Master's thesis

NTNU
Norwegian University of Science and Technology
Faculty of Information Technology and Electrical Engineering
Dept. of Information Security and Communication Technology

Julie Fylkesnes Halland

Public Key Cryptography for 5G Private Identification

Master's thesis in Communication Technology
Supervisor: Stig Frode Mjølsnes and Ruxandra-Florentina Olimid
June 2020

NTNU
Kunnskap for en bedre verden

Julie Fylkesnes Halland

Public Key Cryptography for 5G Private Identification

Master's thesis in Communication Technology
Supervisor: Stig Frode Mjølsnes and Ruxandra-Florentina Olimid
June 2020

Norwegian University of Science and Technology
Faculty of Information Technology and Electrical Engineering
Dept. of Information Security and Communication Technology

NTNU
Kunnskap for en bedre verden

Title: Public Key Cryptography for 5G Private Identification

Student: Julie Fylkesnes Halland

Problem description:

Privacy continues to be a main concern for mobile network users, although several proposed enhancements for the protection of the mobile user identifiers based on public key cryptography exist. Standards organisations have rejected the use of public key methods so far, but these are now finally accepted in 5G standards [1]. A feasibility study recently published on the usage of elliptic curve based cryptography computations for the paging procedure, claims that public key techniques are feasible [3].

This thesis examines the problem of private identification, and how this occurs in mobile systems [2]. In addition to this, the results claimed by Jiménez et al. [3] are validated, by reproducing the experiments and performance measurements. Lastly, if time allows, the feasibility measurements are extended to other cryptographic libraries that can be of interest and relevance.

References:

[1] 3GPP Technical Specification, TS 33.501: "Technical Specification Group Services and System Aspects; Security architecture and procedures for 5G system", Available online: https://portal.3gpp.org/desktopmodules/Specifications/SpecificationDetails.aspx?specificationId=3169
[2] Mjølsnes and Olimid. "Private Identification of Subscribers in Mobile Networks: Status and Challenges." IEEE Communications Magazine. Volume: 57, Issue: 9, September 2019.
[3] Jimenez, Enrique Cobo, et al. "Subscription identifier privacy in 5G systems." 2017 international conference on selected topics in mobile and wireless networking (MoWNeT). IEEE, 2017. https://ieeexplore.ieee.org/abstract/document/8045947

Responsible professor: Stig Frode Mjølsnes, IIK

Supervisor: Stig Frode Mjølsnes, IIK

Co-supervisor: Ruxandra-Florentina Olimid, IIK

Abstract

The next generation of mobile networks is called 5G. 5G introduces several improvements from the previous generation known as Long Term Evolution (LTE). Some of these improvements are related to subscriber privacy. A new protection scheme for the permanent subscriber identifier has been included. This identifier is known as the International Mobile Subscriber Identity (IMSI). This master's thesis investigates subscriber privacy in the new 5G network. This includes an experiment where the protection scheme for the IMSI is implemented in software, and its performance is tested on several mobile devices. The objective of the experiment is to replicate the work previously conducted by researchers at Ericsson Research. The results from the experiment confirm that the protection scheme is fast enough to be used in 5G. Some weaknesses with the scheme are discussed, and alternative schemes are presented. Other aspects of subscriber privacy are also discussed. This discussion uses examples of weaknesses from previous generations of mobile networks to highlight the improvements in 5G. Despite the improvements, 5G is still vulnerable to attacks such as downgrade attacks. In this type of attack, the device is forced to use older network generations, which opens up for other vulnerabilities. It may also be possible to track the location of a subscriber in some cases.

Sammendrag

Den neste generasjonen mobilnett blir kalt 5G. 5G introduserer flere forbedringer i forhold til den forrige generasjonen, kjent som Long Term Evolution (LTE). Noen av disse forbedringene gjelder personvernet til abonnentene. En ny løsning som beskytter den permanente abonnentidentifikatoren er inkludert. Denne identifikatoren er kjent som International Mobile Subscriber Identity (IMSI). Denne masteroppgaven undersøker personvernet til abonnenter i det nye 5G-nettet. Dette inkluderer et eksperiment der løsningen som beskytter IMSI blir implementert som programvare, og ytelsen blir testet på flere mobile enheter. Målet med eksperimentet er å replikere arbeidet som tidligere har blitt gjort av forskere fra Ericsson Research. Resultatene fra eksperimentet bekrefter at beskyttelsesløsningen er rask nok til å brukes i 5G. Noen svakheter ved løsningen blir diskutert, og alternative løsninger presenteres. Andre aspekter ved personvernet til abonnentene blir også diskutert. Denne diskusjonen bruker eksempler på svakheter fra tidligere generasjoner av mobile nettverk for å fremheve forbedringer som er gjort i 5G. Til tross for disse forbedringene er 5G fortsatt utsatt for angrep, slik som nedgraderingsangrep. Denne angrepstypen innebærer at mobilenheten blir tvunget til å bruke eldre nettverksgenerasjoner, noe som åpner opp for flere svakheter. Det er også mulig å spore lokasjonen til abonnenter i enkelte tilfeller.

Preface

This Master's thesis is written as part of the 5-year master programme in Communication Technology at The Department of Information Security and Communication Technology (IIK) at the Norwegian University of Science and Technology (NTNU).

Thank you to Kim Aksel Tahuil Borgen, Arild Sørensen Dalsgård and Hanna Fylkesnes for lending me your Android devices throughout the semester to help me with my experiments. Thank you to Yan Bekkemoen, Nora Hobæk Hovland and Sigrid Andersen Syverud for additional help with the experiments.

A sincere thank you to my supervisors Stig Frode Mjølsnes and Ruxandra-Florentina Olimid for guiding me through this process.

Contents

List of Figures

List of Tables

Acronyms

5G-GUTI 5G Globally Unique Temporary Identifier.

5G-TMSI 5G Temporary Mobile Subscription Identity.

ABE Attribute-Based Encryption.

adb Android Debug Bridge.

AES Advanced Encryption Standard.

AKA Authentication and Key Agreement.

AMF Access and Mobility Management Function.

AUSF Authentication Server Function.

CPU Central Processing Unit.

CTR Counter.

DoS Denial of Service.

DRX Discontinuous Reception.

eBACS ECRYPT Benchmarking of Cryptographic Systems.

ECCDH Elliptic Curve Cofactor Diffie-Hellman.

ECDH Elliptic Curve Diffie-Hellman.

ECIES Elliptic Curve Integrated Encryption Scheme.

EPS Evolved Packet System.

GSM Global System for Mobile Communications.

GUAMI Globally Unique AMF Identifier.

HN Home Network.

HSS Home Subscriber Server.

ICB Initial Counter Block.

IMSI International Mobile Subscription Identity.

IoT Internet of Things.

IP Internet Protocol.

KA Key Agreement.

KDF Key Derivation Function.

KG Key Generation.

LI Lawful Interception.

LTE Long Term Evolution.

MAC Message Authentication Code.

MCC Mobile Country Code.

MME Mobility Management Entity.

MNC Mobile Network Code.

MSIN Mobile Subscription Identification Number.

NAI Network Access Identifier.

NF Network Function.

NG-RAN Next Generation Radio Access Network.

PIP Private Identification Protocol.

PKI Public Key Infrastructure.

PO Paging Occasion.

SAE System Architecture Evolution.

SHA Secure Hash Algorithm.

SIDF Subscriber Identity De-concealing Function.

SN Serving Network.

SUCI Subscription Concealed Identifier.

SUPI Subscription Permanent Identifier.

TMSI Temporary Mobile Subscription Identity.

UDM Unified Data Management.

UDR Unified Data Repository.

UE User Equipment.

UICC Universal Integrated Circuit Card.

UMTS Universal Mobile Telecommunications System.

USIM Universal Subscriber Identity Module.

USRP Universal Software Radio Peripheral.

Chapter 1
Introduction

The first version of the 5G specification, known as 3GPP Release 15 [2], was finalised last year, and the first public 5G networks have recently been deployed. 5G offers technological enhancements that makes it useful in a wide variety of areas, such as Internet of Things (IoT), smart cities, and healthcare [10]. All of the new applications of 5G make privacy increasingly more important, and subscriber privacy has been one of the focus areas in the development of the specification [14]. Before diving into subscriber privacy in 5G, the following section aims to give an overview of the previous generations of mobile networks.

1.1 Evolution of Mobile Networks

Mobile networks have existed for around 40 years, with a new generation for each decade. These are introduced in this section.

1.1.1 First Generation

The first generation of mobile networks was introduced in the early 1980s. This generation used analog technology, and the implementations varied between countries due to the lack of international standardisation. Common for all implementations was that the security was poor [27]. They were vulnerable to eavesdropping, and an adversary could easily gather information such as the mobile device identifier and the telephone number. This lead to an issue known as cloning, where an adversary could program their phone to become a "cloned" version of another phone [39]. With the cloned phone, the adversary could use services of the network, and make the legitimate phone owner pay for it [27].

1.1.2 Second Generation

In the early 1990s, Global System for Mobile Communications (GSM) was introduced. GSM was a great improvement to the first generation of mobile networks, and it

has been deployed across the globe. This was the first digital mobile network, which made it possible to secure the transmission channel using advanced cryptographic techniques. GSM greatly improved the subscriber privacy [27].

GSM introduced the SIM card, which stored the subscription identifier known as the International Mobile Subscription Identity (IMSI) and a permanent key [1]. The SIM was one of the main components of GSM's security architecture, and it was both tamper resistant and portable. The SIM made it possible for the network to authenticate the subscriber. To perform the authentication, the network sent the subscriber a challenge, and the subscriber responded to this challenge using their permanent key to prove their identity. The authentication process also generated a session key, which was used to encrypt the radio interface [27].

As already mentioned, GSM used IMSI to identify the subscribers. Before authenticating the subscribers, the IMSI had to be sent to the network to identify them. During identification, it was possible for an attacker to eavesdrop on the communication and track the location of the subscriber with their IMSI [62]. To prevent further tracking of the subscribers, they were assigned a temporary identifier after authentication. This identifier is known as the Temporary Mobile Subscription Identity (TMSI), and it was updated regularly [4].

GSM had a robust security architecture, and several elements have been reused by the later generations of mobile networks. Despite this, there were several security issues. The algorithms used to encrypt the communication were poor [22]. Additionally, it was possible for an attacker to set up false base stations that users would unknowingly connect to. Such false base stations could be used to gather IMSIs from several users within an area. This is known as IMSI catching, and the first IMSI catcher was presented in 1996 [62]. Variants of this issue has remained for decades later [27].

1.1.3 Third Generation

The third generation of mobile networks, known as the Universal Mobile Telecommunications System (UMTS), was released in the early 2000s. UMTS was based on the security architecture of GSM, with several improvements. One of the areas that saw improvements was the issue of false base stations [27]. Several protection mechanisms were introduced to combat these. One of these was to integrity protect signalling traffic. Additionally, protection against replay attacks on the Authentication and Key Agreement (AKA) procedure was added. This was done by including a sequence number in the procedure, as well as protecting the integrity with a MAC tag [9]. While these measures were effective against eavesdropping, active attacks, such as IMSI catching, were still possible [38]. The UMTS standards organisation considered encrypting the IMSI during subscriber identification, but concluded that both sym-

metric key and public key techniques were unsuited at the time. With symmetric cryptography, there was a risk that legitimate users could in some cases be shut out of the network. Public key cryptography was too expensive [27]. Mutual authentication was also added with UMTS, and was effective against false base stations. It did not prevent IMSI catching, however. This is because authentication happened after identification, so the IMSI would already have been sent to the network [62]. UMTS also used stronger encryption algorithms than GSM, due to liberalisation of the export restrictions [27].

1.1.4 Fourth Generation

Long Term Evolution (LTE) was introduced around 2010. This fourth generation of mobile networks saw several changes in the architecture [27]. The core network was Internet Protocol (IP) based, which introduced a new set of vulnerabilities [39]. Several concepts and elements from the previous generations were reused, but were adapted to the updated architecture [27]. LTE strengthened the signalling protocols, and required authentication and encryption in more situations [58]. It also introduced new cryptographic algorithms, and a new key structure [39].

It should be noted that it is slightly inaccurate to refer to the fourth generation as "LTE". LTE originally refers to the new radio technology introduced in the system, while SAE/LTE refers to the system as a whole. SAE stands for System Architecture Evolution. Evolved Packet System (EPS) is the technical term for the system. However, LTE has become the brand name of the system, and is most used today [27]. We talk more about LTE later in the thesis.

1.2 Motivation and Objectives

As previously mentioned, subscriber privacy has been prioritised in 5G. Several of the privacy issues in LTE have been mitigated, as we will see in the following chapters. One of the major improvements in this regard is the introduction of a protection scheme for the permanent subscription identifier IMSI. The current specification includes the Elliptic Curve Integrated Encryption Scheme (ECIES) as the protection scheme [11]. This is a public key encryption scheme based on elliptic curve cryptography.

In 2017, researchers Jiménez et al. at Ericsson Research implemented ECIES and tested its performance [31]. They wrote several applications using two popular cryptographic libraries, and measured how fast each cryptographic primitive was executed on several Android devices. Their results were good, and they concluded that ECIES was a feasible method for encrypting the IMSI in a mobile network setting. More information about this study can be found in Section 3.2.

This thesis is motivated by the recent release of the first 5G specification [2], and the improvements it introduces with regards to subscriber privacy. Special attention is drawn towards the use of ECIES to encrypt the IMSI, as leakage of the permanent identifier was called a key issue in the early development of 5G [14]. The thesis therefore includes a replication of Jiménez et al.'s ECIES performance studies, and my results are compared to theirs. The thesis is a continuation of the specialisation project [28].

To summarise, the objectives for this thesis are:

- Review subscriber privacy in 5G
- Replicate ECIES performance studies

1.3 Changes from Original Problem Description

The plan for the thesis included to replicate Jiménez et al.'s ECIES performance measurements [31] using the same cryptographic libraries, i.e., OpenSSL and Nettle. Due to lack of experience with low level programming and cross-compilation, several adjustments were made to meet the deadline for the thesis. The scope for the OpenSSL implementations were adjusted to include the parts of ECIES that were measured to be the most computationally expensive in Jiménez et al.'s studies. With regards to the Nettle implementations, some attempts were made, but it was soon deemed too difficult to continue, as it would require proficiency in C programming. This has not been included in the master programme. When contacting Jiménez et al. about their implementations, they were not willing to disclose them, as the code is proprietary to Ericsson Research.

After replicating the studies, the measurements would then be extended to include other libraries as well. Some attempts were made, but due to some initial difficulties and limited time, this objective was abandoned.

