Security Study and Monitoring of LTE Networks

Dominik Ernsberger[1] and K. Jijo George[2]

[1]*University of Applied Sciences Munich, Germany*
[2]*NEC Technologies India Private Ltd. Chennai, India*
E-mail: ernsberger.dominik@gmail.com

Received 22 November 2018;
Accepted 05 January 2019

Abstract

Mobile communication systems are ubiquitous nowadays. The main requirements of these networks are privacy and security of the subscriber as well as a high performance. To provide these properties the 3GPP (Third Generation Partnership Project) developed the LTE (Long Term Evolution) mobile communication network which is deployed worldwide.

In this paper, we give a brief overview of the LTE Network Architecture as well as a look on the security mechanism as defined by 3GPP. We describe the security architecture and discuss possible threats and attacks on the core and on the access network. Due to these possible attacks we developed a program which is able to extract certain security relevant information out of the message flow in real time and to detect a possible attach flood attack. Finally, we validate the function of the program with three test cases and discuss the impact of such flood attacks on the LTE network and other future work to extend it to other protocol exchanges.

Keywords: LTE, LTE Security, Blockmon.

1 Introduction

A worldwide and permanent connection to mobile communication networks is taken for granted in our day-today life. It is no longer sufficient to just make calls and surfing in the Internet. The number of multimedia offers such as audio and video streaming or online gaming are increasing. Consequential the number of subscribers is rising and based on the growing field of application, new properties and opportunities are required for the wireless communication. In order to accommodate the new needs, 3GPP developed the LTE network where the latency time has been reduced as well as the data rate and coverage region increased.

Since there are several well-known security vulnerabilities in the 2G and 3G networks for instance DoS (Denial of Service) attacks [1] and MitM (Man-in-the-Middle) attacks [2] it is important that the new generations of mobile communication networks provide strengthened security. In the new deployed LTE network, they react to the previously known attacks and implemented new security mechanism.

In this paper we address the possibility of an attach flood attack in the network and explain the developed toolkit for analysing corresponding message flows. Flooding is a form of DoS attack that can be used to bring down a network or at the least disrupt the service for other users by choking the resources of the network. In the case of an attach flood attack, an attacker sends a lot of attach requests to the network and therefore tries to overload it.

The paper is organised as follows. Section 2 talks about the 3GPP defined LTE architecture from the standards perspective. In Section 3 we explain the established security mechanism, based on the new security architecture and describe the authentication procedure as defined by 3GPP. In Section 4 we discuss several vulnerabilities in the LTE network as well as possible attacks and threats on the core and access network. Due to the possibility of flood attacks in the LTE network, we developed a monitoring and analysing program based on the open-source tool Blockmon [11]. This allows us to extract and monitor important security relevant information such as unique IDs (e.g. ENB-UE-S1AP-ID, M-TMSI, IMSI) and security settings. Furthermore, we were able to check the flow order of the receiving packets and to discover flaws like missing or duplicated packets. Based on this we were able to detect a possible attach flood attack and monitor the collected information. In Section 5 we give an overview of Blockmon itself and the structure of the program. Furthermore, we explain the functionality and the implementation of the new added blocks. In Section 6 we prove the functionality of our program with the help of three test cases in our testbed. The test cases are concentrated to detect flaws and possible flood attacks in the EPS (Evolved Packet System) attach procedure, as shown in Figure 2. We developed the program as generic as possible to allow the use with other EPS procedures as well. Finally, we give a conclusion of this paper and a prospect to the future work.

2 LTE System Architecture

Figure 1 shows a briefly overview of a non-roaming EPS architecture. This can be split in two major areas. First the Access Network (AN) which includes the E-UTRAN (Evolved Universal Terrestrial Radio Access Network), GERAN (GSM/EDGE Radio Access Network) and UTRAN (UMTS Terrestrial Radio Access Network) and second the Core Network (CN) also called EPC (Evolved Packet Core).

The E-UTRAN is the evolved radio access network which contains several evolved base stations (eNodeB/eNB). It manages the wireless communication with the UE (User Equipment) and forwards the data via secured wired connections (S1 interface) to the EPC (MME/SGW). The eNodeBs are interconnected with each other via IPsec secured wired connections (X2 interface). For the radio communication to the UE the eNodeB uses a set of access network protocols, called Access Stratum (AS). These AS messages are also known as Radio Resource control (RRC) protocol messages [3].

The EPC is a completely IP-based and packet-switched (PS) core network. It includes several entities. The MME (Mobility Management Entity), SGW (Serving Gateway), PDN GW (Packet Data Network Gateway), PCRF (Policy and Charging Rules Function), SGSN (Serving GPRS Support Node) and the HSS (Home Subscriber Server) [4].

