5GReasoner: A Property-Directed Security and Privacy Analysis Framework for 5G Cellular Network Protocol

Syed Rafiul Hussain
Purdue University
hussain1@purdue.edu

Mitziu Echeverria
University of Iowa
mitziu-echeverria@uiowa.edu

Imtiaz Karim
Purdue University
karim7@purdue.edu

Omar Chowdhury
University of Iowa
omar-chowdhury@uiowa.edu

Elisa Bertino
Purdue University
bertino@purdue.edu

ABSTRACT

The paper proposes 5GReasoner, a framework for property-guided formal verification of control-plane protocols spanning across multiple layers of the 5G protocol stack. The underlying analysis carried out by 5GReasoner can be viewed as an instance of the model checking problem with respect to an adversarial environment. Due to an effective use of behavior-specific abstraction in our manually extracted 5G protocol, 5GReasoner's analysis generalizes prior analyses of cellular protocols by reasoning about properties not only regarding packet payload but also multi-layer protocol interactions. We instantiated 5GReasoner with two model checkers and a cryptographic protocol verifier, lazily combining them through the use of abstraction-refinement principle. Our analysis of the extracted 5G protocol model covering 6 key control-layer protocols spanning across two layers of the 5G protocol stack with 5GReasoner has identified 11 design weaknesses resulting in attacks having both security and privacy implications. Our analysis also discovered 5 previous design weaknesses that 5G inherits from 4G, and can be exploited to violate its security and privacy guarantees.

CCS CONCEPTS

• **Security and privacy → Formal security models**; **Security protocols**; **Denial-of-service attacks**; Mobile and wireless security.

KEYWORDS

Cellular Network, 5G, Model Checking, Vulnerabilities, Attacks

ACM Reference Format:
Syed Rafiul Hussain, Mitziu Echeverria, Imtiaz Karim, Omar Chowdhury, and Elisa Bertino. 2019. 5GReasoner: A Property-Directed Security and Privacy Analysis Framework for 5G Cellular Network Protocol. In *2019 ACM SIGSAC Conference on Computer and Communications Security (CCS '19), November 11–15, 2019, London, United Kingdom.* ACM, New York, NY, USA, 16 pages. https://doi.org/10.1145/3319535.3354263

CCS '19, November 11–15, 2019, London, United Kingdom

ACM ISBN 978-1-4503-6747-9/19/11…$15.00
https://doi.org/10.1145/3319535.3354263

1 INTRODUCTION

The imminent deployment of the fifth generation (5G) cellular network has created a lot of enthusiasm in both industry and academia particularly due to its promise of enabling new applications such as smart vehicles and remote robotic surgery. 5G is not only envisioned as a replacement of home broadband Internet but also is expected to have impact in the military battlefield and emergency management by improving situational awareness. All these potential novel and critical applications of 5G can be attributed to its following enhancements over 4G LTE: (1) Improvements in the physical-layer technologies enabling the support of large numbers of devices with substantially improved bandwidth; (2) Robust security posture due to the introduction of security measures in the upper-layer of the 5G protocol stack. The 5G standard, however, has opened the door to a wide array of new security challenges stemming from: (i) New security policies that are not formally verified against adversarial assumptions; (ii) Retaining security mechanisms from 4G Long Term Evolution (LTE) and its predecessors. *This paper thus aims to develop highly automated approaches enabling property-guided formal verification of control-plane protocols of the 5G protocol stack.*

Scope and Problem. The 5G control-plane consists of a number of critical procedures (e.g., initial registration, deregistration, paging) which are leveraged by fundamental cellular services, such as, voice call, SMS, data and billing. For example, vulnerabilities in the initial registration procedure may have serious consequences on those services, such as man-in-the-middle attacks [10, 39] and spurious mobile billing [34]. In this paper, we, therefore, address the following concrete research question: *Is it possible to formally verify six 5G NAS layer procedures— initial registration, deregistration, paging, configuration update, handover, and service request procedures and the corresponding five RRC layer procedures— spanning across two control-plane layers against relevant security and privacy properties?*

Challenges. To achieve our goal, however, we need to address the following challenges. (C_1) *Specification*: The 5G protocol lacks a formal specification [4–6] and hence is prone to ambiguity and underspecification. (C_2) *Protocol Complexity*: 5G comprises of multiple sub-protocols across multiple layers that are inter-dependent and *stateful* in nature [49]. Multiple types of protocol participants and messages containing data with a large domain further contribute to the complexity. (C_3) *Obtaining Requirements*: The standard [4–6] often states security and privacy requirements in an abstract way and thus careful considerations and complex assumptions are required to formulate formal properties from such implicit requirements. Particularly, the conformance test suites [7] prescribed by

the 3GPP standard encompass only primitive security requirements lacking both completeness and the consideration of adversarial environments. Finally, the current 5G test suites do not include conformance requirements of the core network components.

Existing Efforts on 5G formal verification. Only two previous efforts have formally analyzed the 5G protocol [14, 22]. These analyses [14, 22], however, focus only on a small part of the protocol, i.e., authentication and key agreement (AKA) protocol of the initial registration procedure. Also, the analyses are performed in isolation without considering their interaction with other procedures.

Approach. The most relevant to our approach is the effort by Hussain et al. [28] that proposed a framework called LTEInspector for adversarially testing three 4G LTE control-plane protocols. LTEInspector lazily combines a symbolic model checker and a cryptographic protocol verifier using an abstract-refinement principle. Their instantiation of the LTEInspector framework, however, suffers from the following limitations: (1) They consider only a single layer of the protocol stack in isolation and thus LTEInspector misses critical interactions with protocols of other layers; (2) For the sake of scalability, they only model packet type and do not model critical data or packet payload, missing out on interesting data-/payload-dependent protocol behavior; (3) Their adversary instantiation cannot handle protocols spanning across different layers of the stack. The coverage issues of LTEInspector instantiation by Hussain et al. [28] can be attributed to their coarse-grained abstraction during protocol modeling.

Our 5GReasoner can be viewed as an alternative instantiation of the general LTEInspector framework but with a different protocol modeling discipline. In our 5G protocol model covering six 5G procedures (i.e., initial registration, deregistration, paging, configuration update, handover, service request, and radio bearer establishment procedures), each protocol layer (Network Access Stratum or NAS [4] and Radio Resource Control or RRC [5]) contributes a single state machine. The NAS layer and RRC layer state machines of the same entity communicate with each other through a private channel. The RRC layer state machines residing on different entities (i.e., cellular device and the base station), however, communicate through a public, adversary-controlled channel. The adversary is also modeled as a state machine. The adversary's state machine sits in the middle of the communicating RRC layer state machines. It mimics a Dolev-Yao adversary by non-deterministically dropping, modifying, or injecting messages/payloads while respecting certain well-formedness conditions. These conditions guide the adversary's state machine to perturb message types/payloads while avoiding parsing errors. Our 5G protocol model has a total of **27** states and **238** transitions.

In our 5G protocol model, the NAS layer protocol packets are considered as payloads of RRC layer protocol packets. It is thus essential for us to model packet payloads. Directly capturing all packet payloads in our model, however, impedes the scalability of our analysis. To address this, our model only captures those packet payloads that impact the security- and privacy-specific behavior of the NAS and RRC layer protocols mentioned above. For addressing the state-explosion problem of the model checking step due to payloads, we use *behavior-specific predicate abstraction*. In this approach, rather than directly modeling data we model predicates over data which are sufficient to reason about the protocol behavior. In addition to packet payloads, there are some other data, such as wrapping link counters, which we model faithfully. Some of these counters are also included in the protocol packet for performing well-formedness check. Our model also includes different timers that are missing from the LTEInspector framework.

We instantiate 5GReasoner with two infinite-state model checkers (i.e., Kind 2 [19] and nuXmv [18]) and a cryptographic protocol verifier (i.e., ProVerif). The reason for using two model checkers is that different model checkers are more effective in reasoning about different types of temporal properties (e.g., safety and liveness properties). For gathering properties, we first use the conformance test suite suggested by the standard. We augment these properties with other properties that are implicit in the standard. For our evaluation, in total we gather and verify **187** properties of the 5G protocol model. For each observed counterexample provided by the model checker while verifying properties, we confirm its feasibility with ProVerif before reporting it as a design weakness.

Findings. Significant among our findings is the *Exposing the Device's TMSI and Paging Occasion* attack which enables an adversary to track a victim device's location, hijack the paging channel, and learn the TMSI which further enables other attacks. Another significant finding is the *Installing Null Ciphering and Null Integrity* attack which can lead to a UE in the limited service mode to expose its SUPI which breaks one of the critical security requirements.

Contributions. In summary, the paper has the following technical contributions.

(1) We propose the 5GReasoner framework for property-guided formal verification of 5G control-plane protocols.
(2) We construct a formal model of the 5G protocol covering six 5G NAS layer procedures and five RRC layer procedures. Our model contains sufficient information to allow one to reason about temporal properties referring to both packet payload, data, and timer behavior. Our 5G protocol model is of independent interest as it can be used to test 5G-enabled devices.
(3) Our evaluation of the 5G protocol model against **187** properties with 5GReasoner revealed 11 new exploitable protocol design weaknesses. It has also discovered 5 prior attacks which 5G inherits from 4G LTE. Our findings have severe security and privacy implications including downgrade and SUPI catching.

2 BACKGROUND

We now discuss the basics of 5G NR (New Radio). We first discuss the 5G network architecture and its components followed by the relevant 5G control-plane procedures needed for this paper.

2.1 5G System Architecture

The 5G architecture can be partitioned into the following three main components: User Equipment (UE), the 5G radio access network (5G-RAN) and the 5G core network (5G-CN).

UE: The "User Equipment" is a device (e.g., a smartphone) equipped with a *USIM (Universal Subscriber Identity Module)*. Each USIM is uniquely identified by its *SUPI (SUbscription Permanent Identifier)*, similarly to how previous generations (3G and 4G) USIMs were identified by their *IMSI (International Mobile Subscriber Identity)*. A significant difference between 4G and 5G USIMs is that the new 5G USIMs are capable of generating random nonces.

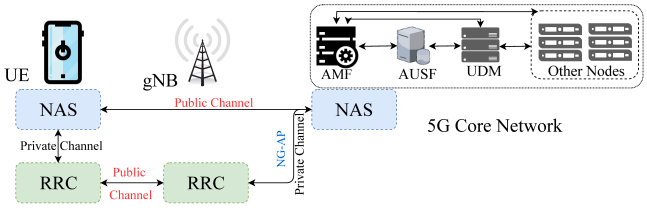

Figure 1: Simplified 5G Architecture

5G-RAN: In 5G, a geographical area is partitioned into hexagonal cells, where each cell is serviced by a **gNB** (5G base-station). A key difference between 4G and 5G cells is that the latter are powered by low-power base stations which cover smaller geographical areas while providing extremely fast coverage with low latency. The 5G-RAN can be seen as the network between a UE and a gNB and also between communicating gNBs.

5G-CN: The 5G core network (5G-CN) can be considered as a mesh of interconnected services (see Figure 1). The Access and Mobility Management Function (AMF) manages registration, detach procedures, paging and services related to registration, connection, and mobility. The authentication and key agreement for registration procedure is completed with the help of **AUSF** (Authentication Server Function)–which stores the UE's identities, keys and subscription data and **UDM** (Unified Data Management) –responsible for generating Authentication and Key Agreement (AKA) credentials. Other nodes of 5G-CN are not relevant to our discussion.

2.2 NAS Layer Procedures

We now briefly discuss the NAS layer procedures that are the most relevant in the context of our paper.

Initial Registration. After rebooting, a UE performs a radio setup procedure through which it gets assigned a C-RNTI (Cell Radio Network Temporary Identifier). After the radio setup, the UE establishes communication through the RRC layer following the RRC Setup procedure (discussed in Section 2.3). After completing the RRC setup, the UE starts the NAS registration procedure by sending the `reg_request` message. The `reg_request` message includes its *SUCI (SUbscription Concealed Identifier)*– an encryption of its identifier (SUPI) with a random nonce. The AMF completes the authentication procedure with the help of AUSF. After successful authentication, AMF initiates negotiation of ciphering and integrity algorithms through the security mode procedure. At this point, the NAS level security context is established between the UE and AMF, and the selected encryption and integrity protection algorithms will be applied to the subsequent NAS messages. The AMF concludes the registration procedure by sending the `reg_accept` message containing UE's TMSI and the UE responds with `reg_complete` message.

