An Experimental Testbed for 5G Network Security Assessment

Karim Baccar, Abdelkader Lahmadi

An Experimental Testbed for 5G Network Security Assessment

Karim Baccar* and Abdelkader Lahmadi*
*Université de Lorraine, CNRS, Inria, LORIA, 54000 Nancy, France
karim.baccar@inria.fr, abdelkader.lahmadi@loria.fr

***Abstract*—The Fifth Generation (5G) mobile networks are designed to provide a large range of services with stringent requirements in robustness and security. Thus, it is important to ensure that these networks fulfill these requirements and are resilient against attacks. To meet this challenge, experimental testbeds and tools are required to test and evaluate the security of 5G networks. This work presents an experimentation testbed and support tools for generating and injecting on the fly 5G packets to realize multiple security assessing tasks in particular fuzzing operations for vulnerabilities discovery. Our tool is implemented and tested within a controlled testbed environment built on top of a 5G standalone core server provided by a hardware base station (gNb) and uses an Software Defined Radio(SDR) card for radio transmission. We validate our testbed and the developed tools by successfully injecting at the 5G air interface and the network control levels different messages including RRC and NGAP/NAS over already established communications.**

***Index Terms*—5G protocols, 5G Security, Fuzzing**

I. Introduction

Fifth Generation (5G) mobile networks have changed the landscape of mobile networking and opened new horizons in the way industries and people can utilize the newly introduced services of 5G to innovate and create new business opportunities. 5G networks are already deployed and massively used in most countries worldwide [1] and their applications are extremely diverse ranging from 4.0 industries, transportation systems to precision agriculture. Many of these applications are mission critical with high security and trust requirements. For instance, 5G mobile networks are candidate for autonomous cars transportation system, and thus, these requirements have to be evaluated and assessed for 5G protocols and their associated implementations. The complexity of 5G networks has increased compared to previous mobile networks with more programmability and virtualization which also enlarged their attack surface.

Given the novelty of the technology, 5G protocols stack lacks the supporting tools to assess its security although it should be a priority given the criticality of its applications. Moreover the changes brought to 5G by the 3GPP organisation in terms of architecture and procedures have made conformity evaluation of the network to the standard requirements a complex task. Techniques such as fuzz [2] and conformance testing would be therefore interesting to apply as they have proven their results when it comes to testing the robustness of a complex system. This is in fact a requirement specified by 3GPP in Technical Specification(TS) 33.117 [3], and it is required that vendors and mobile network operators alike must test their available 5G protocol implementation through fuzzing techniques and vulnerability discovery prior to network deployment.

Testing on a network like 5G can be challenging since sending packets to the base station through the Radio Access Network (RAN) requires having a full implementation of the 5G protocol stack and also being able to synchronize an uplink channel with the cell through the random access procedure in order to activate an RRC connection. Fortunately open source 5G implementations of User Equipment's(UE) exist, most prominently the implementation of srsUE developed within the srsRAN [4] project. The library offers a functional New Radio(NR) UE client that is compatible with 5G Standalone(SA) and most of the available SDR cards. However the library was not developed for a security testing use case but mostly to emulate operations of 5G networks.

Several existing open source software tools used by researchers to test the operations of 5G protocols and networks, such as OpenAirInterface(OAI) [5], Open5GS [6] and previously mentioned srsRAN, are still not yet adapted for security testing tasks such as protocol fuzzing, vulnerability discovery or attack replay as they lack custom packet injection functions in particular at the radio level. Moreover, the main purpose of these tools is to implement 5G network functionalities instead of security testing functions.

To fit this gap between available tools for testing 5G functionalities and security testing, this work aims to develop an experimental testbed and support tools for generating and injecting on the fly 5G packet at the radio access network. We tested our tools on an experimental setup using Amarisoft Callbox (hardware base station) [7] that implements both a Next Generation NodeB (gNB) and a full 5G core network.

The remainder of this paper is organized as follows. Section II presents related work on 5G security and the use of tools for its assessment. Section III provides preliminaries and background elements of 5G protocols relevant to our experimental testbed. Section IV details the architecture of our injection tool and its integration and validation within our proposed testbed. Section V will be dedicated to the experimental results. Section VI concludes the paper.