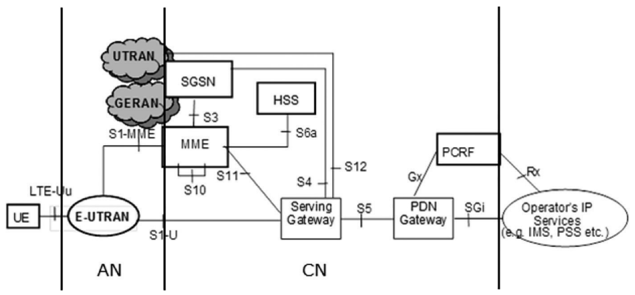


Figure 1 LTE system architecture [5].

3 LTE Security Mechanism as per 3GPP

To ensure the security and privacy of subscribers in modern mobile communication networks, security mechanism and features are defined by 3GPP. In the LTE network the security aspects are summarized in a five-group security architecture, which is shown in Figure 2 and defined as follows [5].

- Network access security (I): the set of security features that provide users with secure access (integrity protection and ciphering) to services, and which in particular protect against attacks on the (radio) access link.
- Network domain security (II): the set of security features that enable nodes to securely exchange signaling data, user data (S1 and X2 interface), and protect against attacks on the wireline network.
- User domain security (III): the set of security features that secure access to mobile stations.
- Application domain security (IV): the set of security features that enable applications in the user and in the provider domain to securely exchange messages.
- Visibility and configurability of security (V): the set of features that enables the user to inform himself whether a security feature is in operation or not and whether the use and provision of services should depend on the security feature.

The LTE Security Architecture gives an overview of all included security features. The following describes the security mechanisms which are used for the network access and for the secured communication between the UEs and the EPC.

A. Security in the Authentication

Every UE who wants a connection via the E-UTRAN to the EPC has to execute an attach procedure that includes a mutual authentication and key agreement

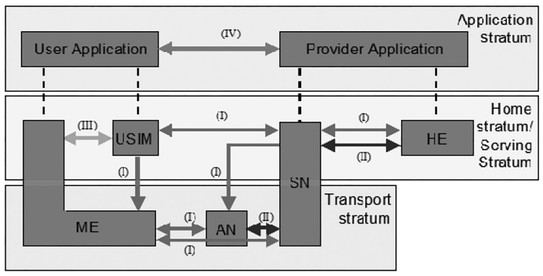


Figure 2 LTE security architecture [5].

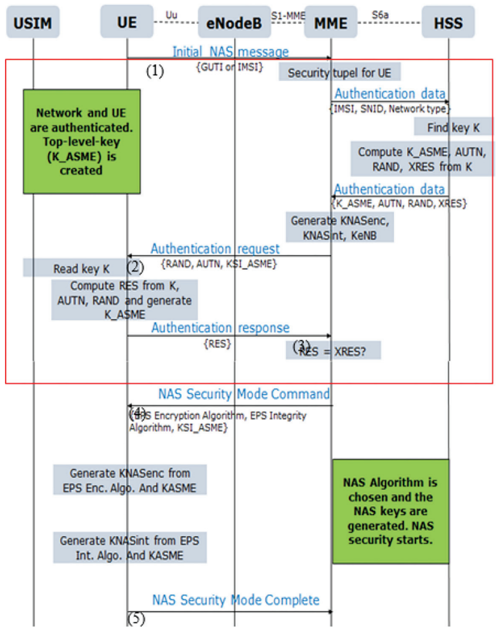

USIM
UE
Uu
eNodeB
S1-MME
MME
S6a
HSS
Initial NAS message
(1)
{GUTI or IMSI}
Security tupel for UE
Authentication data
{IMSI, SNID, Network type}
Find key K
Compute K_ASME, AUTN, RAND, XRES from K
Authentication data
{K_ASME, AUTN, RAND, XRES}
Network and UE are authenticated. Top-level-key (K_ASME) is created
Generate KNASenc, KNASint, KeNB
Authentication request
{RAND, AUTN, KSI_ASME}
Read key K
(2)
Compute RES from K, AUTN, RAND and generate K_ASME
Authentication response
{RES}
(3)
RES = XRES?
NAS Security Mode Command
(4)
{EPS Encryption Algorithm, EPS Integrity Algorithm, KSI_ASME}
Generate KNASenc from EPS Enc. Algo. And KASME
NAS Algorithm is chosen and the NAS keys are generated. NAS security starts.
Generate KNASint from EPS Int. Algo. And KASME
NAS Security Mode Complete
(5)

Figure 3 EPS attach procedure and AKA (red square).