Deregistration. To disconnect from the network, the UE can initiate this procedure by sending a `ue_dereg_request` message to which the network is expected to respond with a `dereg_accept` message.

Configuration Update. Due to tracking area update or a successful service request procedure invoked as a response to a paging request, the network can initiate a configuration update procedure to update the UE configuration, e.g., assign a new TMSI. To initiate this procedure, the AMF sends a `config_update_command` message which the UE acknowledges with a `config_update_complete` message.

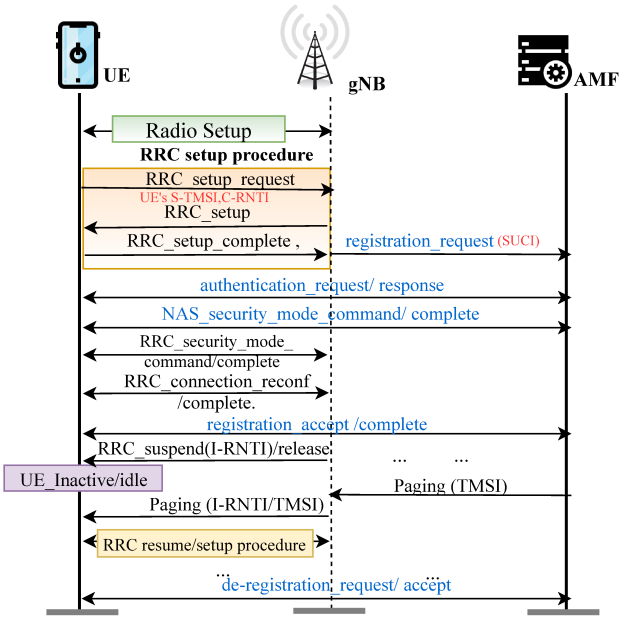

Figure 2: Important UE procedures

Service request. The UE invokes this procedure when it receives a paging request from the network or the UE has pending uplink data and it is in idle mode. The UE initiates this procedure by sending the `service_request` messages to the AMF and network responds with `service_accept` marking the completion of the procedure.

2.3 RRC Layer Procedures

We discuss the relevant RRC procedures for the sake of easy exposition of our findings.

RRC Setup. RRC setup procedure is the backdrop for NAS registration. The purpose of this procedure is to establish an RRC connection and to transfer the initial NAS dedicated information/ message from the UE to the network.

RRC Security Activation. The purpose of this procedure is to activate security upon RRC connection establishment.

RRC Release. This procedure is used by the network to release the established radio bearers as well as all radio resources or to suspend the RRC connection, which includes the suspension of the established radio bearers. When it suspends the radio bearers it assigns a I-RNTI (Inactive-Radio Network Temporary Identifier) to the UE.

RRC Connection Reconfiguration. The purpose of this procedure is to modify a RRC connection, e.g. to establish/modify/release radio bearers (RBs). As part of the procedure, NAS dedicated information may be transferred from the network to the UE. After this RRC procedure the UE completes the initial registration. To begin this procedure, the network sends a `rrc_reconfiguration` message to which the UE replies with `rrc_reconfiguration_complete` to acknowledge the reconfiguration.

RRC Connection Resume. A significant difference in the RRC layer between 4G LTE and 5G NR is–apart form RRC connected and RRC idle states, that 5G NR has introduced a new RRC state named RRC inactive. When a UE is powered up it is in disconnected/idle

state; it can move to RRC connected with an initial RRC setup or with connection establishment. If there is no activity from UE for a short time, it can suspend its session by moving to the RRC inactive state. The motivation for this state is to reduce system access, save power and optimize mobility. For a UE to transition from RRC inactive to RRC connected, a new procedure named RRC Resume has been setup. The UE initiates the procedure when responding to NG-RAN paging (while the UE is in RRC inactive) and requests the resume of a suspended RRC connection using this procedure. The process is quite similar to RRC setup with `rrc_resume_request`, `rrc_resume` and `rrc_resume_complete`.

RRC Connection Re-establishment. A UE in RRC idle state follows this procedure to transition from RRC idle to RRC connected. However, a UE, which is in RRC connected state and for which security has been activated, may initiate the procedure in order to continue the RRC connection. The connection re-establishment succeeds if the network is able to find and verify a valid UE context.

2.4 Paging Procedure

Whenever a registered UE has no data to send, it goes to low energy/idle mode and wakes up periodically according to a defined paging occasion to check for paging massages. As discussed earlier, in 5G an extra state has been added to RRC idle and active named inactive, thus there are two types of paging.

RAN Initiated Paging. In RRC Inactive mode, paging is initiated by 5G-RAN and uses I-RNTI (Inactive-Radio Network Temporary Identifier) as a unique identifier for the UE. The paging is triggered by the last serving gNB. If the UE has been successfully reached, it attempts to resume from RRC inactive to RRC connected following the RRC connection resume procedure.

CN Initiated Paging. When in RRC idle mode, paging is initiated by 5G-CN and uses TMSI as a unique identifier for the UE. In this case the paging is triggered through the CN and RRC setup procedure is invoked to move the UE from idle to connected state.

3 OVERVIEW OF 5GREASONER

We now present the architecture of 5GReasoner (see Figure 3) and describe its major components. We then present a working example to describe the verification workflow of 5GReasoner. Before delving into the details, we first present our threat model.

3.1 Threat Model

For our analysis, we consider the following communication channels to be private and free of adversarial influence: (1) channel between NAS and RRC layers in UE; (2) channel between the core network and the base station. The communication channels between the UE and base station, and between the UE and core network, are subject to adversarial influence from a Dolev-Yao-style network adversary [24] who can impersonate a legitimate protocol participant and can also drop, inject, or modify any packet while adhering to cryptographic assumptions (e.g., it can decrypt an encrypted message *only if* he possesses the decryption key). Also, cryptographic constructs are considered to be perfectly secure.

We also consider the core network components, target user's UE, and the USIM to be part of the trusted computing base and free of adversarial influence. The adversary, however, may possess USIMs provided by network operators which the adversary can compromise to learn the master secret key and symmetric session keys of that USIM along with network operators' public keys.

3.2 High-Level Approach

Our approach at a high-level is similar to Hussain et al. [28]. Both approaches follow the counterexample-guided abstraction-refinement principle (CEGAR) [21] with one subtle difference. Before pointing out this difference, we first give a brief introduction to CEGAR.

In the general CEGAR framework, the verification inputs are a concrete model/program M_c and the property to verify ϕ. The aim is to check whether M_c satisfies ϕ. The verification starts with abstracting the concrete model to obtain an abstract model M_a and verified against ϕ. If the verification goes through (i.e., M_a satisfies ϕ), due to the use of abstraction (i.e., the number of executions in M_c is a subset of M_a), then it entails that M_c satisfies ϕ. If the verification, however, does not go through (i.e., a counterexample σ is generated) then there are two possibilities: (1) M_c violates ϕ; (2) M_a violates ϕ due to the use of abstraction but M_c actually may not violate ϕ. To figure out which case it is, one check to see whether σ is a realizable execution in M_c. If it is, then we are in case (1) and the counterexample is returned as evidence of verification failure. If σ is, however, not realizable in M_c, then M_a is modified to obtain M_{a1} which rules out σ (and, possibly its generalization). This cycle continues until either the verification goes through or a realizable counterexample is found.

Example. Now let us consider a very simple program M with the following four statements. The property we are interested in verifying is $\phi \equiv out = (x1 + x2) * (y1 + y2)$ where $x1, x2, y1, y2$ are 32-bit machine integers and "+" (resp., "$*$") represents addition (resp., multiplication). Suppose reasoning about both "+" and "$*$" operations together in a bit-precise manner is hard for an automated reasoner like a Satisfiability Modulo Theory (SMT) solver [13].

```
1 x1, x2, y1, y2: int32 // input declarations
2 u1  := x1 + x2
3 u2  := y1 + y2
4 out := u1 * u2
```

To make the analysis amenable to an automated reasoner like an SMT solver, let us assume that we abstract "+" and "$*$" with two uninterpreted binary functions $F : \text{int32} \times \text{int32} \rightarrow \text{int32}$ and $G : \text{int32} \times \text{int32} \rightarrow \text{int32}$, respectively. Roughly, an uninterpreted function is a function symbol whose signature is known but not its semantics. We also expect that both these uninterpreted functions satisfy the congruence property—required for an SMT solver to reason about uninterpreted functions, that is, $\forall s1, s2, t1, t2.(s1 = t1 \wedge s2 = t2) \rightarrow F(s1, s2) = F(t1, t2)$ and $\forall s1, s2, t1, t2.(s1 = t1 \wedge s2 = t2) \rightarrow G(s1, s2) = G(t1, t2)$. After applying our uninterpreted-function abstraction, the abstract property becomes $\phi_{\text{abs}} \equiv out = G(F(x1, x2), F(y1, y2))$. Furthermore, the abstract counterpart of the above program becomes the following.

```
1 x1, x2, y1, y2: int32 // input declarations
2 u1  := F(x1 , x2)
3 u2  := F(y1 , y2)
4 out := G(u1 , u2)
```

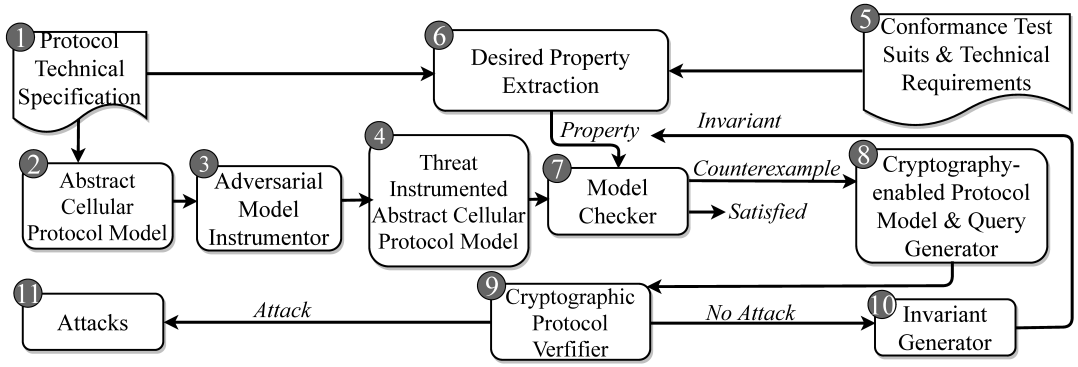

Figure 3: 5GReasoner Architecture.

Expanding u1 and u2 in the abstract program will result in the exact expression warranted by ϕ_{abs} discharging the verification obligation. This shows the potential power of CEGAR-based approach as we did not require to reason about multiplication or addition of machine integers at all.

For the sake of argument, suppose that the property we want to verify now is the following: $\phi' \equiv out = (y1+y2)*(x1+x2)$ with its abstract form being $\phi'_{\text{abs}} \equiv out = G(F(y1, y2), F(x1, x2))$. As for all uninterpreted functions $G(\cdot,\cdot)$, it is not the case that G(x,y)=G(y,x) (i.e., not commutative) and hence the verification will fail because from the program we know "out = G(F(x1, x2), F(y1, y2))" which does not entail ϕ'_{abs}. Suppose the refinement we add to the model and property is to replace all occurrences of $G(\cdot,\cdot)$ with the actual multiplication function. In that case, the verification will pass as "$*$" is commutative. Also, we did not need to reason about addition at all, showing the advantage of a CEGAR-based approach.

5GReasoner. In our context, during verification, abstraction is achieved by replacing cryptographically-protected messages with their plaintext counterpart. As the 5G protocol model is fixed so to rule out infeasible counterexamples—due to abstraction of cryptographically-protected messages—we refine the property instead of the model. Our refined properties are of the form $\alpha \rightarrow \beta$ ($\rightarrow$ signifies logical entailment) where α is a formula used to rule out the infeasible counterexample. This is to focus the verification so that it only considers traces that satisfy α.

