People's Democratic Republic of Algeria
الجمهورية الجزائرية الديمقراطية الشعبية

Ministry of Higher Education and Scientific Research
وزارة التعليم العالي والبحث العلمي

ESI ECOLE NATIONALE SUPÉRIEURE D'INFORMATIQUE

المدرسة الوطنية للإعلام الآلي
(المعهد الوطني للتكوين في الإعلام الآلي سابقا)
Higher national school of computer science ESI
ex. INI (National IT training institute)

Engineer's Dissertation

For the Award of the Engineer's Degree in Computer Science

Option : Computer Systems and Software Engineering

A Blockchain and Zero Trust Framework for Secure Roaming in 5G Networks

Realised by :
Rezazi Mohamed Abdessamed

Supervised by :
Dr. Benabidallah Rymel (ESI)
Dr. Ghamri Douane (L3i)
Dr. Bouchiha Amine (L3i)
Mr. Bendada Moncef (L3i)

2024 / 2025

Acknowledgment

In the name of Allah, the All-Wise, the All-Knowing. Praise be to Allah who has guided us to this path, and without his guidance, we would not have found our way. All success is from him, and to him belongs all gratitude.

First and foremost, I wish to express my deepest and most heartfelt gratitude to the two pillars of my life, my beloved father and mother. Their unconditional love, endless sacrifices, and unwavering prayers have been my source of strength throughout this journey. Their constant encouragement and patience gave me the determination to persevere, and every achievement I reach is, above all, a reflection of their dedication. To my brothers and sister, I extend my warmest thanks for their support, care, and companionship. They have been by my side through challenges and triumphs alike, offering not only their time and energy but also their encouragement and faith in me.

I would also like to extend my appreciation to my teachers, mentors, and all those who have guided and enlightened me throughout my academic journey. Each of them, in their own way, has shaped my thinking and helped me grow. In particular, I would like to acknowledge my supervisors, Dr. Benabidallah Rymel and Dr. Amine Bouchiha, for their invaluable guidance, insightful feedback, and tireless support. Their expertise, patience, and encouragement have been instrumental in shaping this work, and I am profoundly grateful for the time and effort they invested in me.

To my friends and peers, thank you for the moments of collaboration, exchange, and shared learning. Your companionship made the demanding moments lighter and the successes more meaningful. I am equally grateful to everyone who, in one way or another, has had a positive impact on my personal and academic journey. Even the smallest words of encouragement have played an important role in helping me move forward.

Abstract

The evolution of fifth-generation (5G) mobile networks introduces unprecedented capabilities such as ultra-low latency, massive device connectivity, and enhanced bandwidth. These advancements, while enabling transformative applications across industries, also bring new challenges to traditional roaming systems. In particular, roaming settlement between mobile network operators (MNOs) remains complex, involving delayed reconciliation, high operational overhead, and growing concerns about fairness, transparency, and user privacy. As the number of 5G roaming connections is projected to increase significantly in the coming years, addressing these challenges is crucial for ensuring seamless and trustworthy connectivity.

To overcome these limitations, blockchain technology has been explored as a promising foundation for decentralized 5G roaming settlement. By leveraging smart contracts and distributed ledgers, blockchain-based approaches provide tamper-proof record-keeping, transparent billing, and automated settlement between operators without reliance on centralized intermediaries. Nevertheless, existing solutions face critical challenges related to scalability, due to the high volume of roaming transactions, and privacy, since sensitive user and billing data may be exposed on-chain. These issues limit the practicality of current blockchain-based roaming frameworks in real-world deployments.

In this thesis, we propose B5GRoam, a blockchain-based framework for secure and privacy-preserving roaming settlement in 5G networks. Our solution introduces a decentralized architecture that ensures trust, fairness, and transparency between operators while minimizing settlement delays and operational overhead. We design and implement the framework with a focus on scalability and privacy, and validate its feasibility through a proof-of-concept implementation and performance evaluation. The results demonstrate that B5GRoam offers an efficient, secure, and practical approach to handling roaming settlements in future 5G deployments.

Keywords : 5G, Roaming, Blockchain, Scalability, Privacy

ملخص

شهدت شبكات الاتصالات المتنقلة من الجيل الخامس (5G) نقلة نوعية من خلال إدخال قدرات جديدة مثل الكمون المنخفض جدًا، والقدرة على ربط عدد هائل من الأجهزة في وقت واحد، وزيادة عرض النطاق الترددي. هذه الميزات تفتح الباب أمام تطبيقات ثورية في قطاعات النقل، الصحة، الصناعة والترفيه. غير أن أنظمة التجوال التقليدية ما زالت تواجه تحديات متزايدة، خصوصًا على مستوى التسوية المالية بين مشغلي الشبكات، والتي تتسم بالتأخير، وارتفاع التكاليف التشغيلية، والاعتماد على وسطاء مركزيين، إضافةً إلى الإشكاليات المتعلقة بالشفافية، الإنصاف، وحماية خصوصية المستخدمين. ومع توقعات بارتفاع هائل في حجم اتصالات التجوال بالجيل الخامس في السنوات المقبلة، فإن معالجة هذه التحديات تُعد أمرًا حاسمًا لنجاح المنظومة الرقمية المستقبلية.