(AKA) [5], which is shown in Figure 3. This procedure contains the exchange and derivation of the new key hierarchy [5]. Both are one of the important security features in the LTE security framework.

During the attach procedure the MME is the End-point for the UE and is managing the integrity and confidentiality protection for the NAS (Non-Access Stratum) [17]. Every attach request is forwarded via the eNodeB to the MME to identify the UE. For this purpose, the MME is requesting information from the HSS based on the sent IDs and collecting these data.

4 Threats and Practical Attacks

As mentioned above a new security architecture is implemented in the LTE network. In this section we want to show, that there are still vulnerabilities available and give a survey on the security aspects for LTE networks [6].

A. Tested practical Attacks

We tried three practical attacks on our LTE testbed, to exploit the security mechanism.

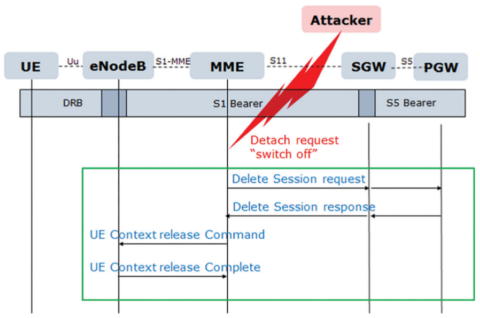


Figure 4 Rouge detach request attack. Green box is normal behaviour.

The first attack was with a rouge detach request, as shown in Figure 4. This test case shows if it's possible to deactivate an EPS bearer by sending a modified detach request to the MME without knowledge of the subscriber. If the core network accepts the wrong detach message and deactivates the bearer, then the user may not be able to use the network services (Denial of Service). For this purpose, we generated a fake detach request with the GUTI and ID's of the subscriber and set the detach type to "switch off". The result was that the MME doesn't accept the fake request due to a wrong timestamp. Furthermore, the EPS bearer was still active, and no messages were sent.

The second attack was to check the UE attach procedure without integrity protection. This test case shows the impact of a wrong configured MME by executing the LTE Security procedure with a UE. The MME response to the attach request of the UE, after authentication, with the security mode command, as shown in Figure 3, which includes the NAS security algorithm to use for the whole connection [17]. In this test case the MME was forced to send this message not integrity protected (EIA0). The result was that the UE receives the not integrity protected message and sends a security mode reject with the cause #24 security mode rejected, unspecified. The MME receives the reject and abort the ongoing procedure. No connection was established.

We also tested the network with an attach flood attack by sending many such spoofed attack request from impersonating UEs. We used Blockmon to monitor the network to ensure it is able to identify the correct UEs from the fake ones using a flow check method. The implementation and results can be found in Sections 5 and 6.

B. IP based Threats

Due to the aspect of the completely IP-based core network, it includes now some security risks which are already known in other IP-based networks like the Internet. One possible attack is IP Spoofing, which can cause for instance fast draining of the UEs battery, abnormal charging in the payoff and can reduce the usability of the UEs, as accurately described in [7]. Another attack is gaining control of the cellular traffic accounting by spurious TCP retransmission [8].

Furthermore, it is possible to attack the LTE access network. As clearly shown in [9] it is possible to locate a specified subscriber in the network and attack the UE based on a rogue eNodeB to deny services or forcing the user to use lower secured networks.

A rogue eNodeB is not always necessary for DoS attacks to affect the availability of LTE mobile networks [10]. One of these possible attacks is an attach flood attack. In the LTE network it is possible for an attacker to send a lot of attach requests to the MME. It increases the network traffic drastically and forces the MME, due to the above descripted security mechanism in the attach procedure, to collect data for the entire number of requests. Based on the included identifier in the attach request, the MME and HSS must compute the keys and IDs. This additional work can cause an outage of the whole system.

5 Blockmon for Security Monitoring

We showed above that it is still possible to attack the LTE network, especially with flooding attacks. For this reason, it is important to monitor and analyse the traffic in mobile communication networks. In this section we talk about a tool called Blockmon, a high-performance network monitoring tool, which we extended to detect such flooding attacks in an LTE network. It is developed under the BSD license and is open-source [11].

Blockmon is modular, which means it is based on small executable blocks. For instance, one block is just for counting the number of packets that it receives from another block. The different blocks are interconnected using input and output gates and communicating through these gates with different types of “messages”. A set of blocks is called “composition” and is an XML file which allows configuring the order and the settings of the blocks. The core and the blocks itself are developed in C++. The whole system gets controlled by a python-based command line interface at runtime.

