From Control to Chaos: A Comprehensive Formal Analysis of 5G's Access Control

Mujtahid Akon
Pennsylvania State University
mba5773@psu.edu

Md Toufikuzzaman
Pennsylvania State University
mpt5763@psu.edu

Syed Rafiul Hussain
Pennsylvania State University
hussain1@psu.edu

***Abstract*—We develop CoreScan, a comprehensive formal analysis framework for analyzing the access control mechanism of 5G core networks. In doing so, we build the first comprehensive formal model for the access control mechanism of 5G core network that considers the indirect communication mode and 5G roaming. Given a global property, CoreScan employs the *compositional verification technique* that leverages the *assume-guarantee* style reasoning approach to decompose the system model into multiple disjoint components and applies the *split assertion* principle to identify local assumptions and guarantees. The model's global security property holds if and only if all local guarantees derived from the global property are verified in their respective components. CoreScan features a configurable adversary model, enabling the evaluation of access control properties under diverse adversary capabilities. We tested 61 access control properties with CoreScan and uncovered five new classes of exploitable privilege escalation vulnerabilities in the 5G standards. Additionally, we found that most previously known overprivilege vulnerabilities in direct communication also extend to indirect communication and roaming settings.**

1. Introduction

The fifth-generation cellular network (5G) presents the latest advancement in wireless technology. It is now being deployed by operators worldwide. To deliver on its promises, such as enhanced Mobile Broadband (eMBB) and Ultra-Reliable Low Latency Communication (URLLC), 5G has undergone significant transformations from its predecessors. A central component of this evolution is the 5G core network (5GC), which manages communication, connectivity, and data routing for 5G devices and services. Unlike previous generations that relied on dedicated, proprietary hardware for specific network services, the 5GC employs a service-based architecture (SBA). This architecture breaks the core into multiple functional components, known as Network Functions (NFs). The NFs are implemented as software-based services capable of running in virtualized or cloud environments. Additionally, 5G supports software-defined networking (SDN) and invites third-party businesses and enterprises to leverage and contribute to the 5GC. However, the integration of these new technologies, along with the involvement of diverse vendors and third-party tenants, makes the security of the 5GC more critical than ever.

A fundamental security challenge in the 5GC is the issue of over-privilege, which typically occurs when a malicious or malfunctioning NFs controlled by either service providers or third-party tenants gains unintended or illegitimate access to resources or services of other NFs. To address this, the 3rd Generation Partnership Project (3GPP)— the organization responsible for developing cellular network specifications— has adopted OAuth (Open Authorization) 2.0, a state-of-the-art access control framework, to authorize NF access. However, OAuth design has been proven to be insecure and susceptible to various attacks [1]. This signifies that the integration of OAuth in 5GC may also entail critical flaws. Hence, malicious or compromised NFs controlled by third-parties can exploit the flaws to obtain unauthorized access to sensitive data, modify it, or cause denial-of-service (DoS).

Since the introduction of access control mechanisms in 5GC, two prior efforts attempted to systematically analyze its security [2], [3]. These analyses, however, fall short because of the following reasons: (i) static program analysis performed on the open-source 5GC implementations to compute least privilege permissions is imprecise, and the use of incomplete open-source 5GC implementations as reference is error-prone [2]; (ii) formal analysis is incomplete due to not accounting for *indirect communication modes* and *5G roaming*, which are integral to 5GC operations and introduce unique access control challenges due to multi-hop interactions and cross-network trust boundaries. Specifically, multi-hop interactions introduced by the indirect communication mode involve data passing through intermediary nodes such as a proxy, also known as SCP in 5GC. All these expand the attack surface and increase the risk of unauthorized access, while cross-network trust boundaries in roaming scenarios require secure coordination between two different network operators, complicating access control enforcement; and (3) finally, the analysis cannot effectively manage state explosion problems. Therefore, in this paper, we pose the following research question: *Is it possible to develop a formal analysis framework to comprehensively verify the access control mechanism of a 5GC, including indirect communication mode and 5G roaming service?*

However, we run into the following challenges to address the above research question. First, the standard body [4] does not provide any 5G reference implementations. Ex-

isting open-source implementations [5], [6], [7] are incomplete, error-prone, and do not strictly follow the specification requirements, making them unsuitable for formal analysis. Most importantly, the attribute-based access control mechanism in the 5G core [8] involves over 50 types of NFs, hundreds of services, and thousands of APIs, resulting in a vast space of attributes to consider in each access control decision. Formally verifying security properties in a system of this size is, therefore, challenging due to the state explosion problem. Additionally, the 5G access control design is specified in natural language, which introduces ambiguity, lacks formal rigor, and complicates precise interpretation and formalization.

The most closely related work to our approach is 5GCVerif [3], which introduced the first formal model of 5GC's access control mechanism and its formal analysis. This framework formulates the verification of 5G access control mechanism as a model-checking problem. It leverages the *small model theorem* [9] to reduce the model's state space, effectively limiting it to a smaller, representative subset (e.g., reducing from 50 NFs to 5 NFs) without compromising accuracy. The model is then tested against various access control properties using a model-checker to uncover counterexamples indicative of over-privilege attacks. However, 5GCVerif has several limitations. (A) It only considers the *direct communication mode*, where an NF interacts directly with another without a proxy. With the introduction of 5G standards in Release 16, the indirect communication mode, which involves a proxy between two interacting NFs, becomes highly critical. This mode significantly alters 5G access control and warrants additional security analysis. (B) The framework overlooks 5G roaming services, a critical 5GC feature where NFs from different networks interact. Since these networks reside in separate trust domains, they may introduce security risks that 5GCVerif fails to address. (C) There are flaws in 5GCVerif's implementation of certain authorization logic, potentially causing it to miss some attacks. For example, we identified a novel attack (§7.1.1) that 5GCVerif failed to detect due to an oversight in verifying an attribute during a service grant. (D) Finally, 5GCVerif does not scale while capturing and analyzing both direct and indirect communication modes and roaming scenarios.

Our approach. To address the above limitations, we design `CoreScan`, which can be viewed as an alternative instantiation of the general 5GCVerif framework but employs different modeling and verification approaches to tackle additional challenges. To address the scalability challenge, we adopt a *compositional verification approach* based on *assume-guarantee reasoning* [10]. In this approach, the 5GC's access control system's model is logically divided into smaller *components*, each represented as a finite state machine (FSM). A component assumes a local property (*assumption*) and verifies another (*guarantee*). These components are connected such that each assumption corresponds to a guarantee of another, like in a chain, and the global property holds if all local guarantees derived from it are satisfied.

However, deriving meaningful local assumptions and guarantees from a global property can be computationally expensive and often requires manual intervention [11], [12]. We observe that the 5GC access control system consists of multiple functionally disjoint features, each defined by a unique set of attributes, with only a small set of common attributes (§3). We leverage this structure by modeling a bare-bone component using only the common attributes, requiring no assumptions. Additional components are built on top, each incorporating a disjoint feature-specific attribute set. Since these feature-specific components are disjoint and their local guarantees target only component-specific attributes, they can be independently verified, rendering their assumptions trivial (i.e., $true$). `CoreScan` consists of five such components—*nf-domain*, *slicing*, *roaming*, *indirect*, and *other*—in addition to the bare-bone one. Following the *split assertion* principle [13], global security properties are decomposed into local guarantees, each verified in its corresponding component, while the local assumptions are reduced to triviality. Thus, to verify a global security property in `CoreScan`, we test the corresponding local guarantees in their respective components. The global property holds if and only if all local guarantees are satisfied. Because components and their guarantees are disjoint, any local violation directly reflects a flaw in the overall system.

`CoreScan` adopts a modular design not only in terms of components but also across NF components, attributes, key API requests/responses, and communication subjects and objects. This modularity enhances model extensibility and customization. The threat model is highly flexible, allowing any communicating entity to be treated as an adversary with varying degrees of capability. Specifically, adversaries can inherit partial or full capabilities of consumer NFs, producer NFs, SCPs, or even the serving network during roaming, enabling the analysis of diverse security properties from multiple perspectives. To further improve scalability, we apply abstraction techniques across components, simplifying data types, attribute values, and behaviors. These abstractions significantly reduce complexity, allowing efficient modeling of the intricate interactions involved in 5G access control.

Findings. Using `CoreScan`, we tested 61 security properties across six model components, revealing 10 distinct types of access control attacks, including five newly discovered attacks. Additionally, we found that a 5G core utilizing indirect communication via SCP and/or roaming also suffers from most access control flaws identified in prior studies [3], [14]. Our experimental results show that `CoreScan` significantly improves both scalability and runtime. Compared to the baseline [3], `CoreScan` achieves a 9-27 times speedup for reachability property checks and uses 55% fewer states despite supporting more complex features such as indirect communication and roaming.

Contributions. In summary, this paper has the following main contributions:

- We introduce the first comprehensive formal model for the access control mechanism of 5G core network that considers *indirect communication mode* and *5G roaming*.
- We develop `CoreScan`, an *assume-guarantee* style compositional verification framework for formally analyzing the access control design of the 5G core network.

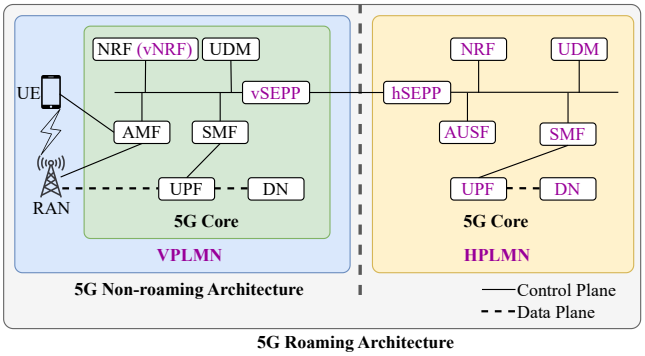


Figure 1: Simplified 5G system architecture. In non-roaming (blue), the UE connects to the 5G Core (green) via RAN. In roaming, the UE connects to the HPLMN (orange) through the VPLMN (blue). Roaming-specific elements are shown in purple text.

`CoreScan` operates on a formal model to rigorously verify a wide range of security properties. Leveraging the *split assertion* principle, it decomposes both the system model and security properties into independent *components*, enabling scalable and effective model-checking. A configurable adversary component further allows `CoreScan` to evaluate properties under diverse adversarial conditions.

- We tested 61 security properties with `CoreScan` and discovered five new classes of exploitable privilege escalation attacks in 5G standards. We also confirmed that most previously known overprivilege vulnerabilities also apply to indirect communication and 5G roaming. The model, properties, and findings are open source at [15].

Responsible disclosure. We shared our findings and proposed patches with GSMA [16] which agreed to acknowledge them under CVD-2025-0101. We are collaborating with GSMA to draft and submit multiple Change Requests (CRs) to 3GPP [4] for updating 5G security specifications.

2. Background

This section briefly describes the 5G access control mechanism and the necessary concepts to understand our model, methodology, and findings.

2.1. 5G & 5G Roaming Architecture

A basic 5G non-roaming architecture comprises three key components (blue region of Figure 1): User Equipment (UE), i.e., are end-user devices; Radio Access Network (RAN); and the 5G Core or 5GC (green region). RAN connects UEs to the core via radio links. To support diverse services, the 5GC adopts a Service-Based Architecture comprising a set of modular functional blocks, called network functions (NFs), each responsible for performing a specific set of tasks. 3GPP defines over 50 *types* of standard NFs (denoted by NFType) in the 5GC, such as the Access and Mobility Management Function (AMF), Session Management Function (SMF), and Unified Data Management (UDM). These NFs collectively manage mobility, session handling, and subscriber data. Depending on demand, multiple NF instances of the same NFType may run concurrently, each identified by a unique NFInstanceID. For simplicity, since we typically assume a single NF instance per NFType in our analysis, we use the terms: *NF*, and *NF instance* interchangeably.

5G roaming connects networks from different operators, enabling a seamless user experience across networks. Each network is identified by a unique Home Public Land Mobile Network (PLMN) ID. A typical roaming scenario (Figure 1) involves the Home PLMN (HPLMN) (orange region), to which the UE is subscribed, and the Visited PLMN (VPLMN) (green region), which the UE connects to while roaming. In this setup, the UE attaches to the RAN of the VPLMN, which provides local services and interacts with its own core network. To retrieve subscriber information and other required data, the VPLMN communicates with the HPLMN through secure inter-PLMN interfaces—typically via the Inter-PLMN Backbone (IPX)—using vSEPP (Visited Security Edge Protection Proxy) and hSEPP (Home Security Edge Protection Proxy) to ensure secure and authenticated inter-operator communication.

2.2. Network Isolation

Network slicing is a core feature of 5G, enabling the creation of multiple *virtual networks* on shared physical infrastructure. Each slice is an end-to-end network customized to specific service requirements, such as bandwidth, latency, and security. Identified by a unique ID called sNssai, a network slice spans resources across the RAN, transport network, and core network, ensuring logical separation between slices. This separation is key to prevent performance and security issues in one slice from impacting others.

2.3. 5G Access Control

To facilitate communication between NFs in the 5GC, each NF exposes a set of standard API functions, called *NFOperations*, that other NFs can invoke. NFOperations are grouped in *NFServices*, with each NFService consists of several NFOperations managing a specific set of resources. An NF can offer one or more NFServices. Each NFService and, optionally, NFOperation is assigned with a *permission scope* called *service-level* scope and *operation-level* scope, respectively. Scope refers to the specific set of permissions granted to an NF to access an NFOperation during NF communications. For example, AMF provides an NFService *namf_comm*, which contains NFOperations, providing communication services with a UE. Among these NFOperations, *CreateUEContext* allows the creation of UE context for signaling and communication. The AMF defines both the service-level *namf-comm* scope and the operation-level *namf-comm:ue-contexts:mobility* scope for this interaction.