برزت تقنية البلوكتشين كأحد الحلول الواعدة للتغلب على هذه الإشكاليات، إذ توفر من خلال سجلاتها الموزعة وعقودها الذكية آلية لتوثيق المعاملات وضمان الشفافية، مع أتمتة عمليات التسوية المالية بين المشغلين دون الحاجة إلى وسطاء مركزيين. ومع ذلك، فإن الحلول الحالية القائمة على البلوكتشين ما زالت تعاني من تحديات رئيسية، أهمها مشكلة قابلية التوسع بسبب العدد الكبير من المعاملات، إضافةً إلى مخاطر الخصوصية، حيث يمكن أن تُكشف بيانات حساسة متعلقة بالاستخدام أو الفوترة على السلسلة. هذه القيود تجعل من الضروري تطوير أطر جديدة أكثر كفاءة وأمانًا لتمكين التجوال في بيئة الجيل الخامس.

في هذا البحث، نقترح إطارًا مبتكرًا أطلقنا عليه اسم B5GRoam ، وهو إطار يعتمد على البلوكتشين لتحقيق تسوية مالية للتجوال بشكل آمن، عادل، وشفاف، مع الحفاظ على خصوصية المستخدمين وتقليل التعقيدات التشغيلية. يقوم الإطار على بنية لامركزية تدعم الثقة بين المشغلين، وتوفر آليات تحقق قابلة للتدقيق، كما تم اختباره من خلال نموذج أولي يبرز جدواه العملية من حيث الأداء، الكفاءة، وقابلية التوسع. أظهرت النتائج أن B5GRoam يشكل مقاربة عملية وواعدة لإدارة التجوال في شبكات الجيل الخامس على نطاق عالمي.

الكلمات المفتاحية : شبكات الجيل الخامس، البلوكتشين، قابلية التوسع، الخصوصية.

Table of contents

General Introduction 1

I Background **3**

1 5G Technology **4**

1.1 Introduction 5

1.2 The evolution of Cellular networks from 1G to 5G 5

1.2.1 First Generation (1G) 5

1.2.2 Second generation (2G) 5

1.2.3 Third Generation (3G) 6

1.2.4 Fourth Generation (4G) 6

1.2.5 Fifth Generation (5G) 7

1.3 5G Technologies 7

1.3.1 Network Functions Virtualization 7

1.3.2 Software Defined Networking 8

1.3.3 SDN and NFV 9

1.4 5G Architecture 10

1.4.1 General Architecture 10

1.4.1.1 Component Perspective 10

1.4.1.2 Data flow perspective 11

1.4.2 5G core functions 11

1.4.3 Network Slicing 13

1.5 Summary 14

2 Roaming in 5G Networks **15**

2.1 Introduction 16

2.2 Roaming evolution 16

2.3 Types of Roaming 17

2.3.1 National/Internal Roaming . . . 17
2.3.2 International Roaming . . . 17
2.4 Involved 5G Core Network Functions in Roaming . . . 18
2.5 Roaming Approaches . . . 19
2.5.1 Home-Routed (HR) . . . 19
2.5.2 Local-Breakout (LBO) . . . 20
2.6 Roaming Agreements and Charging . . . 21
2.7 Summary . . . 23

3 Blockchain Technology 24
3.1 Introduction . . . 25
3.2 Definition . . . 25
3.3 Blockchain Key Elements . . . 27
3.3.1 Transaction . . . 27
3.3.2 Block . . . 27
3.3.3 Distributed Ledger . . . 27
3.3.4 Consensus Protocol . . . 28
3.3.4.1 Permissionless Consensus . . . 28
3.3.4.2 Permissioned Consensus . . . 29
3.4 Blockchain Properties . . . 29
3.4.1 Decentralization . . . 30
3.4.2 Immutability . . . 30
3.4.3 Transparency . . . 30
3.4.4 Security and Fault Tolerance . . . 30
3.5 Categories of Blockchain . . . 30
3.6 Smart Contracts . . . 31
3.7 Limitations and Challenges . . . 31
3.8 Summary . . . 32

4 Review of Blockchain-based 5G Roaming Contributions 33
4.1 Introduction . . . 34
4.2 Threat Model . . . 34
4.3 Literature Review . . . 35
4.4 Security Analysis . . . 37
4.5 Challenges and Limitations . . . 39

4.6 Conclusion 40

II Contribution **42**

5 Contribution **43**

5.1 Introduction 43
5.2 Design Objectives and Requirements 44
5.2.1 Design Objectives 44
5.2.2 Functional Requirements 44
5.2.3 Non-Functional Requirements 45
5.3 System Design 45
5.3.1 System Actors 46
5.3.1.1 User Equipment (UE) 46
5.3.1.2 Home Mobile Network Operator (HMNO) 46
5.3.1.3 Visited Mobile Network Operator (VMNO) 47
5.3.2 System Layers 47
5.3.2.1 5G Core Layer 47
5.3.2.2 5G Roaming Interaction Layer 47
5.3.2.3 Prover Layer 48
5.3.2.4 Blockchain Layer 49
5.3.3 Framework Modules 49
5.3.3.1 5G Roaming Modules 50
5.3.3.2 TEE Module 50
5.3.3.3 ZK Module 51
5.3.3.4 Networking Provider 51
5.3.3.5 Smart Contracts Module 51
5.3.3.6 Node Core 51
5.4 System Flow 52
5.4.1 Roaming Agreements Creation 52
5.4.2 Subscriber Roaming Session Initiation and Authorization 55
5.4.3 Usage Commitment via TEE 56
5.4.4 Zero-Knowledge Billing and Settlement Verification 58
5.4.5 ZK-Rollups 59

