Setup_Request` command. The Malicious UE can send its own command, and the integrity of the message is not verified, exploiting `RRC_Setup_Request` to force the gNB to erase the victim's security context. The malicious UE must send a strong enough signal for the victim UE to attach to. [26] The security

context from the viewpoint of the base station includes a set of keys, algorithms, identifiers, and procedural information essential for maintaining secure communication with the UE.

C. Lullaby Attack Assumptions

The adversary knows the victim's C-RNTI and can connect the victim to a fake base station. This attack is also possible with a MitM relay that can modify, inject, drop, or view legitimate messages. This attack denies service and drains the battery of the Victim UE. This attack is classified as a Lullaby attack because it sends the victim UE into a perpetual IDLE state, causing additional battery drain.

RRC_reconfiguration Vulnerability: After initial connection, the `RRC_Setup` procedure produces a security context with a particular $MAC = \alpha$ and the UE is then attached to the base station. A malicious base station then sends a `RRC_reconfiguration` command with $MAC = \beta$. The UE is unable to verify the integrity of the MAC and moves into an IDLE state by releasing the RRC connection. The malicious base station can then release connection, allow connection to legitimate base station, and then send another `RRC_reconfiguration` with an arbitrary MAC to put the UE into another IDLE state. By perpetuating the IDLE loop an adversary can potentially cause the UE to expend more resources on cryptographic processes and searching for network connections.

D. Incarceration Attack Assumptions

The adversary knows the victim's C-RNTI or TMSI, and can establish two fake base stations to send alternating commands from. The primary goal of this attack is to put the victim in a DoS state. It is similar to a Distributed DoS attack because of the multiple actors or base stations involved, but it is better classified as an Incarceration attack. As shown in Fig.5.b. The main characteristics of this attack are forced connection loops, multiple base stations, and message manipulation.

RRC_reject / RRC_release Vulnerability: The vulnerability is when the UE is in the `RRC_idle` state. In this state, the UE accepts `RRC_reject` messages without integrity protection. The malicious message also sets the mobility backoff timer; the victim then waits in an IDLE state for a max of 16 seconds until trying to reconnect to a base station. If a malicious base station lures a UE to send a `RRC_setup_request` to itself, it can then send non-integrity protected `RRC_reject` messages to the victim. This attack requires two malicious base stations. The UE will only try reconnecting to the same base station for a maximum of 4 tries while luring the UE between the two to maintain a perpetual incarceration loop. TableII shows a summary of detected vulnerabilities.

V. Discussions

Initially, the Prompt-based Inspector showed promise in generating NuXmv verifiable protocols. Early attempts had syntax errors, requiring multiple iterations for a valid protocol. Once accurate protocols were produced, generating them for other RRC commands was straightforward. These

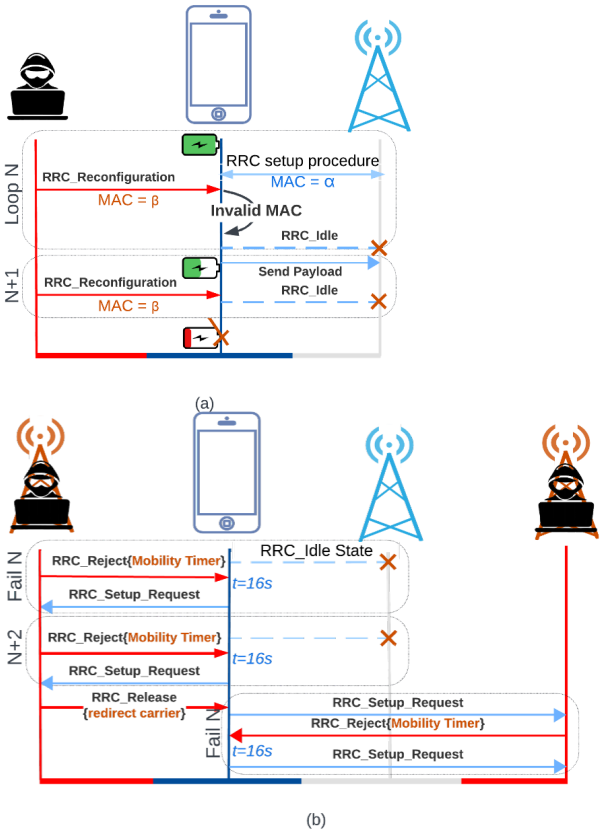

Fig. 5. (a) Lullaby Attack: Adversary acts as a base station, sending `RRC_reconfiguration` with an arbitrary MAC, forcing the victim UE into an IDLE loop, draining its battery. (b) Incarceration Attack: Using two malicious base stations, adversary alternates sending `RRC_reject` and `RRC_release` messages, keeping the UE in a perpetual connection attempt loop.