Concretely, given a 5G protocol model and a property to check, we first replace all encrypted/integrity-protected messages with their plaintext counterpart in both the model and the property. We then enhance the model to include a Dolev-Yao-style adversary. We then use a general-purpose model checker (**MCheck**) [18, 19] to check whether the (cryptography-abstracted) model satisfies the property. If this is the case, then we adjudicate the property to be satisfied by the model. If, however, a counterexample is generated, like above, there are two possibilities: (a) the 5G model violates the property; (b) due to the abstraction of cryptographic-constructs, a spurious counterexample is generated. To check which of the cases is true, we consult a symbolic cryptographic protocol verifier (**CPVerif**). If the **CPVerif** confirms that all the steps conform to the cryptographic assumptions, then the counterexample (alternatively, the attack) is reported by 5GReasoner. If **CPVerif**, however, adjudicates one of the steps taken by the adversary to be infeasible, then we refine the property to ensure that the adversary does not exercise the offending action in the future iterations of the verification. The verification loop continues until either the property is satisfied by the model or a realizable counterexample is found.

3.3 Major Components of our Framework

We now describe the major components of 5GReasoner.

Protocol Model. We model the protocol abstractly as a set of communicating state machines (**SM**). Each of these state machines $\mathcal{M}_1$ communicate with another state machine $\mathcal{M}_2$ with two unidirectional (private or public) channels, one carrying messages from $\mathcal{M}_1$ towards $\mathcal{M}_2$, and another from $\mathcal{M}_2$ to $\mathcal{M}_1$. Each state machine is a tuple $(\mathcal{I}, \mathcal{O}, \mathcal{V}, \text{Init}, \mathcal{A})$ where $\mathcal{I}$ is a finite set of input variables; $\mathcal{O}$ is a finite set of output variables; $\mathcal{V}$ is a finite set of state variables; Init is a set of initial states; and $\mathcal{A}$ is a finite set of assignments to variables in V. Assignments define how state variables are updated, and thus define the transition relation of the system.

Adversarial Model Instrumentor. The adversarial model instrumentor takes as input a general protocol model $\mathcal{M}$ and it returns another model $\mathcal{M}_{\text{adv}}$ which is an extension of $\mathcal{M}$ containing explicit adversarial influence. Given a public channel c_1 from $\mathcal{M}_1$ to $\mathcal{M}_2$, the instrumentor introduces a new state machine $\mathcal{M}_{\text{a}}$ capturing the behavior of the adversary. It then replaces c_1 with two channels c_{1a} and c_{a2}. Channel c_{1a} carries data from machine $\mathcal{M}_1$ to $\mathcal{M}_{\text{a}}$ whereas channel c_{a2} carries data from $\mathcal{M}_{\text{a}}$ to $\mathcal{M}_2$.

$\mathcal{M}_{\text{a}}$ mimics a Dolev-Yao-style adversary, that is, given an input message, it non-deterministically decides either to drop the message (no_operation), let the message go, or change the message (or, its payload). The non-deterministic behavior is needed to let the adversary choose any arbitrary strategy to attack the protocol.

Model Checker (MCheck). MCheck takes as input the adversary included protocol model and a temporal trace property (i.e., safety and liveness), and checks whether there is an execution of the model which violates the property. If such a violation is not found, then it outputs that the model satisfies the property. If it finds a violating execution of the model, it presents a counterexample as evidence. Due to abstraction, this counterexample may be spurious and hence it is validated with a cryptographic protocol verifier.

Cryptographic Protocol Verifier (CPVerif). For each adversary action in the **MCheck** provided a counterexample, we query the **CPVerif** to check its feasibility. If all adversarial actions can be

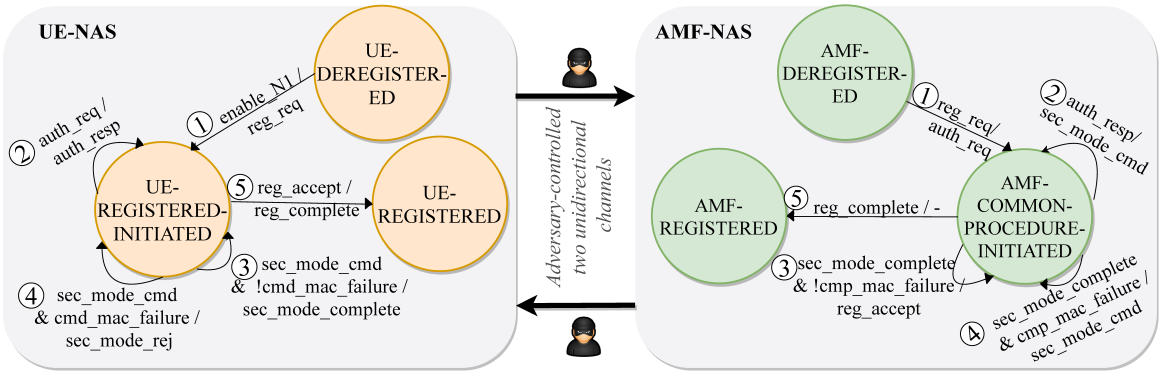

Figure 4: A simplified 5G Model for NAS layer

proven to feasible then the counterexample is presented as a feasible attack. Otherwise, we manually generate an invariant—ruling out the infeasible adversarial action—to refine the property.

3.4 Working Example

We now walk the reader through our approach using an example. For ease of exposition, we rely on an overly-simplified model of the 5G NAS ecosystem (See Figure 4).

Example model. In $\mathcal{M}_{\text{Adv}}$, the UE and the AMF are represented with two SMs communicating through two unidirectional, attacker-controlled public channels—UE to AMF, and AMF to UE. Transitions labels are of the form "condition/actions" in which condition is a logical formula specifying the condition under which the transition will be triggered whereas the actions component refers to an action sequence to be performed (in their appearance order) by the SM after the transition is taken. An empty actions component is denoted with –. Initial states for UE and AMF SMs are UE-DEREGISTERED and AMF-DEREGISTERED, respectively. The SMs have the following environmental variables: enable_n1 (enable 5G connectivity); cmd_mac_failure (improper MAC for `sec_mode_command` message); cmp_mac_failure (improper MAC for `sec_mode_complete` message).

Desired example property. The property φ we want to check is: "*It is always the case that whenever the UE SM is in the registered-initiated state and the UE has authenticated the AMF, the UE will reply with* `sec_mode_complete` *only if the AMF sent a* `sec_mode_command` *message.*" The property signifies that whenever the UE initiates the registration procedure and authenticates the core network, it will eventually be able to move on to the next stage of the registration procedure where the UE successfully negotiates the security algorithms with the core network using `sec_mode_command` and `sec_mode_complete` messages while passing all sanity checks (i.e., valid MAC and same security capabilities). This property is desired as any violation of this could signify a privacy or service disruption attack.

Verification with MCheck. Checking $\mathcal{M}_{\text{Adv}}$ against φ generates a counterexample π_1 in which the adversary sends a `sec_mode_command` to the UE. After the UE receives it, the UE sends `sec_mode_complete`.

Verification with CPVerif. To ensure the validity of π_1, 5GReasoner using **CPVerif** verifies whether the adversary can inject a fake `sec_mode_command` message. We verify a *injective-correspondence* [16] property (with infinite parallel sessions) which asserts that every `sec_mode_command` message received by the cellular device should be preceded by a unique `sec_mode_command` message sent by the AMF. The **CPVerif** provides one counterexample for this property. In the attack trace, the adversary can replay a `sec_mode_command` message captured from a previous session to the UE. Upon receiving the message, the UE responds with `sec_mode_complete` message.

Invariant Generation. 5GReasoner rules out π_1 by refining φ with the following additional invariants— (1) cmd_mac_failure will not happen; and (2) the adversary can only inject `sec_mode_command` only if it has seen a previous `sec_mode_command` message sent by the AMF.

Second Iteration. After this iteration, 5GReasoner checks φ_{ref} with **MCheck** which generates another counterexample π_2 in which the adversary replays a `sec_mode_command` message that was sent earlier by the core network to the UE. After the UE receives it, the UE sends `sec_mode_complete`. This is a feasible attack and demonstrates the effectiveness of 5GReasoner.

4 MODELING DISCIPLINES

We now first explain how we extract security requirements and turn them into formal properties. We then discuss the high-level protocol modeling discipline. Finally, we present the different types of behavior-aware predicate abstraction techniques we employ.

4.1 Extracting Formal Properties

The set of properties that 5GReasoner aims to check include *authenticity* (e.g., disallowing impersonation), *availability* (e.g., preventing service denial), *integrity* (e.g., restricting unauthorized messages), *secrecy* of user's sensitive information (e.g., preventing location data and activity profiling), and *replay protection* (e.g., restricting reception of same messages more than once). We, therefore, first identify and extract the precise and formal security goals from the informal and high-level descriptions given in the conformance test suites [7], the technical specification (TS) [4–6], and the technical requirement (TR) [2] documents provided by the 3GPP and then translate them into formal properties.

Conformance Test Suites. To test a UE's correct behavior in different control-plane procedures, the 3GPP standard defines conformance test cases [7] from the UE's point of view. The test cases, however, are defined at a very high-level and do not consider all

with {The UE in 5GMM-REGISTERED-INITIATED state}
ensure that{
when { the SS sends an EAP-request/AKA'-challenge within AUTHENTICATION REQUEST}
then { the UE sends an EAP-response/AKA'-challenge message within AUTHENTICATION RESPONSE } }

Figure 5: A conformance test case for authentication procedure [7].

possible use cases. For instance, the conformance test case listed in Figure 5 only considers the case when a UE is in the *registered_initiated* state, possesses a valid credential, and then receives an `authentication_request` message. This test case, however, does not consider the cases when the UE has already failed to verify the integrity of a previous authentication_request message and fails the sequence number checking for the current authentication_request message. To address this challenge, we break down the high-level test case into fine-grained sub test-cases by enumerating all possible conditions, and then translating them into formal properties.

TS 24.501, clause: 4.4.3.2: Replay protection assures that one and the same NAS message is not accepted twice by the receiver.

Figure 6: An example of replay protection requirement [1].

Technical Specifications and Requirements. Similar to conformance test suites, the technical specifications [4–6] and the technical requirement [2] documents define the security requirements at a high-level, albeit, in an abstract way. For instance, the *replay protection* requirement in Figure 6 extracted from the technical specification [4] needs to be interpreted and translated into formal properties for each NAS layer message only after the integrity protection has been activated.

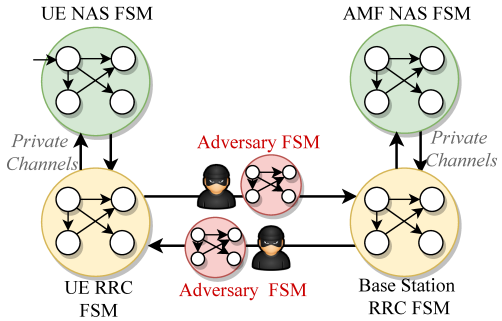

Figure 7: Threat instrumented cross-layer model.

4.2 High-Level Protocol Modeling Discipline

We model the 5G protocol as a set of communicating state machines (SM) (see Figure 7). Our 5G model considers four communicating state machines for three participants— $\mathcal{M}_{\text{UE}}^{\text{NAS}}$ and $\mathcal{M}_{\text{UE}}^{\text{RRC}}$ for the UE, $\mathcal{M}_{\text{BS}}^{\text{RRC}}$ for the base station, and $\mathcal{M}_{\text{AMF}}^{\text{NAS}}$ for the AMF.

We model each communication channel between two FSMs, for instance, $\mathcal{M}_{\text{UE}}^{\text{RRC}}$ and $\mathcal{M}_{\text{BS}}^{\text{RRC}}$ with two uni-directional channels; one from $\mathcal{M}_{\text{UE}}^{\text{RRC}}$ to $\mathcal{M}_{\text{BS}}^{\text{RRC}}$ and another from $\mathcal{M}_{\text{BS}}^{\text{RRC}}$ to $\mathcal{M}_{\text{UE}}^{\text{RRC}}$. Such design choice of using two unidirectional channels instead of a single bidirectional channel provides better flexibility (e.g., one direction of the public channels to be adversary controlled whereas the other to be reliable) in reasoning specific scenarios and filtering spurious counterexamples.