In the basic configuration there are already several blocks included, which are covering all the basic needs for a functional network monitoring tool. For instance, blocks for fast capturing of packets from a live network-interface or of a stored pcap-file (e.g. PcapSource, PFQSource), blocks for processing data (e.g. PacketCounter, FlowMeter) as well as blocks for exporting the collected data into a file or sending it to a server (e.g. IPFIXExporter, IDMEFExporter). Due to the fact that the blocks are designed in C++ it is simple to develop additional blocks and connect them to the already existing ones. No matter if completely from the scratch or importing an already existing functional C/C++ code like in [12].

To monitor such an attach flood attack we developed blocks for Blockmon which are able to extract certain information from captured packets in real live and to detect a possible flood attack. In this section we want to introduce all the implemented blocks and how they work together. We developed these as generic as possible to expand the functionality for other user cases as well. For instance, to use it with the detach procedure.

A. Overall Architecture of new Composition

To detect an attach flood attack it is necessary to monitor and analyse the messages which are flowing via the S1 interface from the UE to the MME. Especially the Attach procedure is of prime importance, shown in Figure 3.

The basic idea is firstly to extract uniquely the UEIDs and security properties of the captured packets to identify the UE and to collect information. After this we have to check for flaws in the flow order. For instance, missing or duplicated packets. Finally, we can make a measurement based on the previous collected information and decide if there is a possible flood attack or not.

By default, Blockmon only supports the transport protocols TCP (Transmission Control Protocol) and UDP (User Datagram Protocol). The transport protocol which is used in the LTE network on the S1 interface is SCTP (Stream Control Transmission Protocol). Due to this condition we also needed to implement a SCTP filter and dissector beside of the three new blocks (ValueExtractor, FlowControl and FloodDetection). An overview of the complete implementation is given in Figure 5 and is described as followed.

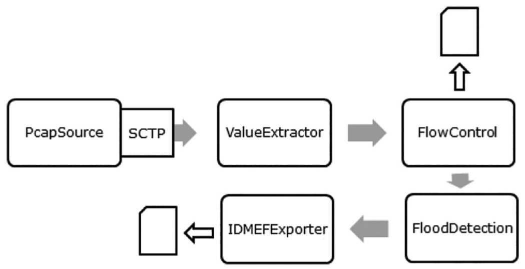


Figure 5 Implementation flood detection in Blockmon.

The already defined block “PcapSource” uses the implemented SCTP filter to forward only the SCTP packets which it sniffs. The next block “ValueExtractor” receives the SCTP packets, extract with the help of the SCTP dissector the payload, which contains the S1AP (S1 Application Protocol), collect the desired data and send the information to the next block, called “FlowControl”. This block receives the data and checks the correct order of the packets based on their identifier. It stores the results in a local log-file as well as forwards these to the next block. The “FloodDetection” block counts finally the number of flaws in a pre-set time interval and creates an alert if necessary. These alerts can either be stored in a local log-file or can be forwarded to the “IDMEFExporter” which creates an IDMEF (Intrusion Detection Message Exchange Format) [13] file. This can be stored local as well or can be forwarded to a specified server.

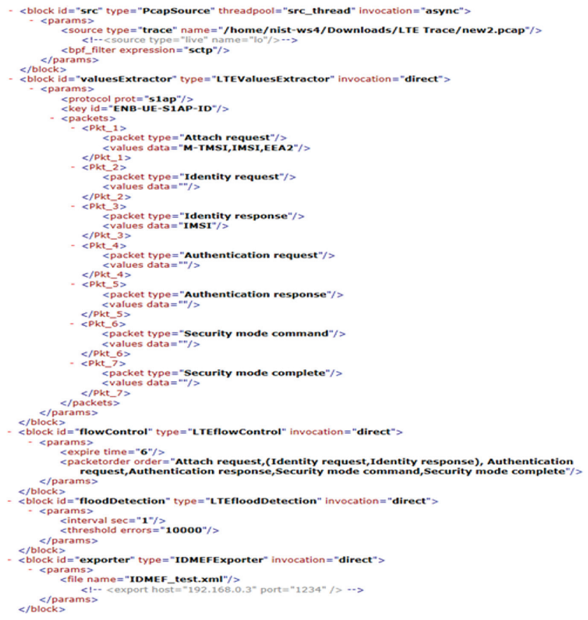
B. SCTP Filter and Dissector

As mentioned above, the SCTP Filter and Dissector are necessary to expand the functionality of Blockmon to support SCTP. We used the SCTP header-file of lksctp-tools [14]. Based on this main header-file we developed the dissector for Blockmon and implemented the functions to extract the payload with the LTE specified protocols (S1AP for instance) and all other important information. The filter can be added as an option by the pcap-source block, as shown in Figure 6.