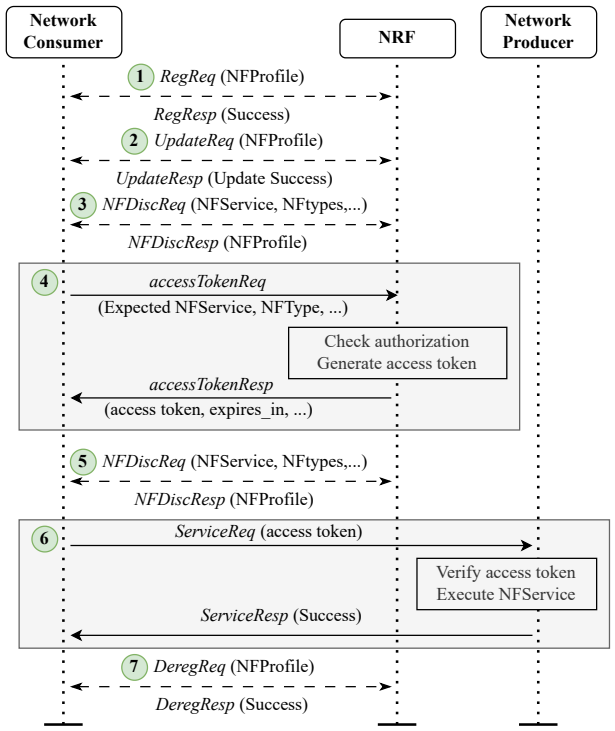


Figure 2: Access control interactions between NFs. Refer to Figure 5 for corresponding API usage in the 5GC access control model.

To limit unwanted resource access during NF interactions, 3GPP adopts the *Client Credentials* mode [17] of *OAuth 2.0* protocol. In this mode, to access any resources by invoking an API call, the *client* must present a valid *access token* (AT) to the *resources owner* that the owner will verify before granting resource access to the client. The client can obtain the AT from the *authorization server*, a trusted entity to both the client and the resource owner. In the 5GC, a special NF acts as the authorization server called Network Repository Function (NRF). The client NF invoking the API (i.e., NFOperation) is called *consumer NF* or $\mathcal{N}_C$ in short, whereas the resource owner is the NF that receives, verifies, and grants/rejects the NFOperation, also known as *producer NF* or simply $\mathcal{N}_P$.

Authorization APIs. A simplified diagram of how NFs interact each other before NFService access is shown in Figure 2. (i) *NF Register, Update and Deregister Request:* Each NF maintains a multi-level key-value pair data structure, called an NFProfile, which contains its attributes, e.g., NFInstanceID, NFType; states, e.g., NFStatus (nf registered or not); and authorization rules: allowedNfTypes, allowedNssais. Each NF instance is identified by its NFInstanceID attribute. To register a new NF instance to the 5GC, its NFProfile is sent to the configured NRF via *RegReq* API (1). Since NFProfile is dynamic, any changes to it are reported to NRF via *UpdateReq* API (2). This is crucial for 5G access control since NRF may use NFProfiles to authorize an *accessTokenReq* (4) and grant AT for the $\mathcal{N}_C$. To scale the network down, e.g., due to inactivity, an NF can also deactivate itself by requesting NRF to remove its NFProfile via *DeregReq* API (7). (ii) *AT Request (accessTokenReq or ATR):* A consumer NF ($\mathcal{N}_C$) invokes *accessTokenReq* (4) to ask for an AT from NRF. Based on the request parameter values and the NFProfiles of the prospective producer NFs ($\mathcal{N}_P$s), NRF verifies the request and, if authorized, grants an AT to the $\mathcal{N}_C$. AT contains important information regarding permissions, e.g., NFInstanceIDs of the allowed $\mathcal{N}_C$s & $\mathcal{N}_P$s, allowed scopes, etc., for the $\mathcal{N}_P$ to verify NFService Request (6). Note that AT is considered a resource for *accessTokenReq*. (iii) *NF Discovery Request (NFDiscReq or DR):* Any time the $\mathcal{N}_C$ wants some resource access or services from another NF, it first needs to identify the endpoints, (e.g., IP address) of the potential producer NFs in the 5GC that may serve its need. As NFs are dynamic, the consumer needs to query NRF to get this information. NRF exposes NF Discovery Request (*NFDiscReq*) API (3, 5) for this purpose. *NFDiscReq* can be invoked before or after the *accessTokenReq* as required. Both *accessTokenReq* and *NFDiscReq* require authorization checks via NRF and thus are called *authorization request*. Note that discovered NFProfiles are considered a resource for *NFDiscReq*. (iv) *Service Request (ServiceReq):* Once the consumer NF has both AT and the potential producer's endpoint, it finally invokes resource access or service request via this API (6). Upon receiving the request, $\mathcal{N}_P$ verifies the AT information against its NFProfile to determine if $\mathcal{N}_C$ is authorized for resource access and, if allowed, grants resource access.

Authorization verification process. Each NFProfile contains a special set of attributes that NRF uses to authorize the consumer's *authorization requests*. These attributes are called *authorization attributes* and can be defined at multiple levels and thus provide some flexibility in describing a fine-grained access control. For instance, *allowedNfTypes* is an authorization attribute in the NFProfile of the producer NF and refers to a list of NFs of certain NFTypes that are allowed to access the producer NF's resources. It may be defined globally for all NFServices in the producer's NFProfile, or separately for each NFService, or even for each NFOperation. If defined, NFOperation level parameter dominates over NFService level and so on. For example, if in its NFProfile, an AMF instance defines *allowedNfTypes* as {AMF, SMF} (NF-level), but one of its NFServices, *amf-comm*, defines it as {AMF} only (NFService level), then no NF instance of NFType except AMF, even SMF, can access any NFOperations under *amf-comm.* Some other notable authorization attributes are allowedNFInstances, allowedNssais, allowedNFDomains, and allowedPlmns. From a producer NF's point of view, they respectively define lists of NFTypes, sNssais, fqdns[1] and plmns of NF instances that may be considered for resource access. Resource access is granted only if all authorization attribute checks are pass.

1. a *Fully Qualified Domain Name (FQDN)* is a DNS-resolvable name used for service discovery and routing between NFs.

2.4. Communication Types

The 5GC supports three primary modes of communication [18]. In *direct communication* (Figure 2), NFs interact directly with each other, without a proxy, making it ideal for scenarios where low latency is crucial. However, this method may expose NFs to security risks and scalability issues. To address these concerns, the Service Communication Proxy (SCP) facilitates *indirect communication*, which intermediates between NFs, handling all requests and responses, i.e., *accessTokenReq*, *NFDiscReq*, and *ServiceReq* (❶-❺ in Figure 3 in § 3.2). This approach is expected to enhance security through policy enforcement, load balancing, and traffic monitoring while also improving scalability by decoupling NF interactions. Lastly, *hybrid communication* also involves SCP but combines both direct and indirect modes, allowing operators to balance performance and security based on specific network needs. In this mode, *NFDiscReq* always bypasses SCP for performance, while *ServiceReq* always routes through SCP for enhanced security. Depending on the mutual authentication between the consumer NF and NRF, *accessTokenReq* may or may not route through SCP in this mode (❻-❽ in Figure 3, *accessTokenReq* is routed through SCP assuming lack of mutual authentication between $\mathcal{N}_C$ and NRF), depending on the mutual authentication between NF and NRF at the transport layer [8]. Details about communication modes are available in Appendix A.

3. Motivating Example

In this section, we present our threat model and provide a motivating example that highlights the need for formal analysis of 5G access control. We then discuss the high-level workflow and insights underpinning the design of `CoreScan`.

3.1. Threat Model

Since the 5GC system facilitates third-party services and the NFs are deployed as cloud-based microservices, the 5GC access control system is vulnerable to a wide range of threats. For instance, external NFs operated by third-party tenants such as Mobile Virtual Network Operators or MVNOs [19], [20], or compromised NFs due to cloud misconfigurations [21], [22] may act as malicious NFs and illegitimately access unauthorized resources. While the ultimate goal of a malicious NF is often privilege escalation, our analysis considers distinct threat models based on communication types, as well as the capabilities and intentions of different NFs.

Adversary Capabilities Our threat model assumes that an attacker compromises an NF/SCP of the 5GC in some way discussed above and thus has full control over the NF/SCP. Consequently, it can forge and issue arbitrary messages to other legitimate NFs in the network. Also, a compromised NF, i.e., a consumer or producer NF, with the supervision of the Operations, Administration, and Management (OAM) System, may modify and/or misreport its NFProfile attributes to the NRF via *UpdateReq*. Besides, a rogue consumer NF may alter message parameter values arbitrarily while sending an authorization request. During indirect communication, a compromised SCP may impersonate legitimate NFs and alter, replay, or spoof requests/responses, tricking other components into trusting unauthorized access requests, leading to data leaks, unauthorized service access, or even manipulation of user data. Finally, while roaming, a visiting NRF, being part of a foreign network from the perspective of the home network and UE, may exhibit malicious intent, thereby altering authorization request parameters before forwarding them to arbitrary destination NRFs, i.e., home NRFs. The above adversary assumptions are inferred from prior security weaknesses observed in the cloud microservices [21], [22], [23], [24], [25], [26], [27] and are consistent with 3GPP TR 33.875 [28].

Other Assumptions. (1) We assume all single-hop network packets are mutually authenticated. Thus, our threat model does not question the mutual authenticity of the network packets as opposed to the Dolev-Yao [29] adversary model. (2) We assume the malicious NF does not drop, modify, intercept, or eavesdrop a request/response sent from a legitimate authenticated NF, as these capabilities do not affect authorization. (3) All tokens and certificates, e.g., AT, CCA, etc., are integrity-protected and cannot be modified by any party except the originator. (4) For 5G roaming, all intermediate nodes, such as SEPP and IPXs between the vPLMN and the hPLMN, are benign since the communicating nodes are mutually authenticated to each other, providing an end-to-end implicit authentication and dealing with encrypted and integrity-protected payloads only. (5) NRF (resp., hNRF for roaming) acting as an authorization server is not considered malicious because of the elevated trust of the operator in it.

3.2. An Example of Privilege Escalation Attack

Let us walk through an example to highlight the necessity of our enhanced model-checking approach. Consider a scenario in Figure 3 where an AMF instance in the 5GC wants to create a session management context with an SMF instance via indirect communication. The AMF instance sends a *ServiceReq* (*PostSmContexts*) with relevant parameters, which is routed through an SCP (❶). Upon receiving the request, the SCP, acting on behalf of the AMF, obtains an access token (AT_{SMF}) using *accessTokenReq* and response (❷), discovers the appropriate SMF instance using *NFDiscReq* from the NRF (❸), and then makes the *PostSmContexts* API call to the SMF (❹). Since the SCP is a proxy, the AMF includes a signed token called Client Credentials Assertion (CCA_{AMF}) token in the service request to SCP. NRF and/or a producer NF use the CCA token in indirect or hybrid communication mode to identify a legitimate request. The CCA_{AMF} contains ⟨AMF's NFInstanceID, an expiration time, and aud:'NRF' and 'PRODUCER_NF', since the CCA is for both the NRF and the expected producer⟩. The NRF or SMF verifies the CCA token's authenticity each time it

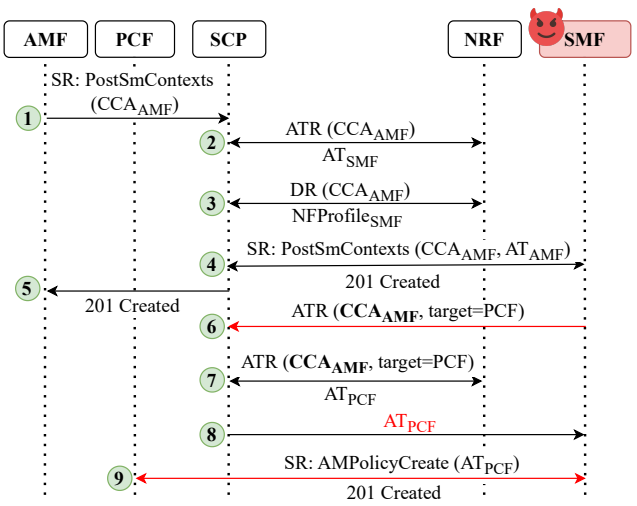


Figure 3: Message flow demonstrating the motivating attack example. Here, AMF is the consumer, and SMF in red acts as a malicious actor exploiting CCA_{AMF} via hybrid communication to obtain AT_{PCF}. NRF is the authorization server and SCP is the proxy used for hybrid communication, assuming mutual authentication between SMF and PCF is not established in transport layer [8]. Here, CCA_{AMF} = ⟨id: AMF_ID, aud: 'NRF', 'PRODUCER_NF'⟩ and AT_{PCF} = ⟨consumer: AMF_ID, producer: PCF⟩, DR: *NFDiscReq*, ATR: *accessTokenReq*, SR: *ServiceReq*.

receives a request, ensuring that SCP can only act on behalf of AMF.