TABLE II
RRC Layer Vulnerabilities and Attacks

RRC Layer Attacks				
Attack Implication	Vulnerability	LLM Replicated	OTA Validated	TS 38.331
Null Cipher	No integrity protection in RRC Sec Mode Failure	Yes	Future Work	5.3.3
Masquerade DoS	No integrity protection in RRC Setup Request	Yes	Yes	5.3.3.4 / 5.3.5.3
Lullaby Attack	No integrity protection in RRC Reestablish Reject	Yes	Future Work	5.3.5.3
Incarceration Attack	UE in IDLE state accepts non-integrity protected commands	Yes	Future Work	5.3.3.5

protocols, when compared to those from the 5GReasoner [4], were accurate but lacked guidance on adversary channels. Using adversary settings from human-generated protocols, we achieved similar results, with NuXmv identifying state loops. However, starting from a fresh chat window led to syntax issues and overreliance on examples, resulting in non-authentic protocols. The LLM reused provided states rather than generating new ones, likely due to non-deterministic output [17]. We have reported our findings to 3GPP through the Coordinated Vulnerability Disclosure (CVD) [27] program and are awaiting their response.

VI. Conclusion

The research presented demonstrates the effectiveness of integrating LLMs into the formal verification of 5G protocols, specifically targeting the RRC layer of 5G release 17. Utilizing the Prompt-Based Inspector Aided Formal Verification (PAVE) framework, we significantly reduced the manual effort and complexity in verifying telecommunication specifications. With variants, our approach successfully identified and validated seven vulnerabilities including variations of DoS style, Lullaby, Incarceration, and Null Cipher attacks, using both formal verification and over-the-air experimental platforms. This comprehensive framework, which spans from virtualization to real-world testing, highlights the critical importance of robust protocol verification and system validation for secure 5G infrastructure. Future work will focus on extending this methodology to include additional inherited attacks and broader protocol layers, further enhancing the security and reliability of next-generation telecommunications.

VII. ACKNOWLEDGMENT

This effort was sponsored by the Defense Advanced Research Project Agency (DARPA) under grant no. D22AP00144. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of DARPA or the U.S. Government.

References

[1] A. Algarni and V. Thayananthan, "Autonomous vehicles: The cybersecurity vulnerabilities and countermeasures for big data communication," *Symmetry*, vol. 14, no. 12, p. 2494, 2022.

[2] G. J. Holzmann, "The model checker spin," *IEEE Transactions on Software Engineering*, vol. 23, no. 5, pp. 279–295, 1997.

[3] K. G. Larsen, P. Pettersson, and W. Yi, "Uppaal in a nutshell," *International Journal on Software Tools for Technology Transfer*, vol. 1, no. 1-2, pp. 134–152, 1997.

[4] S. R. Hussain, M. Echeverria, I. Karim, O. Chowdhury, and E. Bertino, "5greasoner: A property-directed security and privacy analysis framework for 5g cellular network protocol," in *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*, 2019, pp. 669–684.

[5] S. Hussain, O. Chowdhury, S. Mehnaz, and E. Bertino, "Lteinspector: A systematic approach for adversarial testing of 4g lte," in *Network and Distributed Systems Security (NDSS) Symposium 2018*, 2018.

[6] J. Yang, Y. Wang, T. X. Tran, and Y. Pan, "5g rrc protocol and stack vulnerabilities detection via listen-and-learn," in *2023 IEEE 20th Consumer Communications & Networking Conference (CCNC)*. IEEE, 2023, pp. 236–241.

[7] J. Yang, Y. Wang, Y. Pan, and T. X. Tran, "Systematic meets unintended: Prior knowledge adaptive 5g vulnerability detection via multi-fuzzing," *arXiv preprint arXiv:2305.08039*, 2023.

[8] Y. Wang, A. Gorski, and A. P. da Silva, "Development of a data-driven mobile 5g testbed: Platform for experimental research," in *2021 IEEE International Mediterranean Conference on Communications and Networking (MeditCom)*. IEEE, 2021, pp. 324–329.