II. Related Work

Recently, there has been efforts to test attacks and assess the robustness of the 5G RAN through packet injection tools.

In [8], the authors presented attacks on the Evolved Packet Core(EPC) handover in 5G networks. Their implementation used both srsLTE and srsUE [4] along with SDR devices. They also modified parts of srsUE to obtain the P-CSCF IP address for their validation of IP Multimedia Subsystems(IMS) attacks. In [9], Evangelos et al. demonstrated attacks on the public warning system(PWS) by sending and removing CMAS/ETWS messages. This was done by modifying the configuration of Amarisoft callbox to send emergency warning messages. In [10], Jochem et al. presented a downlink fuzzer for UE Long Term Evolution(LTE) devices. The implementation of their fuzzer uses a modified version of srsEnb to send the messages. In another work, Hongxin et al. [11] implemented downlink and uplink fuzzing on the open source RAN implementation of OpenAirInterface. They used code instrumentation to modify the UE and gNB programs. The novelty in their work consists in implementing a stateful RRC fuzzer. In [12], authors developed an open source tool named 5GReplay that offers packet forwarding and modification through deep packet inspection. The tool was tested on the NGAP protocol for fuzzing and testing replay attacks.

Although, as described above, some notable works and tools exist for 5G packets injection, none of them offers a generic tool able to inject on the fly generated 5G packets to network components. Usually, they rely on mutated and already captured messages instead of injecting messages generated from 3GPP specifications. Hongxin et al. [11], for example used packet mutation of existing messages in OAI gNB tool while Jochem et al. [10] only tested fuzzing with the RRC Setup Request message. The only work that offered the flexibility of injecting generated RRC messages based on the ASN.1 specification was the work of Srinath et al. [13]. While it was possible to make such injections according to the authors, their tool was not tested on 5G air interfaces but rather only on SRS LTE software components.

III. Preliminaries

3GPP have brought a lot of changes to the 5G networks in order to offer enhanced functionalities over LTE, better security and more applications. This section we will briefly introduces the 5G protocol stack for the RAN level, its key procedures and an analysis of its security issues that are relevant to the development of our experimental testbed and tools.

A. Protocol Stacks

Figure 1 shows the protocol layers for the 5G NR architecture in the control plane. The establishment and management of sessions and the authentication procedures occur at the highest layer on the control plane called **Non-Access Stratum**. In the same plane, **Radio Resource Control** (RRC) is used to exchange control information with the device to set important parameters for the session. The main services of RRC include the broadcast of system information related to NAS and AS, Paging, security key function management, management of SRB's and DRB's, mobility function management, QoS management, UE reporting and encapsulation of NAS messages.

Protocols of layer 2 include **Radio Link Control** (RLC), **Packet Data Convergence Protocol** (PDCP), and **Medium Access Control** (MAC) and provide roles that includes ciphering, mapping between logical and transport channels QoS flow management, protocol error detection and segmentation, resegmentation.

Layer 1 includes the **PHY** layer responsible for modulation, demodulation of physical channels and frequency and time synchronization for both planes.

Communication between the Access and Mobility Function (AMF) and the gNB happens in the N2 interface which transports **NGAP** protocol messages encapsulated within Stream Control Transmission Protocol (SCTP). The NG Application Protocol has two main sets of services, UE-associated services which are responsible for the communication between the AMF and the UE and Non UE-associated services related to the management of the NG-RAN node itself.

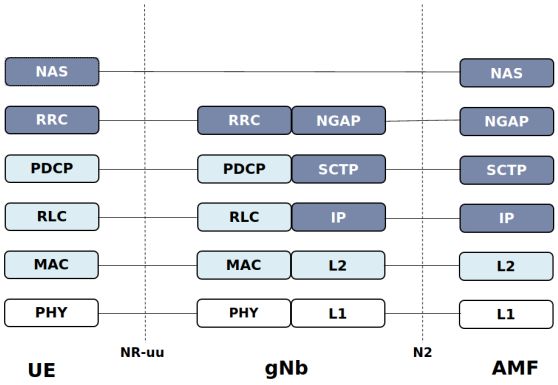

Fig. 1. 5G architecture protocol stacks for the control plane.