We consider the communication channel between $\mathcal{M}_{\text{UE}}^{\text{NAS}}$ and $\mathcal{M}_{\text{UE}}^{\text{RRC}}$, and $\mathcal{M}_{\text{BS}}^{\text{RRC}}$ and $\mathcal{M}_{\text{AMF}}^{\text{NAS}}$ to be private. This is because our threat model assumes that neither the cellular device's firmware nor the secure communication channel (with SCTP connection) between the base station and AMF is controlled by an adversary. We model the public communication channel between $\mathcal{M}_{\text{UE}}^{\text{RRC}}$ and $\mathcal{M}_{\text{BS}}^{\text{RRC}}$, through which all NAS and RRC layers' messages are transmitted over the air, to be Dolev-Yao adversary-controlled.

4.3 Capturing Data and Packet Payload

One of the most challenging aspects of precisely capturing the 5G protocol behavior is to model the protocol packet payload. If one were to directly represent all the packet payloads, due to the large domain of the payloads, the resulting model is unlikely to be amenable to automated reasoning. Our initial attempt of directly capturing packet payload resulted in a model that was not amenable to analysis by any of the state-of-the-art model checkers. To address the issue with state-explosion, we employed the notion of *behaivor-aware predicate abstraction*. Intuitively, rather than capturing the data directly we model a predicate over the data and such predicates are directly inspired by the protocol behavior. The following are only a few of the representative predicates our 5G model employs.

Validity predicate. As most of the 5G protocol behavior relies only on the validity of particular message authentication codes (MAC) instead of the precise MAC value, we use a predicate for each such MAC value called valid_MAC(·) whose truth value signifies the validity of the MAC.

Presence predicate. In the paging procedure of 5G, the behavior of the device relies on the type of identity (TMSI or C-RNTI) used in the received paging message. Instead of capturing the exact identity, we use two mutually exclusive predicates isPresentTMSI(·) and isPresentCRNTI(·). The first (resp., second) predicate is true only if the identity used is TMSI (resp., C-RNTI).

Grouping predicate. When the core network denies the device connection, it sends the device a reject message which contains a payload signifying the reason for such rejection. There are 70 possible rejection reasons and the device reacts differently based on the rejection reasons. Our inspection of the standard revealed that the device behavior can be used to divide these 70 reasons into three mutually exclusive groups. We thus capture the group membership of the rejection reason with the following three mutually exclusive predicates: ReasonGroup1(·); ReasonGroup2(·); ReasonGroup3(·) where predicate ReasonGroup$i$(·) is true only if the rejection reason belongs to group $i \in \{1, 2, 3\}$.

In addition to the payload, we also faithfully model counters with a finite domain. Some of these counters (e.g., link counters) are also used as payloads. Some of these counters are modeled as they are without employing any sort of predicate abstraction.

4.4 Modeling Timers

In our 5G model, some transitions are dependent on various timers. The protocol behaves differently based on the various state of the timer. We do not model timers counting down, that is, given a timer for 5 time units we do not model every clock tick as this would severely impede the analysis scalability.

For each timer in our model, we maintain two Boolean variables timer_started and timer_expired. The state variable timer_started signifies the start of the timer whereas the environmental variable timer_expired signifies the expiration of the timer. Other timer behavior such as timer is ongoing can be captured in the following way: timer_started $\wedge$ $\neg$timer_expired.

4.5 Capturing Multi-Layer Adversary Behavior

As we model multiple layers of the protocol stack, we have to handle packets generated by protocols of both layers. Conceptually, NAS layer protocol packets are sent as payloads of the RRC layer packets. Moreover, there are RRC layer packets which may not carry any NAS layer packet. If we were to let the adversary change arbitrary data of the packet, it may generate packets that are not compliant to any protocol packet and will result in parsing errors.

To ensure that all packets after adversarial modification result in compliant protocol packets, we include additional well-formedness properties. For instance, whenever a NAS layer packet is sent from the UE to the AMF, it is embedded as a payload of an RRC packet of type `ul_info_transfer`. When such a packet is encountered, the adversary would only modify the NAS layer packet, not the RRC layer packet. We add similar rules for other relevant protocol packets.

5 IMPLEMENTATION

We now discuss our realization of 5GReasoner.

5.1 Formal Property Gathering

Conformance Requirement Documents. After carefully reviewing the conformance requirement document [7], we extract 74 conformance tests for the NAS layer that are within the scope of the paper. We formalize and write them as properties which our model adheres to[1]. Out of these 74, 9 are liveness properties and 65 are safety properties. We also extract 63 conformance tests for the RRC layer and then formalize and write them as properties. Out of these 63, 6 are liveness properties and 57 are safety properties. The excluded conformance tests fall under one of the following categories: (1) Performance related tests; (2) Refers to procedures not modeled (e.g., Mobile Initiated Connection Only, Wifi-calling).
Technical Requirement and Specifications. We extracted, formalized, and verified a total of 50 implicit properties from the technical specification and requirement documents [4–6] for both NAS and RRC layers that are within the scope of the paper. Out of these 50, 33 are liveness properties and 17 are safety properties.

5.2 Protocol Abstraction

We have developed a model generator that takes as input the state machine of the protocol written in a graphviz-like language and outputs a Lustre [27] or SMV [38] description of the model. The UE state machine at the NAS layer has 7 states and 76 transitions whereas the AMF state machine has 5 states and 66 transitions. Our RRC layer UE state machine consists of 6 states and 54 transitions and the base station state machine is comprised of 9 states and 42 transitions. The current models and the respective properties are available at https://github.com/relentless-warrior/5GReasoner.git.

[1]The model's compliance with the concretized test suites is crucial to establish our model's correctness concerning the specification.

5.3 MCheck Component Instantiation

To model check our state machine representation, we use two different tools, NuXmv [18], and Kind 2 [19]. We decide to use both these tools to complement each other and overcome their weaknesses.
NuXmv. NuXmv allows one to check for liveness properties which are essential for reasoning about some of the requirements, however, it is not capable of handling data from infinite domains. Our NuXmv model consists of 847 lines of code for NAS in isolation, 629 for RRC in isolation, and 1476 for the cross-layer.
Kind 2. Kind 2 is able to handle data from an infinite domain; however, it is unable to check for liveness properties. Our Kind 2 model consists of 2098 lines of code for NAS in isolation, 1494 for RRC in isolation and 3771 for the cross-layer.

5.4 CPVerif Implementation

We use ProVerif [15] as the cryptographic protocol verifier. We verified 23 secrecy (for confidentiality), 52 injective-correspondence (for strong authentication), 52 correspondence (for weak-authentication), and 16 observational equivalence (for linkability) properties.

5.5 Formal Protocol Analysis in Isolation

To reduce a wide array of spurious counterexamples generated due to a large number of environment variables spanning across two control-plane protocol layers, we first reason about each layer in isolation with 5GReasoner.
NAS Layer. We first instantiate our analysis with $\mathcal{M}_{\text{UE}}^{\text{NAS}}$ and $\mathcal{M}_{\text{AMF}}^{\text{NAS}}$ and the corresponding NAS layer properties ϕ_{NAS} relevant to the NAS layer procedures. While analyzing the NAS layer test suites and procedures (e.g., registration, configuration update, service request, handover) in isolation, we consider the following aspects for protocol model abstraction. (A) The UE and the base station reliably perform the relevant RRC layer procedures, such as, RRC connection setup, resume, reestablish, reconfiguration, and negotiation of security algorithms that are prerequisites to initiating respective NAS layer procedures. For instance, to reason about the NAS layer registration procedure, 5GReasoner assumes that the UE and the base station set up the required RRC layer connection with the given security context by following the corresponding RRC layer procedures. (B) An incoming event from RRC to NAS layer is represented by a boolean environment variable. As an illustration, when the UE's RRC layer releases the connection with the base station, the RRC layer notifies the NAS layer to move the UE to the *idle* state. We abstract such inter-layer communications as boolean environment variables while analyzing in isolation.
RRC Layer. For analyzing the RRC layer procedures in isolation, we similarly instantiate 5GReasoner with $\mathcal{M}_{\text{UE}}^{\text{RRC}}$, $\mathcal{M}_{\text{BS}}^{\text{RRC}}$ and ϕ_{RRC}. In our instantiation, we abstract the incoming and outgoing events for the RRC layer in the following ways. (A) We abstract the invocation of RRC layer procedures from the NAS layer using boolean

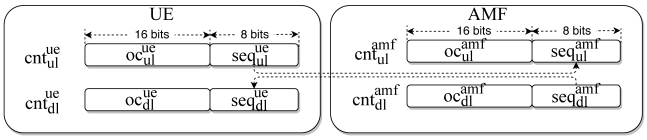

Figure 8: Counters maintained by UE and AMF

environment variables. (B) We model the RRC layer to consume the outgoing events generated for the NAS layer. (C) Though RRC layer uses the PDCP (packet data convergence protocol) layer for ensuring encryption and integrity protection of the RRC layer's control-plane messages, we incorporate the functionalities for encryption of integrity protections into RRC layer for ease of reasoning and minimizing state explosion problems.

5.6 Formal Cross-layer Protocol Analysis

As all unexpected interactions cannot be realized by reasoning about procedures in isolation, we also model and reason about the control-plane procedures considering two layers. For cross-layer analysis, we model a channel to carry at most one message at each step between two participants to avoid any non-determinism. This, however, poses additional challenges because a participant may occasionally require to send both NAS and RRC layer messages at the same time when a particular condition of the participant's next state transition becomes true. For instance, if AMF receives a `ue_dereg_request` message from a UE, it sends `dereg_accept` message to UE-NAS and sends a *context release request* to BS-RRC for releasing the RRC layer connection. To address this challenge, 5GReasoner first lets the AMF-NAS send `dereg_accept` message to the UE and then sets a boolean state variable "context_release_required" to true which triggers the AMF-NAS to send the context_release_request message to the base station at the next state.

6 FINDINGS

This section highlights the vulnerabilities and attacks uncovered by 5GReasoner. We also discuss how the uncovered vulnerabilities can be further exploited along with some domain knowledge to perform detrimental attacks. We summarize our findings in Table 1.

6.1 Attacks in NAS Layer

We first present our findings on the 5G NAS layer.

6.1.1 ***NAS Counter Reset.*** With this attack, the adversary exploits a potential vulnerability lurking in the handling of NAS counter values typically used in generating/verifying the message authentication codes (MAC) for replay protection of NAS layer messages. This may allow an adversary to replay particular messages which can reset and desynchronize the counter values between the device and network, and may further enable the adversary to cause over-billing to a user.

Adversary Assumptions. The adversary knows the C-RNTI [45] (i.e., the layer-2 identity of the victim device) and uses it to follow and eavesdrop on victim's downlink messages from the AMF. The adversary is also capable of setting up a fake base station or a Man-in-the-Middle relay [28, 45] that can additionally replay `sec_mode_command` and `sec_mode_complete` messages.

Vulnerability. The UE and AMF independently maintain a set of NAS counters— $\mathsf{cnt}^{\mathsf{ue}}_{\mathsf{ul}}$ and $\mathsf{cnt}^{\mathsf{ue}}_{\mathsf{dl}}$ at UE, and $\mathsf{cnt}^{\mathsf{amf}}_{\mathsf{ul}}$ and $\mathsf{cnt}^{\mathsf{amf}}_{\mathsf{dl}}$ at AMF (as shown in Figure 8)— to prevent replay of both uplink (UE→AMF) and downlink (UE←AMF) transmissions. A NAS counter (cnt) is a 24-bit unsigned integer comprised of a 16-bit overflow counter (oc) concatenated with a 8-bit sequence number (seq).

The sender ($\mathcal{S}$) uses its locally stored NAS counters (e.g., $\mathsf{cnt}^{\mathsf{ue}}_{\mathsf{ul}}$ for UE or $\mathsf{cnt}^{\mathsf{amf}}_{\mathsf{dl}}$ for AMF) as input to the integrity protection algorithm for generating message authentication code whereas the receiver ($\mathcal{R}$) uses the sequence number included in the received message and estimates the overflow counter to compute the corresponding counter used as input to the integrity verification algorithm. If verification passes, in case of downlink transmissions, the receiver updates its $\mathsf{cnt}^{\mathcal{R}}_{\mathsf{dl}}$ with the estimated $\mathsf{cnt}^{\mathcal{S}}_{\mathsf{dl}}$.