At first glance, this process seems secure. However, we discovered a critical vulnerability after modeling and verifying the indirect communication mode of 5GC's access control specified by the 3GPP Release 17 [30]. Suppose the SMF acts maliciously. Upon receiving the *PostSmContexts* request with the CCA_{AMF} (4), it could exploit CCA_{AMF} by immediately sending a fabricated *accessTokenReq* to SCP using hybrid communication mode to obtain an access token AT_{PCF}:⟨consumer: AMF_ID, producer: PCF⟩ from the NRF (6–8). The goal of the malicious SMF is to illegitimately obtain some services from a PCF (Policy Control Function) instance such as forge/modify/delete the Access and Mobility Policy data (defining rules for network access, mobility management, and service prioritization), which the AMF has legitimate access to originally (Figure 3). SCP, being the proxy for SMF, forwards the request to the NRF along with CCA_{AMF} obtained from SMF (7). The NRF verifies the replayed CCA_{AMF} and, seeing it as valid, grants access token AT_{PCF}:⟨consumer: AMF_ID, producer: PCF⟩ to the SMF assuming that the request has come from the AMF (similar to 2). With AT_{PCF}, the SMF can then illegitimately forge AMF-related resources, e.g., Access and Mobility Policy data, and send them to PCF by invoking the *AMPolicyCreate* operation via direct communication (9).

The vulnerability arises because the CCA and SCP combined allow the SMF to misuse the granted AT, which has a sufficient lifetime, allowing the SMF to exploit it indefinitely. Besides, when SMF sends the malicious *ServiceReq* (9), it signs this message with its signature that contains SMF's NFInstanceID as the consumer. However, while verifying this request, PCF does not verify that SMF's NFInstanceID in the request signature does not match the consumer field in the AT_{PCF} (i.e, AMF's NFInstanceID). As evident from the above example, this sophisticated attack is difficult to detect manually, underscoring the need for systematic formal analysis. Previous work [3] failed to detect this type of vulnerability because it could not model and analyze indirect communications due to state explosion.

4. Design Overview

4.1. Design Philosophy of `CoreScan`

To tackle such scaling limitations, we design `CoreScan` based on compositional verification by leveraging the *assume-guarantee* style reasoning paradigm. As the 5G core can include numerous producer and consumer NFs, each with various services and operations, similar to 5GCVerif [3], at first, we employ the *small model theorem* to reduce the verification of an unbounded parametrized system to a verification of a small constant parameter value system. The small model theorem suggests that if all interactions in the large system are represented in the smaller system, any property valid for the smaller system will also be valid for the larger one. Based on that, we can project the set of reachable states in a large 5GC onto a reduced set of key producers and consumers while preserving critical system behaviors within this simplified model.

While the small model theorem effectively reduces the number of consumer and producer NFs in the model, it alone cannot address the additional complexities introduced in `CoreScan`. For instance, the small model theorem requires `CoreScan` to model two entities of NRFs: visiting (vNRF) and home (hNRF) NRFs of serving and home networks, respectively, for inter-network communications (e.g., Vodafone in the UK with T-Mobile in the USA), whereas 5GCVerif required none. Additionally, 2 out of 4 NF communication modes [18] require the modeling of an SCP component. `CoreScan` also aims for a comprehensive model by incorporating additional security-sensitive parameters (e.g., fqdn, plmn), further increasing its complexity.

We address this problem via *compositional verification technique* [31]. Here, we disintegrate the model into several small *components* and verify *local properties* of individual components to imply that these properties hold in the entire system. This technique leverages the *assume-guarantee* reasoning proposed by Pnueli [10]. Let a system consists of two *components* M_1 and M_2. Similar to Hoare triple, Pnueli's system uses $\langle A \rangle M \langle G \rangle$ form, meaning that if a component M is part of a system that satisfies property A (*assumption*), then the system must satisfy property G (*guarantee*). The simplest form of compositional reasoning based on the

transitivity principle can be expressed as follows:

$$\frac{\langle true\rangle M_0\langle A\rangle \quad \langle A\rangle M_1\langle G\rangle}{\langle true\rangle M_0||M_1\langle G\rangle}$$

The above rule states that if $\langle true\rangle M_0\langle A\rangle$ and $\langle A\rangle M_1\langle G\rangle$ hold, then $\langle true\rangle M_0||M_1\langle G\rangle$ must hold. Here $M_0||M_1$ denotes the composite state space of M_0 and M_1.

Let, the set of model components be $\mathcal{M} = \{M_0, M_1, \ldots, M_n\}$, the set of all assumptions, $\mathcal{A} = \{A_0, A_1, \ldots, A_n\}$, and the set of all guarantees $\mathcal{G} = \{G_0, G_1, ..., G_n\}$. To verify if a model M of the system satisfies a desired property G, i.e., $\langle true\rangle M\langle G\rangle$, compositional verification asks to satisfy a series of proof: $\langle A_i\rangle M_i\langle G_i\rangle$ for all $i = 0, 1, \ldots, n$. However, a critical challenge is to devise compositional rules to find the appropriate set of triplets $\{(A_i, M_i, G_i) \mid i = 0, 1, \ldots, n\}$ such that the *local assumption* A_i satisfy the *local guarantee* G_i where $G_i = A_{i+1}$ for all i. Here, $G = G_n$ denotes the *global* property to be satisfied in the system.

To identify the appropriate set of triplets and the local assumptions, we leverage the following observation. The authorization verification for a service request in the 5GC system depends on multiple authorization attributes, and the checks on different authorization attributes are independent. This observation enables us to employ the principle of *split assertion* [13] to identify appropriate triplets and the local assumptions. A split invariant is a conjunctively *separable assertion*, which is a global inductive invariant and is a Boolean combination of local invariants. It has the form $G_1(V_1) \wedge G_2(V_2) \wedge \ldots G_n(V_n)$, one for each component $M_1, M_2, \ldots M_n$. On the other hand, a local invariant to a component M_k is defined only over the variables V_k that belong to M_k.

Employing split invariant rules to compositional reasoning enables us to decompose a global property of the 5GC access control system into several smaller local properties in such a way that each local property targets only a small part of the whole system that is independent of the rest. With this approach, we can divide the large model of the 5GC access control system and each security requirement (i.e., a global property) into n smaller and disjoint components and n local properties so that each local property depends on only one smaller model component. Each decomposed model corresponds to distinct authorization attributes and is significantly smaller than the combined model of the system consisting of other disjoint features (see §4.2.2). This significantly reduces the state space and makes it manageable for the state-of-the-art model checkers to verify the local properties of the smaller model components in a reasonable time. Indeed, this approach enables us to detect the attack example in Section 3.2 by modeling only the small component associated with the indirect communication.

4.2. High-level Overview of `CoreScan`

4.2.1. 5GC Model Decomposition. We observe that the 5GC access control behaviors can be systematically decomposed into several unique functionalities called *features*, each defined by a distinct set of NF attributes. A feature functionally represents a high-level operational capability or service offered by the network, while its attributes define feature identity, configure authorization policies, and manage related resources. For instance, the *(network) slicing* feature corresponds to a logical network in 5GC and includes feature-specific attributes:{sNssais, allowedNssais,...} to configure and authorize slice-specific access. Similarly, the *roaming* feature includes feature-specific attributes:{plmns, allowedPlmns,...}. Note that the feature-specific attributes are disjoint from one another.

Some attributes, e.g., NFType and allowedNfTypes are, however, fundamental and required by default for all valid NF instances, regardless of features. These shared attributes form the *base feature*. Consequently, *roaming* and *slicing*, both features, must include *base* feature attributes: {NFType, allowedNfTypes, ...}, apart from their corresponding feature-specific attributes.

Formally, let the *base* feature be $F_0 = \{c_1, c_2, ..., \}$ where each c_i is a core attribute of F_0. Let, F_1 (e.g., *slicing*) and F_2 (e.g., *roaming*) be two additional features with attribute sets: $F_1 = \{a_1, a_2, ...\}$, and $F_2 = \{b_1, b_2, ...\}$. Since F_1 and F_2 are distinct (i.e, $F_1 \neq F_2$) and both include the *base* feature F_0, the following holds: $(F_1 - F_0) \cap (F_2 - F_0) = \emptyset$, or equivalently, $F_1 \cap F_2 = F_0$. For any feature F_i where $i \neq 0$, we call attributes in $F_i - F_0$ *feature-specific attributes* and those in F_0 *base or common attributes*. For simplicity, we refer to feature-specific attributes as *feature attributes* or attributes in short, unless stated otherwise.

The key insight is that if a 5GC system is decomposed into features, and each feature's authorization logic is verified independently, then the overall access control logic of the system is also correctly verified. This is supported by two observations: (1) except for the *base* feature, other features are optional (e.g., private 5G networks may lack *slicing* or *roaming*); and (2) feature-specific attributes are disjoint, so cross-feature authorization logic does not arise. For example, a consumer NF's sNssais is checked against allowedNssais (both under *slicing*), while plmns is checked against allowedPlmns (both under *roaming*).

Leveraging this insight, we model the 5GC system by decomposing it into features, each represented as a separate model *component* instantiated as FSMs. This modular approach transforms the large, complex state space of the monolithic access control system into several smaller, feature-specific components, each with significantly reduced state space— an essential step toward mitigating state explosion. Further details about modeling components are provided in §5.

4.2.2. 5GC Property Decomposition. Given the decomposed components of the monolithic 5G access control system, if a global security property can be partitioned into local properties aligned with individual components, then, by the split-assertion principle (§4.1), verifying each local property within its respective component suffices to establish the global property for the system.

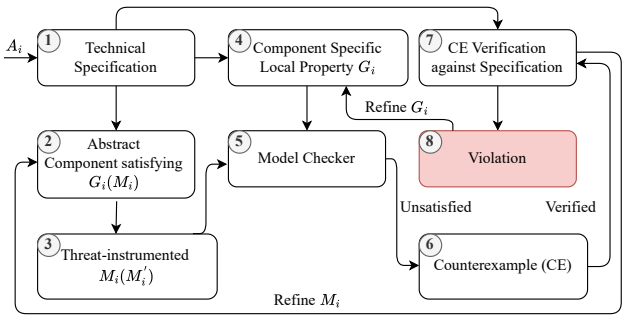


Figure 4: `CoreScan` **workflow for an iteration** i

A global property applies to the entire 5GC system and spans multiple components, whereas a local property is restricted to a single component. For example, consider the high-level global property: *A consumer NF can access services/resources only if it is authorized to do so* [3], where services/resources refer to response packets such as the NFProfile (in response to *NFDiscReq*), an AT (in response to *accessTokenReq*), or a service response (in response to *ServiceReq*). A more concrete global property φ states: *If an AT for a producer NF is granted to a consumer NF in a accessTokenReq, then the consumer's NFProfile attributes must match the authorization attributes in the producer NF's NFProfile*. This property is global because it leaves unspecified which specific attributes should be checked, implying that the monolithic system must verify all relevant attributes (e.g., sNssais, plmns) from all available features (e.g., *slicing*, *roaming*).

Deriving a local property from a global one is a *specialization problem*: the global property is refined by adding component-specific elements. For example, in the *slicing* component, the local property φ^{slicing} specifies that *the consumer's sNssais must be authorized by the producer's allowedNssais*. Specifically, local property φ^{slicing} is: *if an AT for a producer NF is granted to a consumer NF in a accessTokenReq, then the consumer's sNssais attribute must match the allowedNssais attribute in the producer NF's NFProfile*. Similarly, for *roaming*, the local property φ^{roaming} requires that *the consumer's plmns appear in the producer's allowedPlmns*.

4.2.3. `CoreScan` **Workflow.** Leveraging the model and property decomposition techniques, at first, we construct the bare-bone component M_0, instantiating the *base* feature (§4.2.1) and verify local guarantee property G_0 against it (§4.1). M_0 is independent of any particular assumption; thus $A_0 = true$. The rest of the components are constructed on top of this assuming $\langle true \rangle M_0 \langle G_0 \rangle$ valid, i.e., for all $i, j = 0, 1, 2..., n-1$ and $i \neq j$, $M_i \cap M_j = M_0$.

If $\langle true \rangle M_0 \langle G_0 \rangle$ is valid, we can construct M_1 by extending M_0 with a new feature and verify G_1 against M_1 targeting only the component-specific authorization attributes. Thus, $\langle A_1 \rangle M_1 \langle G_1 \rangle$ being valid implies M_1 satisfies both local properties G_0 and G_1, i.e., $G_0 \wedge G_1$. Similarly, M_2, being an extension of M_0 and satisfying $\langle A_2 \rangle M_2 \langle G_2 \rangle$ implies that M_2 satisfies both G_0 and G_2. Moreover, since $M_1 \cap M_2 = M_0$, i.e., all attributes between $M_1 - M_0$ and $M_2 - M_0$ are disjoint, by construction, M_2 also satisfies G_1 provided M_1 satisfies G_1. Thus, we get M_2 satisfies $G_0 \wedge G_1 \wedge G_2$. In this manner, we can show that if we can construct a component $M_i \in \mathcal{M} - M_0$ as an extension of M_0 and verify $G_i \in \mathcal{G}$ against $M_i \in \mathcal{M}$, then the composite state space $M = M_0 || M_1 || \ldots || M_n$ will satisfy the target property $G = G_0 \wedge G_1 \wedge G_2 \wedge \cdots \wedge G_n$.

Since G_i targets only the component-specific authorization attributes in M_i, and G_0 is tested against M_0, we can ignore all assumptions A_i since assumptions only flow from M_0 to M_i. Global property G will be satisfied if only if for all i, G_i satisfies M_i individually. The above insight hints at the following approach for systematically analyzing 5G access control as presented in `CoreScan`: for each feature F_i, `CoreScan` constructs a model component M_i. Incorporating the threat model gives a threat-instrumented model M_i'. For ease of exposition, in the rest of the paper, we use M_i and M_i' interchangeably to denote the threat-instrumented model. Based on the model, we construct a local security property G_i targeting only the component-specific attributes and verify it against M_i. If M_i satisfies G_i, the analysis terminates. Figure 4 presents the system architecture for an iteration of `CoreScan`.