1.3.1 Description of the Work

This thesis includes a replication of the experimental results presented by Jiménez et al. [31], using the OpenSSL cryptographic library. The performance measurements in this study are conducted on more recent hardware, that better reflect the mobile devices that are currently available. This has likely contributed to the better results achieved in this study. The study also offers full transparency of the methods used to achieve the results to make them easily replicable. In addition to the experimental studies, a review of subscriber privacy in 5G is conducted.

1.4 Outline

The thesis is structured as follows. Chapter 2 provides background for the thesis. Chapter 3 introduces the related work. In Chapter 4, the method for the thesis is explained, before the results are presented and discussed in Chapter 5. Chapter 6 concludes the thesis and proposes future research directions.

Chapter 2
Background

2.1 The 5G System

The 5G System, as defined by 3GPP, consists of three main parts. These are the access network, the core network, and the User Equipment (UE) [15]. In this section, we give a brief introduction to the 5G System and its architecture, focusing on the aspects that are most relevant to subscriber privacy. We refer to the 5G System simply as "5G" for the rest of the thesis.

For more details on 5G, see TS 23.501 [15] for a description of the architecture. The security aspects of 5G are described in TS 33.501 [11].

2.1.1 User Equipment

The User Equipment (UE) is the device that a user uses to connect to the network. This is commonly a mobile phone, but it can also be any IoT device. The UE communicates with the network over the radio interface [16].

The UE uses the Universal Subscriber Identity Module (USIM) to access services from the network [16]. This is an application that resides on the Universal Integrated Circuit Card (UICC), which is often referred to as the SIM card. The USIM stores the subscription credentials that are used in the identification and authentication procedures between the UE and the network [11]. This includes the public key of the Home Network (HN).

Each USIM is identified with a permanent identifier. In 5G, this is called the Subscription Permanent Identifier (SUPI), and in most cases this is identical to the earlier generations of mobile network's permanent identifier IMSI. In addition to this, the UE shall also support the temporary identifier 5G-GUTI [11]. Read more about the identifiers in 5G in Section 2.2.

2.1.2 Network Functions

The 5G core network consists of several Network Functions (NF), where each NF has a defined functional behaviour [15]. Some of the NFs that are most relevant to subscriber privacy are described in this section. An overview of the relevant NFs are shown in Figure 2.1.

Access and Mobility Management Function

The Access and Mobility Management Function (AMF) is responsible for managing the connection to the UE, amongst other things. It handles tasks such as initial registration of the UE to the network, and (re-)allocation of temporary identifiers to the UE [15]. The AMF is the entry point to the core network of 5G, and it may deny services to the UE if it is not able to confirm its identity [11]. It resembles LTE's Mobility Management Entity (MME).

Unified Data Management

The Unified Data Management (UDM) is the subscriber database in the 5G core network. It is located in the home network of the subscriber, and it stores subscription and authentication data in the Unified Data Repository (UDR) [15].

The Subscriber Identity De-concealing Function (SIDF) is a service that is part of the UDM. It is responsible for de-concealment of the SUCI to the SUPI, which it decrypts using the home network's private key [11]. Read more about the SUPI and SUCI identifiers in Section 2.2.

Authentication Server Function

The Authentication Server Function (AUSF) is responsible for authentication between the UE and the network. The AUSF, together with the UDM, resemble LTE's Home Subscriber Server (HSS).

2.2 Private Identifiers

There are many private identifiers in mobile networks. This section gives an overview of the identifiers that are most relevant to subscriber privacy.

2.2.1 Subscription Permanent Identifier

All subscribers in 5G have a permanent subscription identifier called the Subscription Permanent Identifier (SUPI). This identifier is independent from the identifier of the UE, and it is stored on the USIM. The SUPI can have one of two formats. It can either be an IMSI, or the network specific identifier called the Network Access

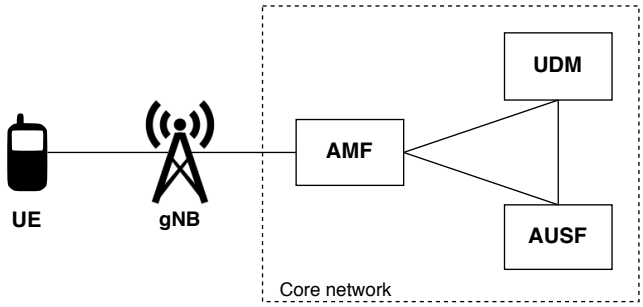

Figure 2.1: Part of the 5G architecture. Note that only a few of the network functions in the core network are shown. See TS 23.501 [15] for the complete architecture.

Identifier (NAI). The latter is used in private networks, but we only consider IMSI here. The SUPI shall not be transmitted in clear text between the UE and the network [15]. However, certain special cases are excluded from this requirement. See Section 2.2.2 for a list of these cases.

International Mobile Subscription Identity

The International Mobile Subscription Identity (IMSI) is a globally unique number used to identify a mobile subscription. This number is stored in the USIM of the UE, as well as in the UDM [11]. The IMSI in 5G is the same as the IMSI in earlier generations of mobile networks.

The IMSI is composed of the 3 digit Mobile Country Code (MCC) and the 2-3 digit Mobile Network Code (MNC). These identify the home country and the home network of the subscription respectively, and are used for roaming purposes. The remaining part of the IMSI is the 9-10 digit Mobile Subscription Identification Number (MSIN), which identifies the subscription itself [4]. Figure 2.2 illustrates the IMSI format.

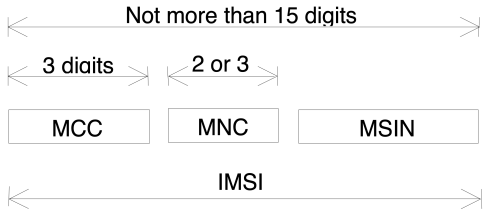

Figure 2.2: The IMSI format [4].

2.2.2 Subscription Concealed Identifier

The Subscription Concealed Identifier (SUCI) contains the concealed SUPI to preserve privacy, and it is described in TS 33.501 [11]. The format of the SUCI can be seen in Figure 2.3. The concealed SUPI is located in the Scheme Output part of the SUCI.

The SUCI is generated by using one of the protection schemes specified in TS 33.501 [11], or a proprietary protection scheme specified by the HN. In the first case, there are currently two options available. Both are variants of ECIES (see Section 2.4), and take the MSIN as input. The Scheme Output consists of the ECC ephemeral public key, the encrypted MSIN and a MAC tag (see Figure 2.4). To generate the SUCI, the UE uses the HN's public key, which should be securely provisioned by the HN [11]. The SUCI is generated by the USIM or the ME.

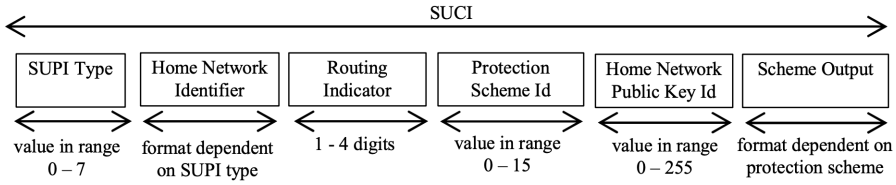

Figure 2.3: The SUCI format [4].

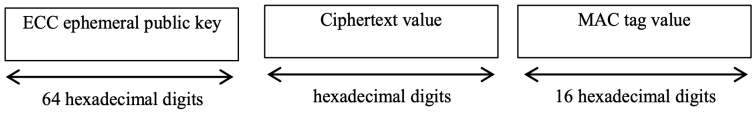

Figure 2.4: Scheme Output of using ECIES Profile A to encrypt the IMSI. This is the last field of the SUCI. Note that when ECIES Profile B is used, the ECC ephemeral public key is 66 hexadecimal digits [4].

The SUCI is only included in the following 5G messages from the UE to the network [11]:

- In the Initial Registration message, which is a type of Registration Request, if the UE has not yet been assigned a 5G-GUTI.
- In the Identity Response message, which is a response to the Identity Request message.
- If the UE sends a De-Registration Request message during the initial registration procedure where the UE did not receive a registration accept message with the 5G-GUTI. In that case, the UE should include the same SUCI as was used in the initial registration.

The UE can in some specified cases generate the SUCI by using the null scheme. This scheme does not offer any protection, and simply returns the clear text SUPI. Generating SUCI with the null scheme is only allowed in the following cases [11]:

- If the UE wants to initiate an unauthenticated emergency session, and does not have a 5G-GUTI in the network.
- If the HN specifies that the null scheme is to be used.
- If the UE has not been provisioned with the HN's public key.

2.2.3 5G Globally Unique Temporary Identifier

The 5G Globally Unique Temporary Identifier (5G-GUTI) is the temporary identifier for the subscription. It is constructed from the Globally Unique AMF Identifier (GUAMI) and the 5G Temporary Mobile Subscription Identity (5G-TMSI). The GUAMI identifies the relevant AMF and network for routing purposes, and the 5G-TMSI identifies the subscriber. The 5G-GUTI is assigned to the UE by the AMF after the security context has been established, and is refreshed in the following situations [11]:

- After the UE sends a Registration Request message of the type Initial Registration
- After the UE sends a Registration Request message of the type Mobility Registration
- When the UE responds to a Paging message with a Service Request message

The specification also recommends that the 5G-GUTI is refreshed in the following situation [11]:

- After the UE sends a Registration Request message of the type Periodic Registration Update

Temporary Mobile Subscription Identity

Variants of the TMSI have been used since GSM, and its purpose is to unambiguously identify the subscriber without revealing its permanent identifier. It is local to the geographical area where the subscriber is located. TMSI reallocation should not occur until after the security context has been set up. This is to prevent it from being sent in clear text [4].

In 5G, the TMSI is known as the 5G-TMSI. The specification requires it to be generated as an unpredictable identifier [11]. The 5G-S-TMSI is a shortened form of the GUTI, and it enables more efficient radio signalling procedures, such as Paging [15]. It consists of parts of the GUAMI, as well as the 5G-TMSI [4].

2.3 Cryptographic Primitives in ECIES

ECIES, the encryption scheme used to encrypt the IMSI in 5G, consists of several cryptographic primitives. Each primitive has certain cryptographic properties, and this section introduces these primitives, and explains their purpose in ECIES. Information about the primitives was gathered from Stallings' book *Cryptography and Network Security* [61] and the relevant standards. The explanations of the primitives are generally simplified, and the reader should refer to the standards for more accurate descriptions.

2.3.1 Elliptic Curve Cryptography

Elliptic Curve Cryptography (ECC) is a form of public key cryptography that uses elliptic curves over finite fields, and it has been standardised in SECG SEC 1 [64] and NIST SP 800-56A [20].

A finite field $\mathbb{F}_p$ is a set of p elements, together with the arithmetic operations addition and multiplication, as well as their inverse operations, subtraction and division. Elliptic curves are defined by the set of solutions $P = (x, y)$ to their defining equation, together with the point at infinity $\mathcal{O}$ [64]. Figure 2.5 shows a visual representation of an elliptic curve.

An elliptic curve is described by its domain parameters, which are specified by the tuple $T = (p, a, b, G, n, h)$. p specifies the finite field. $a, b \in \mathbb{F}_p$ are parameters of the defining equation specifying the elliptic curve. $G = (x_G, y_G)$ is the base point of the curve, and this is also known as the generator. The prime n is the order of G, while the integer h is the cofactor [64]. A set of domain parameters can be reused to generate multiple keys over an extended period [20].

Key Generation

A public key in elliptic curve cryptography is a point on an elliptic curve [64]. An ECC key pair can be generated as follows [20]:

1. Choose the private key $d \in [1, n-1]$. d should be selected as a random integer on the interval.

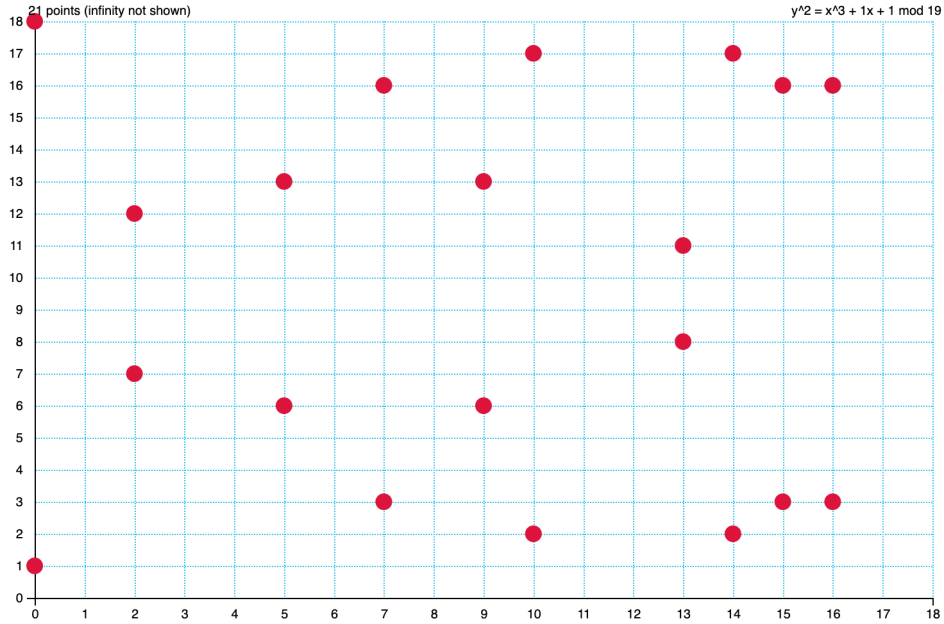

Figure 2.5: Visual representation of the elliptic curve defined by the equation $y^2 = x^3 + x + 1 \bmod 19$.[1]

2. Generate the public key $Q = (x, y) = dG$, where G is the base point of the curve
3. Output the key pair (d, Q)

For cryptographically strong elliptic curves, it is computationally infeasible to find the private key d given the public key Q and the generator G with traditional computers. This is known as the elliptic curve discrete logarithm problem, which forms the basis for elliptic curve cryptography [61].

Named Elliptic Curves

While elliptic curves can be described by their domain parameters, this can sometimes be impractical. Therefore, some of the curves that are commonly used have been standardised and given shorter names, and are usually referred to by their names instead. The two named curves secp256r1 and Curve25519 are used with 5G's ECIES:

– secp256r1 is a named curve specified in SECG SEC 2 [65]. This is a 256-bit elliptic curve over a prime finite field $\mathbb{F}_p$, where p is an odd prime. The curve

[1] Generated with this website: https://graui.de/code/elliptic2/

was initially specified by NIST as Curve P-256 [49]. It is also known by the name prime256v1.

– Curve25519 is a popular alternative to the NIST curves, and it is specified in RFC 7748 [41]. It is designed to perform fast execution of ECDH [21].