6 Zero Knowledge Proof Generation **62**

6.1 Introduction . . . 63
6.2 ZKP Statement and Circuit Logic . . . 63
6.2.1 Circuit Structure and Constraints . . . 64
6.3 Selection of Zero-Knowledge Proof Construction . . . 64
6.3.1 Comparison of Modern ZK Constructions . . . 65
6.4 Proof Generation and Verification Workflow . . . 66
6.5 ZK SNARKs Security Properties . . . 68
6.5.1 Soundness . . . 68
6.5.2 Completeness . . . 68
6.5.3 Zero-Knowledge . . . 68

7 Implementation and Proof of Concept **69**
7.1 Introduction . . . 69
7.2 Used Frameworks and Tools . . . 69
7.3 Programming Languages . . . 70
7.4 Libraries . . . 70
7.5 SNARK Proofs . . . 71
7.6 Roaming Sessions Management . . . 72
7.7 Conclusion . . . 77

8 Tests and Results **78**
8.1 Introduction . . . 78
8.2 ZK Snarks Constructions Benchmark . . . 78
8.2.1 Proof Generation Overhead: Memory and Time Analysis . . . 78
8.2.2 On-Chain Proofs Verification Overhead: Gas Consumption Analysis 80
8.3 B5GRoam Benchmark . . . 80
8.3.1 Benchmark Environment . . . 80
8.3.2 Benchmark Configuration . . . 81
8.3.2.1 Caliper Network Configuration . . . 81
8.3.2.2 Caliper Benchmark Configuration . . . 82
8.3.2.3 Caliper Workload Configuration . . . 83
8.3.3 Results and Discussions . . . 84
8.4 Conclusion . . . 88
General Conclusion . . . 90

List of Figures

1.1 NFV reference architectural framework (Blanco et al. 2017) 8
1.2 SDN architecture layers 9
1.3 5G Core Network Functions (Open5Gcore 2022) 10
1.4 Illustration of 5G network slices operating over a shared multi-vendor, multi-access infrastructure (Ordonez-Lucena et al. 2017) 13

2.1 Home-Routed architecture 20
2.2 Local-Breakout architecture 21

3.1 Blockchain Architecture 25
3.2 Merkle Tree Structure 26
3.3 Blockchain Consensus Protocols Categorization 28

5.1 System Layers 48
5.2 Framework modules 50
5.3 B5GRoam flow 52
5.4 Creating Agreement Sequence Flow 54
5.5 Starting Roaming Session Sequence Flow 56
5.6 UE Commitment Diagram 57
5.7 Terminating Roaming Session Sequence Flow 59
5.8 ZK-Rollup Illustration 61

6.1 ZK Proof Generation Flow 66

7.1 SNARK Circuit Inputs 71
7.2 SNARK Circuit Logic 72
7.3 AgreementFactory States 73
7.4 Create Agreement Code 73
7.5 Agreement Contract States 74

7.6 Start Roaming Session Code 75
7.7 Submit CDR Code 76
7.8 Terminating Roaming Session Code 77

8.1 ZK Proof Memory Consumption for each construction 79
8.2 ZK Proof time for each construction 79
8.3 ZK Proof Verification's Gas Consumption 80
8.4 Caliper Network Configuration 82
8.5 Caliper Benchmark Configuration 83
8.6 Caliper Workload Configuration 84
8.7 L1 Latency under different transaction loads 85
8.8 L1 TPS under different transaction loads 86
8.9 L1 Latency and Throughput under different transaction loads for on-chain settlement 87

List of Tables

4.1 Security analysis of existing frameworks against the proposed threat model. 39

6.1 Summary of existing SNARK constructions 66

8.1 Blockchain Benchmark Environment 81

8.2 Benchmark results comparing Dual Layer and Single Layer transaction costs. Batch size represents the number of L2 TXs within each batch. . . . 88

List of Acronyms

1G First Generation
2G Second Generation
3G Third Generation
4G Fourth Generation
5G Fifth Generation
AAA Authentication, Authorization and Accounting
ABI Application Binary Interface
AF Application Function
AMF Access and Mobility Management Function
AMPS Advanced Mobile Phone System
AMTS Advanced Mobile Telephone System
AUSF Authentication Server Function
B5GRoam Blockchain-enabled 5G Roaming Framework
BS Base Station
CDMA Code Division Multiple Access
CDMA2000 Code Division Multiple Access 2000
CN Core Network
CRS Common Reference String
CU Central Unit
DLT Distributed Ledger Technology
DN Data Network
DSL Domain-Specific Language
DU Distributed Unit
EDGE Enhanced Data rates for GSM Evolution
EVM Ethereum Virtual Machine
FDMA Frequency Division Multiple Access
gNB gNodeB
GSM Global System for Mobile Communications
HMNO Home Mobile Network Operator
HR Home-Routed
HSS Home Subscriber Server
IMT2000 International Mobile Telecommunications-2000
IMTS Improved Mobile Telephone Service
IoT Internet of Things
IPX IP Exchange
JSON JavaScript Object Notation
L1 Layer 1
L2 Layer 2
LBO Local Breakout