Model Checking Process. We use nuXmv [32], a state-of-the-art model checker, against individual components for the property verification. The model checking process leverages the Counter-Example Guided Abstraction Verification (CEGAR) principle [33] to verify the properties systematically. Here, a component M_i is tested against a local property G_i to find if it satisfies M_i, i.e., $M_i \models G_i$. If it is true, it means that the property is verified in the concrete system under verification. Consequently, we proceed to verify the next component M_{i+1} with the corresponding local property G_{i+1}. However, if $M_i \not\models G_i$, then the model checker gives a counterexample. Now, we have two considerations. If the counterexample also exists in the concrete system, then we find a security violation and report it. Otherwise, it is a spurious counterexample (false positive) because of the over-approximation due to abstraction applied during model construction. So, we refine M_i to eliminate the spurious counterexample. This process continues until the local property is satisfied in the component. Additionally, in case we encounter a counterexample that is indeed a violation, to explore further violations, we refine G_i to suppress the counterexample by adding/modifying constraints on model attributes. The above approach is repeated until all model components are tested.

Workflow Summary. (i) Following the 5G standard, we construct six components of the system, equivalent to six FSMs, instead of modeling the 5GC access control system as a whole (② in Figure 4). (ii) Then we instrument our threat model in each component (③). (iii) After that, we take a high-level global access control property and disintegrate it to obtain the local property for each disjoint model component (④). (iv) Now, we pass to verify the local property

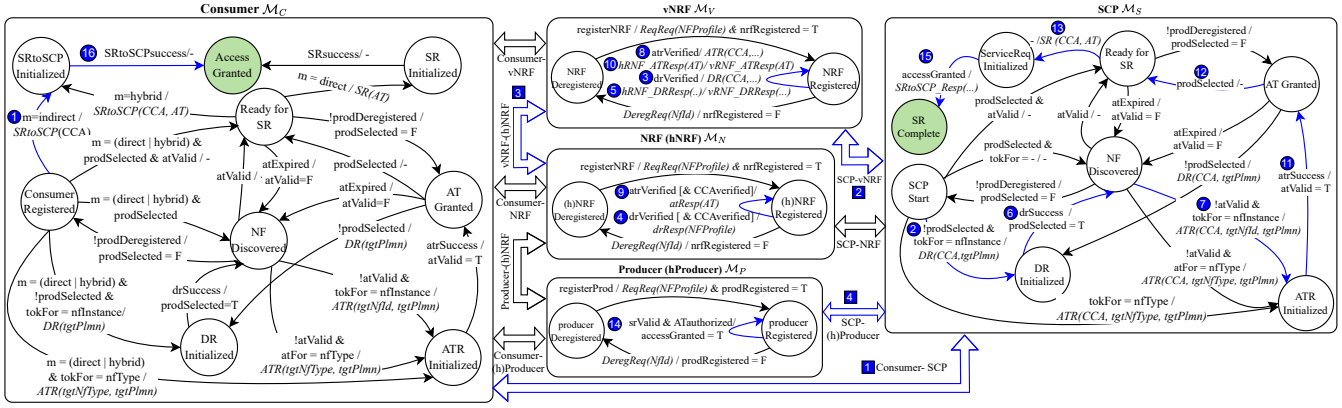


Figure 5: Simplified Abstract model $\mathcal{M}$ of the *comprehensive* 5GC access control mechanism combining all features. The visiting and the home network producer instances are co-located for simplicity. Similarly, hNRF and NRF (non-roaming) are co-located. Blue-colored transitions and interfaces depict an authorization flow involving both indirect communication and roaming.).

in the local component via a model checker (⑤). If the property is satisfied, we relax the property to explore other paths in the model. Otherwise, we follow the steps from §4.2.3 to find each attack(⑥-⑧).

5. Decomposed Model Construction

5.1. Abstract model

The abstract model of the 5GC can be represented as a set of FSMs interacting with each other. An FSM is represented as a 5-tuple $\{S, I, O, S_0, F\}$, where S denotes a finite set of states, I is a finite set of inputs, O is a finite set of outputs, S_0 is the set of initial states, and F represents the finite set of state transitions that define the system's transition relation.

Figure 5 illustrates the comprehensive abstract model $\mathcal{M}$ combining all features of the 5GC access control design as several interacting FSMs representing different 5GC entities communicating via respective common interfaces. The interacting FSMs include: consumer ($\mathcal{M}_C$), producer ($\mathcal{M}_P$), NRF/hNRF ($\mathcal{M}_N$), SCP ($\mathcal{M}_S$) and vNRF ($\mathcal{M}_V$). Blue-colored transitions and interfaces depict an authorization flow example involving both indirect communication and roaming, and involve all five FSMs. In this example, a consumer starts from *Consumer Registered* state by sending a *ServiceReq* to SCP with CCA token attached (①) through interface 1. SCP starts from *SCP Start* state, constructs and sends *NFDiscReq* to vNRF (②) via interface 2. and reaches *DR Initialized* state. vNRF at *NRF Registered* state, verifies the packet and forwards it to the correct hNRF (③) via interface 3. hNRF verifies the request along with the CCA token, and if successful, responds with discovered NFProfiles to SCP via path: ④-3-⑤-2. SCP selects a producer from the discovered ones and reaches *NF Discovered* state. From here, SCP issues *accessTokenReq* to vNRF and receives an AT to reach *AT Granted* state via the path: ⑦-2-⑧-3-⑨-3-⑩-2-⑪. Now SCP reaches state *Ready for SR* (⑫) and initiates *ServiceReq* to hProducer and, if hProducer approves, is granted service access to reach state *ST Complete* via path: ⑬-4-⑭-4-⑮. Then, SCP forwards the obtained resource to the consumer via interface 1. The consumer finally reaches *access granted* state upon receiving the resource (⑯).

Although the 5GC abstract model presented in Figure 5 is highly simplified, it is evident that all aspects of 5GC access control design are difficult to model and formally verify due to their complexities and large state space.

5.2. Features to Components

To manage the state space efficiently, the concept of features is introduced in §4.2.1. Instantiation of a feature involves constructing an *abstract model component* (*component* in short) consisting of only the respective feature attributes from the 5GC access control design. Thus, a component is a subset of the comprehensive model $\mathcal{M}$. Each component is constructed by carefully going through the technical documents [8], [18], [34], [35], [36] specified in 3GPP Release 18 [30]. By following the model decomposition principle (§4.2.1), we first start by constructing the *bare-bone* component M_0, instantiating the *base* feature: {NFType, allowedNfTypes,...}. It does not include network entities such as SCP or vNRF or any attributes related to network slices (e.g., sNssais, allowedNssais) or inter-PLMN communications (e.g., plmns, allowedPlmns). Specifically, M_0 contains a pair of $\mathcal{N}_C$ and $\mathcal{N}_P$ instances, an NRF instance, all important access control packets (Figure 2) such as *UpdateReq*, *accessTokenReq*, and *ServiceReq*, and finally, the corresponding response packets, i.e., discovered NFProfile and constructed AT. Component M_0 is entitled to verify a local property G_0 (§4.1).

The following components are constructed on top of M_0. For instance, M_1 is constructed by adding attributes related

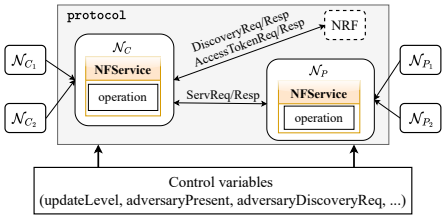


Figure 6: Schematic of the bare-bone component M_0. The gray region encapsulates the `protocol` module, while the dotted rectangle denotes the abstracted NRF instance with its NF-Profile not modeled.

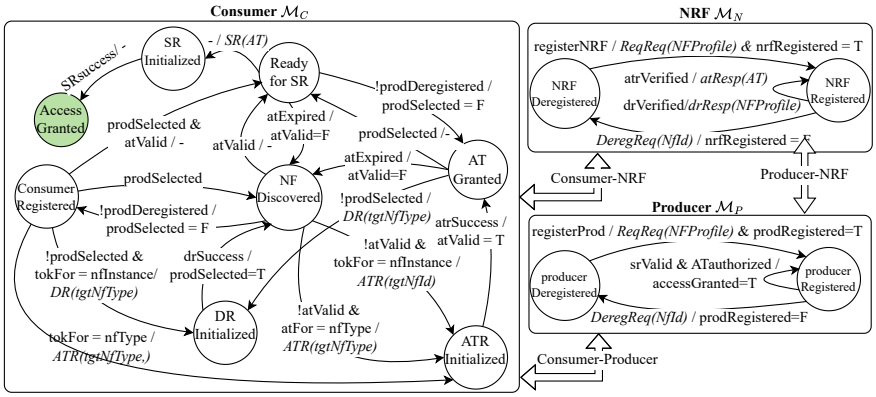


Figure 7: Simplified abstract model of the bare-bone component M_0.

to feature *nf-domain*:{fqdns, allowedNFDomains,...}. M_1 is entitled to test the local property G_1. Similarly, M_2, M_3, M_4, and M_5 are built upon M_0 by adding features and authorization attributes corresponding to network *slicing*: {sNssais, allowedNssais,...}, *roaming*: {plmns, allowed-Plmns,...}, *indirect (communications)*: {CCA,...} and finally *other* consisting of miscellaneous attributes of the NFProfile: {load, priority,...}, respectively. $M_2 - M_5$ are entitled to verify target local properties $G_2 - G_5$, respectively.

5.3. Construction of Bare-bone Component (M_0)

We present the bare-bone model component M_0 to illustrate the construction of each component. M_0 simulates a simplified 5GC as a set of communicating FSMs (Figure 6) and includes five entities: two consumer instances: $\mathcal{N}_{C_1}$ and $\mathcal{N}_{C_2}$, two producer instances: $\mathcal{N}_{P_1}$ and $\mathcal{N}_{P_2}$, and a single NRF instance. Each consumer and producer FSM maintains an NFProfile via state variables (e.g., NFInstanceID, NFType) and needs to register it to NRF to enable NF intercommunication in the network via *RegReq* or *UpdateReq* (not shown in Figure 6). A registered consumer NF initiates network interaction through a pair of authorization requests, i.e., *NFDiscReq* and *accessTokenReq*, before requesting services or resources via *ServiceReq*. The NRF, modeled as a trusted singleton, verifies authorization requests based on the NFProfiles of consumers and expected producers. Since its own NFProfile does not directly participate in authorization checks, its NFProfile is abstracted away and, thus, not modeled. Instead, we capture its essential behaviors using dedicated model variables, e.g., *isNFDiscovered* which encodes the authorization logic for NF Discovery and stores NRF's authorization decision upon receiving an *NFDiscReq*.

We design a parameterized module called `protocol` (gray region in Figure 6) to model access control-related protocol interactions between a consumer-producer pair. This module includes a generic consumer ($\mathcal{N}_C$) and a producer ($\mathcal{N}_P$), one instance of each request/response type between them, and models variables capturing NRF's authorization logic. We represent the protocol module with synchronous communicating FSMs. Each FSM, defined using state variables, transitions, and authorization logic, represents the access control-related stateful functionalities of the protocol participants. We separately model each API request and response in the `protocol` module by capturing only the relevant authorization-related HTTP header and body attributes as state variables. For instance, out of 119 attributes in the *NFDiscReq* packet, we model only five authorization-relevant ones in M_0: {*requesterNFType*, *serviceName*, ...}. The corresponding *NFDiscResp* is modeled as an NFProfile representing the discovered producer. Other requests, like *accessTokenReq* and *ServiceReq*, are modeled similarly. Figure 7 depicts the abstract model of the `protocol` module.

This modular and parameterized design choice also significantly reduces model complexity while providing the flexibility to instantiate multiple `protocol` runs with different consumer–producer pairs, each capable of running different authorization requests. For instance, at runtime, any consumer instance (e.g., $\mathcal{N}_{C_1}$ or $\mathcal{N}_{C_2}$) in M_0 can act as $\mathcal{N}_C$ by populating its NFProfile attribute-values. Similarly, one of the two producer instances ($\mathcal{N}_{P_1}$, $\mathcal{N}_{P_2}$) can be selected as $\mathcal{N}_P$. In one protocol run, one can instantiate the `protocol` module with $\mathcal{N}_C = \mathcal{N}_{C_1}$ and $\mathcal{N}_P = \mathcal{N}_{P_2}$ and can model $\mathcal{N}_{C_1}$ issuing an *ServiceReq* to access resources/services from $\mathcal{N}_{P_2}$. In another run, the module can be instantiated with $\mathcal{N}_C = \mathcal{N}_{C_2}$ and $\mathcal{N}_P = \mathcal{N}_{P_1}$, modeling $\mathcal{N}_{C_2}$ issuing an *NFDiscReq* to access resources/services from $\mathcal{N}_{P_1}$.

Finally, M_0 includes a construct called `modelParam` to capture configurable features and corresponding control logic. For example, the 3GPP specification [8] allows operators to reject authorization requests missing optional API attributes with the *requester-* prefix. We model this behavior using a non-deterministic, optionally configurable boolean variable, *requesterInfoReq*, defined under `modelParam`.