2.3.2 Elliptic Curve Diffie-Hellman

ECIES uses Elliptic Curve Diffie-Hellman (ECDH) for key agreement. ECDH is a cryptographic primitive that allows two parties to agree on a shared secret that corresponds to both of their elliptic curve keys, and it is described in SECG SEC 1 [64] and NIST SP 800-56A [20]. The secret value derived with ECDH is never transmitted over the communication channel, making it suitable for key agreement over insecure channels. Before deriving the secret value, the two parties need to share their public keys Q with each other. These keys are generated with elliptic curve cryptography, where both parties use the same domain parameters. Each party can then derive the secret as follows:

1. Calculate the point $P = (x, y) = dQ$, where d is its private key, and Q is the other party's public key.
2. If $P = \mathcal{O}$, the point is invalid
3. If $P \neq \mathcal{O}$, set $Z = x$
4. Output the shared secret Z

Elliptic Curve Cofactor Diffie-Hellman (ECCDH) is a variant of ECDH where the cofactor h is also used to derive the shared secret value [64]. In ECCDH, $P = (x, y) = hdQ$, where $h \neq 1$. If $h = 1$, this primitive is the same as the ECDH primitive. ECCDH provides protection against certain attacks, such as small subgroup attacks [64].

X25519 is another variant of ECDH that can be used with Curve25519. This variant has some computational advantages compared to regular ECDH, making it a faster alternative.

2.3.3 ANSI X9.63 Key Derivation Function

The ANSI X9.63 Key Derivation Function (KDF) was specified by ANSI in 2001 [57]. It uses a hash function to derive keying data to be used with the encryption and MAC functions. The derived key is based on the shared secret value from the key agreement function [64].

2.3.4 Advanced Encryption Standard

The Advanced Encryption Standard (AES) is a symmetric block cipher, and it is specified in NIST FIPS 197 [50]. It is a strong encryption method that runs efficiently on a wide range of processors.

AES operates on blocks of data in several rounds. The block size is 128 bits, meaning that it takes 128 bits of data to encrypt as input, and gives 128 bits of encrypted data as output. There are three possible key sizes of 128, 192 and 256 bits, where a larger key gives a stronger encryption. In 5G, AES with a 128 bit key, known as AES-128, is used with ECIES. This variant uses 10 rounds to convert the plaintext input to the encrypted output, as well as an initial round where parts of the key is added. Each round consists of one or several stages that alter the input. AES-128 works as follows:

1. Input the 128 bit plaintext and 128 bit key
2. Expand the key to eleven 128 bit round keys
3. In the initial round, only the AddRoundKey stage is performed, where the current block is bitwise XORed with the round key
4. For the next nine rounds, the following four stages are performed:
    a) SubBytes: Substitutes the bytes in a block by using a fixed lookup table called the S-box
    b) ShiftRows: Shifts each row by a certain offset
    c) MixColumns: Substitutes each column by performing a matrix multiplication of the column with a fixed matrix
    d) AddRoundKey
5. The final round consists of the three stages SubBytes, ShiftRows and AddRoundKey
6. Output the 128 bit ciphertext

Decryption with AES is similar to the encryption process, an can be thought of as the inverse operation. Since it is a symmetric cipher, the same key is used for both encryption and decryption.

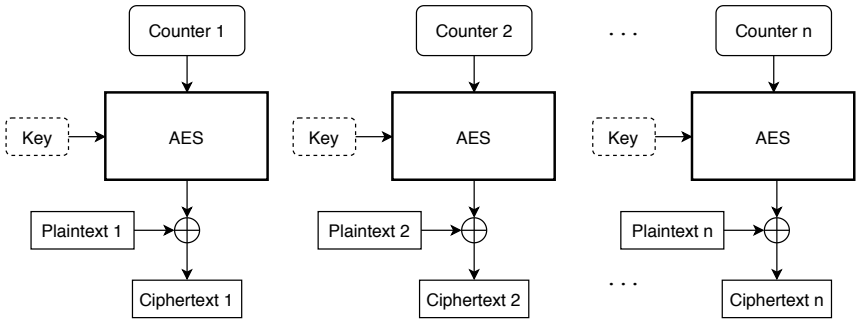

Figure 2.6: AES in counter mode.

Counter Mode

5G uses AES-128-CTR, which means AES in Counter (CTR) mode with a 128 bit key. CTR mode is one of several modes of operation defined in NIST SP 800-38A [25]. A mode of operation is an algorithm that can be used with a block cipher to extend it to several blocks, as a block cipher in itself is only suited to encrypt data that does not exceed its block size. Several vulnerabilities arise when the same key is used to encrypt multiple blocks of data with a block cipher [61].

CTR mode uses unique counters per block, and it works by encrypting the counter with the block cipher, before XORing the result with the plaintext to produce the ciphertext. The same counter should not be used twice with the same key, as this will weaken the security. In 5G, the Initial Counter Block (ICB) is set to parts of the ephemeral encryption key generated with the key derivation function. The counter is then incremented using the standard incrementing function described in Appendix B.1 of NIST SP 800-38A [11, 25].

To encrypt message P with CTR mode, calculate $C = P \oplus E(K, T)$, where C is the resulting ciphertext, E is the block cipher, K is the key, and T is the counter. $\oplus$ is the bitwise XOR operation. To decrypt the ciphertext, calculate $P = C \oplus E(K, T)$. See Figure 2.6.

2.3.5 Secure Hash Algorithm

SHA-256 is a hash function that is part of the Secure Hash Algorithm (SHA) family, and it is specified in NIST FIPS 180 [48]. It is used to construct a condensed representation of the input data, and the output is called a message digest. In SHA-256, the message digest is 256 bits. SHA-256, as other good hash functions, has the following properties [24]:

- **Collision resistance:** It should be infeasible to find two different inputs that return the same output
- **One-way property:** It should be infeasible to find the input that relates to a given output
- **Second preimage resistance:** Given an input, it should be infeasible to find another input that results in the same output

Message digests are often compared to fingerprints, and they are used to assure the integrity of data [48]. Instead of transferring only the message, the hashed value gets appended to the message. The recipient can then re-calculate the hash of the message and compare it to the received message digest. If they are not equal, the recipient knows that either the message or the hash value has been altered during transmission [61].

2.3.6 HMAC

HMAC is a hash-based Message Authentication Code (MAC), and it is described in RFC 2104 [40]. HMAC provides authentication of messages. To do this, it uses a hash function H to generate a message digest of a message m, together with a MAC key K and the two paddings opad and ipad. These paddings are set to specific values. In 5G, SHA-256 is used as the hash function, and this variant is called HMAC-SHA-256. It works as follows:

$$\mathrm{HMAC}(K, m) = \mathrm{H}\Big((K \oplus opad) \,\|\, \mathrm{H}\big((K \oplus ipad) \,\|\, m\big)\Big)$$

2.4 Elliptic Curve Integrated Encryption Scheme

The Elliptic Curve Integrated Encryption Scheme (ECIES) has been proposed as a solution to the private identification problem, and it is included in 3GPP's security specification for 5G [12]. In this section, we see how ECIES works, and how it can be applied in 5G.

2.4.1 Architecture

ECIES consists of several cryptographic functions. The Key Generation (KG) function generates the UE's ephemeral public-private key pair. The HN also has a public-private key pair, but this is not ephemeral. The public key of the HN is pre-provisioned to the user on the USIM, while the private key is securely stored in the core network. The UE's private key and the HN's public key are inputs to the Key Agreement (KA) function, which outputs the ephemeral shared key. Both the

Table 2.1: ECIES parameters in 5G [11].

	Profile A	**Profile B**
EC domain parameters	Curve25519	secp256r1
ECDH primitive	X25519	ECCDH primitive
Point compression	N/A	True
Key derivation function	ANSI-X9.63-KDF	ANSI-X9.63-KDF
Hash function	SHA-256	SHA-256
MAC function	HMAC-SHA-256	HMAC-SHA-256
MAC key length	32 octets (256 bits)	32 octets (256 bits)
MAC length	8 octets (64 bits)	8 octets (64 bits)
Encryption function	AES-128-CTR	AES-128-CTR
Encryption key length	16 octets (128 bits)	16 octets (128 bits)
ICB length	16 octets (128 bits)	16 octets (128 bits)

KG and KA functions use elliptic curve cryptography, and ECDH is used in KA. The ephemeral shared key is then used in the Key Derivation Function (KDF) to derive the keying data, using ANSI-X9.63-KDF. In 5G, the keying data is a concatenation of the encryption key, the ICB, and the MAC key [11]. The MSIN is encrypted with AES-128-CTR, using the encryption key and the ICB. Next, the MAC tag is calculated with HMAC-SHA-256 on the encrypted MSIN, using the MAC key. The output of the encryption scheme is a concatenation of the UE's ephemeral public key, the encrypted MSIN, and a MAC tag. This is known as the Scheme Output [64].

Decryption with ECIES is similar to encryption. To decrypt the MSIN in the HN, the UE's public key is used as input to the KA, along with the HN's private key. The output is the ephemeral shared key, and this has the same value as in the encryption process. It is used to derive the keying data. The MAC tag is verified with the MAC key, and the MSIN is decrypted with the encryption key and the ICB [11, 64].

An overview of the scheme can be seen in Figure 2.7.

2.4.2 Usage in 5G

Implementations of ECIES for use in 5G should adhere to SECG's specification of ECIES [11, 64, 65]. There are currently three protection schemes specified in the security specification of 5G. The first is the null scheme, which does not provide any

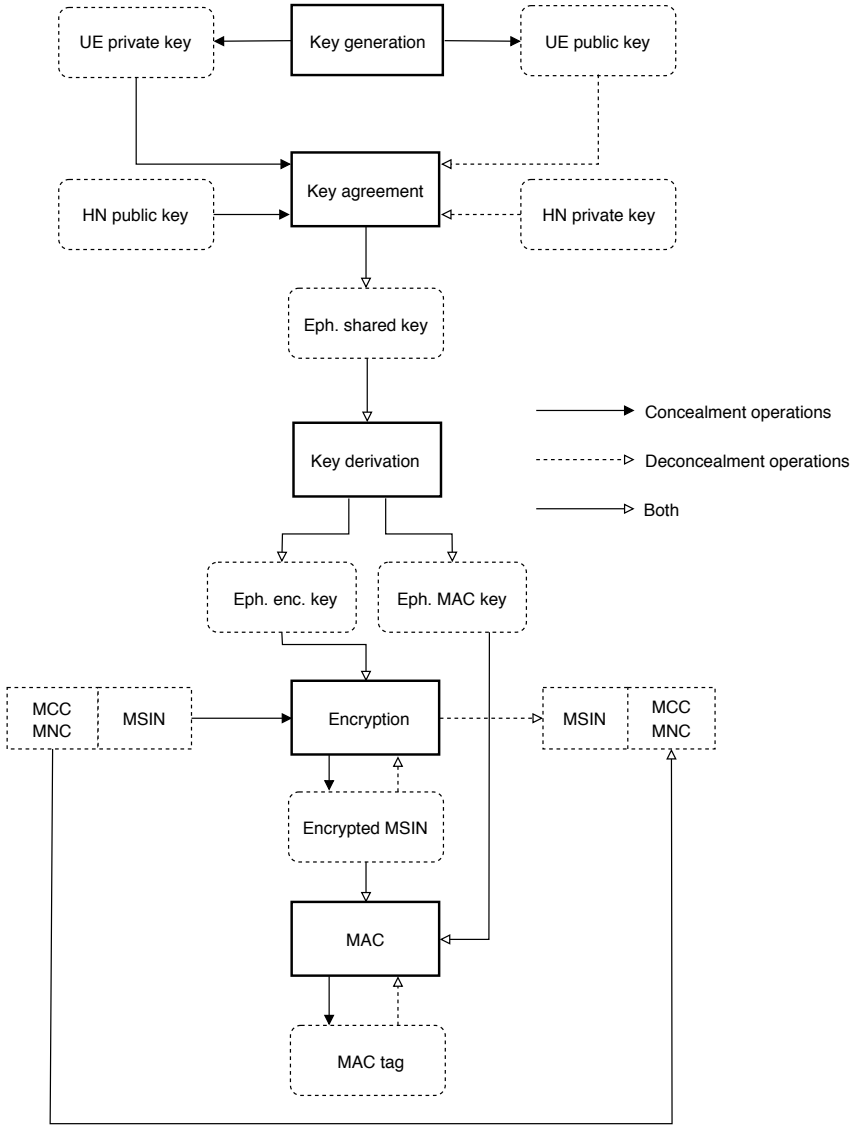

Figure 2.7: ECIES architecture. Adapted from [31] and [11].

protection. There are also two variants of ECIES called Profile A and Profile B. See Table 2.1 for the specifics about these Profiles. Other protection schemes may be included in the future [11].

2.4.3 ECIES*

In TR 33.899 [14], 3GPP proposed to use ECIES without the MAC function. We will call this version ECIES*. Their reason for not using MAC was that an integrity tag does not serve a purpose in this setting, since the UE's key is ephemeral. By removing the MAC the message size is also decreased [14]. This variant of ECIES was used in Jiménez et al.'s experiments [31], see Section 3.2. ETSI SAGE has commented on the initial proposal of ECIES, and stated the usage of MAC is essential to the security proof, and it therefore should be included [26]. The current specification includes MAC.

Chapter 3
Related work

3.1 Evaluations of ECIES

Shortly after 3GPP published their proposal of using ECIES in 5G, ETSI-SAGE, which is ETSI's expert group on cryptography, responded with some comments on the scheme [26]. Overall they agreed on the selection of the scheme, but they also identified some weaknesses and possible attacks. They found several weaknesses related to the cryptographic primitives specified in the proposal, and requested 3GPP to adhere to the standard specifications of ECIES. The requested changes have later been accepted and are now part of the 5G specification, so these are not mentioned here.

ETSI-SAGE also identified several possible attacks on ECIES in 5G. Some of the attacks are still possible, even if ECIES with the highest security options is used. The identified attacks include [26]:

- Chosen plaintext attacks. In this type of attack, the adversary selects an IMSI, encrypts it with the scheme, and sends it to the network. The network will give different responses based on the validity of the IMSI, and from this, the adversary can learn whether a specific subscriber is present in the cell.

- Replay attacks. These types of attacks are similar to chosen plaintext attacks, and are possible because ECIES does not provide any freshness guarantees.

- Denial of Service (DoS) attacks. These attacks can be performed against the home network by flooding it with SUCIs to decrypt, or against the UE by tricking them to generate a large number of SUCIs. IoT devices, and other constrained devices, can be especially vulnerable to these types of attack, as it can drain the battery.

– Downgrade attacks. In this type of attack, the adversary forces the user to use a pre-5G mobile network, which introduces the set of vulnerabilities from that network. See Section 3.5.5 for more information on this type of attack.

ETSI-SAGE also criticised how difficult it is for the home network to update its public key. There are several situations in which the home network needs to do so, for instance if they are hacked. ETSI-SAGE requested a method for the home network to quickly update its public key.