LTE Long Term Evolution
MANO Management and Orchestration
MNO Mobile Network Operator
MTS Mobile Telephone System
NEF Network Exposure Function
NF Network Function
NFV Network Functions Virtualization
NFVI NFV Infrastructure
NSSF Network Slice Selection Function
NSA Non-Standalone
PCF Policy Control Function
PBFT Practical Byzantine Fault Tolerance
PK Proving Key
PLONK Permutations over Lagrange-bases for Oecumenical Noninteractive arguments of Knowledge
PoC Proof of Concept
PoS Proof of Stake
PoW Proof of Work
PTT Push to Talk
P2P Peer-to-Peer
QoS Quality of Service
R1CS Rank-1 Constraint System
RAN Radio Access Network
REE Rich Execution Environment
SA Standalone
SDK Software Development Kit
SDN Software Defined Networking
SIM Subscriber Identity Module
SMF Session Management Function
SMS Short Message Service
SNARK Succinct Non-Interactive Argument of Knowledge
TDMA Time Division Multiple Access
TD-SCDMA Time Division Synchronous Code Division Multiple Access
TEE Trusted Execution Environment
UDM Unified Data Management
UDR Unified Data Repository
UE User Equipment
UMTS Universal Mobile Telecommunication System
UPF User Plane Function
URLLC Ultra-Reliable Low Latency Communication
VK Verifying Key
VM Virtual Machine
VMNO Visited Mobile Network Operator

VNF Virtualized Network Function
WCDMA Wideband Code Division Multiple Access
ZKP Zero-Knowledge Proof
zk-SNARK Zero-Knowledge Succinct Non-Interactive Argument of Knowledge
zk-STARK Zero-Knowledge Scalable Transparent Argument of Knowledge

General Introduction

The evolution of mobile communication has reached a pivotal stage with the deployment of fifth-generation (5G) networks. Compared to its predecessors, 5G introduces transformative capabilities, including ultra-low latency, enhanced bandwidth, and the ability to support massive device connectivity. These advances are expected to revolutionize diverse industries, including transportation, healthcare, manufacturing, and entertainment, thereby establishing 5G as a cornerstone of the digital society.

Within this new generation of mobile networks, roaming continues to play a fundamental role. Roaming enables subscribers to seamlessly access mobile services outside the coverage of their home operator, whether traveling across regions, nationally, or internationally. Although roaming has been supported since earlier generations, the scale, heterogeneity, and demanding performance requirements of 5G introduce new layers of complexity.

Traditional roaming systems, while effective in earlier contexts, face significant limitations in the 5G era. Settlement between mobile network operators (MNOs) often suffers from delayed reconciliation, high operational overhead, and dependence on centralized intermediaries. Furthermore, ensuring fair billing, maintaining service quality, and safeguarding user privacy have become increasingly challenging in a globally interconnected telecom ecosystem. With projections estimating hundreds of millions of 5G roaming connections in the coming years, addressing these challenges is essential to fully realizing the potential of 5G.

Decentralized technologies have emerged as promising enablers in this context. Blockchain, with its immutable ledger and programmable automation through smart contracts, offers transparency, accountability, and trust without relying on centralized parties. However, while blockchain integration brings important benefits, it also introduces new challenges. Chief among them are scalability, since the number of participants and transactions in real-world 5G roaming scenarios can be extremely large, and privacy, given the inherent transparency of blockchain ledgers that may expose sensitive information among participants.

These unresolved challenges motivate the central research question of this thesis: How can roaming settlement in 5G be made secure, privacy-preserving, and scalable while leveraging blockchain as a foundation for trust and transparency?

The objective of this work is to design and implement B5GRoam, a blockchain-based framework grounded in the principles of decentralization and zero trust, that enables secure, transparent, and privacy-preserving roaming settlement between mobile network

operators.

This thesis is structured into two major parts. The first part presents the theoretical and contextual background, including an overview of roaming evolution, the architecture of the 5G core network, and a review of blockchain-based roaming solutions. The second part details the design and implementation of B5GRoam, its underlying cryptographic proof mechanisms, and a proof-of-concept evaluation that assesses feasibility, efficiency, and scalability.

Part I

Background

Chapter 1

5G Technology

1.1 Introduction

Mobile Networks have evolved rapidly over the last two decades, enhancing the user experience and the quality of service, unlocking new horizons of devices and people connectivity, and changing how people interact and communicate in each generation. 5G is the newest and latest of the mobile network generations, it is a huge evolution of the previous generations designed to endure the demands of a fully connected world, as it sets the bar very high compared to its predecessors regarding its architecture, performance, energy efficiency, and cost efficiency, introducing new capabilities, and enabling new use cases.

In this chapter, we introduce the evolution of mobile networks to 5G, the benefits and goals of 5G, its capabilities and new use cases, its architecture and enabling technologies, and the challenges it presents.

1.2 The evolution of Cellular networks from 1G to 5G

Mobile networks are evolving rapidly due to increased demands and the growing number of users; the transition from 1G to 5G took only around 40 years. Every generation of mobile networks, from the 1980s when they initially emerged to the first introduction of 5G in 2020, has brought about new developments and unlocked new capabilities.