5.4. Construction of M_1-M_5

Components M_1 and M_2 are structurally similar to M_0 and do not require modeling additional FSMs, however, require modeling component-specific attributes as discussed in §5.2. M_3 includes an additional FSM, vNRF, to model the behaviors of visiting network's NRF. Finally, to incorporate

indirect and hybrid communication modes, another FSM called SCP is added in M_4. To simplify the modeling and overcome state explosion, M_4 includes a special *ServiceReq* message specific to SCP to supply important attributes so that SCP can use those to form valid authorization requests on behalf of the consumer NF. Some important parameters include CCA token, important NFProfile attributes for authorization requests, etc.

For components M_0, M_1, M_2, M_3, and M_5, we consider a threat model where a single registered consumer NF or a producer NF, compromised by a malicious actor, sends fabricated network packets (*NFDiscReq*, *accessTokenReq*, etc.) to seek unauthorized access to the services or resources provided by the benign entities in 5GC (benign NFs, NRFs). Additionally, M_3 considers that the vNRFs, being an entity in a foreign network, can also be malicious and thus can exhibit malicious intent. Finally, for component M_4, our threat model considers SCP to be malicious and can create and alter network packets before forwarding to arbitrary destinations in search of unauthorized access. We incorporate the above adversary capabilities into the respective M_i to obtain the threat-instrumented component M_i'.

6. Implementation

We use nuXmv [32] model checker to model and verify access control properties for `CoreScan`. nuXmv is an infinite-state symbolic model checker used for the formal verification of finite- and infinite-state systems. It facilitates modeling through an intuitive and high-level language SMV (Symbolic Model Verifier). In this language, the system under consideration can be described using modules that define states, transitions, and variables. The modeling process involves specifying the system's behavior and its various components, including their interactions and constraints. Property checking in nuXmv is performed by specifying formal properties using temporal logic such as LTL (Linear Temporal Logic). Once the system model and properties are defined, nuXmv uses symbolic techniques to explore all possible states and transitions within the model. It checks whether the specified properties hold true in all possible scenarios or if there are counterexamples where the properties might be violated.

7. Evaluation

With `CoreScan`, we modeled the access control mechanism specified by 3GPP Release 18 [30] and analyzed it with 61 properties. We first present `CoreScan`'s effectiveness in detecting new and prior attacks. We also discuss the nuances and our observations while constructing models alongside the recommendations to make 5GC access control more robust and complete. We then demonstrate the efficiency and performance of `CoreScan` with respect to the time `CoreScan` takes to verify a property and compare that with that of the prior work 5GCVerif. We use a machine with Intel i5-7200U CPU and 8GB DDR3 RAM for evaluation.

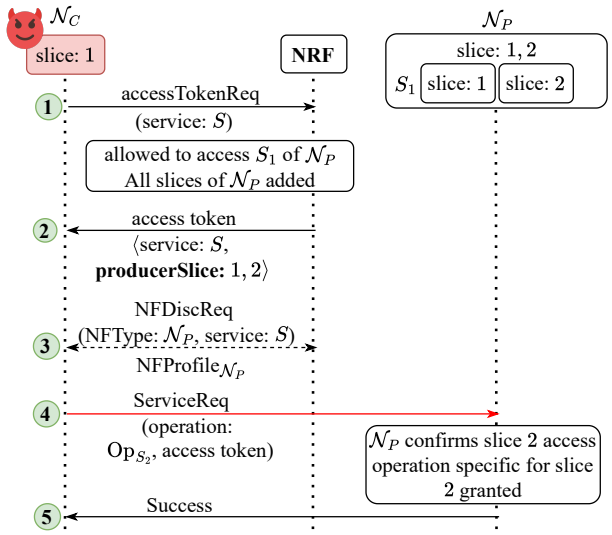


Figure 8: Message flow for *Coarse Scope Attack*

7.1. New Attacks

`CoreScan` uncovered five new access control attacks. The implications of these attacks range from privilege escalation and sensitive information exposure to denial-of-service. In what follows, we discuss the uncovered vulnerabilities, attack details, and their impacts.

7.1.1. *Coarse Scope Attack* (A1). This attack exploits `producerSlice`, an optional attribute of the access token, and affects all communication modes and 5G roaming.
Assumption. In this attack, a consumer NF ($\mathcal{N}_C$) acts as an adversary and thus can forge/modify and send any network packets on behalf of the consumer. We also assume that the adversary NF has authorized access to at least one network slice of the victim producer NF ($\mathcal{N}_P$) but not others.
Vulnerability. The root cause of this attack lies in the incorrect handling of the `producerSlice` attribute in the access token issued by the NRF after a successful *accessTokenReq* authorization. Specifically, instead of including only the authorized slices for the requested NFService, the NRF mistakenly adds *all* slices of the producer NF to the `producerSlice` attribute. Consequently, a malicious consumer NF can use this access token to access NFService operations across all slices served by the producer NF, beyond those to which the malicious NF is authorized to. Additionally, there are issues during *ServiceReq* validation by the producer NF. First, the `producerSlice` attribute is optional and may not be present in the access token. Even if it is included, some producer NFs may not support this attribute, making the attack trivial in such cases [8], [35]. In other cases, during *ServiceReq* validation, the producer NF cannot verify whether the NRF has incorrectly included all slices, leading it to approve service requests for unauthorized slices from the consumer NF.
Attack Description. Figure 8 illustrates *Coarse Scope Attack* using a simplified direct communication model. Consider a scenario where the adversarial consumer NF ($\mathcal{N}_C$)

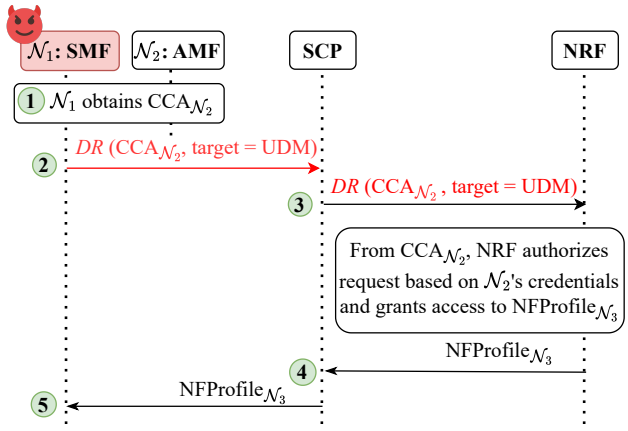


Figure 9: Message flow for *NFProfile Leakage Attack*

is part of slice 1 and is authorized to access NFService S of a producer NF ($\mathcal{N}_P$). The NFService S is divided into two logical slices, S_1 (serving slice 1) and S_2 (serving slice 2). Thus, $\mathcal{N}_C$ is only permitted to access S_1 and not S_2. However, `CoreScan` identifies a counterexample where the consumer NF gains unauthorized access to S_2 via *ServiceReq*. The attack proceeds as follows: The malicious consumer NF invokes *accessTokenReq* to request an access token for NFService S from the NRF (1). The NRF authorizes this request because S_1 serves slice 1, which $\mathcal{N}_C$ is allowed to access. However, while generating the access token, the NRF incorrectly includes all slices of $\mathcal{N}_P$, i.e., both slice 1 and slice 2, as per the 5G specification TS 33.501, which states that the `producerSlice` attribute should be the superset of all network slices served by the producer NF's NFServices. With the access token in hand (2), the adversarial consumer retrieves the endpoints (IP addresses) of the producer NF by invoking *NFDiscReq* (3). The consumer NF then uses the access token to invoke an API operation (Op_{S_2}) under NFService S_2 via *ServiceReq*, requesting a service or resource from slice 2 (4). The producer NF, upon validating the access token, finds that the `producerSlice` attribute includes slice 2. Consequently, it mistakenly grants access, allowing the consumer NF to obtain the unauthorized service (5).

Impact. With this attack a malicious consumer NF can gain unauthorized access to NFService operations across multiple slices, bypassing slice-specific access controls. This exploit compromises the isolation between network slices, potentially leading to data leakage, service disruption, and unauthorized resource usage. Although the attack is demonstrated above for direct communication, `CoreScan` discovered the same attack for indirect communication, hybrid, and 5G roaming.

7.1.2. *NFProfile Leakage Attack* (A2). In this attack, a malicious NF exploits a vulnerability in the CCA token-based hybrid communication mode (§3.2) to extract NFProfiles of any NF instances that the legitimate NF, which signed the CCA token, has access to.

Assumptions. Beyond the assumptions of *Coarse Scope Attack*, this attack additionally requires that the 5GC network supports hybrid communication via SCP, and that a CCA token is used for authentication and authorization.

Vulnerability. This attack is enabled by two key issues: (1) During indirect or hybrid communication, a benign NF leaks its CCA token to the adversarial serving NF without ensuring replay protection. The malicious NF can then use this token to impersonate the legitimate consumer NF and send authorization requests to the NRF via SCP. Since SCP signed and forwarded the request, the NRF cannot detect that an adversary replayed the token. Although the 5G specification states that the CCA token should be short-lived, it does not mention potential lifespans. Moreover, time-based replay protection is not reliable, and due to network reliability issues, it is safe to assume that the malicious consumer NF will have sufficient time to send at least one authorization request using the CCA, which is sufficient to exploit the attack. Other than a 'short' lifespan, 3GPP has not provided any other mechanism for replay protection for CCA tokens. (2) SCP does not verify whether the CCA token was originally signed by the sender NF, allowing the malicious NF to exploit this lack of validation.

Attack Description. Consider a scenario (Figure 9) where a benign NF instance $\mathcal{N}_2$ of NFType:AMF has authorized access to services from two NF instances: $\mathcal{N}_1$ of NFType:SMF and $\mathcal{N}_3$ of NFType:UDM (not shown in Figure 9). When $\mathcal{N}_2$ makes a service request to $\mathcal{N}_1$ via SCP (in either hybrid or indirect communication mode), it reveals its CCA token: ⟨id: $\mathcal{N}_1$, aud: NRF, SMF ⟩ to $\mathcal{N}_1$, which acts as the adversary (1). The adversarial NF $\mathcal{N}_1$ then crafts a malicious *NFDiscReq* impersonating $\mathcal{N}_2$ to search for the NFProfiles of UDM instances accessible to $\mathcal{N}_2$ (e.g., $\mathcal{N}_3$). It forges *NFDiscReq* by setting the target NFType to UDM, attaches $\mathcal{N}_2$'s CCA token, and sends it via SCP (2). SCP repackages, signs, and forwards the request to NRF (3). Now, NRF, mistakenly identifying the request as originating from $\mathcal{N}_2$, authorizes it based on $\mathcal{N}_2$'s access rights and retrieves the NFProfile of $\mathcal{N}_3$. It then sends this sensitive NFProfile information to SCP, which forwards it to $\mathcal{N}_1$ (4-5), thereby leaking sensitive information to the adversary under the guise of legitimate access.

Impact. The impact of this attack is that a malicious NF can gain unauthorized access to sensitive NFProfiles, potentially exposing critical configuration and service information of other NF instances. This leakage compromises network security, potentially enabling further exploitation and unauthorized service usage.

7.1.3. *CCA Replay Attack* (A3). This attack shares similarities with the demo attack illustrated in Figure 3 and discussed in § 3.2. We initially discovered the demo attack in 2023 while analyzing Release-17 [37]. In 2024, we switched to Release-18 [30] and found that the vulnerability was patched in Release-18 [8] because- (1) The producer NF now verifies, in the direct communication, that the consumer's NFInstanceID in the access token matches the NFInstanceID in the request signature, preventing unau-

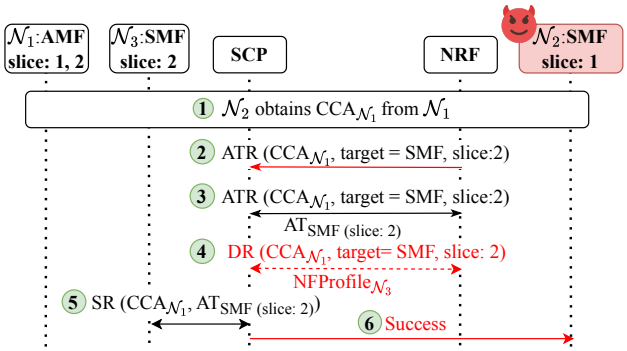


Figure 10: Message flow for *CCA Replay Attack*

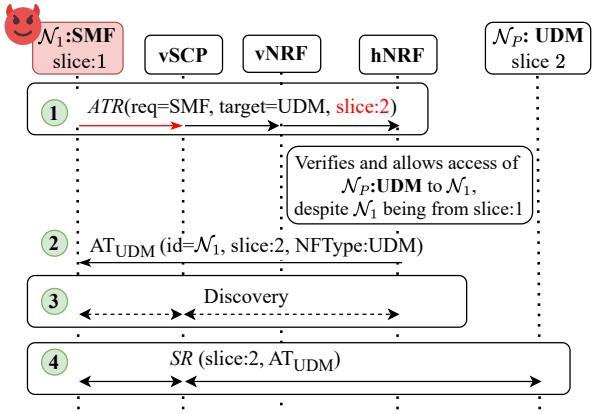


Figure 11: Message flow for *CCA Evasion Attack*

thorized service requests from malicious NFs (⑨ in Figure 3). (2) CCA token now includes the expected producer's NFType instead of the generic 'PRODUCER_NF' string. This ensures that the demo attack fails in indirect or hybrid communication due to the NFType mismatch between the access token (e.g., PCF) and the CCA token (e.g., SMF). However, our 2024 analysis reveals that the demo attack persists due to a new flaw, leading to the discovery of *CCA Replay Attack*, which remains exploitable despite the Release-18 updates.