C. Value Extractor

As shown in Figure 6, you can define several options for this block in the XML composition file. The first entry is for the LTE protocol type which will be used for decoding. In our case, for the attach procedure, it is S1AP, which already includes NAS-EPS. The second entry is for the identifier. This ID is used for mapping all the packets to the correct UE. The third part contains all the message types where we are interested in and which information we want to extract of the given message.

This block receives on its input gate an already implemented “Packet” message of the transport protocol type SCTP. It extracts the DATA-Chunk, if available, and checks if the payload contains the correct LTE protocol type. If it is correct it extracts the payload and decodes it in the next step. To make the block as generic as possible and due to the fact of no available open-source ASN.1 compiler, which can generate decoder for LTE protocols, we use external open-source programs. For decoding we use “tshark” which is a command-line network protocol analyser [15]. For calling “tshark” we need



```
- <block id="src" type="PcapSource" threadpool="src_thread" invocation="async">
  - <params>
      <source type="trace" name="/home/nist-ws4/Downloads/LTE Trace/new2.pcap"/>
        <!--<source type="live" name="lo"/>-->
      <bpf_filter expression="sctp"/>
    </params>
  </block>
- <block id="valuesExtractor" type="LTEValuesExtractor" invocation="direct">
  - <params>
      <protocol prot="s1ap"/>
      <key id="ENB-UE-S1AP-ID"/>
    - <packets>
      - <Pkt_1>
          <packet type="Attach request"/>
          <values data="M-TMSI,IMSI,EEA2"/>
        </Pkt_1>
      - <Pkt_2>
          <packet type="Identity request"/>
          <values data=""/>
        </Pkt_2>
      - <Pkt_3>
          <packet type="Identity response"/>
          <values data="IMSI"/>
        </Pkt_3>
      - <Pkt_4>
          <packet type="Authentication request"/>
          <values data=""/>
        </Pkt_4>
      - <Pkt_5>
          <packet type="Authentication response"/>
          <values data=""/>
        </Pkt_5>
      - <Pkt_6>
          <packet type="Security mode command"/>
          <values data=""/>
        </Pkt_6>
      - <Pkt_7>
          <packet type="Security mode complete"/>
          <values data=""/>
        </Pkt_7>
      </packets>
    </params>
  </block>
- <block id="flowControl" type="LTEflowControl" invocation="direct">
  - <params>
      <expire time="6"/>
      <packetorder order="Attach request,(Identity request,Identity response), Authentication
        request,Authentication response,Security mode command,Security mode complete"/>
    </params>
  </block>
- <block id="floodDetection" type="LTEfloodDetection" invocation="direct">
  - <params>
      <interval sec="1"/>
      <threshold errors="10000"/>
    </params>
  </block>
- <block id="exporter" type="IDMEFExporter" invocation="direct">
  - <params>
      <file name="IDMEF_test.xml"/>
        <!-- <export host="192.168.0.3" port="1234" /> -->
    </params>
  </block>
```

Figure 6 Composition Blockmon blocks.

a pcap-file which is generated with "text2pcap" [16]. It uses the stored hex values and pipe the result directly to "tshark". Both programs are called directly in the source code of the block. If the decoded message type is equal to one of the stored message types, then the block extracts the desired information and send it as an already implemented "TicketMsg", otherwise it will discard the packet. A "TicketMsg" is a map which stores the given identifier and the message type as Header ID. As mentioned before, all messages (Figure 3, packets (1)–(4), including the Identity request and response) are not ciphered. Based on this it is possible to extract the packet types. However, the "Security Mode Complete" (5) is ciphered with the new EPS-NAS-Algorithm and it is not possible to read the packet type. Indeed, in this message the flag for the "Security Header Type" is different compared to all other ciphered packets [17]. This allows us to detect the Security Mode Complete packet. The payload contains all the names of the information as keys with the corresponding value.

D. Flow Control

In the composition file an expire-time in seconds must be set, as shown in Figure 6. The timer is used to detect and delete flows with no response, because in an attach flood attack the attacker sends normally only an attach request and no response to the identity or authentication request. The standard time of every packet in the MME for EMM (EPS mobility management) is 6 seconds [17]. The next entry is the order of the packets for this block. Packets in the flow which are not always existing (for instance the identify request and response in the attach procedure) are surrounded with brackets.