1.2.1 First Generation (1G)

In the 1980s, the first-generation (1G) cellular networks were introduced using analog technology, operating at frequencies between 800 and 900 MHz with a channel capacity of 30 KHz. These networks employed Frequency Division Multiple Access (FDMA), assigning specific frequency bands to channels, which limited the number of simultaneous users. The first mobile phones, based on Advanced Mobile Phone System (AMPS) technology and completed in the early 1990s, provided basic voice services but faced numerous challenges, including poor voice quality, short battery life, large device size, limited capacity, unreliable handoffs, low security, and inefficient spectrum use. AMPS also introduced technologies like Mobile Telephone System (MTS), Advanced Mobile Telephone System (AMTS), Improved Mobile Telephone Service (IMTS), and Push to Talk (PTT). (Vora 2015)

1.2.2 Second generation (2G)

The second-generation (2G) cellular networks, introduced in the early 1990s with GSM as the leading standard, marked the transition to digital communication. Compared to 1G, 2G provided improved voice quality, faster data transfer rates, and enhanced network capacity through technologies like Time Division Multiple Access (TDMA) and Code Division Multiple Access (CDMA), which allowed multiple users to share a single channel.

One of the key innovations was Short Message Service (SMS), enabling users to send text messages of up to 160 characters, revolutionizing quick and efficient communication. Additionally, Enhanced Data rates for GSM Evolution (EDGE) brought basic mobile internet access with data transfer rates up to 384 kbps, and the introduction of SIM cards allowed users to switch between devices easily while maintaining personal information. Despite data speeds up to 64 Kbps in standard 2G, it could not support complex data like videos. However, advancements with 2.5G introduced packet-switched and circuit-switched domains, achieving data rates up to 144 Kbps and enabling phone calls, emails, web browsing, and MP3 downloads, laying the groundwork for mobile internet use. (Vora 2015)

1.2.3 Third Generation (3G)

Launched in 2000, 3G aimed to deliver high-speed data services and introduced significant advancements in mobile connectivity. Built on the GSM platform, it utilized packet switching to achieve data speeds of up to 14 Mbps and employed a Wide Band Wireless Network to enhance call clarity. Operating at 2100 MHz with a bandwidth of 15-20 MHz, 3G enabled services like high-speed internet, video streaming, global roaming, and video chatting. Its key features included speeds of up to 2 Mbps, compatibility with smartphones, increased bandwidth for web applications and multimedia, faster communication, support for large email transfers, and more versatile connectivity.

Despite its advantages, 3G faced various challenges such as expensive licensing fees, infrastructure development hurdles, high bandwidth requirements, and costly phones with large sizes. The European version was known as UMTS (Universal Mobile Telecommunication System), the American variant as CDMA2000, and China introduced its own standard called TD-SCDMA under IMT2000. WCDMA served as the air-interface technology for UMTS. (Vora 2015)

1.2.4 Fourth Generation (4G)

Introduced in late 2009, 4G significantly enhanced data rates, offering download speeds of up to 100 Mbps, building on the capabilities of 3G. It introduced new services like Multi-Media Newspapers and high-quality video streaming with improved clarity, ensuring faster and more efficient data transmission.

Powered by LTE (Long Term Evolution) technology, 4G was designed to meet the Quality of Service (QoS) and speed demands of emerging applications such as wireless broadband, video chat, mobile TV, HDTV content, Digital Video Broadcasting (DVB), and advanced voice and data services.

Key features of 4G include speeds ranging from 10 Mbps to 1 Gbps, high-quality video streaming, enhanced security, the ability to deliver services anytime and anywhere, expanded multimedia options, and a low cost per bit.

Despite all these advantages, 4G presents downsides such as high battery usage on mobile devices, complex hardware implementation, and the need for expensive equipment to establish the network. (Vora 2015)

1.2.5 Fifth Generation (5G)

The latest generation of wireless communication technology, 5G, delivers improved network capacity, lower latency, and faster data transfer rates compared to 4G. Designed to support diverse applications, 5G caters to enhanced mobile broadband, large-scale machine communication, and ultra-low-latency use cases (Vora 2015). It offers three key features (Dangi et al. 2021): **Enhanced Mobile Broadband (eMBB)**, which provides higher bandwidth, high-speed internet connectivity, and advanced capabilities using a non-standalone design; **Massive Machine Type Communication (eMTC)**, introduced in the 3GPP's 13th specification, enabling low-cost, high-bandwidth communication with minimal energy consumption, ideal for IoT applications due to its excellent data service, extended coverage, and low-power operation; and **Ultra-Reliable Low Latency Communication (URLLC)**, which ensures extremely low latency, exceptional reliability, and superior Quality of Service (QoS), surpassing traditional mobile network architecture.

1.3 5G Technologies

5G relies heavily on advanced enabling technologies, transforming traditional network infrastructure to be more flexible, efficient, and scalable. Technologies like Software-Defined Networking (SDN) and Network Functions Virtualization (NFV) are the critical ones in effectively improving network management, resource allocation, and service delivery. These technologies enable a wide range of 5G use cases, from enhanced mobile broadband up to ultra-reliable low-latency applications, thanks to decoupling network functions from hardware and thus enabling dynamic network programmability. Individually, they form the backbone of 5G's adaptability and performance in meeting modern connectivity demands.

1.3.1 Network Functions Virtualization

Network Function Virtualization is the new paradigm in designing and deploying network services, whereby network functions are decoupled from dedicated hardware appliances and run as virtualized software on commodity hardware. This increases flexibility, reduces costs, and accelerates the deployment of services. The modular architecture enables service providers to install, scale, and optimize network services with great efficiency. This decreases dependency on proprietary hardware and enables innovation in network management.

As illustrated in Fig. 1.1, the NFV, according to the European Telecommunications Standards Institute, consists of three major elements that include the following (Blanco et al. 2017):

- **Virtualized Network Functions (VNFs):** These are software programs that implement respective network functions and are executed in virtualized environments instead of physical hardware, for example, firewall, load balancer, and router.
- **NFV Infrastructure (NFVi):** NFVI is the physical and virtual hardware and software environment in which VNFs are deployed. It consists of computing, storage,

and network resources that are abstracted to facilitate flexible and efficient resource allocation.