An additional weaknesses of the scheme is that it is not quantum resistant. ECIES uses elliptic curve cryptography, which is thought to be secure because of the assumed hardness of the elliptic curve discrete logarithm problem, as described in Section 2.3.1. With some advancements in the field of quantum computing, this can be broken with Shor's algorithm [59]. This is an algorithm which can compute discrete logarithms and perform integer factorisation in polynomial time on a quantum computer, which would have major implications on today's public key cryptography. 3GPP state in TR 33.899 [14] that this vulnerability could be resolved by swapping out the elliptic curve primitives in the future. In a recent blog post, security experts at Ericsson have shared their thoughts on post-quantum cryptography [43]. One of the points they make is that it is unlikely that quantum computers will become powerful enough to break modern public key cryptography in many decades. This is based on the fact that NIST currently uses conventional cryptography to protect their top secret information, and will not move to post-quantum cryptography until probably the mid to late 2020s. Note that 'top secret' here means that the information should be kept secret for several decades, up to 75 years.

3.2 ECIES performance study

In 2017, Jiménez et al. studied the performance of using ECIES* (see Section 2.4.3) for encrypting the IMSI [31]. They also discussed privacy issues with the LTE Paging procedure, and proposed a new protection mechanism to protect the IMSI during both the downlink Paging and uplink Initial Network Attach procedures. In their study, they made the assumption that the architecture of 5G would be similar as the architecture of LTE. Thus, the LTE architecture was used for the analyses. In this section, we present a summary of their work.

Motivation

In LTE, temporary subscription identifiers are mostly used, but there are still situations where the IMSI is used instead. One of these situations is during the Initial Network Attach procedure, in which the UE registers with the network. The portable nature of mobile devices make location tracking a possibility. Additionally,

Table 3.1: Devices used in Jiménez et al.'s experiments [31].

CPU specification	Android version	Market name
2.2 GHz Qualcomm MSM8974	5.1.1 Lollipop	Sony Xperia Z1 Compact
1.7 GHz MediaTek MT6592	4.4.4 KitKat	Aquaris E10 (Tablet)
1.3 GHz MediaTek MT6582	5.0 Lollipop	Aquaris E5 HD
1.0 GHz Qualcomm QSD8250	2.3.7 Gingerbread	Nexus 1

the communication between the UE and the network is sent wirelessly over the radio interface, making it vulnerable to eavesdropping. Temporary subscription identifiers partially protect against passive attacks on privacy, but active attacks have remained unaddressed. With new technology such as Software Defined Radio, active attacks have become an even larger threat [31].

While working on 5G, 3GPP released a report summarising the security aspects of this new system [14]. In it, they proposed to use ECIES* to encrypt the IMSI. Jiménez et al. state that there had not been conducted any practical feasibility studies of ECIES* with commodity devices yet, and it was uncertain whether the proposed enhancements were sufficient.

Method

To evaluate the performance of ECIES*, the researchers wrote two test applications using the C programming language. One application used the secp256r1 elliptic curve for the key generation, and the other used Curve25519. For key agreement, ECDH was used, and ANSI-X9.63-KDF with SHA-2 was used for key derivation. To encrypt the MSIN, they tested with both AES-CTR and XOR, which is the bitwise exclusive OR operation.

ECIES* was implemented using the cryptographic libraries OpenSSL and Nettle, and recommendations defined in SECG SEC 1 [64] were followed. Both applications were cross-compiled from Ubuntu to Android using the `gcc-arm-linux-gnueabi` and `binutils-arm-linux-gnueabi` toolchains. For the Nettle application, the dependent library GMP v6.1.1 also needed to be cross-compiled.

The applications were run 10.000 times each on four Android devices (see Table 3.1), and the average computation times were measured. The measurements were performed per cryptographic function. To record the time, they used the function `clock_gettime()` with the flag `CLOCK_PROCESS_CPUTIME_ID` from the C `time.h` library. This returns the per-process time from the CPU.

Table 3.2: secp256r1 computation performance comparison, measured in milliseconds [31].

CPU specification	OpenSSL		Nettle		eBACS	
	KG	KA	KG	KA	KG	KA
2.2 GHz Qualcomm MSM8974	4.62	4.69	1.39	1.56	0.62	2.11
1.7 GHz MediaTek MT6592	6.37	6.50	2.36	2.94	1.01	3.43
1.3 GHz MediaTek MT6582	8.73	8.90	3.22	4.03	1.33	4.49
1.0 GHz Qualcomm QSD8250	10.28	10.47	3.16	4.38	1.37	4.65

Table 3.3: Curve25519 computation performance comparison, measured in milliseconds [31].

CPU specification	OpenSSL		Nettle		eBACS	
	KG	KA	KG	KA	KG	KA
2.2 GHz Qualcomm MSM8974	0.43	1.18	1.31	1.24	0.19	0.19
1.7 GHz MediaTek MT6592	0.97	2.77	2.10	2.23	0.55	0.54
1.3 GHz MediaTek MT6582	1.34	3.79	2.87	3.06	0.72	0.71
1.0 GHz Qualcomm QSD8250	1.27	3.51	2.93	3.51	0.42	0.41

Findings

The added computational overhead introduced by the key derivation and encryption functions were 100-300 times faster than the key generation, and were not considered further. The computation times for key generation and key agreement when using the secp256r1 elliptic curve can be seen in Table 3.2. The results for Curve25519 are presented in Table 3.3. The computation times are presented along with values reported by ECRYPT Benchmarking of Cryptographic Systems (eBACS). The eBACS measurements were not performed on the same hardware, so the values in the table represent results with similar processors.

They saw that Nettle performed the best for secp256r1 with a computational overhead of 2.95 ms. This is the sum of the KG and KA results. OpenSSL performed the best for Curve25519 with an overhead of 1.61 ms. Both overheads were measured on the device with the 2.2 GHz Qualcomm MSM8974 CPU. They argue that the overheads are well within the delay of a typical call setup, and are therefore acceptable. The computational overhead when decrypting on the network side is negligible,

because the HSS has more computational resources than the UE, and it does not need to run the key generation function. This is because the HSS's public key is static [31]. There is some added bandwidth overhead introduced by the encryption, but this is also acceptable [14]. Their conclusion was that ECIES* was well suited to protect the IMSI, and that the overhead in terms of computation time and bandwidth was acceptable.

3.3 The Private Identification Problem

Mjølsnes and Olimid have requested a solution to what they call the private identification problem. A solution to this would answer the question of "how can a device identify itself to the (mobile) network while never disclosing its (permanent) identity to an adversary?" [46]. They call this solution a Private Identification Protocol (PIP). They state that transmission of the clear text private identifier is a frequent cause for privacy disclosure, and that this behaviour can be forced by the adversary. This type of privacy disclosure is relevant for all systems in which the devices need to identify themselves while hiding their identity.

3.3.1 Definition of the Problem

The private identification problem relates to the communication between the set of subscribers and the service provider. The subscribers need to identify themselves to the system to gain access to resources and services from the service provider. In mobile networking terms, the set of subscribers is equivalent to the UEs, and service provider is equivalent to LTE's Home Subscriber Server (HSS) [46], which closely resembles 5G's UDM.

Each subscriber has a permanent identifier that needs to be protected by PIP. It also has a key, and both of these parameters are also stored by the service provider. The subscribers are capable of performing cryptographic computations, but may have limited memory and computing power. They may also have private credentials that are pre-shared with the service provider. Both entities may have certificates. The service providers have more memory and computing power than the subscribers, and may perform advanced cryptography. They are capable of handling several requests from the subscribers at the same time [46].

3.3.2 Desired Properties of the Solution

PIP is perfectly secure if it does not reveal any additional information about the subscribers after the protocol is run. This will ensure the user's privacy. The protocol should also have certain other properties. It should run efficiently, and preferably run in constant time. There are many constrained devices connected to the mobile

network, and an efficient protocol will extend their battery life. It should also be scalable. This means that a solution that does not depend on a trusted central authority is preferable, and the communication should be bidirectional between the subscriber and the service provider [46]. It is assumed that the permanent identifier and key are stored securely in both the USIM and in the home network, so PIP is intended to secure the parameters in transit [45]. Additionally, the solution should make no, or minimal, changes to the architecture. The performance should be measured against 5G's capacity. It should stand against location disclosure and movement tracking. There is also a desire for unlinkability, meaning that an attacker cannot link the originator of one message with the originator of another [46].

3.3.3 Approaches to the Problem

There are several possible approaches to solving the private identification problem [45], which are summarised in this section.

Pseudonyms and Temporary Identifiers

A pseudonym is an alternate name that the subscriber uses to identify itself to the service provider instead of using the permanent private identifier. The pseudonym may be temporary. It is important to keep the pseudonym synchronised between the subscriber and the service provider, or else the subscriber may lose access to the network [36]. There are many different pseudonyms and temporary identifiers currently in use in mobile networks, such as the TMSI, which has been used for this purpose since 2G. See Section 2.2 for more information about identifiers. Previous to 5G, is was the network operator's decision how often the temporary identifiers were reallocated. This has been shown to sometimes be done too infrequently, which may affect the privacy. See Section 3.5.1 for more on this.

Symmetric Key Cryptography

With a symmetric key solution, each subscriber has a permanent key that is stored in the USIM. The same key is also stored in the core of the home network. The permanent key is used to derive the ephemeral keys that secure the communication [45].

A symmetric solution can be either stateless or stateful. If it is stateless, only the IMSI can be used for identification. When receiving an encrypted IMSI, the home network does not know whose key was used to encrypt it. In this case, it would have to use a brute force approach with all the subscriber keys in its database to find the one that decrypts to the correct IMSI. This procedure takes linear time, and is too slow. A stateful solution can also use temporary identifiers for identification. In this type of solution, the TMSI will not be refreshed in certain situations, and there may be issues with state synchronisation, leading to the IMSI being sent in cleartext [45].

Khan et al. have proposed a stateful symmetric scheme [33], which is described in Section 3.4.

Symmetric schemes are not as vulnerable to quantum computing attacks as public key schemes, and are therefore often considered as more secure for the future. With symmetric cryptography, quantum resistance can be achieved by simply increasing the key size [23].

Public Key Cryptography

A solution based on public key cryptography, also known as asymmetric encryption, can run in constant time. In public key cryptography, the service provider has a public-private key pair. The public key is issued to the UE, which uses it to encrypt the private identifier before it is sent to the service provider. The service provider can then decrypt the identifier with its private key. Asymmetric schemes are elegant, in the sense that they do not rely on a common secret between the UE and the network. However, they risk to be broken with quantum computing attacks in the future. This risk is discussed further in Section 3.1. Many asymmetric solutions require that the public key of the service provider is certified in a Public Key Infrastructure (PKI), which adds to the complexity [45]. Attribute-Based Encryption (ABE) is a type of asymmetric solution which does not require a PKI. This has been considered to protect the IMSI, but ECIES was selected instead [14].

ECIES is a hybrid encryption scheme that uses public key cryptography to derive the key used in the symmetric encryption function. It has been accepted in the security standard for encrypting the permanent identifier [11]. This scheme was explained in Section 2.4.

3.4 Survey on Subscriber Privacy

Khan and Martin recently published a survey paper on subscriber privacy on the radio interface in 5G [34]. In the paper, they looked at which vulnerabilities already existed in previous generations of mobile networks. Then they looked into which improvements 5G provides. Finally, they list the remaining vulnerabilities, and describe new attacks that have emerged. This section gives a summary of the paper.

3.4.1 Vulnerabilities in Previous Generations

From previous mobile network generations, IMSI catching seems to be the most powerful attack against subscriber privacy. We get back to IMSI catching in Section 3.5.4.

Another weakness is what they refer to as GUTI persistence. In LTE, there are some recommendations on when network operators should update the temporary GUTI identifier, but ultimately this is up to the serving network's decision when they want to update it. The UE does not take part in these updates. There are examples of network operators who have been shown to have poor practices when it comes to GUTI reallocation, and this can open up for location tracking. GUTI persistence is further described in Section 3.5.1.

IMSI based paging has been another issue. In most cases GUTI is used in the paging messages, but if the serving network has lost its context with the UE, this may not be an option anymore. In such cases IMSI can be used instead, meaning that the clear text IMSI is transmitted over the radio interface, where it is easily eavesdropped. A false base station can be used to send fake IMSI based paging messages in an area, and the responses to these will reveal who are present in the area. This method can be used to find a correlation between the IMSIs and GUTIs. Further, it is possible to correlate these to the MSISDN by placing a phone call to the particular MSISDN, and see which GUTI responds. As can be seen, IMSI based paging can be devastating to the subscriber's privacy. ToRPEDO is another type of attack that exploits a timing weakness with the paging procedure. This attack is described in Section 3.5.2.

3.4.2 Improvements in 5G

The survey paper lists several improvements for subscriber privacy in the 5G specification. These improvements are described elsewhere in the thesis, but are summarised in the following list:

- IMSI concealment with ECIES (see Section 2.4)
- More frequent GUTI refreshment (see Section 3.5.1)
- Detection framework for false base stations (see Section 3.5.4)
- Decoupling of IMSI from the Paging procedure. Temporary identifiers are used instead
- GUTI based Paging Occasion (see Section 3.5.2)

3.4.3 Remaining Challenges in 5G

The authors describe some attacks against the AKA procedure in 5G. These are out of the scope of this thesis, but interested readers may read more about this in the paper [34]. In addition, they referred to some of the weaknesses with ECIES, as were described in Section 3.1. Several of the weaknesses can be eliminated by using

symmetric cryptography instead. Khan et al. have proposed a symmetric scheme as an alternative to protect the identity [33]. This scheme mostly uses symmetric cryptographic primitives that are already a part of the 5G specification, and many of them are from the AKA procedure. The symmetric keys are updated in the identification procedure, so there is no need for a special update mechanism. The scheme does not protect against downgrade attacks [34]. Downgrade attacks are described in Section 3.5.5, which also describes a proposed protection scheme against these. Khan and Niemi have proposed an Identity Based Encryption (IBE) scheme as an alternative to ECIES [35], but this scheme is not quantum resistant. It also has more computation overhead and bandwidth overhead than the alternatives [34].

3.5 Attacks and Vulnerabilities

All generations of mobile networks have vulnerabilities that make them vulnerable to attacks. This section highlights some of the vulnerabilities that have been identified in the literature, as well as some possible attacks, focusing on LTE and 5G.