B. UE Procedures

Figure 2 details the registration procedure of a UE in the 5G network. First the UE synchronizes with the gNB to obtain an uplink Common Control Channel (CCCH) in order to initiate the RRC Setup Request message, this is called the Random Access Procedure. Secondly, in the RRC Connection Procedure, the UE is able to communicate with the AMF over the Dedicated Control Channel (DCCH) after sending RRC Setup Complete. Thirdly, after piggybacking the registration request message over the NAS protocol, the UE validates the authentication challenge that is sent to it and then negotiates security methods that will be used during communication. Finally, after security configuration in AS, the UE is then able to initiate a PDU session request with NAS to the AMF to allocate an Uplink PDU session and is then able to send user data.

C. Security Issues

The 3GPP specification TS 33.501 [14] details the main security requirements related to security in 5G mobile net-

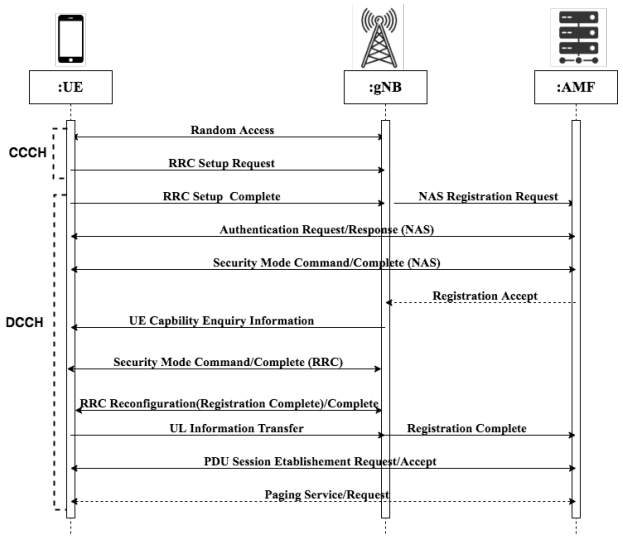

Fig. 2. Overall steps of the registration procedure of a UE in a 5G mobile network.

works. The standard mentions enhanced security against International Mobile Subscriber Identity(IMSI) catching with the use of Subscription Concealed Identifiers(SUCI) during initial registration. However according to the work of Chuan et al. [15]; in which the authors studied the feasibility of 4G attacks in 5G, they demonstrate that 5G networks are still vulnerable to some of the attacks of its predecessor. This is mainly due to a lack of security of the messages that are prior to the creation of the security context as detailed in figure 2 and to the fact that security features are still optional at the user and control planes. It is worthy to note that the TS 33.501 requirements related to message confidentiality and integrity are the following:

"Confidentiality protection of the user data between the UE and the gNB is optional to use."

"Confidentiality protection of the RRC-signalling, and NAS-signalling is optional to use."

"Integrity protection of the user data between the UE and the gNB is optional to use."

"Integrity protection of the RRC-signalling, and NAS-signalling is mandatory to use, except in the lists mentioned in TS 24.501, TS 38.331 and during unauthenticated emergency sessions specified in clause 10.2.2"

The implementation of the security features within the network is therefore a decision to be made by network operators and is not mandatory according to the specification. This has been proven to be a security risk [15], [16] since 5G networks are open in nature which makes them vulnerable to various attacks and threats, in particular at the RAN level where attackers are able to inject malicious messages. Therefore, we built an experimental testbed based on a hardware base station and we developed an injection tool of 5G packets at the RAN level to provide features for security testing tasks. A list of the most relevant features includes (i) protocol based fuzz testing of 5G networks through the RAN level, which allows us to assess attacks from the UE side or rogue small cells, (ii) assessing the compliance of the RAN components through behaviour based evaluations, (iii) testing and finding novel attacks on the network and vulnerabilities in the already deployed 5G implementations.

IV. Testbed Implementation

In this section we will present the architecture of the testbed and the implementation of our packets injection tool with the modification that we carried on the tool srsUE to send and generate NR messages.