- **NFV Management and Orchestration (MANO):** MANO is the framework responsible for the management and orchestration of both VNFs and NFVI, ensuring the seamless orchestration of network functions. This is generally done using orchestration tools like Kubernetes.

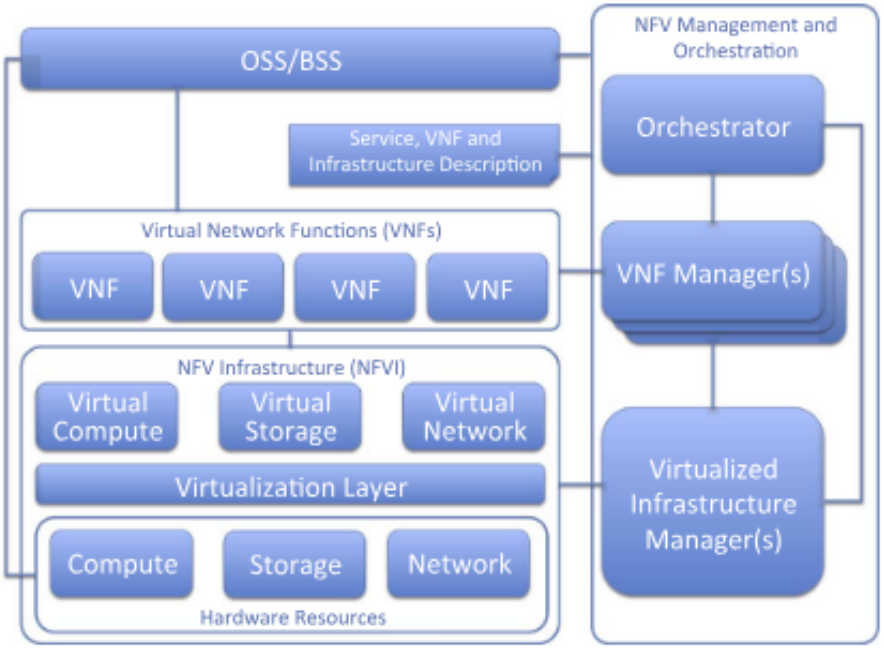


Figure 1.1 – NFV reference architectural framework (Blanco et al. 2017)

1.3.2 Software Defined Networking

Software-Defined Networking (SDN) emerged as a response to the limitations of traditional network architectures, which relied on specialized hardware and tightly coupled control and data planes, making networks rigid and hard to manage. SDN technology utilizes software-defined centralized network management and dynamic network configuration programming to give scalability and flexibility for 5G network provisioning and end-to-end network performance and monitoring.

SDN architecture is structured on three primary layers (Blanco et al. 2017): the Data plane (or called infrastructure layer) which consists of the network elements, the Control plane which acts as the central Controller, and the Application plane which consists of the different applications. These layers communicate with each other through protocols like OpenFlow (Fig. 1.2). The basic SDN principle is based on the separation of the data plane from the control plane and the logical centralization of all the control functions.

In conventional networks, the control plane is integrated with the data plane within each device. Centralizing the control plane in SDN brings several advantages, including simplified network management, enhanced programmability, and streamlined service upgrades. Unlike traditional networks that require manual configuration for each device, SDN enables network administrators to easily modify and upgrade services from a

centralized source, which is the SDN controller.