This block receives a "TicketMsg" on its input gate and implements a state machine based on the given packet-order. For every message which arrives, it checks if the current captured message type is equal to the expected type for this ID. If it is correct, it stores the whole message with the current time and the next expected state in an unordered map. Otherwise it forwards the captured ticket with an error flag and error message to the output gate, stores it in an external log-file and sends this message to the next block. Every second, based on a periodic timer, it checks all stored entries in the map to detect flows where the EMM timer is expired. All discovered flows are getting deleted and send with a special error flag and message to the output gate and log-file.

E. Flood Detection

In the composition file, as shown in Figure 6, the options are a time interval for a periodic timer and a threshold, for the number of errors, before it will raise an alert.

This block receives the messages with the flow information and sends "Alert-messages" if necessary. An alert contains the unique ID, in our case the ENB-UE-S1AP-ID, the M-TMSI, the IP-Address of the eNodeB and the number of errors for this UE. Every message which arrives and where the error flag is set, gets stored and counted in the given time interval. Every time the periodic timer expires it checks if the number of all received errors is equal or higher than the threshold. In this case it will delete the stored information and send an Alert for every different ENB-UE-S1AP-ID. Otherwise it will just delete the data.

6 Results and Validation

In this section we demonstrate the functionality of the new blocks which we developed for Blockmon. The test cases are executed in our testbed [18] with a pcap-file as well as with real network flows. The Blockmon-master is running in our testbed on the same machine as the MME and the eNodeB and can capture the traffic of the S1 interface.

Test case 1 – Normal Attach

In the first test case we tested the correct attach procedure. Two different UEs which are doing the attach procedure are captured in a pcap-file. The first UE is unknown to the MME and receives the Identity request and sends the Identity response with his IMSI. The second UE is already known to the MME and consequently these two messages are skipped, like in Figure 3. The composition-file was the same as showed in Figure 6.

The value extractor block, decodes and reads all information correctly and the flow-order block detects the correct flow in both packets. We got two entries in our output file, for every UE one, with all the specified information. The second UE didn't send his IMSI value, because it was already known to the network. So, in the output file the value is empty. The flood detection block didn't raise an alert.

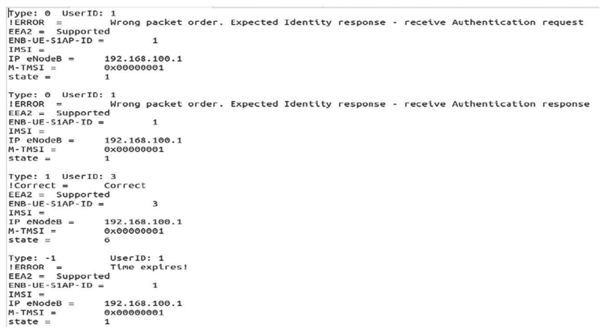
Test case 2 – Wrong Attach

In the second test case we tested a wrong attach procedure. We used the same pcap-file as in test case 1 (A). In the first case we changed the packet-order in the composition file to simulate a wrong order by the arriving packets. In this case, all two UEs were affected. In the second case we changed the order in the program by discarding some packets in the "ValueExtractor" block to simulate missing messages. We discarded only the Identity request and Identity response of the first UE.

In both cases it sends for every packet, which was unexpected (out of order), an "error" message with the error flag "Error – Out of Flow". In the first scenario both flows expired after 6 seconds due to the missing packets. In the second scenario only the first flow expired with the message "Error – Time expired" for the unfinished flow, as shown in Figure 7. Both behaviours are correct and expected.

Test case 3 – Flood detection in Attach

In the third test case we tested the detection of an attach flood attack. We used the same pcap-file as in test case 1 (A). We changed the behavior of the "ValueExtractor" block by inserting a loop which sends the same Attach request in an infinite loop. For test purposes we set the timer for the flow control to the standard value of 6 seconds. The time interval of the flood detection was 1 second and the threshold was 10000.