A. Testbed Architecture

Figures 3 and 4 show the architecture and the setup that we implemented in our testbed. Our testbed is composed of several components to provide a full end-to-end 5G network. The main component of our testbed is the base station with is able to provide RAN functionalities and embeds also a core 5G server. We make the choice to use a commercial off-the-shelf base station able to offer a full-stack 5G network instead of an existing open source software solutions that only emulate partially a 5G network and its protocols stack. However, our architecture is open to rely on other 5G base stations or 5G open source software tools emulating 5G networks. The second major component of our experimental testbed is the UE emulator. We chose to use the srsUE [4] tool which is able to provide a 5G NR UE modem implemented entirely in software. This tool is implemented in C++ and is actively maintained by an open source community, it also supports multiple SDR cards including USRP B2x0/X3x0 families, BladeRF, LimeSDR. However, this tool is mainly developed to emulate a UE and not dedicated to inject on the fly custom 5G packets. Thus, we carried necessary modifications to allow us to make such injections possible. The wireless communications between the srsUE and the Amarisoft Box are realized through the Ettus USRP B210 SDR card.

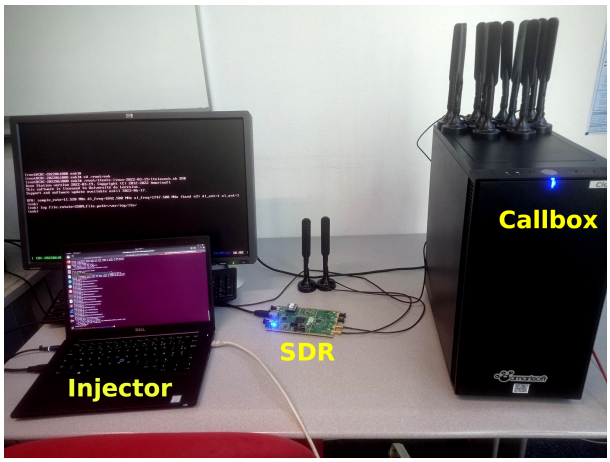

Fig. 3. The experimental setup of our testbed including the Amarisoft call Box (base station), the SDR card Ettus USRP B210 connected to a laptop running our injection tool coupled with srsUE.

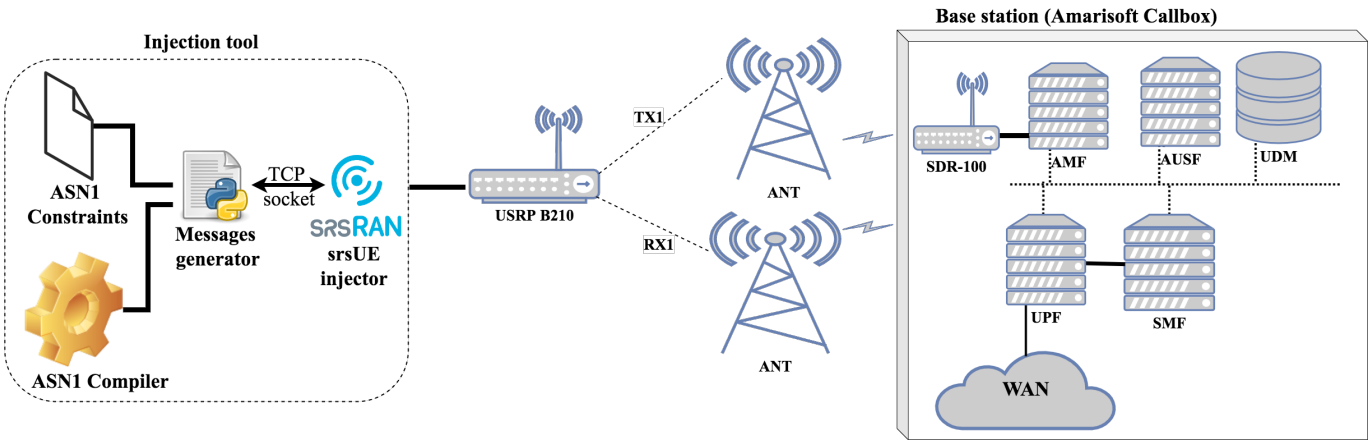

Fig. 4. Our injection tool and testbed architectures with their respective components.