If the estimated sequence number wraps around, the overflow counter is incremented by one. According to TS 33.501, clause: 6.4.3.1, for detecting wrap around, the AMF checks whether its $\mathsf{seq}^{\mathsf{amf}}_{\mathsf{dl}}$ or $\mathsf{seq}^{\mathsf{amf}}_{\mathsf{ul}}$ is *close* to 2^8 [6]. Primarily, the vulnerability stems from the lack of specification [4, 6] when a currently received message has a sequence number (e.g., $\mathsf{seq}^{\mathsf{S}}_{\mathsf{dl}}$) smaller than that of the last accepted message ($\mathsf{seq}^{\mathsf{R}}_{\mathsf{dl}}$). Due to such underspecified policy of "wrap-around", the receiver may handle the overflow counter ($\mathsf{oc}^{\mathcal{R}}_{\mathsf{dl}}$) in one of the two possible ways: (i) do not increment the overflow counter ($\mathsf{oc}^{\mathcal{R}}_{\mathsf{dl}}$) to estimate $\mathsf{cnt}^{\mathcal{S}}_{\mathsf{dl}}$ when neither the received sequence number nor the locally stored number is close to 2^8; (ii) increment $\mathsf{oc}^{\mathcal{R}}_{\mathsf{dl}}$ assuming that ($\mathsf{seq}^{\mathcal{S}}_{\mathsf{dl}} + 2^8$ -$\mathsf{seq}^{\mathsf{R}}_{\mathsf{dl}}$) number of messages between the received sequence number ($\mathsf{seq}^{\mathcal{S}}_{\mathsf{dl}}$) and the receiver's stored sequence number ($\mathsf{seq}^{\mathcal{R}}_{\mathsf{dl}}$) have been lost. Both of these interpretations entail two different potential vulnerabilities. The case (i) enables the adversary to replay the integrity-protected `sec_mode_command` and `sec_mode_complete` messages used for changing the current cipher suite or for resynchronizing the uplink NAS counters between UE and AMF. The case (ii), on the other hand, allows the adversary to stealthily drop packets without getting noticed. On top of that, if the network operator and device manufacturer interpret this "wrap around" policy differently, there will be inter-operability issues which may disrupt the regular services. In what follows, we discuss a potential attack for case (i) in detail and briefly outline the potential adversarial impact for case (ii).

Detection. We modeled $\mathcal{M}^{\mathsf{NAS}}_{\mathsf{UE}}$ and $\mathcal{M}^{\mathsf{NAS}}_{\mathsf{AMF}}$ considering the case (i). We check the $\mathcal{M}^{\mathsf{NAS}}_{\mathsf{adv}}$ against the following property implicitly mentioned in TS 24.501: "*Replay protection assures that the same NAS message is not accepted twice by the receiver. Specifically, for a given 5G NAS security context, a given NAS COUNT value shall be accepted at most one time and only if message integrity verifies correctly*". This is violated by a counterexample in which the adversary sends a `sec_mode_command` and receives `sec_mode_complete` message. Since `sec_mode_command` message has integrity protection, we resort to ProVerif to reason the following *correspondence* property: *If a UE sent* `sec_mode_complete` *message, then the AMF previously sent a* `sec_mode_command` *message.* ProVerif provided an attack strategy which allows the adversary to replay the `sec_mode_command` message. ProVerif also provided a similar attack strategy for replaying the `sec_mode_complete` message to the core network.

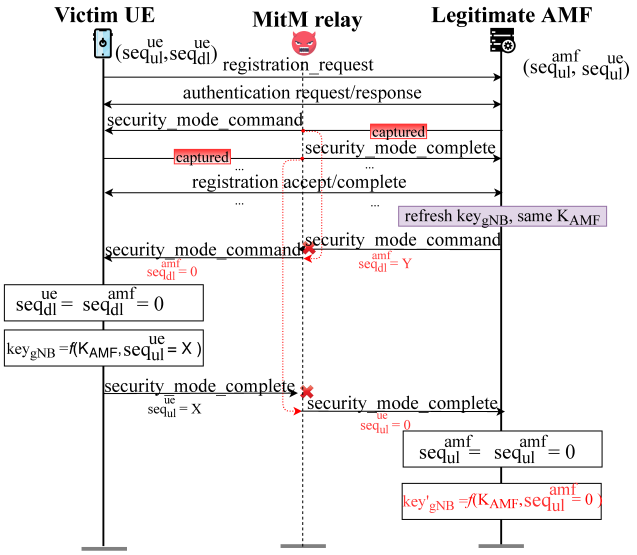

Figure 9: Counter reset attack

Attack Description. The adversary using a fake base station or a MitM relay transmits unauthenticated initial broadcast (`sysinfoblock1` and `sysinfoblock2`) messages with higher signal strength [30] and forces the victim UE to connect to itself. The adversary captures both `sec_mode_command` and `sec_mode_complete` message with $\text{seq}_{\text{dl}}^{\text{amf}}$ and $\text{seq}_{\text{ul}}^{\text{ue}}$, respectively being 0 during the initial registration procedure as shown in Figure 9. Whenever the UE completes the registration with a legitimate AMF, the adversary waits for the `sec_mode_command` and the `sec_mode_complete` massage from AMF and UE, respectively. If the AMF again sends `sec_mode_command` message for refreshing RRC layer keys but long before the $\text{seq}_{\text{dl}}^{\text{amf}}$ gets close to 2^8, the adversary can selectively drops the message (by guessing the type of the encrypted message from the packet length, context, and the DCI values [45]) and replays the previously captured `sec_mode_command` and `sec_mode_complete` messages with count values set to 0 (shown in Figure 9). Both the replayed command and the complete messages will be successfully verified because the overflow counter values (e.g., $\text{oc}_{\text{dl}}^{\text{amf}}$ and $\text{oc}_{\text{dl}}^{\text{ue}}$) at the legitimate UE and AMF will still be the same, i.e., 0 since the $\text{seq}_{\text{dl}}^{\text{ue}}$ and $\text{seq}_{\text{ul}}^{\text{amf}}$ are not close to 2^8. After successful verification, the AMF and UE updates its locally maintained $\text{seq}_{\text{ul}}^{\text{amf}}$ and $\text{seq}_{\text{dl}}^{\text{ue}}$, respectively with the received $\text{seq}_{\text{ul}}^{\text{ue}}$ and $\text{seq}_{\text{dl}}^{\text{ue}}$, respectively. Therefore, if the adversary can replay `sec_mode_command` and `sec_mode_complete`, it will be able to reset $\text{seq}_{\text{ul}}^{\text{amf}}$ and $\text{seq}_{\text{dl}}^{\text{ue}}$. Note that, with this attack, the adversary, however, cannot reset $\text{seq}_{\text{ul}}^{\text{ue}}$ and $\text{seq}_{\text{dl}}^{\text{amf}}$ locally maintained by UE and AMF, respectively.

Impact. With this attack, the adversary is able to desynchronize the uplink counter values between the victim UE and the legitimate AMF. These counters are crucial parameters which the UE and the AMF use to generate the key_{gNB}, later on, share with the base station for providing RRC/PDCP layer security. Since the attack desynchronizes the values of $\text{seq}_{\text{ul}}^{\text{ue}}$ and $\text{seq}_{\text{ul}}^{\text{amf}}$, the key_{gNB} generated by UE and AMF will be different and thus induce the victim UE to re-establish the connection with the network using another registration procedure. To make matters worse, the adversary may replay the same attack numerous times to force the victim UE to keep re-establishing the connection, causing DoS and battery depletion together. It is also possible to replay the same user-plane packets over and over again with same (key_{gNB}) to cause over billing to the client.

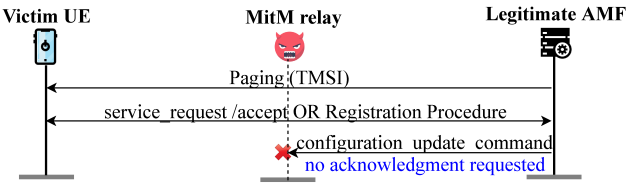

Figure 10: Neutralizing TMSI Refreshment

Attack Considering Case (ii). The adversary with a MitM relay can drop an arbitrary number of packets without getting detected since the receiver accepts a packet even if the received sequence number is smaller than the stored sequence number at the receiver end.

6.1.2 ***Uplink NAS Counter Desynchronization.*** In this attack, the adversary exploits the lack of rate-limiting on the number of failed *security mode procedure* to desynchronize UE's uplink NAS counters. Impacts of the attack are prolonged desynchronization and DoS between UE and AMF.

Adversary Assumptions. Unlike the *counter reset* attack discussed above, we assume that the adversary knows the victim UE's C-RNTI [45], but does not require it to eavesdrop on or capture `sec_mode_command` message sent in the previous session.

Vulnerability. The root vulnerability of this attack is the lack of attempt counter for the security mode command procedure. Neither the UE nor the AMF checks how many times it fails to verify the `sec_mode_command` or `sec_mode_complete` message in a row. This allows the adversary to inject an arbitrary number of invalid `sec_mode_command` messages to the victim device which induces the UE to send `sec_mode_reject` messages as response while incrementing its $\text{seq}_{\text{ul}}^{\text{ue}}$ and $\text{oc}_{\text{ul}}^{\text{ue}}$.

Detection. We model check the $\mathcal{M}_{\text{adv}}^{\text{NAS}}$ against the following property: *the AMF will correctly verify a legitimate* `sec_mode_complete` *message sent by the UE in response to a* `sec_mode_command` *message sent by the AMF.* This is trivially violated by a counterexample in which the adversary sends/receives at most 2^8 arbitrary `sec_mode_command` messages to the target UE triggering the increment of the uplink overflow counter $\text{seq}_{\text{ul}}^{\text{ue}}$ by 1. After that any uplink message will violate our sanity check of the uplink counter.

Attack description: The adversary using a fake base station or a MitM relay connects to a victim UE and sends `sec_mode_command` messages containing arbitrary values for MAC. Since the UE fails to verify the MAC, it sends `sec_mode_reject` to the fake base station and increments its $\text{seq}_{\text{ul}}^{\text{ue}}$ by 1 (i.e., consistent behavior with respect to the reception of a uplink message). The attacker may continue this process until he observers a $\text{seq}_{\text{ul}}^{\text{ue}}$ wraps around. This signifies that the $\text{seq}_{\text{ul}}^{\text{ue}}$ at the UE has reached a value of 2^8 and UEs locally stored $\text{oc}_{\text{ul}}^{\text{ue}}$ has been incremented by 1. This desynchronizes the

uplink NAS counters (cnt^{ue}_{ul} and cnt^{amf}_{ul}) between the UE and the legitimate AMF.

Impact.: As a result of such desynchronization, though the victim UE can correctly verify the downlink messages from AMF, but the legitimate AMF will discard any uplink messages (both control plane and data) sent from the victim UE. Even if the legitimate AMF tries to resynchronize the uplink NAS counters with a new `sec_mode_command` message, the AMF cannot verify the `sec_mode_complete` message because of the mismatch of uplink overflow counters. This allows the adversary to carry out a prolonged DoS and service disruption to the user. The victim UE will have to either drop the connection and re-authenticate again with the legitimate AMF or the AMF will initiate a `tracking_area_update` procedure (default timer for which is 54 mins [4]) to re-synchronize with the UE.

6.1.3 ***Exposing NAS Sequence Number.*** In this attack, the adversary passively monitors the uplink and downlink transmissions of the victim UE and reveals the cnt^{ue}_{ul} and cnt^{amf}_{dl}. The adversary then uses this value to monitor victim UE's cellular activity.

Adversary Assumptions. The adversary knows the victim UE's C-RNTI [45] and eavesdrops on the uplink and downlink NAS messages of the victim UE. We, however, assume that the adversary does not know the session keys to decrypt the messages.

Vulnerability. The NAS sequence number, part of the cnt^{ue}_{ul} and cnt^{amf}_{dl}, is exchanged in plaintext between the UE and the AMF as part of the NAS signaling messages. The sender encrypts a NAS message and computes the MAC with the corresponding NAS counter values and then includes the MAC and NAS sequence number in plaintext with the message.

Detection. While checking the observational equivalence property for the prior linkability/coarse-grained location tracking attack (migrated from 4G LTE), we also check the *secrecy* of the NAS sequence numbers sent in the NAS messages. ProVerif returned a violation of this secrecy property in which we observed that the NAS sequence numbers of the UE and AMF have been exposed in the `sec_mode_command` (cnt^{amf}_{dl}) and `sec_mode_complete` (cnt^{ue}_{ul}) messages.