[9] D. Dauphinais, M. Zylka, H. Spahic, F. Shaik, J. Yang, I. Cruz, J. Gibson, and Y. Wang, "Automated vulnerability testing and detection digital twin framework for 5g systems," in *2023 IEEE 9th International Conference on Network Softwarization (NetSoft)*. IEEE, 2023, pp. 308–310.

[10] J. Yang and Y. Wang, "Hyfuzz: A nextg hybrid testing platform for multi-step deep fuzzing and performance assessment from virtualization to over-the-air," in *2023 IEEE 12th International Conference on Cloud Networking (CloudNet)*. IEEE, 2023, pp. 274–280.

[11] J. Yang, S. Arya, and Y. Wang, "Formal-guided fuzz testing: Targeting security assurance from specification to implementation for 5g and beyond," *IEEE Access*, 2024.

[12] M. Cosler, C. Hahn, D. Mendoza, F. Schmitt, and C. Trippel, "nl2spec: Interactively translating unstructured natural language to temporal logics with large language models," *arXiv preprint arXiv:2303.04864*, 2023.

[13] J. Yang and Y. Wang, "Toward auto-modeling of formal verification for nextg protocols: A multimodal cross-and self-attention large language model approach," *IEEE Access*, vol. 12, pp. 27 858–27 869, 2024.

[14] Y. Peng, X. Li, S. Arya, and Y. Wang, "Deft: A novel deep framework for fuzz testing performance evaluation in nextg vulnerability detection," *IEEE Access*, 2023.

[15] Y. Peng, X. Li, J. Yang, S. Arya, and Y. Wang, "Raft: A real-time framework for root cause analysis in 5g and beyond vulnerability detection," in *2024 IEEE 21st Consumer Communications & Networking Conference (CCNC)*. IEEE, 2024, pp. 446–454.

[16] J. Yang and Y. Wang, "Dependency-graph enabled formal analysis for 5g aka protocols: Assumption propagation and verification," in *ICC 2024-IEEE International Conference on Communications*. IEEE, 2024, pp. 715–721.

[17] S. Ouyang, J. M. Zhang, M. Harman, and M. Wang, "Llm is like a box of chocolates: the non-determinism of chatgpt in code generation," *arXiv preprint arXiv:2308.02828*, 2023.

[18] C. Hahn, F. Schmitt, J. J. Tillman, N. Metzger, J. Siber, and B. Finkbeiner, "Formal specifications from natural language," *arXiv preprint arXiv:2206.01962*, 2022.

[19] 3GPP, "Technical Specification Group Radio Access Network; NR; Radio Resource Control (RRC) protocol Specification," 3rd Generation Partnership Project (3GPP), Technical Specification (TS) 38.331, 09 2023, version 17.6.0.

[20] S. Jha, S. K. Jha, P. Lincoln, N. D. Bastian, A. Velasquez, and S. Neema, "Dehallucinating large language models using formal methods guided iterative prompting," in *2023 IEEE International Conference on Assured Autonomy (ICAA)*. IEEE, 2023, pp. 149–152.

[21] 3GPP, "Technical Specification Group Radio Access Network; 5GS; User Equipment (UE) conformance specification; Part 1: Protocol," 3rd Generation Partnership Project (3GPP), Technical Specification (TS) 38.523-1, 01 2024, version 17.5.1.

[22] srsRAN, "Srsransrsran_project: Open source o-ran 5g cu du solution from software radio systems (srs)."

[23] "Models - openai." [Online]. Available: https://platform.openai.com/docs/models/overview

[24] N. F. Liu, K. Lin, J. Hewitt, A. Paranjape, M. Bevilacqua, F. Petroni, and P. Liang, "Lost in the middle: How language models use long contexts," *Transactions of the Association for Computational Linguistics*, vol. 12, pp. 157–173, 2024.

[25] D. Dolev and A. Yao, "On the security of public key protocols," *IEEE Transactions on information theory*, vol. 29, no. 2, pp. 198–208, 1983.

[26] S. Park, I. You, H. Park, and D. Kim, "Analyzing rrc replay attack and securing base station with practical method," in *Proceedings of the 17th International Conference on Availability, Reliability and Security*, ser. ARES '22. New York, NY, USA: Association for Computing Machinery, 2022. [Online]. Available: https://doi.org/10.1145/3538969.3544448

[27] [Online]. Available: https://www.3gpp.org/delegates-corner/coordinated-vulnerability-disclosure