3.5.1 GUTI Persistence

Shaik et al. have identified poor GUTI reallocation procedures with three network operators in LTE [58]. In their experiment, they collected and analysed network messages to find out how often the GUTI changed over time. They found that a stationary UE would often be reallocated the same GUTI as it previously had, and it would sometimes have the same GUTI for up to three days. They also found that a new GUTI would sometimes be very similar to the previous one. The reason for these poor GUTI allocation routines is likely that the LTE specification does not mandate the operators to reallocate the GUTIs frequently. Rather, it is up to each operator to decide their policies on this [58]. Sørseth et al. have also studied how often the GUTI gets refreshed in LTE with two network operators [63]. Their findings were that the operators generally refreshed the GUTI at the expected times. However, one of the operators did not refresh the GUTI for 48 hours when the UE was stationary in a cell, and had no call activity. The other operator refreshed the GUTI at least three times a day under the same circumstances, which is a better privacy practice. This type of vulnerability means that the temporary identifiers are not really temporary after all, and it makes the network more vulnerable to passive attacks [58].

3.5.2 ToRPEDO Attack

The TRacking via Paging mEssage DistributiOn (ToRPEDO) attack was recently described by Hussain et al. [30]. This attack exploits a weakness with the paging procedure to gain information about a subscriber's cell location. In LTE, it can also

be used to reveal parts of the IMSI. The attack is efficient, and can be performed with less than 10 calls. Here, a call means an action that triggers a paging message to the victim's UE, e.g., a telephone call or an SMS.

To understand the ToRPEDO attack, we first need to understand how paging works. The Paging procedure in 5G is described in TS 38.300 [6]. Paging is used by the network to inform the user of a pending service, for instance an incoming call. When a UE receives a paging message, it re-establishes a connection to the network to receive the service. The UE checks for new paging messages at specific times, and enters an idle mode at other times. This is known as Discontinuous Reception (DRX), and allows the UE to save power by turning off the receiver for most of the time. The Paging Occasion (PO) controls at what times the UE should wake up from idle mode and listen for paging messages. The Paging procedure is illustrated in Figure 3.1. In 4G, PO is derived from IMSI [5], while in 5G, it is derived from 5G-S-TMSI [8].

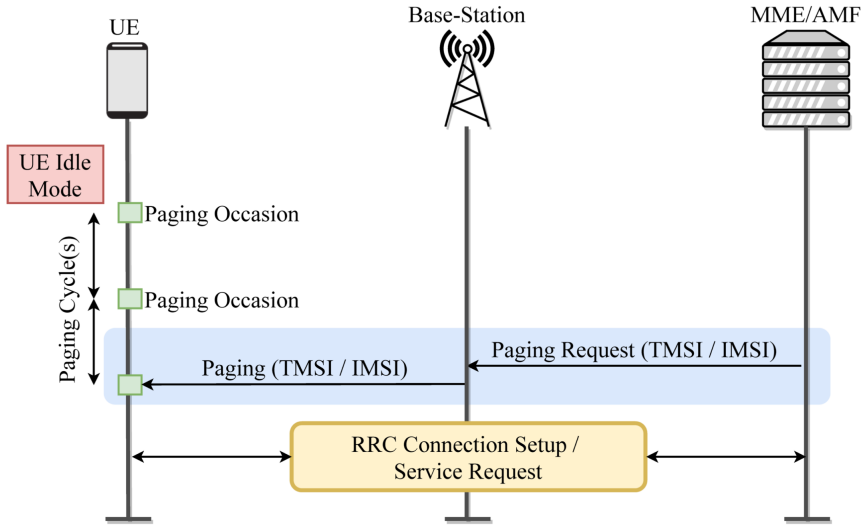

Figure 3.1: The Paging procedure [60].

ToRPEDO in LTE

The ToRPEDO attack in LTE exploits the fixed nature of a device's PO. Since the PO is derived from the fixed IMSI, it is possible for an adversary to use statistical analysis to gain information about both the presence of a subscriber in a given cell, and reveal 7-10 bits of the MSIN part of their IMSI.

To perform the attack, the adversary needs to know a soft identity of the victim, which is an identity that the adversary can call to trigger a paging message. They

also need a sniffer that can listen for paging messages. The sniffer needs to be set up in the cell where the victim is thought to be, so the adversary also needs to estimate where the victim is likely to be. When all the preparations are done, the adversary performs the attack by making several calls to the victim. Then they listen to the paging channel to see if there is a response. Since the same paging channel is shared between several users, the adversary may experience some noise on the channel, and uses a likelihood analysis to find the correct paging occasion, as explained in the paper [30]. Since the paging occasion is derived from the UE_ID, which is calculated as UE_ID = IMSI mod 1024, it is possible to learn the last 7-10 bits of the IMSI [60]. This is under the assumption that the paging parameters are set to certain values, as was the case with three out of four wireless carriers they observed [30]. This enables another attack explained in the paper called IMSI-Cracking [30].

The ToRPEDO attack was validated in a real LTE network, and the researchers were able to perform a successful attack in 2.4–4.3 minutes on average [30].

ToRPEDO in 5G

In the description of the ToRPEDO attack in 5G, the authors referred to a 3GPP specification of the paging procedure which is now outdated [7]. This version of the specification stated that the paging occasion was derived from IMSI, as in LTE. In an updated version of the specification [8], it is derived from the 5G-S-TMSI instead. This means that ToRPEDO cannot be used to reveal the last 7-10 bits of the IMSI in 5G. The original attack was similar as in LTE, but this section explains how the attack can be used with the updated version of the specification.

In 5G, ToRPEDO can in some cases be used to reveal a user's location. The attack is only possible if the network operator refreshes the temporary identifier with a value that is too predictable. This has happened in LTE [29], and it may occur in 5G as well. The authors of the paper [30] also state that the attack is possible if the temporary identifier is not refreshed after each paging, but this is mandatory in the current specification [11], so the statement does not apply anymore.

3.5.3 IMSI-Cracking

IMSI-Cracking is a brute force attack in LTE where the goal is to reveal the IMSI of a user. This was also described by Hussain et al. [30].

IMSI, as explained in Section 2.2.1, consists of the MCC, MNC and MSIN. The MCC and MNC are used for roaming purposes, and are not kept secret. These values can easily be looked up by an adversary, which will reveal up to 18 bits of the IMSI. Through the ToRPEDO attack, the researchers were able to also reveal the last 7 bits of the MSIN. This left 24 bits that they would have to guess, which is almost 17

million combinations. To validate a guess, they sent a paging message on the channel and waited for a response. The response type revealed whether the guessed IMSI was correct or not.

IMSI-Cracking was also validated in a real LTE network, and the successful attack required a little over 200.000 paging messages. It took 74 hours to crack the IMSI [30].

3.5.4 IMSI Catching

IMSI catching has been a persistent problem ever since the IMSI was introduced in GSM. This type of attack can be either *passive* or *active* (see Figure 3.2). In a passive attack, the adversary merely eavesdrops on the communication channel, hoping that a clear text IMSI gets transmitted. In order to do this the adversary needs a device, such as a Universal Software Radio Peripheral (USRP), with the required software [58]. The use of temporary identifiers protects against passive attacks, and they are therefore not very powerful.

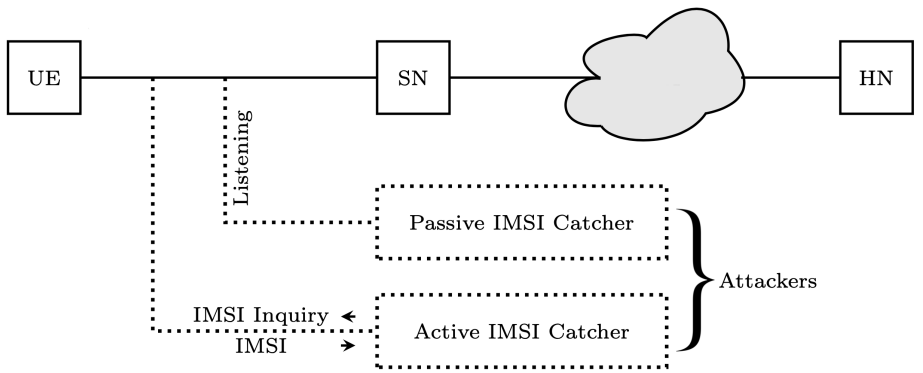

Figure 3.2: Passive and active IMSI catching attacks. Adapted from [37].

Active IMSI catching attacks are more powerful than passive. In this attack, the adversary imitates a base station to gather IMSIs. The adversary uses Identity Request messages to learn about the identities of the users present in a given area. The users then respond with an Identity Response message, containing the clear text IMSI [34]. The Identification procedure is illustrated in Figure 3.3. The IMSI is sent in clear text because the security context between the UE and the Serving Network (SN) has not yet been established, and the UE has not been assigned a temporary identifier. This type of attack has been possible in both GSM, UMTS, and LTE networks. In 5G, SUCI has been added to prevent active IMSI catching attacks, as it prevents the clear text IMSI to be transmitted with the Identity Response message.

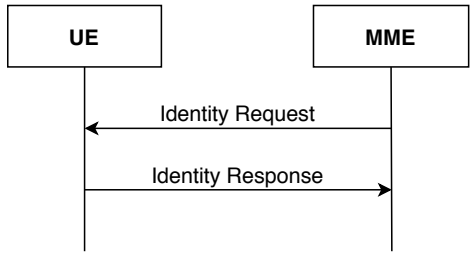

Figure 3.3: The Identification procedure in LTE. Adapted from [3].

IMSI catching attacks can be used in larger attacks, and may have consequences such as location tracking and mass-surveillance [44]. They are possible in LTE, as demonstrated by Shaik et al. [58]. Mjølsnes and Olimid have shown that IMSI catching attacks can be performed by persons without programming skills [44].

3GPP has included a framework for detecting false base stations in the 5G specification [11]. This framework uses measurement reports from the UEs to look for irregularities. The framework is currently informative, and Khan and Martin recommend that 3GPP make it normative instead [34]. This is because the industry has often failed to adopt informative parts of the specification in the past. In addition to the detection framework, 3GPP are also investigating how to enhance the security against false base stations in the future [13].

3.5.5 Downgrade attacks

Khan et al. have made some interesting points about downgrade attacks in 5G, and how they can be used to enable IMSI catching [37]. They have also proposed a pseudonym-based solution to counter this.

A 5G UE will occasionally need to connect to pre-5G networks. This is because it takes time to upgrade the current infrastructure to 5G, so some areas only support GSM, UMTS, or LTE connections. When a UE connects to such networks, it cannot use SUCI to protect the IMSI, as it is not supported. This can be abused by an adversary by impersonating e.g., an LTE SN, and then exploiting the vulnerabilities in LTE to catch the IMSI [37].

In the paper [37], the authors also propose a solution to counter IMSI Catching attacks that downgrade the communication to LTE in order to obtain the IMSI. The solution uses pseudonyms with the same format as IMSI, which the UE uses instead of IMSI to identify itself to LTE SNs. This way, the adversary will not be able to obtain the IMSI, even if the communication is downgraded to LTE. If the UE connects to a 5G network, it identifies itself with SUCI, as normal. The HN

is responsible for the allocation of pseudonyms, and it initially allocates two per UE, stored in the USIM. New pseudonyms are allocated during the AKA procedure. The proposal also uses some techniques to prevent de-synchronisation, as well as a method for re-synchronization. These are described in the paper [37].

The proposal makes use of existing LTE messages, so no additional messages are required. However, some changes would be needed in the UE and networks. To support Lawful Interception (LI), elements in the LTE SN's core network would require a software update. The USIM would need to store the two pseudonyms, as well as a key to decrypt new pseudonyms from the HN. The key is shared with the HN. The HN would also need to store the key, as well as a list of all pseudonyms in use. This is to prevent two UEs from being allocated the same pseudonym [37].

Chapter 4
Method

This thesis consists of two main parts. The first part is a review on the private identification problem in mobile networks, with a focus on 5G. This topic was investigated through a literature review, and the process is further described in Section 4.1. The thesis also includes an experimental part where the key generation and key agreement parts of ECIES have been implemented, and the performance measured. This process is described in Section 4.2.

4.1 Review of Subscriber Privacy in 5G

The subscriber privacy in 5G was reviewed through a qualitative analysis. The review took basis in Mjølsnes and Olimid's description of the private identification problem [45, 46]. This is summarised in Section 3.3. The 3GPP specification was used for information about the 5G architecture, procedures and security practices. The review also includes research papers on the topic. When searching for literature, recent papers were prioritised. The 5G specification is changing fast, so papers can become outdated rather quickly. Most of the papers on 5G that are included in the review were published in the last two years. Important details in the papers were crosschecked with the recent 3GPP documents that are of relevance. The review also includes some papers on important aspects of LTE, which are useful for comparison. The oldest of these papers was published four years ago.

The papers were generally found through two methods. One method was to search with relevant keywords on Google Scholar, or the regular Google search engine. The other method was to find interesting papers in the reference list of other papers. In the early stages of the thesis work, it was up for consideration to perform a structured literature review. This was decided against, as the nature of the topic is rather structured in itself.

4.2 Replication of ECIES Performance Study

Part of the thesis is to replicate and extend the ECIES performance studies conducted by Jiménez et al. in 2017 [31]. This section aims to give insight into how this was done. First, the tools and environments used are described in Section 4.2.1. Next, programs are presented and described in Section 4.2.2, and the cross-compilation process is described in Section 4.2.3. Section 4.2.4 describes how the programs were run. This study has used a slightly different approach than the replicated study, and these deviations are described in Section 4.2.5.

4.2.1 Tools and Environments

This section describes the tools and environments used for the experiments.

The C Programming Language

The programs are written in the C programming language. This is a universal programming language, and it is supported by many different operating systems and computer architectures [32]. It is influenced by the typeless languages BCPL and B, but C itself is a typed language. C uses pointers that ensure machine-independent address arithmetic, meaning that we can access the memory of different computer architectures in the same way. C is a fairly low-level language that maps efficiently to machine instructions, and programs are generally fast. In this thesis, C is used with OpenSSL to implement ECIES.

OpenSSL

OpenSSL is a toolkit for the TLS and SSL protocols. Additionally, it functions as a general-purpose cryptographic library, and it can be used with C [55]. OpenSSL is one of the most popular cryptographic libraries, and it is well-documented. In the thesis, the EVP functions of OpenSSL version 1.1.1d were used for the replication. The programs are largely based on the examples found in the EVP documentation, but has been slightly modified to my needs.

Nettle

Nettle is a low-level cryptographic library that can be used in C programs. The goal of Nettle is to provide a simple, but general interface to several cryptographic primitives, and it is adaptable to many contexts [47].

The plan for the thesis was to use both Nettle and OpenSSL to replicate Jiménez et al.'s experiments, but Nettle was eventually dropped from the plan. In order to implement ECDH with Nettle one needs some proficiency in C, and my master programme has not included any courses on this. I also reached out to Jiménez et al.

and asked if they were able to give me their source code. They were unfortunately not willing to disclose it as it belongs to Ericsson Research, which means that this part was not replicable.