Assumption. Beyond the assumptions in *Coarse Scope Attack*, this attack assumes that a malicious NF, using hybrid or indirect communication, can receive a CCA token from a benign NF that it provides services to.

Vulnerability. This attack exploits the similar vulnerabilities discussed in *NFProfile Leakage Attack* that uses *NFDiscReq* API for hybrid or indirect communication to illegitimately obtain NFProfiles. However, this attack exploits *accessTokenReq* in indirect communication to obtain illegitimate service access potentially from an unauthorized network slice. Despite the security enhancements in Release-18 for CCA and access tokens to mitigate the demo attack in direct communication, a vulnerability persists when a malicious NF makes an indirect or hybrid service request to another NF instance of the same NFType but serves a different network slice.

Attack Description. Consider a simplified scenario in Figure 10, where the 5GC has an AMF instance $\mathcal{N}_1$ operating in both slice 1 and slice 2, along with two SMF instances: $\mathcal{N}_2$ (slice 1) and $\mathcal{N}_3$ (slice 2). Suppose that the malicious instance $\mathcal{N}_2$ intends to exploit this vulnerability. It waits for a service request from $\mathcal{N}_1$ via indirect communication. Upon receiving the CCA token CCA$_{\mathcal{N}_1}$: ⟨id: $\mathcal{N}_1$, aud: NRF, SMF⟩ from $\mathcal{N}_1$ (①), $\mathcal{N}_2$ requests an access token from SCP for accessing a service in slice 2 (②). The SCP forwards this request to NRF, which, due to the CCA token's presence, assumes the request is from $\mathcal{N}_1$ with legitimate access to slice 2 and issues the access token AT$_{\text{SMF}}$ ⟨sub: $\mathcal{N}_1$, aud: SMF⟩ (③).Next, if necessary, the SCP may perform a NFProfile search via *NFDiscReq* (④). Finally, the SCP initiates the service request using *ServiceReq* via indirect communication, attaching CCA$_{\mathcal{N}_1}$ and access token AT$_{\text{SMF}}$. The malicious instance $\mathcal{N}_2$ validates both tokens and completes the unauthorized service request (⑤-⑥). In this way, a malicious SMF instance $\mathcal{N}_2$ of slice 1 achieves illegitimate access to resources of SMF instance $\mathcal{N}_3$ of slice 2.

Impact. NF interactions between NFs of the same type are critical, especially, for 5G handover procedure in the network side during mobility. The attack enables a malicious NF to gain unauthorized access to services across different network slices, disrupting the isolation between slices in 5G networks.

7.1.4. *CCA Evasion Attack* (A4). In this attack, a malicious consumer NF exploits vulnerabilities in certain NFs to gain unauthorized resource or service access through forged service requests in hybrid or indirect communication scenarios, particularly during roaming.

Assumption. In addition to the assumptions in §7.1.1, this attack assumes the use of SCP in inter-PLMN, i.e., internetwork communication, where a consumer NF in the visiting network acts as an adversary. The attack is even possible in some non-roaming cases as well.

Vulnerability. The attack leverages the absence of CCA token validation during 5G roaming. In roaming scenarios, the producer NF in the home network (HPLMN) cannot verify the CCA token received from the visiting network, as 5G standards rely on implicit hop-by-hop authentication, e.g., consumer NF to vSCP (SCP of the visiting network, VPLMN), vSCP to vNRF (NRF of the VPLMN), vNRF to hNRF (NRF of the HPLMN) [8]. However, this method fails to prevent NF impersonation. The attack can also occur within the same PLMN if the producer NF or NRF opts not to use CCA tokens for indirect or hybrid communication, as CCA token-based security is optional and may be replaced by implicit methods based on the operator's policy.

Attack Description. Consider a scenario where a malicious consumer NF $\mathcal{N}_1$ of NFType:SMF and serves slice 1 only, in the visiting network, aims to gain unauthorized access to a UDM instance $\mathcal{N}_P$ serving slice 2, which does not authorize $\mathcal{N}_1$ (Figure 11). To exploit this, $\mathcal{N}_1$ sends a forged *accessTokenReq* by misreporting its slice information (slice 2 instead of slice 1) via the `requesterSlice`

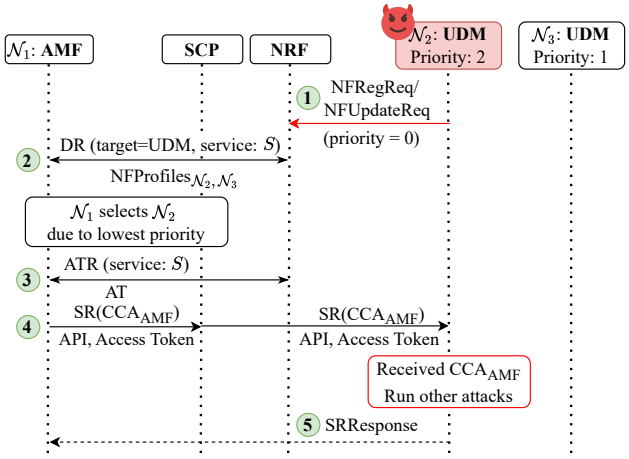


Figure 12: Message flow for *Forced NF Selection Attack*

parameter. The request is forwarded from the vSCP to the vNRF, which cannot validate the authenticity of the request (1). The vNRF then forwards it to the hNRF, which mistakenly assumes that $\mathcal{N}_1$ is part of slice 2, and issues an access token: $\langle$id:$\mathcal{N}_1$, producerSlice:slice 2, target: UDM$\rangle$, granting access to any UDM instances in slice 2 (e.g., $\mathcal{N}_P$) (2). Using this token, $\mathcal{N}_1$ can then retrieve the endpoint information of $\mathcal{N}_P$ (3). Finally, with the access token and endpoint address, the malicious NF makes a *ServiceReq* through SCP, gaining unauthorized access to services and resources in $\mathcal{N}_P$'s slice 2 in the same way as 1.

Impact. A malicious NF can impersonate any consumer NF to gain unauthorized access to services and resources across the network, bypassing proper authentication checks. This breach undermines network security, enabling unauthorized service usage and data exposure.

7.1.5. *Forced NF Selection Attack* (A5). This attack exploits a vulnerability in the 5G access control that opens the door for other overprivileged attacks. Through this, a malicious NF can misreport seemingly security-insensitive metadata of the NFProfile to force a benign consumer NF to select the malicious NF as a producer NF.

Assumption. Besides the assumptions of *Coarse Scope Attack*, this attack assumes that a malicious NF is capable of updating its metadata in the NFProfile, such as priority, load, etc.

Vulnerability. The vulnerability lies in how an NF/SCP selects a producer NFs from the list of NFProfiles obtained from NRF via *NFDiscReq*. According to the 5G standard [35], a consumer NF/SCP uses attributes such as load, and priority (with 0 meaning the highest priority) to select a suitable NF. Thus, a malicious NF can misreport these NFProfile attributes to force other NFs to NRF so that other NFs select the malicious NF as a producer more frequently.

Attack Description. Consider a simplified scenario (Figure 12) where the 5GC contains two UDM instances: $\mathcal{N}_2$ and $\mathcal{N}_3$ with priority 2 and 1 respectively, i.e., $\mathcal{N}_2$ has lower priority than $\mathcal{N}_3$. However, the malicious $\mathcal{N}_2$ misreport this seemingly security-insensitive attribute of the NFProfile to NRF via *UpdateReq/RegReq* (1). Now, A consumer NF instance $\mathcal{N}_1$ looking for some services from UDM searches for available UDM instances via *NFDiscReq* (2) and discovers both $\mathcal{N}_2$ and $\mathcal{N}_3$. However, based on the priority and balance (not shown in this example), it naively selects $\mathcal{N}_2$. In the following communications, it obtains an access token from NRF via *accessTokenReq* (3) and makes service requests via *ServiceReq* to $\mathcal{N}_2$ (4-5), thus $\mathcal{N}_2$'s goal is achieved as it has successfully forced $\mathcal{N}_1$ to send request to it and deprives $\mathcal{N}_3$ which with lack of request will get removed automatically by the network to reduce unnecessary load in the cloud.

Impact. DoS attack is a direct consequence of the vulnerability. However, This vulnerability can exploited by the malicious NF in several ways. For example, If the consumer NF uses a hybrid or indirect communication mode to invoke the service request to the malicious producer NF (4-5), then it may leak its CCA token to the producer in a similar way to *CCA Replay Attack*. However, it solves a problem of *NFProfile Leakage Attack*or *CCA Replay Attack*that these attacks are short-lived because of the short life span of the CCA token. Exploiting this vulnerability, those attacks can be performed repeatedly by a malicious NF.

7.2. Attack Validation

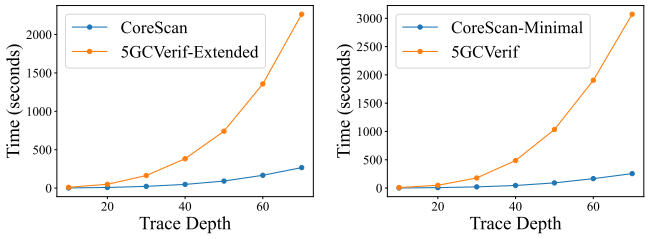
Since commercial 5GC networks are closed-source, we validated our findings using open-source implementations [5], [6], [7]. Among these, only *free5GC* partially implements 5G access control but lacks support for indirect communication and 5G roaming. Thus, only *Coarse Scope Attack* could be effectively validated in free5GC. We found that the root cause of the attack in free5GC is the omission of the optional *producerSlice* attribute in the access token, making exploitation trivial.

For further validation, we submitted `CoreScan` findings to GSMA along with feasible patches (appendix B). As of April 2025, GSMA has agreed to submit change requests (CRs) for attacks A1–A3. For A5, GSMA considers the attack relevant but an implementation issue. So, no CR is required other than notifying GSMA members. A4 remains under discussion. Further updates will be on Github [15].

7.3. Existing Attacks

Apart from the above new attacks, `CoreScan` also confirms some existing attacks found in previous works [3]. (1) *Confused Producer Attack*: Here, a malicious consumer NF exploits an attribute describing allowed producer's NFType in a legitimate access token to gain unauthorized access to a different instance of the same NFType in another network slice, leading to potential overprivileged access and sensitive data leakage. However, we found that this attack is no longer possible in Release-18 since 3GPP patched the attack by modifying the authorization checks. (2) *Token Reuse Attack:* Here, a malicious consumer NF uses an unexpired access

token to access a producer NF's services even after its permissions are revoked, exploiting the lack of real-time validation of the token's revocation status. All communication modes and 5G roaming suffer from this attack. (3) *Default Overprivilege Attack*: Here, a malicious NFC exploits the 'allow by default' rule in 3GPP's specification, gaining unauthorized access to network slices by removing its serving *slice* attribute temporarily from its NFProfile. This attack is also applicable in all communication modes and roaming. (4) *Authorization Bypass Attack*: Here, a malicious NFC exploits the lack of cross-NFProfile check between the consumer and the producer NF, allowing it to gain sensitive information from an unauthorized NF. We found that this attack's possibility depends on the operator's policy since some operator may implement it while others may not. This attack can be carried out across all communication modes, including roaming scenarios.



(a) **(b)**

Figure 13: Performance comparison of (a) `CoreScan` and 5GCVerif-Extended, (b) `CoreScan`-Minimal and 5GCVerif.

7.4. Performance Analysis

Our model consists of six independent components that can verify properties concurrently and thus enhance scalability. We benchmarked our approach against 5GCVerif [3], which supports a limited subset of these components. To ensure a fair comparison, we extended `CoreScan` to cover all components analyzed in our study. Experiments were conducted seven times, varying the trace depth from 10 to 70 in increments of 10.

As shown in Figure 14, the runtime of the six components of `CoreScan` increases with trace depth; however, at each trace depth, the runtime across different components remains similar, with minimal variance observed across repetitions. Since the components execute in parallel, the overall runtime of `CoreScan` is determined by the maximum runtime among them.

Figure 13(a) compares between `CoreScan` and the extended version of `CoreScan`, demonstrating that `CoreScan` achieves significantly lower runtimes, with an average performance improvement of over sevenfold. For the original 5GCVerif, which supports only two components (AC-nftype and AC-snssai), we compare it with the maximum runtime of these specific components in `CoreScan`, referred to as `CoreScan`-Minimal. Figure 13(b) confirms that `CoreScan` delivers an average performance gain of more than ninefold compared to the original 5GCVerif. These results highlight the superior scalability and efficiency of `CoreScan` compared to both standard and extended versions of 5GCVerif.

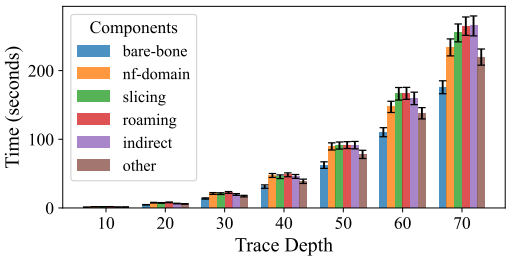


Figure 14: Performance analysis of CoreScan component

TABLE 1: Modeling and analysis efforts in terms of number of states, transitions, and lines of code

Model/Components	#State variables	#states	LoC
bare-bone	179	367	2320
nf-domain	225	548	2518
slicing	243	494	2771
roaming	282	634	2821
indirect	206	424	2517
other	212	437	2670
CoreScan	282	634	15617
5GCVerif [3]	471	1397	5150
Avg	**224.5**	**484**	**2602.83**

7.5. Modeling and Analysis Effort

Table 1 presents the number of state variables and states for each `CoreScan` component, as well as for the full `CoreScan` model and the baseline `5GCVerif` [3]. We implemented `CoreScan` in the SMV language [32], with a total of 15,617 lines of code. Since all `CoreScan` components operate in parallel, the overall number of state variables and states correspond to that of the largest individual component. Among the six components, *roaming* is the most complex, with 282 state variables and 634 states, primarily due to the inclusion of an additional NRF instance (vNRF) and associated forwarding network packets and authorization logic. On average, each `CoreScan` component is modeled with approximately 225 state variables, 484 states, and ∼2603 lines of code. Despite being more feature-rich and having a codebase nearly three times larger than `5GCVerif`, `CoreScan` requires 40% fewer state variables and 55% fewer states, highlighting its scalability and efficiency.