As depicted in Figure 3, the overall setup of our testbed is composed of the following components:

- **The injector**: A Linux Ubuntu 20.4 host with an I7 8650U, 16GB of RAM, the modified version of srsUE and a python program for message generation.
- **USRP B210**: an SDR connected to wireless antennas compatible with band 3 of 5G NR and a mounted GPSDO oscillator.
- **Amarisoft Callbox**: the base station which contains a gNodeB operating with an embedded SDR and a core server that includes most 5G components.

B. Adaptation of srsUE program

srsRAN [4] is a free and an open sourced software suite that offers implementations of LTE and NR components that follows 3GPP specifications. The project offers an implementation of gNB, UE and an EPC core network components. In our work, we only rely on the srsUE since the gNB and core network components are provided by the Amarisoft callbox.

Our first modifications consists in creating a method in the file *ue.cc* that instantiates a socket object in order to open a TCP connection. This socket is used to forward the RRC messages in UPER format from our message generation tool to the srsUE program. We then added two other functions in the file *rrc _nr.cc* to send raw bytes of the forwarded messages to the lower stack layers in order to send them across the network. Depending if the messages are of types CCCH or DCCH, they will be respectively sent to the RLC and the PDCP layers.

We also added a parameter in the configuration file of srsUE to set the logical channel on which the injection will happen. The injection through DCCH is straight forward since it will make srsUE establishes an RRC connection normally. However, with CCCH, the *ue.cc* object will reboot the UE each time a message is sent to srsUE. The message is then sent during the initiation of the random access procedure.

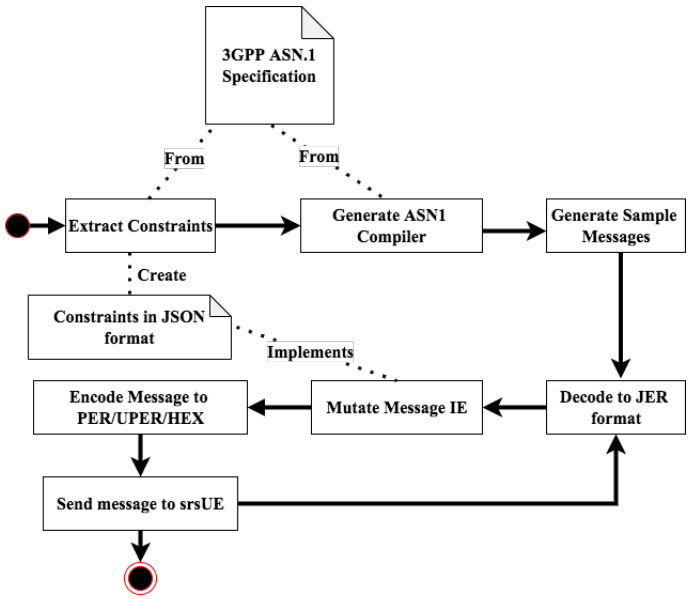

Fig. 5. 5G Messages generation and sending process used by our injection tool.

C. Protocol Messages Generation

Generating 5G NR messages is a necessary task for fuzz testing the 5G protocols and simulating specific attack scenarios. This task is challenging since it requires to rely on message specifications to generate them in vendor-independent way, which is more difficult than simply capturing messages from normal traffic flow.

In order to generate NR messages we used the open source Python library pycrate [17] for Abstract Syntax Notation One (ASN.1) language compilation. The library offers features that include generating a compiler from a given ASN.1 specification and message encoding and decoding. In the current version of our tools, we generate 21 types of uplink RRC CCCH/DCCH messages, all of the 67 types of NGAP messages types and 28 types of NAS messages.

Our mutation strategy of the message samples consists in

changing the Information Elements (IE) without violating the language specification of the message. This will ensure better testing coverage as the messages will not be directly discarded by the System Under Test(SUT) parser. We therefore had to extract all of the constraints in the ASN.1 specification and stored them in a JSON file. Figure 6 gives an example of the constraints of NGAP IE's extracted from its ASN.1 specifications. In total, 1247 IE were extracted from RRC specification and 175 from NGAP specification, the types that we identified are integer, bit string and octet string.


```
{
  "name": "AlternativeQoSParaSetIndex",
  "min": "1",
  "max": "NULL"
},
{
  "name": "AlternativeQoSParaSetNotifyIndex",
  "min": "0",
  "max": "NULL"
},
{
  "name": "AMF-UE-NGAP-ID",
  "min": "0",
  "max": "1099511627775"
},
```


Fig. 6. An examples of constraints extracted from the ASN.1 specifications of NGAP message IE's.