Android Studio

Parts of Android Studio was used to compile and run the OpenSSL programs. Android Studio is an IDE for development of Android applications [19]. It includes Android NDK, which is a toolset that adds support for native code in Android applications [18]. It offers a set of compilers that can be used to cross-compile C programs to run on Android phones. Android Debug Bridge (adb) was used to transfer the programs to the phones and run them. This is a tool that allows for communication with a device by using the command line [17]. See Section 4.2.3 and Section 4.2.4 for a description of how these tools were used.

Devices Used

The devices used in this study are listed in Table 4.1. All devices, except for the Samsung Galaxy S5 Neo, has several CPU cores with varying clock speeds. It is unclear which specific core was used to run the programs, but the highest clock speed is shown in the table. A complete overview of a device's CPUs can be found on the Device Specifications website.[1]

Table 4.1: List of the devices used.

System on Chip (SoC)	Android ver.	Market name
2.7 GHz Exynos 9810	Android 10.0	Samsung Galaxy Note9
2.45 GHz Qualcomm Snapd. 835	9.0 Pie	Sony Xperia XZ1 Compact
2.36 GHz HiSilicon Kirin 659	7.0 Nougat	Huawei Mate 10 Lite
2.36 GHz HiSilicon Kirin 970	8.1 Oreo	Huawei P20 Pro
1.8 GHz Qualcomm Snapd. 652	7.0 Nougat	Samsung Galaxy Tab S2
1.6 GHz Exynos 7580	6.0 Marshm.	Samsung Galaxy S5 Neo

4.2.2 Programs

The source code for the programs can be found on Github.[2]

[1] https://www.devicespecifications.com/
[2] https://github.com/ulebass/openssl-ecc

The programs use OpenSSL's EVP manual pages as a reference [53]. These explain the high level cryptographic functions in the library. The following is a high level description of how the programs work:

```
Pre-calculate HN's key pair

For 10.000 iterations:

    Start KG timer
    Generate UE's key pair with specified curve
    Stop KG timer

    Start KA timer
    Derive shared key with ECDH
    Stop KA timer

Calculate average measurements
Print results
```

The programs check if the EVP functions return the expected value. In most cases, a return value of `1` means that the function was successful. If not, an error statement is called. If the expected return value is an EVP structure, the program throws an error if the returned value is `NULL` instead. The expected return values are listed on the manual pages [53].

The programs experience a bug with the measured timings if the start time and the end time falls on different seconds, for instance if the start time is 0.97 ms and the end time is 1.04 ms. In that case, the measured time becomes negative due to how it is calculated. Therefore, the programs check if the timing values are negative before adding them to the cumulative sum. The `correct_timing` function is used to correct the negative instances.

4.2.3 Cross-compilation of Programs

The OpenSSL programs were cross-compiled using the `armv7a-linux-androideabi18-clang` compiler included in Android NDK r21. This compiler compiles to a 32-bit ARM processor architecture with Android API level 18, and it was chosen to make the programs compatible with both older and newer phones. The programs were cross-compiled on a virtual machine running Linux Mint 19.3.

Before configuring and installing OpenSSL, the following lines were added to `~/.bashrc` to set the Android NDK's path:

```
export ANDROID_NDK_HOME=/path/to/android-ndk-r21
PATH=$ANDROID_NDK_HOME/toolchains/llvm/prebuilt/linux-x86_64/bin:
$PATH
```

OpenSSL was configured with the following options before installation:

```
$ ./Configure android-arm no-shared -D__ANDROID_API__=18
```

The programs were compiled with the following command:

```
$ armv7a-linux-androideabi18-clang -I /usr/local/include program.c
-o program -L /usr/local/lib -lssl -lcrypto -fPIC -pie
```

The file `NOTES.ANDROID` included in the OpenSSL library was used as a reference for the cross-compilation process.

4.2.4 Running the Programs

Android Debug Bridge (adb) was used to transfer and run the programs on the phones. In order to use it, Developer options was enabled on the phones. Three people volunteered to run the programs on their phones, and the instructions they were given can be seen in Appendix A. The other three phones were tested by me, using the same approach. The phones I tested ran the programs several times to test how consistent the results were. They were generally consistent. Some runs differed more than others, but not by a large margin. The volunteers ran the programs once. The results can be seen in Section 5.1.

4.2.5 Deviations From Previous Study

A goal of replication studies is to follow the same methods. The experimental study performed in this thesis had some deviations from the the study it aims to replicate. Some of the deviations are minor, while other deviations are more substantial, and are likely to have influenced the results rather significantly. The deviations are described in the following list:

- **Hardware differences.** This study tested the performance of a different set of devices than the replicated study. The devices used were chosen based on availability. Additionally, the performance of mobile devices has improved a lot since the replicated study was performed, and it is likely that the results achieved in this thesis reflect that.

- **Different compiler.** To compile the programs, the instructions in OpenSSL's NOTES.ANDROID were followed, where a compiler from Android NDK is used. This may have influenced the results somewhat, but it is not clear how much.
- **Different software versions.** This study used a more recent version of OpenSSL. This may have influenced the results, and is discussed further in Section 5.1.1. Additionally, the programs were compiled from a different Linux distribution. It is unlikely that this has influenced the results.
- **Different programs.** Since the researchers of the replicated study were not willing to disclose the source code of their programs, it is likely that my programs differ from theirs. How much they differ is not known, but it is likely that this has influenced the results, potentially to a large degree.

As stated in the list, some of the deviations between this study and the study it aims to replicate may have yielded significantly different results, and the reader should be cautious about comparing the results without this in mind. However, the two studies are also equal in several ways. This includes the programming language used, how timing was measured, and which cryptographic interface from OpenSSL was used.

Chapter 5
Results and Discussion

5.1 ECIES Implementation Results

In this section, the results from the OpenSSL implementations of ECIES with elliptic curves secp256r1 and Curve25519 are presented and discussed. The results are summarised in Table 5.1. Screenshots of the results can be seen in Appendix B. We see that the KG and KA primitives run fast with both elliptic curves, and there is not a significant difference between the two. KA takes 2-3 times longer than KG in all cases.

The consistency in the results between distinct runs were measured on the Huawei Mate 10 Lite. The minimum and maximum results are shown in Table 5.2. A notable outlier was accidentally found when the Curve25519 program was run when the phone had only 1% battery left. This outlier is not inclued in the data, but gave a KG time of 0.37 ms and KA time of 1.00 ms, which is almost twice as high as the

Table 5.1: OpenSSL computation performance of key generation and key agreement using the elliptic curves secp256r1 and Curve25519, measured in milliseconds.

System on Chip (SoC)	**secp256r1**		**Curve25519**	
	KG	KA	KG	KA
2.7 GHz Exynos 9810	0.09	0.26	0.09	0.24
2.45 GHz Qualcomm Snapd. 835	0.28	0.62	0.26	0.69
2.36 GHz HiSilicon Kirin 659	0.21	0.50	0.20	0.55
2.36 GHz HiSilicon Kirin 970	0.20	0.60	0.17	0.49
1.8 GHz Qualcomm Snapd. 652	0.15	0.37	0.14	0.40
1.60 GHz Exynos 7580	0.32	0.75	0.30	0.82

Table 5.2: Consistency measurements on the Huawei Mate 10 Lite from 20 runs.

	secp256r1	**Curve25519**
Min. KG	0.203682	0.201710
Max. KG	0.205703	0.204515
Min. KA	0.499304	0.550846
Max. KA	0.500183	0.552486

other results. This outlier is likely the result from a throttling of the processor to save battery. The other measurements were performed when the phone had at least 10% battery.

5.1.1 Comparison to Previous Study

The results from the experiments differ significantly from Jiménez et al's performance studies (see Section 3.2). As a reminder, their results with OpenSSL and secp256r1 were that KG took 4.62 ms–10.28 ms. KA showed similar results of 4.69 ms–10.47 ms. My results with the secp256r1 curve are significantly faster for both KG and KA. Additionally, my KG and KA results are not as similar to each other.

For Curve25519, their results were that KG took 0.43 ms–1.34 ms, and KA took 1.18 ms–3.79 ms. Their results with Curve25519 are also slower than mine, but the difference is not as large as with secp256r1. When comparing the KG and KA times, they are more consistent with my results.

Since they did not disclose their source code, it is not entirely clear why the results differ so much. It is still interesting to make some educated guesses, based on the information we have available. In Section 4.2.5, several deviations between the two studies were presented. One of the deviations that is thought to have contributed the most to the different results is the set of devices used. It is expected that the newer phones used in this study perform better. Phones have evolved very fast in the last 10-15 years, and the three year gap between the studies is a long time in this sense. Another aspect that may have contributed significantly to the different results is the source code, but it is not clear how much.

A look at the changelog of OpenSSL between version 1.1.0c and version 1.1.1d revealed some updates to the library that can potentially explain some of the differences between the results [51]. Version 1.1.1 introduced many updates, and a few of these are related to how ECC multiplication is performed. It is possible that this has contributed to faster execution of the ECC based KG and KA primitives. It

should be noted that there may have been additional changes in the lower level of the library that could have an indirect impact on the results, but this is not clear.

5.1.2 Discussion

A surprising aspect of the results is that the Curve25519 implementation did not perform better than the secp256r1 implementation. As mentioned in Section 2.3.1, Curve25519 is designed to perform fast execution of ECDH by using the X25519 primitive. This was not reflected in the results, where there were no significant differences between the two curves' performance. The Curve25519 implementation used OpenSSL's X25519 manual page [56] as a reference. It should be noted that the code used to perform ECDH is the same in both the secp256r1 and Curve25519 programs. This is because the X25519 manual page links to another manual page on how to derive a shared secret [54] that is fundamentally the same as the ECDH example found on the OpenSSL Wiki page [52]. This could mean that the computational optimisations that Curve25519 in theory provides, were not utilised in practice.

Despite the unexpected results of the Curve25519 implementation, both curves yielded fast results, and much faster than in the replicated study. The slowest phone in this study took at most 1.14 ms to perform both KG and KA. When we take the other results from the replicated study into consideration, namely that the key derivation and encryption functions were at least 100–150 times faster than KG and KA, it is likely that the total time is only marginally slower. Note that the MAC function of ECIES has not been tested in either study. This will also add to the total computation time, but it is unlikely that it will contribute much. It should also be noted that in a real 5G setting, the SUCI can be computed directly in the USIM, as explained in Section 2.2.2. This will likely be more computationally efficient.

In Jiménez et al.'s study [31], they also considered the bandwidth overhead that SUCI encryption with ECIES* introduces. Their conclusion was that it was well acceptable. With ECIES*, the Scheme Output is 320 bits. This includes the public key and the encrypted MSIN value. In Section 2.2.2, we saw that the MAC tag is 16 hexadecimal digits. This would add 64 bits to the Scheme Output, meaning that it is 384 bits in total with ECIES. This is not a large increase, when we take the added security of including the MAC into consideration.

5.2 Subscriber Privacy in 5G

As we have seen, subscriber privacy has been one of the main challenges in mobile networks. Each generation has introduced new security mechanisms that aim to protect the privacy, but some challenges have remained. In this section, subscriber privacy in 5G is discussed.

5.2.1 Transmission of Clear Text IMSI

One of the major weaknesses in previous generations of mobile networks is that the IMSI is sometimes transmitted in clear text between the UE and the network. In 5G, the SUCI was introduced to conceal the IMSI.

Why is it important to conceal the IMSI? There are several identifiers associated with a mobile subscriber, and all of these could potentially be used by an attacker to harm the subscriber's privacy. In practice, the IMSI is the most vulnerable of them. There are two major reasons for this. First, it is a permanent identifier, so it does not change over time. This means that an attacker does not need to re-learn the IMSI of a known subscriber. There are also other permanent identifiers related to a subscriber, but these are generally never transmitted in clear text to the network. This leads us to the second point. It is the only identifier that needs to be sent to the network before the security context has been established. Unencrypted wireless communication is easy to eavesdrop on, as is has no physical or cryptographic barriers. Fortunately, temporary pseudonyms are preferably used as the identifier instead of the IMSI whenever possible. As long as these are changed relatively frequently, they do not share the same privacy threat as IMSI.

As we have seen, the transmission of the clear text IMSI in earlier generations of mobile networks have made them vulnerable to IMSI Catching attacks. In 5G, the IMSI shall not be transmitted in clear text, except for a few cases. These were presented in Section 2.2.2, but we repeat them here:

- If the UE wants to initiate an unauthenticated emergency session, and does not have a 5G-GUTI in the network
- If the HN specifies that the null scheme is to be used
- If the UE has not been provisioned with the HN's public key

The first exception is necessary to allow users access to emergency services. It is rare that a specific user needs to start an emergency session, so this is not a viable method for an adversary to track them. Additionally, users should not be denied access to emergency services because of strict privacy rules in the network. Several countries also require that emergency phone calls should be possible even if the phone does not have a UICC inserted [27].

As for the second exception, it is unclear why the HN would choose to use the null scheme. One of the requirements in the 5G specification is that the MSIN part of the IMSI should not be transmitted in clear text over the Next Generation Radio Access Network (NG-RAN) [11]. NG-RAN is the radio interface in 5G, which means

that in a pure 5G setting, network operators would not fulfil this requirement if the null scheme is used. However, it is possible that this exception is included to allow the UEs to connect to pre-5G networks.

The final exception is probably included to allow for cases where the HN needs to update its keys, for instance if the key has been compromised. In that case, all subscribers that belong to that network become vulnerable to both passive and active IMSI catching attacks [14]. As mentioned earlier, the specification does not include a clear method on how the HN can securely distribute its updated public key to the UEs. Instead, it is up to the HN to decide how they want to do it. One possible method is to send new (physical) UICCs to all subscribers, which has the updated public key stored on them. In the meantime the UEs would only have access to the outdated key. In that case, it makes more sense to allow use of the clear text IMSI, than to deny the subscribers access to the network.

As seen in Section 3.5.4, 3GPP has included a detection framework against false base stations in the 5G specification, and are also investigating how to further eliminate them. This, together with the use of SUCI in the Initial Registration and Identity Response messages, will make IMSI catching a lot harder in 5G than in previous generations. However, downgrade attacks are still possible with the current specification, and these can have a major impact on privacy. The detection framework presented in the specification can be applied to earlier mobile generations as well, to combat some of the consequences of a downgrade attack. However, as previously mentioned, it is unlikely that the detection framework will be widely adopted by the network operators unless it is transformed to a normative one. Despite the risk of downgrade attacks, it is good that 3GPP have finally included IMSI protection in 5G. As stated in TR 33.899 [14]: "If each generation system is let to pass without IMSI privacy, it means that 'complete' IMSI privacy is delayed by another decade, then another decade, and so on." Downgrade attacks to reveal the IMSI will become less powerful in the future, when older mobile network generations are being phased out.