Attacks. The adversary learns the NAS sequence numbers and may relate this information to infer the number of AKA sessions or the number of change of cipher suites due to either handover from 4G to 5G or the wrap-around of NAS counters. The adversary can check the NAS counter values at different times and loosely infer the engagement level (e.g., service consumption) during each interval.

Impact. This attack may enable the adversary to profile the service usage or monitoring the activity of a target UE.

6.1.4 ***Neutralizing TMSI Refreshment.*** The AMF uses the *configuration update procedure* to assign a new TMSI to a user upon completion of a service, such as a phone call or SMS. In this attack, the adversary prevents the network from changing the victim device's TMSI when the change is required. This further enables the adversary to correlate victim's phone number with the old/new TMSI which is a useful arsenal [37] for location tracking.

Adversary Assumptions. We assume the adversary knows the victim UE's C-RNTI [45], and can eavesdrop and selectively drop victim UE's downlink messages through a MiTM relay [28, 45] consisting of a fake UE and a fake base station. We also assume that the adversary knows the victim's old TMSI using another attack uncovered in the cross-layer analysis (Attack in Section 6.3.1).

Vulnerability. The AMF sends an encrypted and integrity protected `config_update_command` message to initiate the configuration update procedure. This message includes the new TMSI and may indicate if an acknowledgment (i.e., `config_update_complete`) from the UE is requested. If there is no such indication, the adversary may exploit this to disrupt the TMSI refreshing mechanism.

Detection. We checked the following property with **MCheck**: *If the AMF initiates configuration update procedure, it will eventually assign a new TMSI to the device.* We obtained a counterexample in which the adversary drops the `config_update_command` message sent from AMF to carry out the following attack.

Attacks. The adversary with the knowledge of victim device's C-RNTI and using a MitM relay can monitor the downlink messages sent from the AMF. It can also drop the `config_update_command` message (containing no indication of UE's acknowledgment) destined for the victim device. As a result of this, the victim has the old TMSI, but the network will have both the old and new TMSIs until the device reconnects to the network. Now, if there are any incoming services (e.g., phone call or SMS) waiting for the device, the network first tries paging with new-TMSI for a certain number of times and then retries paging with the old-TMSI. Provided that the adversary knows the old TMSI of the victim UE, it can compute the victim's paging occasion and hijack the paging channel, and thus prevent legitimate paging messages from reaching to the victim UE. After a certain number of attempts, the network aborts both the paging and configuration update procedures. The adversary, thus, ceases the changing of the victim's TMSI in that case.

Impact. This attack may enable the adversary to force the network to continue using the same TMSI for a victim long term which makes the target user vulnerable to also location tracking attacks.

6.1.5 ***Cutting off the Device.*** The goal of the adversary is to exploit the lack of integrity protection of the initial `reg_request` or `ue_dereg_request` messages to stealthily disconnect the victim device from the network.

The adversary knowing the C-RNTI [45] or TMSI (Attack in Section 6.3.1) of the victim device can impersonate the victim by sending the `reg_request` or `ue_dereg_request` messages to the AMF. The AMF discards these initial request messages as they do not contain integrity protection. This induces the AMF to implicitly release the existing connection with the victim device by deregistering the device from the network and connecting to the impersonator.

6.2 Attacks in RRC Layer

We now present the findings of 5GReasoner when analyzing the RRC layer in isolation.

6.2.1 ***Denial-of-Service with*** `rrc_setup_request`. In this attack, the adversary exploits the lack of integrity protection of an RRC layer message to stealthily disconnect a target UE from the network.

Adversary Assumptions. We assume that the adversary already knows the TMSI (Attack in Section 6.3.1) of the device and is capable of setting up a fake UE impersonating the victim device.

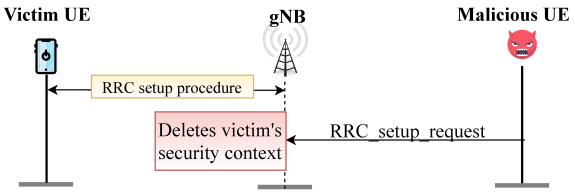

Figure 11: DoS with rrc_setup_request

Vulnerability. The `rrc_setup_request` message, which does not include any integrity protection, is sent by the UE to set up an RRC layer connection with the base station. This lack of integrity protection enables an adversary to spoof this message to mount a potential denial-of-service attack.

Detection and Attack Description. We check the $\mathcal{M}^{\text{RRC}}_{\text{adv}}$ against the following property implicitly mentioned in TS 38.331: *If the base station (i.e., gNB) has an RRC security context in the current state, it will exist forever unless there is a* `rrc_setup_request` *from the UE.* This is trivially violated by a counterexample in which the adversary impersonating a victim UE sends a `rrc_setup_request` (with victim's TMSI) message to the base station to which the victim is already connected (shown in Figure 11). This induces the base station to delete victim UE's current RRC security context by implicitly releasing the connection with the victim device and connecting with the malicious UE. We also confirm this with ProVerif with a *correspondence* property on the attack traces by the model checker.

Attack Variant. The adversary can also perform similar attacks using `rrc_reestablish_request` or `rrc_resume_request` message (containing victim's TMSI) by sending them over the initial signaling channel (i.e., signal radio bearer 0) as these messages also do not have any integrity protection.

Impact. The adversary can stealthily disconnect a victim UE from the network.

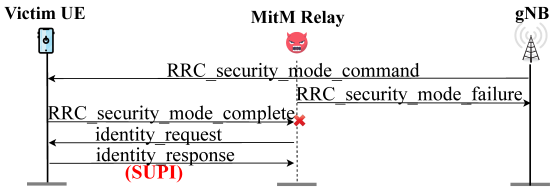

Figure 12: Installing Null Cipher and Null Integrity

6.2.2 ***Installing Null Cipher and Null Integrity.*** The goal of this attack is to trick the victim UE in *limited service mode* to use null cipher and null integrity protections. This unleashes the MitM relay to not only relay the messages, but also to act as a full-form Man-in-the-Middle attacker, i.e., to eavesdrop on, inject, or drop legitimate messages.

Adversary Assumptions. We assume that the adversary already knows the victim's C-RNTI [45] and is able to set up a MitM relay [28, 45] through which the device is connected to a core network.

Vulnerability. RRC layer's *security mode procedure* is typically initiated by the base station when it wants to set up/refresh the RRC/PDCP layer security contexts. In this attack, the adversary exploits the lack of integrity protection in `rrc_sec_mode_failure` message which the UE sends to the base station when the UE cannot verify the `rrc_sec_mode_command` message.

Detection and Attack Description. We model check the $\mathcal{M}^{\text{RRC}}_{\text{adv}}$ against the following property implicitly mentioned in TS 38.331: *If the base station requires the access stratum (i.e., AS or RRC layer) security context to be set up, the base station will eventually establish the security context.* The **MCheck** yields a counterexample in which the adversary impersonating a victim UE sends a `rrc_sec_mode_failure` message in response to the `rrc_sec_mode_command` message. Given the attack trace, we also confirm this with ProVerif using a *correspondence* property which confirms adversary's capability to inject `rrc_sec_mode_failure` message. According to TS 38.331 (clause: 5.3.4.3), the base station and UE continue using the security configuration used prior to the reception of `rrc_sec_mode_command` message, i.e., neither apply integrity protection (NIA0) nor ciphering (NEA0). The null integrity protection algorithm is used only for control-plane messages and for the UE in *limited service mode.* A UE in *limited service mode* may still be allowed to establish an emergency session (for emergency 911 phone calls) by sending the emergency registration request message (TS 33.501, clause: 6.7.3.6). In case the null integrity protection algorithm is used, the 'NULL' ciphering algorithm is also used.

Impact. The adversary forces the victim to use null cipher and null integrity at the RRC and PDCP layer which allows the adversary to see and inject any control-plane messages. In such limited service mode, the UE may expose its SUPI in plaintext if the fake base station sends a `identity_request` message as shown in Figure 12.

6.2.3 ***Lullaby Attack.*** In this attack, the adversary intermittently forces the victim device to release the existing connection with the legitimate network and traps the device into the idle state until the device needs to reconnect with the network. Incessant switching from idle to connected state requires the UE to spend its energy on further interactions and cryptographic operations which may cause the device to deplete its battery faster.

Adversary Assumptions. We assume that the adversary already knows the C-RNTI [45] of the victim device. The victim device is assumed to be connected with a fake base station or to a MitM relay [28, 45].

Vulnerability. The base station sends the encrypted and integrity protected `rrc_reconfiguration` message to the UE to reconfigure the RRC layer connection with the UE. If the UE cannot verify the integrity protection of `rrc_reconfiguration` message, it releases the connection and moves to the idle state. In this attack, the adversary exploits this response behavior to perform the following denial-of-service attack.

Detection and Attack Description. We model check the $\mathcal{M}^{\text{RRC}}_{\text{adv}}$ against the following property: *If a UE is in* RRC connected *state and the base station sends a* `rrc_reconfiguration` *message, it will remain in the RRC connected state after the reconfiguration procedure.* The **MCheck** provided a counterexample in which the adversary impersonating the base station sends a `rrc_reconfiguration` message with an arbitrary MAC. Upon receiving this message, the victim UE cannot verify the integrity and thus moves to the *RRC idle* state by locally/implicitly releasing the RRC connection.

Attack Variants. The adversary may use a similar philosophy, but use the `rrc_resume` and `rrc_reestablish_request` messages with arbitrary invalid MAC to achieve a similar impact on the victim UE.
Impact. This attack enables the adversary to coerce a device to move to the *RRC idle* state by deleting its security context (TS 38.331, clause 5.3.11). Later on, if the UE has any outgoing/incoming message to be sent/received, the UE will again establish the connection and set up the RRC layer security context. The adversary may perform this attack more frequently to quickly drain the battery of the victim UE.

6.2.4 ***Incarceration with*** `rrc_reject` ***and*** `rrc_release`. The overarching goal of the adversary in this attack is to keep the victim device in a connection initiation loop with the base station so that the adversary can hold off the victim device from connecting to a legitimate network as long as possible.
Adversary Assumptions. We assume the adversary already knows the victim device's C-RNTI [45] or TMSI (Attack in Section 6.3.1) and is capable of setting up two fake base stations.
Vulnerability. When a UE is in the *RRC idle* state, it accepts `rrc_reject` messages without integrity protection.
Attack Description. One of the fake base stations lures the victim device to connect and send `rrc_setup_request` message to itself. In response, the fake base station may reply with a non-integrity protected `rrc_reject` message. Since the UE is in the idle mode, it accepts the non-integrity protected *reject* message. If the fake base station sets the *mobility backoff timer* in the *reject* message, the victim waits in the idle state for a maximum 16 seconds and tries reconnecting with the base station. The base station can then force the victim UE to stay in this connection establishment loop by sending the *reject* messages again and again. The victim UE, however, maintains a connection establishment fail counter on the same cell and changes the cell selection criteria once that counter reaches the limit (e.g., 4 trials). To prevent the counter from reaching the maximum limit on the same cell, the fake base station sends `rrc_release` messages interleaving with `rrc_reject` message. The adversary includes the *redirected carrier information* in the `rrc_release` message to persuade the victim to connect with the second fake base station operating on the redirected frequency.
Impact. By turning on/off the fake base stations one at a time, the adversary can keep the victim device in such a malicious connection establishment loop as long as possible.

6.3 Cross-Layer Attacks

5GReasoner's cross-layer analysis uncovered two new attacks. In what follows, we discuss these two new attacks.