The entire modeling process took approximately 13 months, including understanding 5G specifications, designing abstraction techniques, and encoding the models and properties. While 3GPP releases minor updates (e.g., Release-17.3.0 to 17.4.0) roughly every three months, these incremental changes can typically be integrated into `CoreScan` within two weeks. In contrast, major releases (e.g., Release-17 to Release-18) may require 2–3 months to accommodate new features and structural changes.

Verification of the 61 security properties took a total of ∼32 minutes (excluding time for counterexample analysis). On average, CoreScan took 47 seconds to verify each property up to depth 40, but only ∼0.2 seconds when it identifies a counterexample .

8. Related Work

Access control analysis of 5GC. Ensuring robust 5G access control has become essential due to the rapid and extensive deployment of the 5G-based applications [38]. Prior works have formalized existing access control approaches [39], [40], and have introduced novel access control frameworks and enhancements to address emerging security demands [14], [41], [42]. In addition, some research has explored quantum-resistant access control mechanisms [43], too. To analyze existing access control mechanisms, Akon et al. [3] proposed a formal model-checking approach using the small model theorem to identify flaws in the 5G core's access control mechanism. Thorn et al. [2] designed a program analysis framework, performing static analysis on open-source implementations of the 5G access control, and conducted submodular analyses using the least privileged principle to identify potential over-privileges. However, the open-source implementation they considered does not represent reference implementation utilizing OAuth 2.0. In addition, none of these approaches cover indirect and hybrid communication and roaming scenarios, which have been prevalent from release 16 of the 5G standard.

Generic access control analysis. Extensive research has been conducted on analyzing access control, particularly in operating systems [44], [45] and the ARBAC model [46]. Static analysis has also been applied for access control analysis [47], [48], [49]. Gouglidis et al. [50] modeled Google's RBAC (Role-Based Access Control) Identity and Access Management (IAM) based on publicly available information by constructing a temporal logic-based transition system, which was then used to verify specific user-defined properties. Shen et al. [51] proposed a framework that automatically adjusts configurations to resolve access-deny issues by exploring configuration changes that enable minimal necessary permissions. Machine learning (ML) approaches are also being applied to enhance access control, as discussed by Nakamura et al. [52]. In addition, Natural Language Processing (NLP) techniques have been used to extract as well as generate access control policies [53], [54], [55], [56]. In the context of 5G, OAuth 2.0 [17] serves as the reference for access control. Several studies have previously examined OAuth implementations and identified security vulnerabilities for specific use cases like microservice and web applications [57], [58], [59], [60], [61], [62]. Following Akon et al. [3], our work focuses on analyzing OAuth 2.0 within the context of 5GC standards.

9. Discussion

Analysis Accuracy. We leverage CEGAR framework (§4.2.3) to iteratively refine the model and eliminate false positives (FP), i.e., spurious counterexamples. After refinement, we did not observe any spurious counterexamples (FP = 0), yielding a 100% precision (TP / (TP + FP) = 1 where TP denotes true positives). Recall measures how many *actual violations* have been successfully detected, i.e., TP/(TP+FN) where FN denotes false negatives. However, since, being an unknown quantity, the total number of actual violations is undecidable, recall is not measurable. We have carefully constructed our formal models based on a comprehensive examination of the 5G specifications. Similar to previous model-checking works [3], [40], we follow the conventional approach of aiming for soundness instead of completeness, i.e, if our approach reports a violation, it is indeed a violation; we cannot, however, detect all violations.

Analysis Scope. CoreScan's primary focus is to analyze the *design* of the 5G access control mechanism. However, it may not detect issues stemming from NF misconfigurations, implementation bugs, or other runtime vulnerabilities beyond its analysis capabilities. Additionally, our analysis is limited by the 3GPP 5G standards, i.e., if an operator enforces stricter policies beyond 5G standards, our findings may not apply. However, CoreScan's modeling discipline allows operators to mark certain NFs as benign per the operator's policy, making analysis possible for such cases.

CoreScan covers both intra- and inter-PLMN access control, enabling our analysis to apply to both Local Breakout (LBO) and Home Routing (HR) scenarios. In LBO, user traffic is routed locally within the Visited PLMN (vUPF to vDN), while in HR, traffic is tunneled back to the Home PLMN (vUPF to hUPF to hDN). The key distinction lies in Data Network (DN) access [18]: in LBO, vUPF–vDN is intra-PLMN, whereas in HR, vUPF–hUPF is inter-PLMN and hUPF–hDN is intra-PLMN.

Attack Domain. The scope of attacks identified by CoreScan is constrained by the specific security properties tested within the model. Therefore, we do not claim that the set of discovered attacks is exhaustive.

Defenses. We deliberately exclude detailed countermeasures from the main body since adding fixes without carefully accounting for other system factors (e.g., backward compatibility) can result in fragile or ineffective solutions. Moreover, some defenses (§7.1.4) require major design overhauls and are beyond the scope of this work. Instead, we are collaborating with the GSMA CVD panel to formulate long-term recommendations for specification updates. Appendix B outlines potential short-term mitigations.

10. Conclusion and Future Work

We have developed CoreScan, the first comprehensive formal analysis framework for analyzing the access control mechanism of 5GC. CoreScan employs *compositional verification technique* that leverages the *assume-guarantee* style reasoning approach to decompose the system model into disjoint components and uses the split assertion principle identify local assumptions and guarantees. With CoreScan, we tested 61 access control properties and uncovered five new classes of exploitable privilege escalation vulnerabilities

in the 5G standards. In the future, we will develop mitigation techniques for the uncovered issues.

Acknowledgments

We appreciate the anonymous reviewers and the shepherd for valuable feedback and GSMA's support during the vulnerability disclosure process. This work has been supported by the NSF under grants 2145631 and 2215017, the Defense Advanced Research Projects Agency (DARPA) under contract number D22AP00148, and the NSF and Office of the Under Secretary of Defense—Research and Engineering, ITE 2326898 and 2515378, as part of the NSF Convergence Accelerator Track G: Securely Operating Through 5G Infrastructure Program.

References

[1] D. Fett, R. Küsters, and G. Schmitz, "A comprehensive formal security analysis of OAuth 2.0," in *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, 2016, pp. 1204–1215.

[2] S. Thorn, K. V. English, K. R. Butler, and W. Enck, "5gac-analyzer: Identifying over-privilege between 5g core network functions," in *Proceedings of the 17th ACM Conference on Security and Privacy in Wireless and Mobile Networks*, 2024, pp. 66–77.

[3] M. Akon, T. Yang, Y. Dong, and S. R. Hussain, "Formal Analysis of Access Control Mechanism of 5G Core Network," in *Proceedings of the 2023 ACM SIGSAC Conference on Computer and Communications Security*, 2023, pp. 666–680.

[4] *3GPP Standard*, www.3gpp.org.

[5] *Free5GC*, www.free5gc.org.

[6] *Open5GS*, open5gs.org.

[7] *OpenAirInterface*, https://www.openairinterface.org/.

[8] 3GPP, "5G; Security architecture and procedures for 5G System," TS 33.501, 18.7.0. [Online]. Available: {https://www.3gpp.org/ftp/Specs/archive/33_series/33.501/33501-i70.zip}

[9] Arons, Tamarah and Pnueli, Amir and Ruah, Sitvanit and Xu, Ying and Zuck, Lenore, "Parameterized verification with automatically computed inductive assertions?" in *Computer Aided Verification: 13th International Conference, CAV 2001 Paris, France, July 18–22, 2001 Proceedings 13*. Springer, 2001, pp. 221–234.

[10] A. Pnueli, "In transition from global to modular temporal reasoning about programs," in *Logics and models of concurrent systems*.

[11] R. Alur, P. Madhusudan, and W. Nam, "Symbolic compositional verification by learning assumptions," in *International Conference on Computer Aided Verification*. Springer, 2005, pp. 548–562.

[12] J. M. Cobleigh, D. Giannakopoulou, and C. S. Păsăreanu, "Learning assumptions for compositional verification," in *Tools and Algorithms for the Construction and Analysis of Systems: 9th International Conference*. Springer, 2003, pp. 331–346.

[13] E. M. Clarke, T. A. Henzinger, H. Veith, R. Bloem *et al.*, *Handbook of model checking*. Springer, 2018, vol. 10.

[14] A. Security, "White Paper: A slice in time: Slicing security in 5G Core Networks," Tech. Rep., [Online; accessed December 1, 2022]. [Online]. Available: {https://info.adaptivemobile.com/network-slicing-security}

[15] *CoreScan*, https://github.com/SyNSec-den/CoreScan.

[16] *GSMA Coordinated Vulnerability Disclosure Programme)*, https://www.gsma.com/security/gsma-coordinated-vulnerability-disclosure-programme/.

[17] D. Hardt, "The OAuth 2.0 Authorization Framework," RFC 6749. [Online]. Available: {https://doi.org/10.17487/rfc6749}

[18] 3GPP, "System architecture for the 5G System (5GS)," TS 23.501, Version 18.7.0. [Online]. Available: {https://www.3gpp.org/ftp/Specs/archive/23_series/23.501/23501-i70.zip}

[19] "Telecoms: How 5G will revolutionise the MVNO market," https://telecoms.com/opinion/how-5g-will-revolutionise-the-mvno-market/, [Online; accessed July 25, 2023].

[20] "5GRadar: Discover how MVNOs can make the most of 5G," https://www.5gradar.com/features/discover-how-mvnos-can-make-the-most-of-5g, [Online; accessed July 25, 2023].

[21] "Over 900,000 Kubernetes instances found exposed online," https://www.bleepingcomputer.com/news/security/over-900-000-kubernetes-instances-found-exposed-online/, [Online; accessed July 25, 2023].

[22] "Hildegard: New TeamTNT Cryptojacking Malware Targeting Kubernetes," https://unit42.paloaltonetworks.com/hildegard-malware-teamtnt/, [Online; accessed July 25, 2023].

[23] "5G Networks Are Worryingly Hackable," https://spectrum.ieee.org/5g-virtualization-increased-hackability, [Online; accessed July 25, 2023].

[24] "CVE-2016-1906," https://www.cvedetails.com/cve/CVE-2016-1906, [Online; accessed July 25, 2023].

[25] "Kubernetes RBAC Used By Attackers To Deploy Persistent Backdoor," https://phishingtackle.com/articles/kubernetes-rbac-used-by-attackers-to-deploy-persistent-backdoor/, [Online; accessed July 25, 2023].

[26] "Malicious Kubernetes Helm charts can be used to steal sensitive information from Argo CD deployments," https://apiiro.com/blog/malicious-kubernetes-helm-charts-can-be-used-to-steal-sensitive-information-from-argo-cd-deployments/, [Online; accessed July 25, 2023].

[27] "OpenRAN – 5G hacking just got a lot more interesting," https://media.ccc.de/v/mch2022-273-openran-5g-hacking-just-got-a-lot-more-interesting, [Online; accessed July 25, 2023].

[28] 3GPP, "Study on security aspects of the 5G Service Based Architecture (SBA)," TR 33.855, Version 16.1.0. [Online]. Available: {https://www.3gpp.org/ftp//Specs/archive/33_series/33.855/33855-g10.zip}

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

[30] 3GPP, "Release 18 Description; Summary of Rel-18 Work Items," TR 21.918, 2024, 1.0.0. [Online]. Available: {https://www.3gpp.org/ftp/Specs/archive/21_series/21.918/21918-100.zip}

[31] D. E. Long, *Model checking, abstraction, and compositional verification*. Carnegie Mellon University, 1993.

[32] R. Cavada, A. Cimatti, M. Dorigatti, A. Griggio, A. Mariotti, A. Micheli, S. Mover, M. Roveri, and S. Tonetta, "The nuXmv symbolic model checker," in *International Conference on Computer Aided Verification*.

[33] E. Clarke, O. Grumberg, S. Jha, Y. Lu, and H. Veith, "Counterexample-guided abstraction refinement," in *International Conference on Computer Aided Verification*.

[34] 3GPP, "5G; 5G Security Assurance Specification (SCAS);Service Communication Proxy (SCP)," TS 33.522, 18.0.0. [Online]. Available: {https://www.3gpp.org/ftp/Specs/archive/33_series/33.522/33522-i00.zip}

[35] ——, "5G System; Network function repository services; Stage 3," TS 29.510, 18.8.0. [Online]. Available: {https://www.3gpp.org/ftp/Specs/archive/29_series/29.510/29510-i80.zip}

[36] ——, "Procedures for the 5G System (5GS)," TS 23.502, Version 18.7.0. [Online]. Available: {https://www.3gpp.org/ftp/Specs/archive/23_series/23.502/23502-i70.zip}

[37] ——, "Release 17 Description; Summary of Rel-17 Work Items," Tech. Rep. [Online]. Available: {https://www.etsi.org/deliver/etsi_tr/121900_121999/121917/17.00.01_60/tr_121917v170001p.pdf}

[38] J. Kumar, A. Gupta, S. Tanwar, and M. K. Khan, "A review on 5g and beyond wireless communication channel models: Applications and challenges," *Physical Communication*.