When it comes to the other attacks mentioned in Section 3.1, they cannot be used to find the IMSI. In a chosen plaintext attack, the adversary needs to know the IMSI beforehand. From that, they can learn about the subscriber's presence in an area. Similarly, in a replay attack, the adversary needs to know the SUCI of a subscriber to learn about their presence. While these attacks are problematic, they are not as powerful as a downgrade attack. DoS attacks cannot be used to find the IMSI either, but they can have a large impact on the availability in the network. 5G is envisioned to be used with IoT devices, which are particularly vulnerable towards these types of attacks.

5.2.2 Improvements in the Paging Procedure

The LTE paging procedures has several weaknesses that could be exploited by an adversary. One of these weaknesses is that network operators are allowed to perform paging with IMSI instead of TMSI in some cases, for instance before the subscriber has initially registered to the network, and therefore has not yet been assigned a TMSI. Network failures can also cause the IMSI to be used as the paging identifier, and it is possible for an adversary to provoke this type of behaviour. In 5G, IMSI is never used as the paging identifier. SUCI is used instead.

Another weakness with LTE paging is the relationship between the paging occasion and IMSI, where the paging occasion is derived from the IMSI. The static nature of the IMSI means that the paging occasion is also static, and adversaries could perform attacks such as ToRPEDO and IMSI-Cracking to reveal the IMSI. This weakness is mitigated in 5G paging, as the paging occasion is derived from the 5G-S-TMSI instead.

The final improvement in the 5G paging procedure is that it is now mandatory for the network operator to refresh the GUTI after each paging instance. This prevents it from effectively being yet another permanent identifier, which has been observed in LTE, where the same TMSI has been used for several days. When TMSI is updated too rarely, it becomes vulnerable to many of the attacks against permanent identifiers, including location tracking. This issue is discussed further in Section 5.4.

5.3 Protection Scheme

Mobile networks need to support a wide range of use cases, and they have strict requirements on performance. Previous to 5G, mobile networks had no method to protect the IMSI when a subscriber initially registered with a network, due to the complexity this would add to the system. This changed with 5G, when ECIES was included as the protection scheme.

The experiments performed in the thesis have again confirmed that ECIES is fast, and does not add much computational overhead. This is an important feature in a system like 5G, which will be used by devices with constraints on memory usage and battery life. We have also seen that ECIES adds some bandwidth overhead, but is within the limits. Most of the added bandwidth overhead is used to include the ephemeral public key of the UE, which the HN needs to decrypt the MSIN. ECIES uses ECC, which has short keys compared to the other popular public key scheme RSA. RSA keys are typically 2048 bits or more, while ECIES uses 256 bit keys. A study that compared the performance of RSA and ECC also found that ECC was the best option for resource constrained devices [42].

One of the drawbacks of ECIES is that it is not quantum resistant. 5G will likely be in use for several decades, and it is therefore necessary to consider the possibility of quantum computing attacks in the future. Current quantum computers are expensive, and not very powerful. It is difficult to predict when, or if, quantum computers will become a threat to modern public key cryptography, but as mentioned in Section 3.1, it will likely take many decades. 3GPP has proposed to swap out the ECC primitives in ECIES with a quantum resistant alternative in the future to improve the security [14]. It is not clear if this would invalidate the formal security proof of the scheme. As mentioned in Section 3.4.3, a symmetric scheme has been proposed as an alternative to ECIES. Symmetric schemes are not as vulnerable to quantum computing attacks as public key schemes. If the proposed symmetric scheme is used with the mitigation strategy against downgrade attacks (see Section 3.5.5), as proposed in Section 3.4, it may offer better privacy than ECIES does. Additionally, it will have less bandwidth overhead, as it does not need to include a public key in the message. Future releases of the specification should consider to include this.

The specification also opens up for the network operators to use proprietary protection schemes. The obvious risk in this regard is that the schemes used may be weaker than ECIES. This is unfortunate, and future releases should be more standardised to avoid this.

5.4 Persistence of Temporary Identifiers

5G has reduced the situations in which the clear text IMSI is used, which has greatly reduced the risk of IMSI catching attacks. However, as we have seen in Section 3.5.1 and Section 3.5.2, temporary identifiers can also cause privacy issues in some cases.

In LTE, the GUTI is refreshed in the GUTI reallocation procedure. The specification does not have clear guidelines on when the network operators should initiate this procedure [3, 29]. This would in some cases lead to situations where the same temporary identifier was used for several days. This type of persistence makes it possible to identify and track a subscriber, as previously explained. The 5G specification provides clearer instructions on when the GUTI should be refreshed, as explained in Section 2.2. This includes after the UE initially registers in the network, when it updates its location, and after paging. Together with the recommended GUTI refreshment in the Periodic Registration Update, it seems unlikely that location tracking based on GUTI persistence will be a problem.

However, we have seen that GUTI persistence is not the only vulnerability. The GUTI may be refreshed to a similar or predictable value. This behaviour has been shown in LTE, and similar behaviours in 5G would make it possible to perform the 5G ToRPEDO attack (see Section 3.5.2). While the 5G specification requires that

the TMSI is generated to an unpredictable value, it does not specify how this should be done. This could lead to some poor implementations by the network operators. The specification should include clearer instructions on how this can be done in the future.

The specification opens up for SUCI persistence in the De-registration procedure. Remember that if the UE did not receive a registration accept message in the initial registration procedure, it should re-use the SUCI in a De-Registration Request message. An attacker could exploit this by actively hindering the network's registration accept messages from reaching the UE, and trace the subscriber based on the persistent SUCI. This is not a very powerful attack.

Chapter 6
Conclusion

This thesis has investigated how well the 5G specification maintains subscriber privacy. The work consisted in a literature review of the topic, as well as a replication of the ECIES performance study conducted by Jiménez et al. at Ericsson Research.

The replication study in the thesis validated that ECIES runs fast enough to be used with mobile phones and tablets to conceal the SUPI in the 5G network. The results obtained in this study were faster than the results from the previous study. This is likely due to the updated hardware used in this study. Since the source code from the previous study is proprietary to Ericsson Research, the source code was written from scratch in this study. It is possible that this has contributed to the different results as well. The source code used in this study follows the guidelines in the OpenSSL EVP documentation to avoid logical errors that may invalidate the results. The methodology used to obtain the results is well documented. It is likely that ECIES is faster in practice, as the scheme can be implemented and run directly on the USIM.

While ECIES is a fast encryption scheme, it is not quantum resistant. It is difficult to say how big of an issue this currently is. Quantum computers are not very powerful at the moment, and need some advancements before they become a threat to modern public key cryptography. Current predictions indicate that quantum computers will not be powerful enough for a few more decades, so there is not an immediate threat to the security of ECIES. However, the next generation of mobile networks should strive to include a quantum resistant scheme. This should ideally be a symmetric scheme. These are, in general, more quantum resistant than public key schemes, and do not add as much bandwidth overhead. Symmetric schemes are also faster, which is ideal for resource constrained devices. The 5G specification opens up for including additional protection schemes in the future. However, it also allows that network operators use proprietary protection schemes, which may lead to some poor protection schemes being used.

The thesis has also reviewed the subscriber privacy in 5G, and found that 5G offers much better subscriber privacy than previous generations of mobile networks. The major privacy weaknesses that were present in LTE have been mitigated, and it is almost infeasible to find the IMSI of a subscriber in a pure 5G setting. This is largely due to the introduction of the SUCI, as well as the improvements in the Paging procedure. However, it is possible to downgrade the communication to an older network generation, which makes it vulnerable to previous attacks. Another potential issue is persistence of temporary identifiers, which makes it possible to track the location of a subscriber. This has been a problem in previous network generations. 5G requires that the temporary identifiers are generated as an unpredictable value, but it remains to see if this practice is followed by the network operators. While the subscriber privacy in 5G may not be perfect, it is good to see that the issue has been taken seriously, and future releases will hopefully address the remaining issues.

6.1 Future Research Directions

There are several topics that have not been fully covered by this thesis, which are interesting to investigate more in the future. Future research directions include:

- Test the performance of ECIES in a realistic setting, i.e., with a UE that supports 5G, and in a real 5G network
- Test the feasibility of using ECIES with resource constrained devices
- Investigate the possibility of including countermeasures against downgrade attacks in the specification
- Investigate the possibility of including a symmetric scheme in the specification
- Investigate how well network operators adhere to the security specification of 5G in a real network

References

[1] 3GPP. *Digital cellular telecommunications system (Phase 2+); Specification of the Subscriber Identity Module - Mobile Equipment (SIM-ME) interface*. TS 51.011. Version 4.15.0. June 2005. URL: https://portal.3gpp.org/desktopmodules/Specifications/SpecificationDetails.aspx?specificationId=2793.

[2] 3GPP. *Release 15*. URL: https://www.3gpp.org/release-15.

[3] 3GPP. *Technical Specification Group Core Network and Terminals; Non-Access-Stratum (NAS) protocol for Evolved Packet System (EPS)*. TS 24.301. Version 16.4.0. Mar. 2020. URL: https://portal.3gpp.org/desktopmodules/Specifications/SpecificationDetails.aspx?specificationId=1072.

[4] 3GPP. *Technical Specification Group Core Network and Terminals; Numbering, addressing and identification*. TS 23.003. Version 15.8.0. Sept. 2019. URL: https://portal.3gpp.org/desktopmodules/Specifications/SpecificationDetails.aspx?specificationId=729.

[5] 3GPP. *Technical Specification Group Radio Access Network; Evolved Universal Terrestrial Radio Access (E-UTRA); User Equipment (UE) procedures in idle mode*. TS 36.304. Version 16.0.0. Mar. 2020. URL: https://portal.3gpp.org/desktopmodules/Specifications/SpecificationDetails.aspx?specificationId=2432.

[6] 3GPP. *Technical Specification Group Radio Access Network; NR; NR and NG-RAN Overall Description*. TS 38.300. Version 16.0.0. Dec. 2019. URL: https://portal.3gpp.org/desktopmodules/Specifications/SpecificationDetails.aspx?specificationId=3191.

[7] 3GPP. *Technical Specification Group Radio Access Network; NR; User Equipment (UE) procedures in Idle mode and RRC Inactive state*. TS 38.304. Version 15.0.0. June 2018. URL: https://portal.3gpp.org/desktopmodules/Specifications/SpecificationDetails.aspx?specificationId=3192.

[8] 3GPP. *Technical Specification Group Radio Access Network; NR; User Equipment (UE) procedures in Idle mode and RRC Inactive state*. TS 38.304. Version 15.6.0. Dec. 2019. URL: https://portal.3gpp.org/desktopmodules/Specifications/SpecificationDetails.aspx?specificationId=3192.

[9] 3GPP. *Technical Specification Group Services and System Aspects; 3G Security; Security architecture*. TS 33.102. Version 15.1.0. Dec. 2018. URL: https://portal.3gpp.org/desktopmodules/Specifications/SpecificationDetails.aspx?specificationId=2262.

[10] 3GPP. *Technical Specification Group Services and System Aspects; Release 15 Description; Summary of Rel-15 Work Items*. TR 21.915. Version 15.0.0. Sept. 2019. URL: https://portal.3gpp.org/desktopmodules/Specifications/SpecificationDetails.aspx?specificationId=3389.

[11] 3GPP. *Technical Specification Group Services and System Aspects; Security architecture and procedures for 5G system*. TS 33.501. Version 16.1.0. Dec. 2019. URL: https://portal.3gpp.org/desktopmodules/Specifications/SpecificationDetails.aspx?specificationId=3169.

[12] 3GPP. *Technical Specification Group Services and System Aspects; Security architecture and procedures for 5G system*. TS 33.501. Version 15.6.0. Sept. 2019. URL: https://portal.3gpp.org/desktopmodules/Specifications/SpecificationDetails.aspx?specificationId=3169.

[13] 3GPP. *Technical Specification Group Services and System Aspects; Study on 5G Security Enhancement against False Base Stations*. TR 33.809. Version 0.8.0. Nov. 2019. URL: https://portal.3gpp.org/desktopmodules/Specifications/SpecificationDetails.aspx?specificationId=3539.

[14] 3GPP. *Technical Specification Group Services and System Aspects; Study on the security aspects of the next generation system*. TR 33.899. Version 1.3.0. Aug. 2017. URL: https://portal.3gpp.org/desktopmodules/Specifications/SpecificationDetails.aspx?specificationId=3045.

[15] 3GPP. *Technical Specification Group Services and System Aspects; System architecture for the 5G System (5GS)*. TS 23.501. Version 15.8.0. Dec. 2019. URL: https://portal.3gpp.org/desktopmodules/Specifications/SpecificationDetails.aspx?specificationId=3144.

[16] 3GPP. *Technical Specification Group Services and System Aspects; Vocabulary for 3GPP Specifications*. TR 21.905. Version 16.0.0. June 2019. URL: https://portal.3gpp.org/desktopmodules/Specifications/SpecificationDetails.aspx?specificationId=558.

[17] Android Studio. *Android Debug Bridge (adb)*. URL: https://developer.android.com/studio/command-line/adb.

[18] Android Studio. *Getting Started with the NDK*. URL: https://developer.android.com/ndk/guides.

[19] Android Studio. *Meet Android Studio*. URL: https://developer.android.com/studio/intro.

[20] Elaine Barker et al. *Recommendation for Pair-Wise Key-Establishment Schemes Using Discrete Logarithm Cryptography*. Tech. rep. National Institute of Standards and Technology, Apr. 2018. URL: https://csrc.nist.gov/publications/detail/sp/800-56a/rev-3/final.

[21] Daniel J. Bernstein. "Curve25519: New Diffie-Hellman Speed Records". In: *Proceedings of the 9th International Conference on Theory and Practice of Public-Key Cryptography*. PKC'06. New York, NY: Springer-Verlag, 2006, pp. 207–228. ISBN: 3540338519. DOI: 10.1007/11745853_14. URL: https://doi.org/10.1007/11745853_14.

[22] Giuseppe Cattaneo, Giancarlo De Maio, and Umberto Ferraro Petrillo. "Security Issues and Attacks on the GSM Standard: a Review". In: *Journal of Universal Computer Science*. Vol. 19. 16. Oct. 2013, pp. 2437–2452.

[23] Chi Cheng et al. "Securing the Internet of Things in a Quantum World". In: *IEEE Communications Magazine* 55.2 (2017), pp. 116–120.

[24] Quynh Dang. *Recommendation for Applications Using Approved Hash Algorithms*. Tech. rep. National Institute of Standards and Technology, Sept. 2012. URL: https://csrc.nist.gov/publications/detail/sp/800-107/rev-1/final.

[25] Morris Dworkin. *Recommendation for Block Cipher Modes of Operation: Methods and Techniques*. Tech. rep. National Institute of Standards and Technology, Dec. 2001. URL: https://csrc.nist.gov/publications/detail/sp/800-38a/final.

[26] ETSI SAGE. *First response on ECIES for concealing IMSI or SUPI*. Oct. 2017.