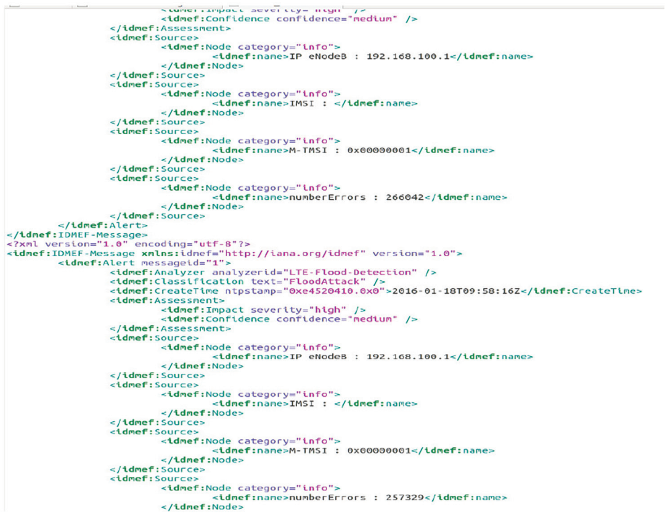
```
Type: 0  UserID: 1
!ERROR  =        Wrong packet order. Expected Identity response - receive Authentication request
EEA2 =  Supported
ENB-UE-S1AP-ID =        1
IMSI =
IP eNodeB =     192.168.100.1
M-TMSI =        0x00000001
state =         1

Type: 0  UserID: 1
!ERROR  =        Wrong packet order. Expected Identity response - receive Authentication response
EEA2 =  Supported
ENB-UE-S1AP-ID =        1
IMSI =
IP eNodeB =     192.168.100.1
M-TMSI =        0x00000001
state =         1

Type: 1  UserID: 3
!Correct =      Correct
EEA2 =  Supported
ENB-UE-S1AP-ID =        3
IMSI =
IP eNodeB =     192.168.100.1
M-TMSI =        0x00000001
state =         6

Type: -1         UserID: 1
!ERROR  =        Time expires!
EEA2 =  Supported
ENB-UE-S1AP-ID =        1
IMSI =
IP eNodeB =     192.168.100.1
M-TMSI =        0x00000001
state =         1
```

Figure 7 Output wrong attach procedure with missing packets.



```
                        <idmef:Impact severity="high" />
                        <idmef:Confidence confidence="medium" />
                </idmef:Assessment>
                <idmef:Source>
                        <idmef:Node category="info">
                                <idmef:name>IP eNodeB : 192.168.100.1</idmef:name>
                        </idmef:Node>
                </idmef:Source>
                <idmef:Source>
                        <idmef:Node category="info">
                                <idmef:name>IMSI : </idmef:name>
                        </idmef:Node>
                </idmef:Source>
                <idmef:Source>
                        <idmef:Node category="info">
                                <idmef:name>M-TMSI : 0x00000001</idmef:name>
                        </idmef:Node>
                </idmef:Source>
                <idmef:Source>
                        <idmef:Node category="info">
                                <idmef:name>numberErrors : 266042</idmef:name>
                        </idmef:Node>
                </idmef:Source>
        </idmef:Alert>
</idmef:IDMEF-Message>
<?xml version="1.0" encoding="utf-8"?>
<idmef:IDMEF-Message xmlns:idmef="http://iana.org/idmef" version="1.0">
        <idmef:Alert messageid="1">
                <idmef:Analyzer analyzerid="LTE-Flood-Detection" />
                <idmef:Classification text="FloodAttack" />
                <idmef:CreateTime ntpstamp="0xe4520410.0x0">2016-01-18T09:58:16Z</idmef:CreateTime>
                <idmef:Assessment>
                        <idmef:Impact severity="high" />
                        <idmef:Confidence confidence="medium" />
                </idmef:Assessment>
                <idmef:Source>
                        <idmef:Node category="info">
                                <idmef:name>IP eNodeB : 192.168.100.1</idmef:name>
                        </idmef:Node>
                </idmef:Source>
                <idmef:Source>
                        <idmef:Node category="info">
                                <idmef:name>IMSI : </idmef:name>
                        </idmef:Node>
                </idmef:Source>
                <idmef:Source>
                        <idmef:Node category="info">
                                <idmef:name>M-TMSI : 0x00000001</idmef:name>
                        </idmef:Node>
                </idmef:Source>
                <idmef:Source>
                        <idmef:Node category="info">
                                <idmef:name>numberErrors : 257329</idmef:name>
                        </idmef:Node>
```

Figure 8 Flood alerts.

We run the program for 20 seconds and received on average over 250.000 attach request per second. All these duplicated packets create an out-of-flow error and every six seconds a time-expired error. All these errors are counted by the flood-detection block which creates an alarm every second, as shown in Figure 8. In the displayed alert messages, the “messageid” is the unique ENB-UE-S1AP-ID.

7 Conclusion and Future Work

A. Conclusion

The LTE mobile communication network was developed by 3GPP to cope with the increasing number of subscribers and consequently with the increasing data traffic and the need of high-speed bandwidth. Furthermore, they implemented new security standards to fix the vulnerabilities of the older mobile network standards.

In this paper we presented a brief overview of the LTE system architecture and described the implemented security architecture as per 3GPP. We showed that there are still security vulnerabilities against the core and access network available. We have further introduced a program which we developed to monitor and analyze security relevant information and to detect a possible attach flood attack in the network. Finally, we showed some test cases to explain and prove the functionality of the developed program. This program is not limited to the current functionality. It can be used as a basis for more blocks and test scenarios.