6.3.1 ***Exposing Device's TMSI and Paging Occasion***. With this attack, the adversary exploits vulnerabilities in both RRC and NAS layers to learn victim device's TMSI as well as the paging occasion (i.e., the time instance or radio frame number at which the network sends paging messages to a UE) which can then be used to track the location of the user.
Adversary Assumptions. For this attack, we assume the adversary to know the victim's C-RNTI [45] and phone number. We also assume that the adversary can selectively drop a message with the use of a MitM relay [28, 45] and also eavesdrop on the paging broadcast channel of the legitimate base station.
Vulnerability The adversary exploits the specification's design weakness of not requiring an acknowledgment of the `rrc_release` message in the RRC layer and also the paging *retransmission* requests with the same TMSI in the NAS layer.
Detection and Attack Description. We model check $\mathcal{M}_{\text{adv}}^{\text{cross-layer}}$ against the following property extracted from TS 24.501 [4]: *Paging containing TMSI will be sent just once between two incoming service notifications*. The **MCheck** yields a counterexample in which the adversary (equipped with a MitM relay) first drops the `rrc_release` message sent for the victim by the legitimate base station. This forces the victim UE not to release the RRC connection and to stay in the RRC connected state. As a result, the network assumes that the victim device is in IDLE mode. Now the adversary makes multiple phone calls (i.e., service notifications) to the victim's phone number. For each call, the network will request the base station to broadcast paging messages containing the victim's TMSI. Since the idle mode paging occasion for a UE is different [3] from that of the connected mode paging occasion, the victim UE will not receive base station's paging message. The adversary, however, with a paging channel sniffer will find a TMSI that appears in more than one paging messages triggered by multiple phone calls and will infer that as the victim's TMSI. Since 5G proposes to compute the idle mode paging occasion using TMSI (instead of IMSI in 4G), the adversary can also learn victim's paging occasion from knowing the TMSI.
Impact. The adversary knowing the victim's TMSI or paging occasion can track the location of the device (in case of infrequent TMSI update policy used by network operators [48] as it is left for operators' implementations by the standard [4]), or hijack the paging channel to broadcast fake emergency alerts or use it as the pre-requisite step for other attacks [28, 50].

6.3.2 ***Exposing Device's I-RNTI***. The adversary follows the similar attack philosophy and exploits similar vulnerabilities to the first cross-layer attack and learns the victim's I-RNTI and paging occasion for tracking the user.
Attack Description. In this attack, if the adversary drops the `rrc_release` message (containing the indication of RRC suspend) sent for the victim by the legitimate base station, it may force the victim to stay in the RRC connected state instead of moving to the inactive state. Now the adversary makes multiple phone calls to the victim's phone number for each of which the base station broadcasts paging messages containing victim's I-RNTI. Thus, the adversary may learn the victim's I-RNTI and the paging occasion at which the base station sent paging containing the victim's I-RNTI. The adversary may use this to further hijack the paging channel and perform stealthy denial-of-service attacks.

6.4 Prior Attacks Detected by 5GReasoner

In addition to the newly discovered attacks, 5GReasoner also identified 5 attacks, the underlying vulnerabilities of which were either detected by prior work [14] or were inherited from 4G LTE. Table 1 summarizes the list of such attacks detected by 5GReasoner for 5G.

Attack	Vulnerability	Assumption & Validation	New Attack?	Notable Implication
NAS Layer				
Counter reset	Generating/verifying integrity using MAC in sec_mode_command and sec_mode_complete messages	Known C-RNTI [45], MitM relay [28, 45]	Y	DoS, over billing
Uplink NAS Counter Desynchronization	Lack of attempt counter for the security mode command procedure and generating/verifying integrity using uplink counters in sec_mode_command and sec_mode_reject messages	Known C-RNTI [45], MitM relay [28, 45]	Y	Prolonged DoS
Exposing NAS sequnece number	cnt_{ul}^{ue} & cnt_{dl}^{amf} transmitted in plain-text	Known C-RNTI [45], session keys unkown	Y	Service profiling
Neutralizing TMSI refreshement	configuration_update_command may not require acknowledgment	Known C-RNTI [45], old TMSI (Attack 6.3.1), MitM relay [28, 45]	Y	Location Tracking
Cutting of the device using `reg_request`	AMF accepts registration_request without integrity	Know C-RNTI [45]	Y	DoS
Cutting of the device using `ue_dereg_request`	AMF accepts de-registration_request without integrity	Known C-RNTI [45]	Y	DoS
Downgrade using reject messages	No integrity in reject message	Known C-RNTI [45] or TMSI (Attack 6.3.1)	Inspired by [48]. [28]	Downgrade from 5G
Linkability using `authentication_failure`	Different response in MAC failure	Known TMSI (Attack 6.3.1)	Inspired by [11] in 3G and [14] in 5G	Tracking
Paging channel hijacking	No integrity check in paging messages	Known S-TMSI (Attack 6.3.1) or I-RNTI	Inspired by [28]	Stealthy DoS
Panic attack	No integrity check in paging messages	Malicious *gNB* [28, 45]	Inspired by [28]	Atificial chaos, mass victimization
Linkability/Tracking using `sec_mode_command`	Generating/verifying integrity using MAC in sec_mode_command message	Known C-RNTI [45], MitM relay [28, 45]	Inspired by [28]	Tracking
RRC Layer				
Denial of service using `rrc_setup_request`	No integrity in `rrc_setup_request`	Known C-RNTI [45]	Y	DoS
Installing null cipher and integrity	Lack of integrity protection in `rrc_sec_mode_failure`	Known C-RNTI [45], MitM relay [28, 45]	Y	SUPI catching
Lullaby attack with `rrc_reconfiguration`	UE 's response to invalid integrity protection to the `rrc_reconfiguration`	Known C-RNTI [45], fake base station [28, 45]	Y	Force state change, battery draining
Lullaby attack using `rrc_reestablish_request`	UE 's reaction to invalid integrity protection to the `rrc_reestablish_request`	Known C-RNTI [45], fake base station [28, 45]	Y	Force state change, battery draining
Lullaby attack with `rrc_resume`	UE's response to `rrc_resume`	Known C-RNTI [45], fake base station [28, 45]	Y	Force state change, battery draining
Incarceration with `rrc_reject` and `rrc_release`	rrc_reject is not integrity protected	Known C-RNTI [45] or TMSI (Attack 6.3.1)	Y	DoS
Incarceration with `rrc_reestablish_reject`	rrc_reestablish_reject is not integrity protected	Known C-RNTI [45] or TMSI (Attack 6.3.1)	Y	DoS
Cross Layer Attacks				
Exposing Device's TMSI and Paging Occasion	Lack of acknowledgment of rrc_release & paging retransmissions	Known C-RNTI [45], MitM [28, 45]	Y	Location Tracking, stealthy DoS, downgrade from 5G, artificial chaos, mass victimization
Exposing Device's I-RNTI	Lack of acknowledgment of rrc_release & paging retransmissions	Known C-RNTI [45], MitM [28, 45]	Y	Stealthy DoS

Table 1: Summary of 5GReasoner's findings.

7 RELATED WORK

In this section, we first discuss the existing efforts on formally analyzing cellular networks and other security protocols and then compare our findings with the known threats (with respect to security, privacy, and availability) on cellular networks.

Formal Verification of Cellular Networks. Two previous efforts [14, 22] that formally analyzed the 5G protocol focus only on the AKA part of the initial registration procedure. As opposed to formal verification, Kim et al. [35] design a stateless dynamic testing framework for 4G RRC and NAS layers. Our proposed framework, on the other hand, combines **MCheck** and **CPVerif** to formally analyze multiple stateful procedures spanning across 5G NAS and RRC layers using enhanced modeling abstractions.

Formal Verification of Other Security Protocols. Bhargavan et al. [25] and Fett et al. [26] develop formal analysis frameworks for TLS 1.3 [25] and OAuth [26]. These approaches, however, either use implementation specific modeling abstractions (e.g., for TLS 1.3 [25]) and proofs, or computational models [26]. It is, therefore, not clear how to apply them to cellular networks.

Linkabillity and Traceability Attacks. Previous work uncover the mapping/linking from TMSI to IMSI [11] and from C-RNTI to TMSI [31, 45] in 3G and 4G networks. Other studies [12, 33, 48] devise the mapping from a user's phone number to TMSI when the network operator does not change the TMSI frequently or randomly enough. Kohls et al. [36] exploit layer two information to launch website fingerprinting whereas Shaik et al. [47] demonstrate device fingerprinting using exposed device capabilities. Our *exposing device's TMSI, I-RNTI and paging occasion* attack, on the other hand, is different since it leverages cross-layer interactions.

IMSI Catching. IMSI catching attacks [9, 17, 23, 32, 41] have been an issue since 2G networks. Hussain et al. [29] recently demonstrate a brute-force IMSI-Cracking attack to retrieve a target UE's SUPI in 5G. Our *installing null cipher and null integrity* attack, on the other hand, does not employ any brute-force technique, but exploits the

unprotected `rrc_sec_mode_failure` to retrieve the *SUPI* when the UE is in limited service mode.

Main-in-the-Middle Relay. Rupprecht et al. [44] and Chlosta et al. [20] devise a MitM and an impersonation attack by exploiting implementation bugs in an LTE dongle and operational networks, respectively. Rupprecht et al. [45] also show how a MitM relay can be leveraged to manipulate the encrypted payload and redirect DNS traffic. Our *counter reset* attack has a MitM flavor, however, it is different since it exploits vulnerabilities in the processing of counter values used for generating/verifying the MAC.

Denial-of-Service. In [40, 42, 43, 46, 48], the authors explore ways to conduct DoS attacks against 3G and 4G subscribers. Kim et al. [35] show several new DoS attacks against both specific users and entire base stations exploiting the vulnerabilities of 4G networks. Our DoS attacks using `rrc_setup_request`, `rrc_resume_request`, `reg_request`, and `ue_dereg_request` messages, however, uncover a different class of vulnerabilities in the initial messages of both NAS and RRC layers of 5G protocol stack.

8 DISCUSSION AND LIMITATIONS

(1) Experimentation in testbed. We currently do not have access to any 5G commercial networks or cellular devices. In addition, there is no open-source 5G protocol stack which prevents us from testing our attacks in a testbed.

(2) Defenses. We do no provide any defenses since it is unclear how to add defenses without altering the protocol. Further investigation is needed for designing defenses and hence is left for future work.

(3) Threat to validity. The manually extracted FSMs from the 3GPP standard are our faithful interpretation; inaccuracies in which may induce false positives. Since commercial and open-source 5G testbed networks not available yet, we could not verify the attacks.

(4) Responsible disclosure. We reported our findings to GSMA through the CVD program [8] and are waiting for their response.

9 CONCLUSION AND FUTURE WORK

We presented the 5GReasoner framework which can formally reason about desired properties of the 5G control-plane-protocols possibly spanning across multiple layers of the stack. Due to the careful use of behavior-specific abstraction, our 5G model—covering 6 NAS and 5 RRC layer control-plane-protocols—is amenable to the highly automated analysis of 5GReasoner. Our evaluation of 5GReasoner with respect to desired properties obtained from the specification revealed 11 new 5G design weaknesses. We also observed that the 5G protocol model also inherits 5 design weaknesses from the 4G LTE protocol. We also show how to take advantage of our findings to devise exploitable attacks against the current version of 5G.

Future work. In the future, we will improve our 5G protocol model to include other key control-layer protocols. To make the 5G model amenable to automated reasoning, we may require to explore different new forms of abstraction which achieves a right balance between behavioral accuracy and analysis scalability.

ACKNOWLEDGEMENT

We thank the anonymous reviewers for their suggestions. This work is supported by NSF grants CNS-1657124 and CNS-1719369, Intel, and a grant by Purdue Research Foundation.