[27] Dan Forsberg et al. *LTE Security*. 2nd ed. John Wiley and Sons Ltd, 2013. ISBN: 9781118355589.

[28] Julie Fylkesnes Halland. *Public key cryptography for 5G private identification*. Project report in TTM4502. Department of Information Security and Communication Technology, NTNU – Norwegian University of Science and Technology, Dec. 2019.

[29] Byeongdo Hong, Sangwook Bae, and Yongdae Kim. "GUTI Reallocation Demystified: Cellular Location Tracking with Changing Temporary Identifier". In: *NDSS '18*. Feb. 2018. ISBN: 1-891562-49-5. DOI: http://dx.doi.org/10.14722/ndss.2018.23349.

[30] Syed Rafiul Hussain et al. "Privacy Attacks to the 4G and 5G Cellular Paging Protocols Using Side Channel Information". In: *NDSS '19*. Internet Society. Feb. 2019. ISBN: 1-891562-55-X. DOI: https://dx.doi.org/10.14722/ndss.2019.23442.

[31] Enrique Cobo Jiménez et al. "Subscription identifier privacy in 5G systems". In: *2017 international conference on selected topics in mobile and wireless networking (MoWNeT)*. IEEE. May 2017, pp. 1–8. DOI: 10.1109/MoWNet.2017.8045947. URL: https://ieeexplore.ieee.org/abstract/document/8045947.

[32] Brian W. Kernighan and Dennis M. Ritchie. *The C Programming Language*. 2nd ed. Prentice-Hall, Inc., 1988.

[33] Haibat Khan, Benjamin Dowling, and Keith Martin. "Identity Confidentiality in 5G Mobile Telephony Systems". In: *Security Standardisation Research*. Springer International Publishing, Jan. 2018, pp. 120–142. ISBN: 978-3-030-04761-0. DOI: 10.1007/978-3-030-04762-7_7.

[34] Haibat Khan and Keith Martin. *A Survey of Subscription Privacy on the 5G Radio Interface - The Past, Present and Future*. Cryptology ePrint Archive, Report 2020/101. Feb. 2020. URL: https://eprint.iacr.org/2020/101.

[35] Mohsin Khan and Valtteri Niemi. "Concealing IMSI in 5G Network Using Identity Based Encryption". In: *Network and System Security: 11th International Conference, NSS 2017*. Aug. 2017, pp. 544–554. ISBN: 978-3-319-64700-5. DOI: 10.1007/978-3-319-64701-2_41.

[36] Mohsin Khan, Valtteri Niemi, and Philip Ginzboorg. "IMSI-based Routing and Identity Privacy in 5G". In: *Proceeding of the 22nd Conference of FRUCT Association*. Apr. 2018.

[37] Mohsin Khan et al. "Defeating the Downgrade Attack on Identity Privacy in 5G". In: *Security Standardisation Research*. Ed. by Cas Cremers and Anja Lehmann. Cham: Springer International Publishing, 2018, pp. 95–119. ISBN: 978-3-030-04762-7. DOI: https://doi.org/10.1007/978-3-030-04762-7_6.

[38] Muzammil Khan, Attiq Ahmed, and Ahmad Reza Cheema. "Vulnerabilities of UMTS Access Domain Security Architecture". In: *2008 Ninth ACIS International Conference on Software Engineering, Artificial Intelligence, Networking, and Parallel/Distributed Computing*. Aug. 2008, pp. 350–355.

[39] Rabia Khan et al. "A Survey on Security and Privacy of 5G Technologies: Potential Solutions, Recent Advancements and Future Directions". In: *IEEE Communications Surveys & Tutorials* 22.1 (July 2019), pp. 196–248. DOI: 10.1109/COMST.2019.2933899. URL: https://ieeexplore.ieee.org/document/8792139.

[40] H. Krawczyk, M. Bellare, and R. Canetti. *HMAC: Keyed-Hashing for Message Authentication*. RFC 2104. Feb. 1997. URL: https://tools.ietf.org/html/rfc2104.

[41] A. Langley, M. Hamburg, and S. Turner. *Elliptic Curves for Security*. RFC 7748. RFC Editor, Jan. 2016. URL: https://www.rfc-editor.org/rfc/rfc7748.txt.

[42] Dindayal Mahto and Dilip Kumar Yadav. "RSA and ECC: A Comparative Analysis". In: *International Journal of Applied Engineering Research* 12.19 (2017), pp. 9053–9061.

[43] John Preuß Mattsson and Erik Thormarker. *What next in the world of post-quantum cryptography?* Mar. 2020. URL: https://www.ericsson.com/en/blog/2020/3/post-quantum-cryptography-symmetric-asymmetric-algorithms.

[44] Stig F. Mjølsnes and Ruxandra F. Olimid. "Easy 4G/LTE IMSI Catchers for Non-Programmers". In: *Computer Network Security*. Ed. by Jacek Rak et al. Cham: Springer International Publishing, 2017, pp. 235–246. ISBN: 978-3-319-65127-9.

[45] Stig F. Mjølsnes and Ruxandra F. Olimid. "Private Identification of Subscribers in Mobile Networks: Status and Challenges". In: *IEEE Communications Magazine* 57.9 (Sept. 2019), pp. 138–144. DOI: 10.1109/MCOM.2019.1800511. URL: https://ieeexplore.ieee.org/abstract/document/8847241.

[46] Stig F. Mjølsnes and Ruxandra F. Olimid. "The Challenge of Private Identification". In: *International Workshop on Open Problems in Network Security (iNetSec)*. May 2017, pp. 39–53. URL: https://hal.inria.fr/hal-01684219/document.

[47] Niels Möller. *Nettle — a low-level cryptographic library*. URL: https://www.lysator.liu.se/~nisse/nettle/.

[48] National Institute of Standards and Technology. *FIPS PUB 180-4: Secure Hash Standard (SHS)*. Aug. 2015. DOI: http://dx.doi.org/10.6028/NIST.FIPS.180-4.

[49] National Institute of Standards and Technology. *FIPS PUB 186-4: Digital Signature Standard (DSS)*. July 2013. DOI: https://doi.org/10.6028/NIST.FIPS.186-4.

[50] National Institute of Standards and Technology. *FIPS PUB 197: Advanced Encryption Standard (AES)*. Nov. 2001. DOI: https://doi.org/10.6028/NIST.FIPS.197.

[51] OpenSSL Software Foundation. *Changelog*. URL: https://www.openssl.org/news/changelog.html.

[52] OpenSSL Software Foundation. *Elliptic Curve Diffie Hellman*. URL: https://wiki.openssl.org/index.php/Elliptic_Curve_Diffie_Hellman.

[53] OpenSSL Software Foundation. *EVP*. URL: https://www.openssl.org/docs/man1.1.1/man7/evp.html.

[54] OpenSSL Software Foundation. *EVP_PKEY_derive*. URL: https://www.openssl.org/docs/man1.1.1/man3/EVP_PKEY_derive.html.

[55] OpenSSL Software Foundation. *OpenSSL*. URL: https://www.openssl.org.

[56] OpenSSL Software Foundation. *X25519*. URL: https://www.openssl.org/docs/manmaster/man7/X25519.html.

[57] *Public Key Cryptography for the Financial Services Industry - Key Agreement and Key Transport Using Elliptic Curve Cryptography*. Accredited Standards Committee X9, 2001.

[58] Altaf Shaik et al. “Practical attacks against privacy and availability in 4G/LTE mobile communication systems”. In: *23rd Annual Network and Distributed System Security Symposium (NDSS 2016)*. Internet Society, 2016. ISBN: 1-891562-41-X. DOI: 10.14722/ndss.2016.23236.

[59] Peter W. Shor. “Algorithms for quantum computation: discrete logarithms and factoring”. In: *Proceedings 35th Annual Symposium on Foundations of Computer Science*. Nov. 1994, pp. 124–134. DOI: https://doi.org/10.1109/SFCS.1994.365700.

[60] Ankush Singla et al. “Protecting the 4G and 5G Cellular Paging Protocols against Security and Privacy Attacks”. In: *Proceedings on Privacy Enhancing Technologies* 2020 (Jan. 2020), pp. 126–142. DOI: 10.2478/popets-2020-0008.

[61] William Stallings. *Cryptography and Network Security: Principles and Practice*. 6th ed. Pearson Education, 2014. ISBN: 978-0-13-335469-0.

[62] Daehyun Strobel. “IMSI Catcher”. In: *Chair for Communication Security, Ruhr-Universität Bochum* (July 2007).

[63] Christian Sørseth et al. “Experimental Analysis of Subscribers’ Privacy Exposure by LTE Paging”. In: *Wireless Personal Communications* 109.1 (Nov. 2019), pp. 675–693. URL: https://link.springer.com/article/10.1007/s11277-019-06585-7.

[64] The Standards for Efficient Cryptography Group. *SEC 1: Elliptic Curve Cryptography*. Version 2.0. May 2009.

[65] The Standards for Efficient Cryptography Group. *SEC 2: Recommended Elliptic Curve Domain Parameters*. Version 2.0. Jan. 2010.

Appendix A

Instructions to Run Programs

Several people volunteered to run the OpenSSL programs on their phones. To do this, they were given the following instructions, along with the programs:

```
-- On your Android phone:

1. Go to Settings > About phone, and locate the "Build number" field

2. Tap "Build number" 7 times, until Developer Options are enabled

3. Connect the phone to the computer with a USB cable

4. Go to Settings > Developer Options

5. Enable USB debugging

6. Set "Select USB configuration" to "Media Transfer Protocol (MTP)"

-- On your computer:

1. Download Android SDK Platform Tools:
https://developer.android.com/studio/releases/platform-tools

2. Extract the ZIP archive to your home folder

3. Download the Android programs to your home folder

-- Linux/MacOS:
4. Open terminal, and type the following commands
```

```
5. Navigate to the platform-tools folder:
$ cd ~/platform-tools

6. Check that the phone is connected:
$ adb devices
OR
$ ./adb devices

>> The output should be something like:
List of devices attached
FFY5T17B24010054 device

If the list is empty, make sure the phone it set up correctly, and
try again.

7. Transfer the programs to the phone:
$ adb push ~/program /data/local/tmp
OR
$ ./adb push ~/program /data/local/tmp

8. Open ADB shell:
$ adb shell
OR
$ ./adb shell

9. Navigate to the folder where the programs are stored:
$ cd /data/local/tmp

10. Make the programs executable:
$ chmod 555 program

11. Run the programs:
$ ./program

-- On your phone:

12. Disable Developer options
```

Appendix B
Performance Results

Screenshots from the performance studies are shown in Figures B.1, B.2, B.3, B.5 and B.6.

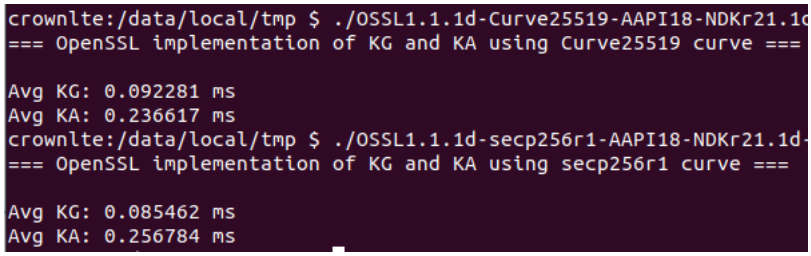

Figure B.1: Results from the Samsung Galaxy Note9.

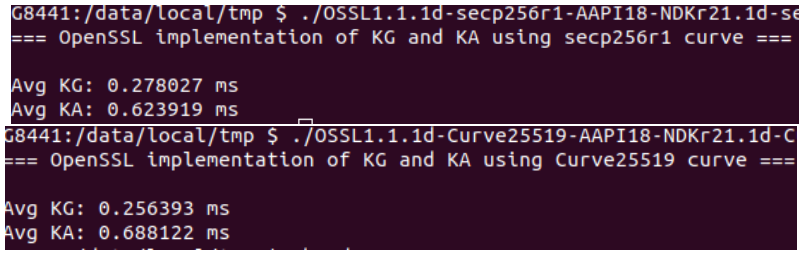

Figure B.2: Results from the Sony Xperia XZ1 Compact.

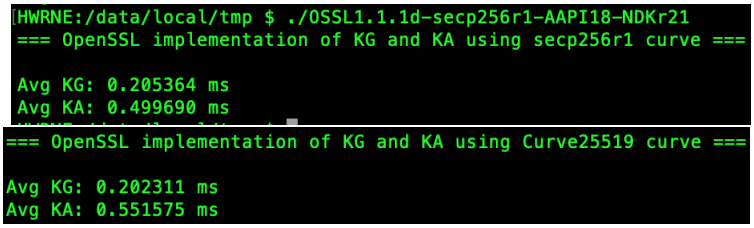

Figure B.3: Results from the Huawei Mate 10 Lite.

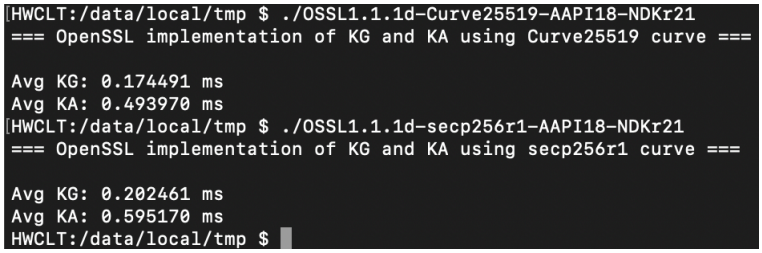

Figure B.4: Results from the Huawei P20 Pro.

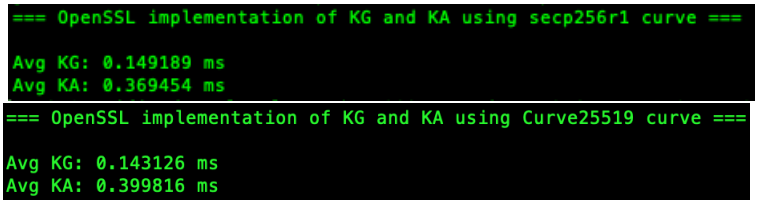

Figure B.5: Results from the Samsung Galaxy Tab S2.

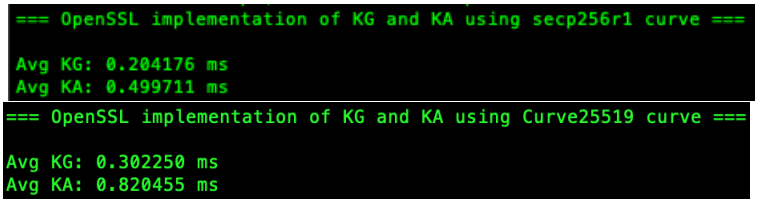

Figure B.6: Results from the Samsung Galaxy S5 Neo.

NTNU

Kunnskap for en bedre verden