B. Future Aspects

Moving forward we plan to develop more blocks for Blockmon to expand the functionality. For instance, to track the TAU (Tracking Area Update) procedure and the detach procedure.

References

[1] Murat Oğul et al.: "Practical Attacks on Mobile Cellular Networks and Possible Countermeasures", Future Internet 2013, 5, pp. 474–489.

[2] Ulrike Meyer et al.: "A Man-in-the-Middle Attack on UMTS", Proceedings of the 3rd ACM workshop on Wireless security, October 2004, pp. 90–97.

[3] 3rd Generation Partnership Project; Technical Specification Group Radio Access Network; Evolved Universal Terrestrial Radio Access (E-UTRA) and Evolved Universal Terrestrial Radio Access Network (E-UTRAN); Overall description; Stage 2 (Release 13) 3GPP TS 36.300 V13.1.0 (2015-09).

[4] 3rd Generation Partnership Project; Technical Specification Group Services and System Aspects; Network architecture (Release 13) 3GPP TS 23.002 V13.4.0 (2015-12).

[5] 3rd Generation Partnership Project; Technical Specification Group Services and System Aspects; 3GPP System Architecture Evolution (SAE); Security architecture (Release 13) 3GPP TS 33.401 V13.1.0 (2015-12).

[6] Jin Cao et al.: "A Survey on Security Aspects for LTE and LTE-A Networks", IEEE Communications Surveys & Tutorials, Vol. 16, No. 1, First Quarter 2014, pp. 283–302.

[7] Dong W. Kang et al.: "A Practical Attack on Mobile Data Network Using IP Spoofing"; Appl. Math. Inf. Sci. 7, No. 6; (2013); pp. 2345–2353.

[8] Younghwan Go et al.: "Gaining Control of Cellular Traffic Accounting by Spuriouse TCP Retransmission"; Conference: NDSS '14, 23–26. Februray 2014.

[9] Altaf Shaik et al.: "Practical attacks against privacy and availability in 4G/LTE mobile communication systems"; arXiv:1510.07563v1; 26 Oct 2015.

[10] Roger Piqueras Jover; "Security Attacks Against the Availability of LTE Mobility Networks: Overview and Research Directions"; WPMC 24–27 June 2013; pp. 1–9.

[11] Blockmon source code; https://github.com/blockmon/blockmon

[12] Maurizio Dusi et al.: "Blockmon: Flexible and High-Performance Big Data Stream Analytics Platform and its Use Cases", Nec Technical Journal Vol. 7 No. 2/2012, pp. 102–106.

[13] The Internet Engineering Task Force (IETF); The Intrusion Detection Message Exchange Format (IDMEF); RFC 4765; March 2007.

[14] SCTP; lksctp-tools; https://github.com/sctp/lksctp-tools

[15] Wireshark Foundation; https://wireshark.org/docs/man-pages/tshark.html

[16] Wireshark Foundation; https://www.wireshark.org/docs/man-pages/text2pcap.html

[17] 3rd Generation Partnership Project; Technical Specification Group Core Network and Terminals; Non-Access-Stratum (NAS) protocol for Evolved Packet System (EPS); Stage 3 (Release 13) 3GPP TS 24.301 V13.4.0 (2015-12).

[18] K. Jijo George et al.: "End-to-End Mobile Communication Security Testbed Using Open Source Applications in Virtual Environment", Journal of ICT Standardization, Vol. 3 Issue 1, July 2015, Article No. 4, pp. 67–90.

Biographies

Dominik Ernsberger received his B.Sc. degree in Computer Science from the University of Applied Sciences Munich, Germany in 2018. He did a six month internship in the NEC India Standardization (NIS) Team at NEC Technologies India Private Ltd. Chennai. His research interest includes Next Generation Networks, Mobile and Network Security as well as Digital Forensic. He is currently pursuing his M.Sc in Computer Science with Embedded Computing as major field of study at the University of Applied Sciences Munich, Germany.

K. Jijo George received his Bachelors in Computer Science and Engineering from Kurukshetra Institute of Technology and Management, India in 2011. He has over 4 years of experience in research and development of mobile communication networks. He worked as Research Engineer in NEC India Standardization (NIS) Team at NEC Technologies India Private Ltd. Chennai. Prior to joining NECI he was associated with IIIT, Bangalore as Research Associate in Context awareness in mobile applications. At NEC he worked on security aspects of telecom networks and testbed development of next generation mobile networks. His research interest includes Next Generation Networks, Mobile and Network Security and Telecom Security. He is currently pursuing his Masters in Cognitive Technical Systems in Albert Ludwigs University Freiburg, Germany.