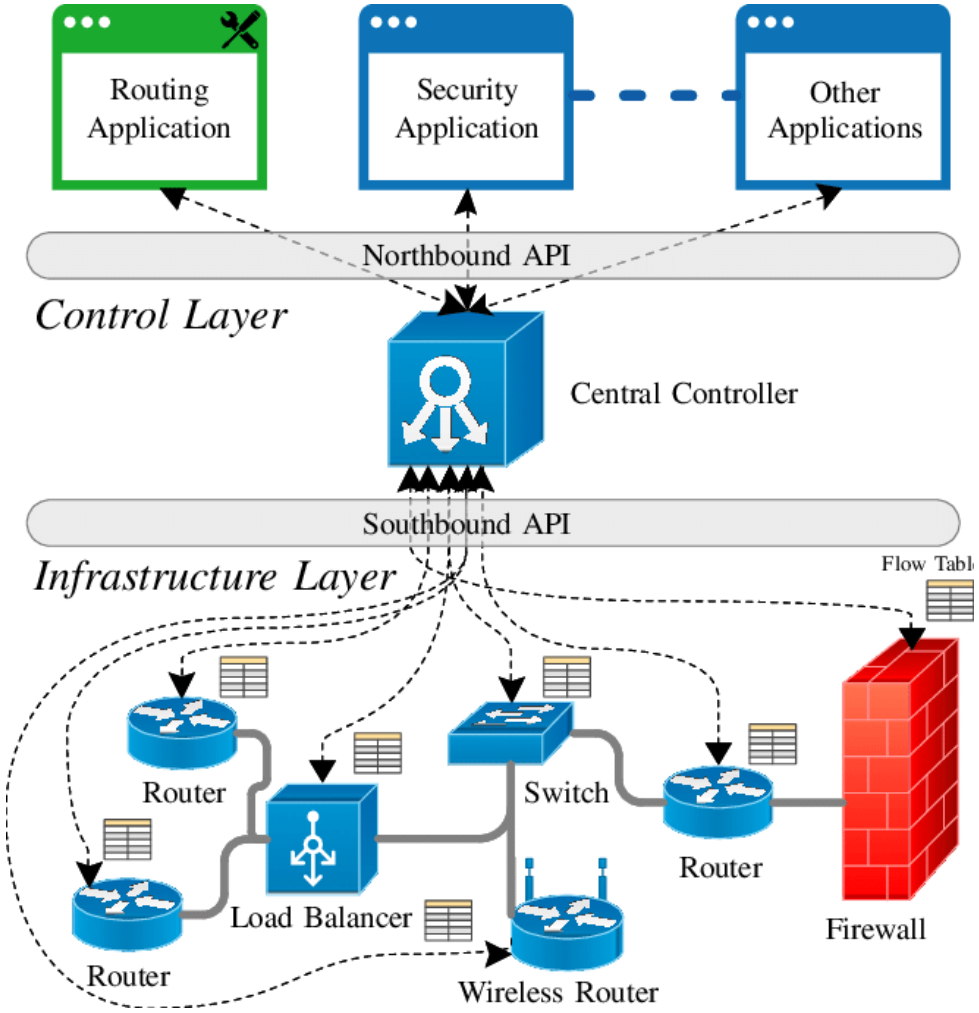


Figure 1.2 – SDN architecture layers

1.3.3 SDN and NFV

NFV and SDN operate independently, although they share certain similarities. Both utilize virtualization and network abstraction, but their methodologies for function separation and resource abstraction are different.

SDN achieves its objectives by isolating network forwarding functions from network control functions, aiming to establish a centrally manageable and programmable network. On the other hand, NFV abstracts network functions from hardware. NFV complements SDN by establishing the network virtual infrastructure on which SDN software can operate.

Combining NFV and SDN is possible, depending on the implementation goals, and both make use of standard, widely available hardware. The combined use of NFV and SDN facilitates the creation of a network architecture characterized by enhanced flexibility, programmability, and efficient resource utilization. (Sharma et al. 2021).

1.4 5G Architecture

In this section, we will illustrate the 5G network's architecture.

1.4.1 General Architecture

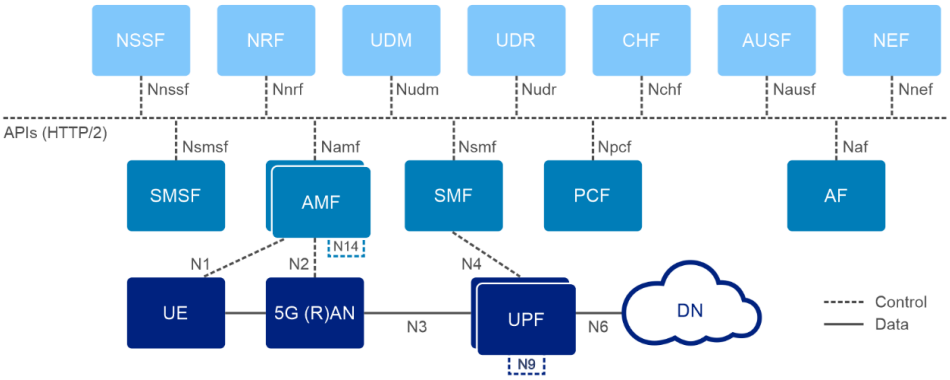


Figure 1.3 – 5G Core Network Functions (Open5Gcore 2022)

As presented in Figure 1.3, the 5G network architecture can be viewed through two different perspectives, the component perspective and the data flow perspective:

1.4.1.1 Component Perspective

The general architecture of mobile networks does not change much between generations; it always contains the same components: the core network, the radio access network, and the user equipment.

- **Core Network:** The core network provides the essential functions needed for a complete network. It manages important tasks such as user access, authentication, security, and mobility, especially when users move between different radio cells. This is why it's often called the control tower of the 5G network. Additionally, the core network oversees Quality of Service (QoS) and ensures smooth interaction with other networks. The core network acts as the central hub, coordinating the key functions that are necessary for the network to operate effectively and provide a good user experience.
- **Radio Access Network:** The Radio Access Network (RAN) is made up of antennas and base stations (BSs) that wirelessly send and receive signals. Besides the hardware, the RAN includes the operations and protocols that manage radio resources such as frequency, power, and modulation. The main components of the 5G RAN are the User Equipment (UE), the gNodeB (gNB), and the Distribution Unit (DU). The base station, also known as a cell or gNB, is responsible for covering a specific geographic area. The DU and the Central Unit (CU) can be separated to form the gNB, and they might be located together or apart. The DU handles

user plane tasks like data processing and transmission, while the CU takes care of control plane tasks such as authentication and mobility management.

There are two primary configurations for the 5G RAN architecture: Non-Standalone (NSA) and Standalone (SA). In NSA mode, the 5G RAN relies on the existing 4G LTE network for certain control features, including mobility and signaling. On the other hand, in SA mode, the 5G RAN operates independently and connects directly to the 5G Core Network (CN), allowing for a fully autonomous 5G experience.

- **User Equipment:** Which represents the user's mobile device.

1.4.1.2 Data flow perspective

Following this perspective, the architecture is divided into two planes, the control plane and the user plane.