Each message sample is then converted to its ASN.1 JER (Json Encoding Rules) format as it is interoperable with Python data structures. Each IE is then mutated according to its max and min value constraints. The overall process of our generation and sending 5G protocol messages is summarized in Figure 5. First, we rely on the 3GPP ASN.1 specification to extract IE constraints and generate the compiler. Our tool then generates the messages using this compiler which are encoded before providing them to srsUE to be sent over the radio channel.

V. Experimentation

This section presents the experimental results of our injection and generation tools by using the implemented 5G testbed. We mainly validated our tool by generating and injecting 3GPP compliant messages. In a first scenario, RRC messages at the radio level are injected through CCCH and DCCH modes. In a second scenario, we inject NGAP and NAS messages through the N2 interface. This second scenario aims at testing our tool by injecting messages in internal control interfaces. Figure 7 gives an overview of the two injection levels used in our experiments with our testbed.

A. Injection of RRC messages

RRC is a control plane protocol used on the air interface to manage the radio resources and establish connections between the UE and the base station. Injecting its messages is realized at the radio level. By using our tools, we successfully injected

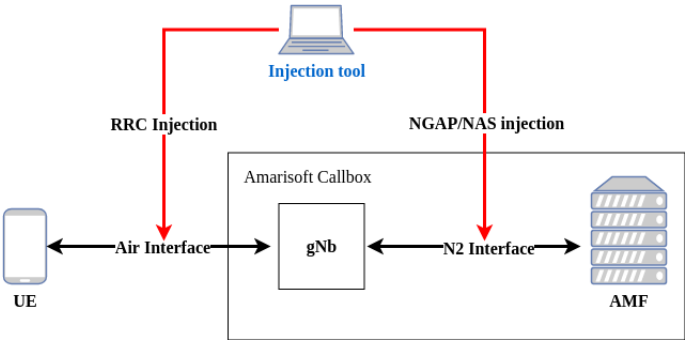

Fig. 7. Message injection levels implemented within our testbed.

different RRC messages over the CCCH and DCCH channels. Figures 8 and 9 show the logs of the Amarisoft Callbox logging application after injecting RRC messages by using our tool and srsUE. We observed that when injecting RRC messages over CCCH, a timelapse of 10 seconds is required for srsUE to reset itself and perform the RAN procedure again. However, the injection over DCCH is almost instantaneous.

Time	Diff	RAN	UE ID	Cell	Info	Message
-		RRC	399	1	DCCH-NR	RRC release
16:54:13.951	+11.976	RRC	400	1	CCCH-NR	RRC setup request
-		RRC	400	1	CCCH-NR	RRC setup
16:54:13.975	+0.024	RRC	400	1	DCCH-NR	RRC setup complete
-		RRC	400	1	DCCH-NR	RRC release
16:54:25.951	+11.976	RRC	401	1	CCCH-NR	RRC resume request
-		RRC	401	1		RRC Resume request: No matching UE
-		RRC	401	1	CCCH-NR	RRC setup
16:54:25.975	+0.024	RRC	401	1	DCCH-NR	RRC setup complete
-		RRC	401	1	DCCH-NR	RRC release
16:54:37.951	+11.976	RRC	402	1	CCCH-NR	RRC setup request
-		RRC	402	1	CCCH-NR	RRC setup
16:54:37.975	+0.024	RRC	402	1	DCCH-NR	RRC setup complete
-		RRC	402	1	DCCH-NR	RRC release
16:54:49.951	+11.976	RRC	403	1	CCCH-NR	RRC setup request
-		RRC	403	1	CCCH-NR	RRC setup
16:54:49.983	+0.032	RRC	403	1	DCCH-NR	RRC setup complete
-		RRC	403	1	DCCH-NR	RRC release
16:55:01.951	+11.968	RRC	404	1	CCCH-NR	RRC setup request
-		RRC	404	1	CCCH-NR	RRC setup
16:55:01.975	+0.024	RRC	404	1	DCCH-NR	RRC setup complete
-		RRC	404	1	DCCH-NR	RRC release
16:55:13.951	+11.976	RRC	405	1	CCCH-NR	RRC reestablishment request

Fig. 8. RRC messages injection over CCCH channel. The red boxes highlight the injected RRC messages by our tool at the radio level.