REFERENCES

[1] [n.d.]. *3GPP. Non-Access-Stratum (NAS) protocol for Evolved Packet System (EPS); Stage 3 Specification 3GPP TS 24.301 version 12.8.0 Release 12.*
[2] [n.d.]. *3GPP. Technical Specification Group Services and System Aspects; Study on the security aspects of the next generation system (3GPP TR 33.899 V1.3.0 Release 14).*
[3] [n.d.]. *3GPP. User Equipment (UE) procedures in idle mode and in RRC Inactive state (3GPP TS 38.304 version 15.1.0 Release 15.*
[4] [n.d.]. *5G; Non-Access-Stratum (NAS) protocol for 5G System (5GS); Stage 3 (3GPP TS 24.501 version 16.0.2 Release 15).*
[5] [n.d.]. *5G; NR; Radio Resource Control (RRC); Protocol specification (3GPP TS 38.331 version 15.5.1 Release 15).*
[6] [n.d.]. *5G; Security architecture and procedures for 5G System (3GPP TS 33.501 version 15.4.0 Release 15).*
[7] [n.d.]. *5G; User Equipment (UE) conformance specification; Part 1: Protocol (3GPP TS 38.523-1 version 15.3.0 Release 15).*
[8] [n.d.]. *GSMA Coordinated Vulnerability Disclosure Programme.* https://www.gsma.com/security/gsma-coordinated-vulnerability-disclosure-programme/.
[9] Dare Abodunrin, Yoan Miche, and Silke Holtmanns. 2015. Some dangers from 2g networks legacy support and a possible mitigation. In *2015 IEEE Conference on Communications and Network Security (CNS).* IEEE, 585–593.
[10] Iosif Androulidakis. 2011. Intercepting Mobile Phone Calls and Short Messages Using a GSM Tester. In *18th Conference on Computer Networks, Ustron, Poland*, Andrzej Kwiecień, Piotr Gaj, and Piotr Stera (Eds.). 281–288.
[11] Myrto Arapinis, Loretta Mancini, Eike Ritter, Mark Ryan, Nico Golde, Kevin Redon, and Ravishankar Borgaonkar. 2012. New privacy issues in mobile telephony: fix and verification. In *Proceedings of the 2012 ACM conference on Computer and communications security.* ACM, 205–216.
[12] Myrto Arapinis, Loretta Ilaria Mancini, Eike Ritter, and Mark Ryan. 2014. Privacy through Pseudonymity in Mobile Telephony Systems. In *NDSS.*
[13] Clark Barrett, Roberto Sebastiani, Sanjit Seshia, and Cesare Tinelli. 2009. Satisfiability Modulo Theories. In *Handbook of Satisfiability*, Armin Biere, Marijn J. H. Heule, Hans van Maaren, and Toby Walsh (Eds.). Vol. 185. IOS Press, Chapter 26, 825–885.
[14] David Basin, Jannik Dreier, Lucca Hirschi, Saša Radomirovic, Ralf Sasse, and Vincent Stettler. 2018. A Formal Analysis of 5G Authentication. In *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security (CCS '18).* ACM, New York, NY, USA, 1383–1396. https://doi.org/10.1145/3243734.3243846
[15] Bruno Blanchet. 2001. An Efficient Cryptographic Protocol Verifier Based on Prolog Rules. In *Proceedings of the 14th IEEE Workshop on Computer Security Foundations (CSFW '01).* IEEE Computer Society, 82–.
[16] Bruno Blanchet. 2009. Automatic Verification of Correspondences for Security Protocols. *Journal of Computer Security* 17, 4 (July 2009), 363–434.
[17] Ravishankar Borgaonkar and Swapnil Udar. 2014. Understanding imsi privacy. In *Vortrag auf der Konferenz Black Hat.*
[18] Roberto Cavada, Alessandro Cimatti, Michele Dorigatti, Alberto Griggio, Alessandro Mariotti, Andrea Micheli, Sergio Mover, Marco Roveri, and Stefano Tonetta. 2014. The nuXmv Symbolic Model Checker. In *CAV.* 334–342.
[19] Adrien Champion, Alain Mebsout, Christoph Sticksel, and Cesare Tinelli. 2016. The Kind 2 model checker. In *International Conference on Computer Aided Verification.* Springer, 510–517.
[20] Chlosta, Merlin, David Rupprecht, Thorsten Holz, and Christina Pöpper. 2019. LTE Security Disabled-Misconfiguration in Commercial Networks.
[21] Edmund Clarke, Orna Grumberg, Somesh Jha, Yuan Lu, and Helmut Veith. 2000. Counterexample-guided abstraction refinement. In *International Conference on Computer Aided Verification.* Springer, 154–169.
[22] Cas Cremers and Martin Dehnel-Wild. 2019. Component-Based Formal Analysis of 5G-AKA: Channel Assumptions and Session Confusion. (2019).
[23] Adrian Dabrowski, Nicola Pianta, Thomas Klepp, Martin Mulazzani, and Edgar Weippl. 2014. IMSI-catch Me if You Can: IMSI-catcher-catchers. In *Proceedings of the 30th Annual Computer Security Applications Conference (ACSAC '14).* 246–255.
[24] Danny Dolev and Andrew C. Yao. 1981. *On the Security of Public Key Protocols.* Technical Report.
[25] E.Rescorla. [n.d.]. The Transport Layer Security (TLS) Protocol Version 1.3. [Online]. Available: https://tools.ietf.org/pdf/draft-ietf-tls-tls13-28.pdf. *Network Working Group, Internet Engineering Task Force (IETF), RFC 8446* ([n. d.]).
[26] Daniel Fett, Ralf Küsters, and Guido Schmitz. 2016. A Comprehensive Formal Security Analysis of OAuth 2.0. In *Proceedings of the 23rd ACM SIGSAC Conference on Computer and Communications Security (CCS 2016).* ACM, 1204–1215.
[27] Nicholas Halbwachs, Paul Caspi, Pascal Raymond, and Daniel Pilaud. 1991. The synchronous data flow programming language LUSTRE. *Proc. IEEE* 79, 9 (1991), 1305–1320.
[28] Syed Rafiul Hussain, Omar Chowdhury, Shagufta Mehnaz, and Elisa Bertino. 2018. LTEInspector: A Systematic Approach for Adversarial Testing of 4G LTE. In *25th Annual Network and Distributed System Security Symposium, NDSS, San Diego, CA, USA, February 18-21.*

[29] Syed Rafiul Hussain, Mitziu Echeverria, Omar Chowdhury, Ninghui Li, and Elisa Bertino. 2019. Privacy Attacks to the 4G and 5G Cellular Paging Protocols Using Side Channel Information. In *26th Annual Network and Distributed System Security Symposium, NDSS, San Diego, CA, USA, February 24-27, 2019*.

[30] Syed Rafiul Hussain, Mitziu Echeverria, Ankush Singla, Omar Chowdhury, and Elisa Bertino. 2019. Insecure connection bootstrapping in cellular networks: the root of all evil. In *Proceedings of the 12th Conference on Security and Privacy in Wireless and Mobile Networks*. ACM, 1–11.

[31] Roger Piqueras Jover. 2016. LTE security, protocol exploits and location tracking experimentation with low-cost software radio. *CoRR* abs/1607.05171 (2016). arXiv:1607.05171 http://arxiv.org/abs/1607.05171

[32] Mohammed Shafiul Alam Khan and Chris J Mitchell. 2017. Trashing IMSI Catchers in Mobile Networks. In *Proceedings of the 10th ACM Conference on Security and Privacy in Wireless and Mobile Networks*. 207–218.

[33] Byengdo Kim, Sangwook Bae, and Yongdae Kim. 2018. GUTI Reallocation Demystified: Cellular Location Tracking with Changing Temporary Identifier. In *25th Annual Network and Distributed System Security Symposium, NDSS, San Diego, CA, USA, February 18-21*.

[34] Hongil Kim, Dongkwan Kim, Minhee Kwon, Hyungseok Han, Yeongjin Jang, Dongsu Han, Taesoo Kim, and Yongdae Kim. 2015. Breaking and Fixing VoLTE: Exploiting Hidden Data Channels and Mis-implementations. In *Proceedings of the 22Nd ACM SIGSAC Conference on Computer and Communications Security (CCS '15)*. ACM, New York, NY, USA, 328–339.

[35] Hongil Kim, Jiho Lee, Lee Eunkyu, and Yongdae Kim. 2019. Touching the Untouchables: Dynamic Security Analysis of the LTE Control Plane. In *Proceedings of the IEEE Symposium on Security & Privacy (SP)*. IEEE.

[36] Kathrian Kohls, David Rupprecht, Thorsten Holz, and Christina Pöpper. 2019. Lost Traffic Encryption: Fingerprinting LTE/4G Traffic on Layer Two.

[37] Denis Foo Kune, John Koelndorfer, and Yongdae Kim. 2012. Location Leaks on the GSM Air Interface. In *NDSS*.

[38] Kenneth L McMillan. 1993. The SMV system. In *Symbolic Model Checking*. Springer, 61–85.

[39] Ulrike Meyer and Susanne Wetzel. 2004. A Man-in-the-Middle Attack on UMTS. In *Proceedings of the 3rd ACM Workshop on Wireless Security (WiSe '04)*. ACM, 90–97.

[40] Benoit Michau and Christophe Devine. [n.d.]. How to not break LTE crypto.

[41] Shinjo Park, Altaf Shaik, Ravishankar Borgaonkar, Andrew Martin, and Jean-Pierre Seifert. 2017. White-Stingray: Evaluating IMSI Catchers Detection Applications. In *11th USENIX Workshop on Offensive Technologies (WOOT '17)*. USENIX Association, Vancouver, BC. https://www.usenix.org/conference/woot17/workshop-program/presentation/park

[42] Roger Piqueras Jover. 2013. Security Attacks Against the Availability of LTE Mobility Networks: Overview and Research Directions. In *Wireless Personal Multimedia Communications (WPMC), 2013 16th International Symposium on*.

[43] Muhammad Taqi Raza, Fatima Muhammad Anwar, and Songwu Lu. 2017. Exposing LTE Security Weaknesses at Protocol Inter-Layer, and Inter-Radio Interactions. In *International Conference on Security and Privacy in Communication Systems*. Springer, 312–338.

[44] David Rupprecht, Kai Jansen, and Christina Pöpper. 2016. Putting {LTE} Security Functions to the Test: A Framework to Evaluate Implementation Correctness. In *10th {USENIX} Workshop on Offensive Technologies ({WOOT} 16)*.

[45] David Rupprecht, Katharina Kohls, Thorsten Holz, and Christina Pöpper. 2019. Breaking LTE on Layer Two. In *IEEE Symposium on Security & Privacy (SP)*. IEEE.

[46] Altaf Shaik, Ravishankar Borgaonkar, Shinjo Park, and Jean-Pierre Seifert. 2018. On the Impact of Rogue Base Stations in 4G/LTE Self Organizing Networks. In *Proceedings of the 11th ACM Conference on Security & Privacy in Wireless and Mobile Networks (WiSec '18)*. ACM, New York, NY, USA, 75–86. https://doi.org/10.1145/3212480.3212497

[47] Altaf Shaik, Ravishankar Borgaonkar, Shinjo Park, and Jean-Pierre Seifert. 2019. New Vulnerabilities in 4G and 5G Cellular Access Network Protocols: Exposing Device Capabilities. In *Proceedings of the 12th Conference on Security and Privacy in Wireless and Mobile Networks (WiSec '19)*. ACM, New York, NY, USA, 221–231. https://doi.org/10.1145/3317549.3319728

[48] Altaf Shaik, Jean-Pierre Seifert, Ravishankar Borgaonkar, N. Asokan, and Valtteri Niemi. 2016. Practical Attacks Against Privacy and Availability in 4G/LTE Mobile Communication Systems. In *23nd Annual Network and Distributed System Security Symposium, NDSS, San Diego, CA, USA, February 21-24*.

[49] Guan-Hua Tu, Yuanjie Li, Chunyi Peng, Chi-Yu Li, Hongyi Wang, and Songwu Lu. 2014. Control-plane Protocol Interactions in Cellular Networks. In *Proceedings of the 2014 ACM Conference on SIGCOMM*. ACM, 223–234.

[50] Hojoon Yang, Sangwook Bae, Mincheol Son, Hongil Kim, Song Min Kim, and Yongdae Kim. 2019. Hiding in Plain Signal: Physical Signal Overshadowing Attack on LTE. In *28th USENIX Security Symposium (USENIX Security 19)*. USENIX Association, Santa Clara, CA, 55–72. https://www.usenix.org/conference/usenixsecurity19/presentation/yang-hojoon

A ACRONYMS

3GPP	Third generation partnership project
5G-RAN	5G Radio Access Network
5G-CN	5G Core Network
AKA	Authentication and Key Agreement
AMF	Access and Mobility Management Function
AUSF	Authentication Server Function
CEGAR	Counterexample-guided abstraction-refinement
CPVerif	Cryptographic Protocol Verifier
C-RNTI	Cell Radio Network Temporary Identifier
gNB	New radio Base station
GUTI	Globally Unique Temporary Identity
I-RNTI	Information Radio Network Temporary Identifier
IMSI	International Mobile Subscriber Identity
LTE	Long Term Evolution
MAC	Message Authentication Code
MCheck	Model Checker
NAS	Non Access Stratum
PDCP	Packet Data Convergence Protocol
RRC	Radio Resource Control
SUCI	Subscription Concealed Identifier
SUPI	Subscription Permanent Identifier
SIB	System Information Block
TMSI	Temporary Mobile Subscriber Identity
UDM	Unified Data Management
UE	User Equipment
USIM	Universal Subscriber Identity Module