[39] L. Suárez, D. Espes, F. Cuppens, P. Bertin, C.-T. Phan, and P. Le Parc, "Formalization of a security access control model for the 5G system," in *2020 11th International Conference on Network of the Future (NoF)*. IEEE, 2020, pp. 150–158.

[40] D. Basin, J. Dreier, L. Hirschi, S. Radomirovic, R. Sasse, and V. Stettler, "A formal analysis of 5g authentication," in *Proceedings of the 2018 ACM SIGSAC conference on computer and communications security*.

[41] I. Ahmad, T. Kumar, M. Liyanage, J. Okwuibe, M. Ylianttila, and A. Gurtov, "Overview of 5G security challenges and solutions," *IEEE Communications Standards Magazine*, vol. 2, no. 1, pp. 36–43, 2018.

[42] A. Sukumar, A. Singh, A. Gupta, and M. Singh, "Enhancing security and privacy implications in 5g network slicing," in *2024 Fourth International Conference on Advances in Electrical, Computing, Communication and Sustainable Technologies (ICAECT)*.

[43] S. Dorozhynskyi, I. Zakutynskyi, M. Ryabyy, and A. Skurativskyi, "Maximizing security and efficiency in 5g networks by means of quantum cryptography and network slicing concepts," in *2023 IEEE 12th International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications (IDAACS)*.

[44] H. Chen, N. Li, C. S. Gates, and Z. Mao, "Towards analyzing complex operating system access control configurations," in *Proceedings of the 15th ACM symposium on Access control models and technologies*.

[45] L. Cheng, Y. Zhang, and Z. Han, "Quantitatively measure access control mechanisms across different operating systems," in *2013 IEEE 7th International Conference on Software Security and Reliability*.

[46] K. Jayaraman, V. Ganesh, M. Tripunitara, M. Rinard, and S. Chapin, "Automatic error finding in access-control policies."

[47] P. Centonze, R. J. Flynn, and M. Pistoia, "Combining static and dynamic analysis for automatic identification of precise access-control policies," in *Twenty-Third Annual Computer Security Applications Conference (ACSAC 2007)*.

[48] X. Li, Y. Chen, Z. Lin, X. Wang, and J. H. Chen, "Automatic policy generation for {Inter-Service} access control of microservices," in *30th USENIX Security Symposium (USENIX Security 21)*, 2021, pp. 3971–3988.

[49] F. Sun, L. Xu, and Z. Su, "Static detection of access control vulnerabilities in web applications," in *20th USENIX Security Symposium (USENIX Security 11)*.

[50] A. Gouglidis, A. Kagia, and V. C. Hu, "Model checking access control policies: A case study using google cloud iam," *arXiv preprint arXiv:2303.16688*.

[51] B. Shen, T. Shan, and Y. Zhou, "Multiview: Finding blind spots in {Access-Deny} issues diagnosis," in *32nd USENIX Security Symposium (USENIX Security 23)*.

[52] S. Nakamura and Y. Tanaka, "Opportunities and challenges in applying machine learning for access control," *Authorea Preprints*.

[53] X. Xiao, A. Paradkar, S. Thummalapenta, and T. Xie, "Automated extraction of security policies from natural-language software documents," in *Proceedings of the ACM SIGSOFT 20th International Symposium on the Foundations of Software Engineering*, 2012, pp. 1–11.

[54] S. H. Jayasundara, N. A. G. Arachchilage, and G. Russello, "Ragent: Retrieval-based access control policy generation," *arXiv preprint arXiv:2409.07489*, 2024.

[55] M. Abdelgawad, I. Ray, S. Alqurashi, V. Venkatesha, and H. Shirazi, "Synthesizing and analyzing attribute-based access control model generated from natural language policy statements," in *Proceedings of the 28th ACM Symposium on Access Control Models and Technologies*, 2023, pp. 91–98.

[56] L. Ma, Z. Yang, Z. Bu, Q. Lao, and W. Yang, "Statement recognition of access control policies in iot networks," *Sensors*, vol. 23, no. 18, p. 7935, 2023.

[57] T. Al Rahat, Y. Feng, and Y. Tian, "Oauthlint: an empirical study on oauth bugs in android applications," in *2019 34th IEEE/ACM International Conference on Automated Software Engineering (ASE)*. IEEE, 2019, pp. 293–304.

[58] ——, "Cerberus: Query-driven Scalable Security Checking for OAuth Service Provider Implementations," *arXiv preprint arXiv:2110.01005*, 2021.

[59] E. Y. Chen, Y. Pei, S. Chen, Y. Tian, R. Kotcher, and P. Tague, "Oauth demystified for mobile application developers," in *Proceedings of the 2014 ACM SIGSAC conference on computer and communications security*, 2014, pp. 892–903.

[60] R. Yang, G. Li, W. C. Lau, K. Zhang, and P. Hu, "Model-based security testing: An empirical study on oauth 2.0 implementations," in *Proceedings of the 11th ACM on Asia Conference on Computer and Communications Security*, 2016, pp. 651–662.

[61] "Oauth2 Access Token Request API of 5G system," https://forge.3gpp.org/swagger/tools/loader.html?yaml=TS29510_Nnrf_AccessToken.yaml, [Online; accessed December 1, 2022].

[62] S. Bhattacharya, M. Najana, A. Khanna, and P. Chintale, "Securing the gatekeeper: Addressing vulnerabilities in oauth implementations for enhanced web security," *International Journal of Global Innovations and Solutions (IJGIS)*, 2024.

[63] "Free5GC's access token implementation'," https://github.com/free5gc/nrf/blob/main/internal/sbi/processor/access_token.go#L23, 2025, accessed: 2025-01-15.

Appendix A.
5G Communication Types

The 5G Core network (5GC) supports three primary communication modes: direct, indirect, and hybrid. In direct communication, Network Functions (NFs) interact with each other without any intermediaries, enabling low-latency exchanges that are ideal for performance-sensitive applications. However, this approach exposes NFs directly to one another, which can introduce security risks such as unauthorized access or data tampering. Besides, the tight coupling of NFs may create scalability challenges in dynamic network environments.

Indirect communication, as illustrated in Figure 15(a), requires all interactions, including access token requests and service requests, to be routed through the SCP. This approach ensures centralized control and helps to enforce traffic monitoring, policy enforcement, and load balancing. By isolating NFs from direct exposure, indirect communication strengthens the network's overall security and reduces potential vulnerabilities.

Finally, hybrid communication strikes a balance between direct and indirect communication by providing more flexibility on how service access token requests are routed. Figure 15(b) and (c) show examples of a hybrid communication mode where discovery is always done directly, and service request is always done through the SCP. Access token requests can be made either directly (as shown in Figure 15(b)) or through the SCP (as shown in Figure 15(c)), depending on the presence of mutual authentication between the consumer NF and the NRF.

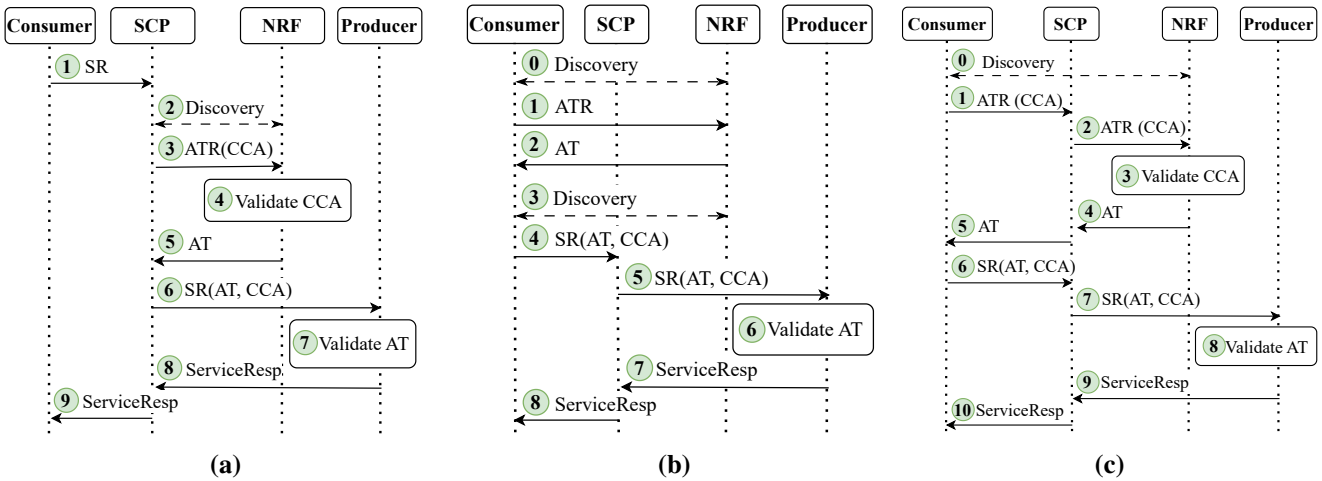


Figure 15: Simplified message flow for different communication models. (a) Indirect communication, where the AT must always be requested through SCP. (b) Hybrid communication, where the AT is directly requested. (c) Hybrid communication, where the AT is requested through SCP. Here, SR:*ServiceReq*, AT:access token, ATR:*accessTokenReq*, and CCA: Client Credentials Assertion token.

Appendix B. Potential Fixes for Identified Vulnerabilities

The potential fixes for the new `CoreScan` attacks (§7.1) are provided below.

B.1. *Coarse Scope Attack*

1) **Clarify `producerSlice` semantics.** The 5G specification should clearly define the `producerSlice` in access tokens. Specifically, it should include only the slices requested and authorized during *accessTokenReq* verification to avoid ambiguity.
2) **Clarify handling of optional attributes.** The specification should (i) clearly define what qualifies as an optional attribute and (ii) state the security implications if such attributes are omitted. Currently, the term 'optional' is inconsistently used, resulting in ambiguity. For instance, optional parameters in *NFDiscReq* (e.g., `targetSnssais`) help refine search results without affecting security. In contrast, omitting `producerSlice` from access tokens, as seen in Free5GC [63], results in a trivial exploit. We recommend `producerSlice` be treated as *conditional*—mandatory whenever the 5GC supports slicing.

B.2. *NFProfile Leakage Attack*

1) **Ensure CCA Token Replay Protection.** Short-lived replay protection is insufficient for CCA tokens, as it can lead to overprivilege or DoS. To prevent this, a nonce should be added to the CCA token by the consumer NF. The NRF then tracks these nonces to guarantee each token is processed only once.
2) **Validate CCA by SCP.** The SCP should verify that the credentials in the CCA token (e.g., sub) correspond to the consumer NF's certificate to prevent impersonation. This validation is currently absent in the 5G specification, making such checks necessary.

B.3. *CCA Replay Attack*

Since the underlying vulnerabilities are similar, the mitigation strategies proposed for *NFProfile Leakage Attack* are sufficient to address this attack as well, requiring no additional measures.

B.4. *CCA Evasion Attack*

We believe this vulnerability cannot be effectively mitigated without a major redesign, which is beyond the scope of this work and left for future investigation. However, we anticipate that 3GPP will introduce a proper authentication scheme for 5G roaming in upcoming specifications.

B.5. *Forced NF Selection Attack*

If no other over-privilege attacks exist in the system, this vulnerability would, at worst, result in a DoS attack. However, to prevent such outcomes, it is essential to patch all related vulnerabilities, particularly those involving CCA and SCP. To further mitigate DoS risks, we propose an approach called **Verifiable Attribute Reporting**:

- **Integrity Checks**: Attributes such as load and priority should be verified by trusted monitoring systems rather than relying solely on NF self-reporting.
- **Telemetry Integration**: The 5G standard should define mechanisms for leveraging network telemetry data to independently assess NF load and performance. The NRF should use this data to update NFProfile attributes, replacing self-reported values.

Appendix C. Meta-Review

The following meta-review was prepared by the program committee for the 2025 IEEE Symposium on Security and Privacy (S&P) as part of the review process as detailed in the call for papers.

C.1. Summary

This paper introduces CoreScan, a compositional verification technique that models the 5g Core network by breaking it into smaller components to perform a comprehensive formal analysis. Formal analysis of the 5g specification is challenging because of missing reference implementations, natural language specification, and state explosion. CoreScan was utilized to test 61 access control properties and found five new classes of exploitable privilege escalation vulnerabilities in the 5G standards.

C.2. Scientific Contributions

- Identifies an Impactful Vulnerability -Provides a Valuable Step Forward in an Established Field

C.3. Reasons for Acceptance

1) This paper identifies multiple impactful vulnerabilities. The formal analysis revealed 10 vulnerabilities and five attacks. Compared against 5GCVerif as a state-of-the-art solution, it shows significantly improved scalability.
2) The paper provides a valuable step forward in an established field. The findings of this research are novel, offering insights that can guide future enhancements to access control protocols.

C.4. Noteworthy Concerns

1) Corescan evaluation is limited to open-source 5G implementations as there was no access to commercial 5G Core systems.

Appendix D. Response to the Meta-Review

We acknowledge that CoreScan's evaluation is limited to open-source 5G Core implementations due to a lack of access to commercial systems. Thus, to compensate for this, we submitted them to GSMA, which has acknowledged all findings (A1-A5) under CVD-2025-0101. We are actively working with GSMA on potential defenses and drafting change requests (CRs) for 3GPP updates.