Time	Diff	RAN	UE ID	Cell	SFN	RNTI	Info	Message
15:51:51.298		**RRC**	**1**	**1**			**DCCH-NR**	**RRC setup complete**
15:51:58.004	+6.706	RRC	1	1			DCCH-NR	Location measurement indication
15:52:00.996	+2.992	RRC	1	1			DCCH-NR	SCG failure information EUTRA
15:52:04.004	+3.008	RRC	1	1			DCCH-NR	Counter check response
15:52:07.004	+3.000	RRC	1	1			DCCH-NR	SCG failure information EUTRA
15:52:10.004	+3.000	RRC	1	1			DCCH-NR	Measurement report
15:55:19.298	+189.294	RRC	2	1			DCCH-NR	RRC setup complete
15:55:23.404	+4.106	RRC	2	1			DCCH-NR	Location measurement indication
15:55:26.413	+3.009	RRC	2	1			DCCH-NR	RRC reconfiguration complete
15:55:29.396	+2.983	RRC	2	1			DCCH-NR	Security mode complete
15:55:32.396	+3.000	RRC	2	1			DCCH-NR	UL information transfer MRDC
15:55:35.396	+3.000	RRC	2	1			DCCH-NR	SCG failure information EUTRA
15:55:38.404	+3.008	RRC	2	1			DCCH-NR	Measurement report
15:55:41.404	+3.000	RRC	2	1			DCCH-NR	UE capability information
15:55:44.396	+2.992	RRC	2	1			DCCH-NR	UL information transfer MRDC
15:55:47.444	+3.048	RRC	2	1			DCCH-NR	UL information transfer MRDC
15:55:50.444	+3.000	RRC	2	1			DCCH-NR	SCG failure information EUTRA
15:55:53.444	+3.000	RRC	2	1			DCCH-NR	RRC reconfiguration complete
15:55:56.436	+2.992	RRC	2	1			DCCH-NR	UE capability information

Fig. 9. A set of RRC messages injected by our tool at the radio level over a DCCH channel.

B. Injection of NGAP and NAS messages

NGAP and NAS protocols are used to register and connect UE and gNB to the 5G core. Their respective messages are sent over SCTP protocol at the N2 interface between the gNB and the AMF components. For NGAP injection, we directly communicate with the N2 interface rather than using srsUE. After initiating an SCTP session we were able to send directly NGAP messages to the AMF component. Figure 10 details a packet capture of injected NGAP messages. NAS messages injection requires an NGAP PDU of type *UEInitialMessage* in order to forward a NAS payload to the AMF. Figure 11 shows a capture of a set of injected NAS messages.

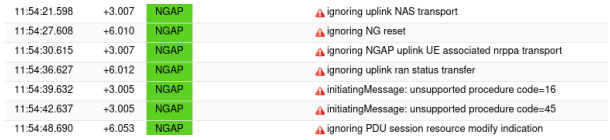

Fig. 10. An example of a set of NGAP messages injected at the N2 interface.

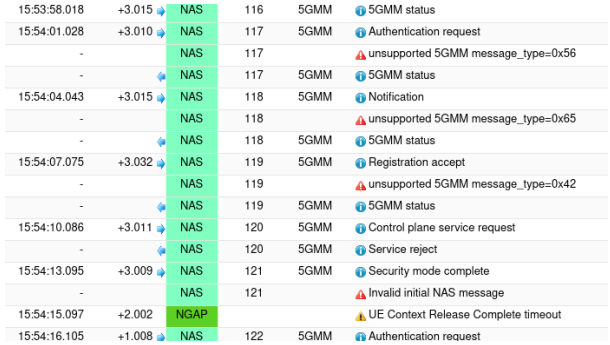

Fig. 11. An example of a set of injected NAS messages.

VI. Conclusions and Future Work

In this paper, we present the implementation of an experimental testbed and a support tool for generating and injecting on the fly 5G protocol stack messages over the air interface or over the interfaces of the core server. We demonstrate their use by injecting RRC, NGAP and NAS messages in already established communications between a UE and a base station. Our injection tool and the testbed are still under active development. In the future we plan to add Fuzz based testing features that mainly require modifications to monitor the state of the tested component and make feedback based mutations. The injection tool will also be used for the implementation and the validation of existing attacks on 5G networks to reproduce them in a controlled environment which provides a better analysis and measurement of their impact and feasibility. The generation and injection tool source code is available for other researchers upon request.

Acknowledgements

This work is partially supported by 5G events Lab, the project IMPACT DigiTrust of "Lorraine Université d'Excellence" and by EU H2020 project AI@EDGE (101015922).

References

[1] K. Buchholz. Where 5g technology has been deployed. [Online]. Available: https://www.statista.com/chart/23194/5g-networks-deployment-world-map/

[2] V. J. Manès, H. Han, C. Han, S. K. Cha, M. Egele, E. J. Schwartz, and M. Woo, "The art, science, and engineering of fuzzing: A survey," *IEEE Transactions on Software Engineering*, vol. 47, no. 11, pp. 2312–2331, 2021.

[3] 3GPP, "Universal Mobile Telecommunications System (UMTS), LTE, Catalogue of general security assurance requirements," Tech. Rep.

[4] srsRAN. [Online]. Available: https://github.com/srsran/srsRAN

[5] F. Kaltenberger, A. P. Silva, A. Gosain, L. Wang, and T.-T. Nguyen, "OpenAirInterface: Democratizing innovation in the 5G Era," *Computer Networks*, vol. 176, pp. 107 284 –, Jul. 2020. [Online]. Available: https://hal.archives-ouvertes.fr/hal-03490924

[6] Open5gs. [Online]. Available: https://github.com/open5gs/open5gs

[7] Amarisoft. [Online]. Available: https://www.amarisoft.com/app/uploads/2022/03/AMARI-Callbox-Advanced.pdf

[8] Z. Cui, B. Cui, J. Fu, and R. Dong, ""Security Threats to Voice Services in 5G Standalone Networks"," *Security and Communication Networks*, vol. 2022, no. 13, 2022.

[9] E. Bitsikas and C. Pöpper, "You have been warned: Abusing 5G's Warning and Emergency Systems," 2022. [Online]. Available: https://arxiv.org/abs/2207.02506

[10] J. Hoes, "Rise of the Machines, On the Security of Cellular IoT Devices," 2021. [Online]. Available: https://www.esat.kuleuven.be/cosic/publications/thesis-400.pdf

[11] H. Wang, B. Cui, W. Yang, J. Cui, L. Su, and L. Sun, "An Automated Vulnerability Detection Method for the 5G RRC Protocol Based on Fuzzing," in *In Proceedings of the 4th International Conference on Advances in Computer Technology, Information Science and Communications (CTISC)*, 2022.

[12] Z. Salazar, H. N. Nguyen, W. Mallouli, A. R. Cavalli, and E. 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*, ser. ARES 21, New York, NY, USA, 2021.

[13] S. Potnuru and P. K. Nakarmi, "Berserker: ASN.1-based Fuzzing of Radio Resource Control Protocol for 4G and 5G," in *2021 17th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob)*, 2021, pp. 295–300.

[14] 3GPP, "Security architecture and procedures for 5g system, technical report 33.501."

[15] C. Yu, S. Chen, F. Wang, and Z. Wei, "Improving 4G/5G Air Interface Security: A Survey of Existing Attacks on Different LTE Layers," *Comput. Netw.*, vol. 201, no. C, dec 2021.

[16] A. Shaik, R. Borgaonkar, S. Park, and J.-P. Seifert, "New Vulnerabilities in 4G and 5G Cellular Access Network Protocols: Exposing Device Capabilities," ser. WiSec '19, New York, NY, USA, 2019, p. 221–231.

[17] Pycrate. [Online]. Available: https://github.com/P1sec/pycrate