The control plane includes all the administrative network functions that handle management tasks such as user authentication. User data packets do not pass through the control plane. It is designed using a service-based architecture, where multiple network functions communicate through service-based interfaces. These interfaces are converted into standard HTTP/2 endpoints with service discovery, allowing each network function to find and interact with the others. This setup is shown with dotted lines in Fig. 1.3.

The data plane, also known as the user plane, includes the parts of the network that handle user data packets. This includes the user equipment (UE), the Radio Access Network (RAN), and the User Plane Function (UPF). In Fig. 1.3, the data plane is shown with straight lines.

A 5G core network consists of all the network functions found in the control plane, along with the UPF. In a typical 5G setup, the core network operates on a centralized server that manages the traffic from the UEs. The data packets are sent from the UE and go through the RAN, which then transfers them to the UPF through potentially long routes. Once the packets reach the UPF, they are directed to the data network (DN). As a result, some of the latency that users experience is caused by the time it takes to transfer packets from the RAN to the UPF.

1.4.2 5G core functions

The components of the 5G Core Network (5G-CN), also known as Network Functions (NFs), have been significantly streamlined to better support various data services and requirements. Most of these components are software-based, which allows them to be easily modified and updated as needed. This software-centric approach makes the network more flexible, adaptable, and scalable.

The 5G core network is composed of many functions (Open5Gcore 2022). In the following, we present the most important ones:

- **AMF (Access and mobility management function):** The Access and Mobility Management Function is responsible for handling connection and mobility-related tasks for user equipment such as smartphones or IoT devices. When a device connects to the 5G network, the AMF manages the initial registration, authentication,

and establishment of the connection. It also oversees mobility management, which includes tracking the device's location as it moves and ensuring seamless handovers between different cell towers or base stations.

- **AUSF (Authentication Server Function):** The AUSF handles the authentication process. When a user device attempts to connect to the network, the AUSF verifies the device's and user's identities to ensure they are authorized to access network services. It works in conjunction with other security functions like UDM to manage authentication credentials and protocols.
- **SMF (Session management function):** The Session Management Function handles the establishment, maintenance, and termination of data sessions for user devices. When a device initiates a data transfer, the SMF is responsible for setting up the necessary network pathways, allocating IP addresses, and managing the data flow to ensure smooth communication. It also monitors active sessions, making adjustments as needed to maintain optimal performance and terminate sessions when they are no longer required.
- **PCF (Policy Charging Function):** The Policy Control Function is responsible for defining and managing network policies that govern how resources are allocated and used within the 5G network. It determines Quality of Service (QoS) levels, prioritizes different types of traffic, and enforces rules related to data usage and access control.
- **AF (Application function):** The Application Function (AF) interfaces with external applications and services to request specific network behaviors or resources.
- **UDM (Unified Data Management):** The Unified Data Management is a centralized unified database for all 5G subscribers' information and permissions, which can use an external DB as UDR (Unified data repository). It holds the secret keys and tracks which AMF each UE is connected to.
- **NRF (Network Repository Function):** The Network Repository Function acts as a central directory within the 5G core network, maintaining information about all available network functions and their capabilities. It allows different network functions to discover and communicate with each other by providing details such as service profiles and supported interfaces.
- **NEF (Network Exposure Function):** The Network Exposure Function serves as an intermediary between the 5G core network and external applications or services.
- **NSSF (Network Slice Selection Function):** The Network Slice Selection Function is responsible for selecting the appropriate network slice for a user or application based on specific criteria and requirements. Network slicing allows the 5G network to be partitioned into multiple virtual networks, each tailored to different use cases such as enhanced mobile broadband, IoT, or mission-critical communications. The NSSF evaluates factors like service type, quality of service needs, and user preferences to assign the most suitable slice.
- **UPFs (User Plane Functions):** The User Plane Function is a critical component that handles the actual transmission of user data within the 5G network. It manages data forwarding, routing, and packet inspection to ensure efficient and high-speed data delivery between user devices and external networks such as the internet. The UPF is responsible for enforcing Quality of Service (QoS) policies, managing traffic flows, and optimizing data paths to minimize latency and maximize throughput.

1.4.3 Network Slicing

Network slicing refers to the creation of distinct, end-to-end virtual networks that operate over a shared physical infrastructure, each tailored to deliver a pre-defined quality of service (QoS). This paradigm enables multiple isolated logical networks to coexist within the same physical environment, with each slice configured to meet the specific performance, security, and functional requirements of its intended application (Ordonez-Lucena et al. 2017), as illustrated in Fig. 1.4.

The fundamental objective of network slicing is to enable service differentiation through the segmentation of resources. By abstracting the underlying infrastructure, operators can assign dedicated or shared resources to each slice, ensuring that its performance remains independent from other slices. This approach can be applied to both the radio access network and the core network, offering a high degree of flexibility in terms of deployment and scalability. Additionally, network slicing can be implemented within infrastructures owned by a single operator or across multiple operators, leveraging virtualization and orchestration technologies to ensure isolation and service reliability.

- **Radio Access Network Slicing:** In 5G systems, the RAN is capable of partitioning available spectrum into sub-frequency blocks. This capability enables operators to allocate radio resources in alignment with the needs of specific services or user groups, ensuring that latency, throughput, and reliability targets are met.
- **Core Network Slicing:** The virtualized nature of the 5G core network allows its functions to be dynamically instantiated and placed across heterogeneous environments. The orchestration process allocates compute, storage, and networking resources in such a way that each core network slice operates in accordance with its defined service-level agreements (SLAs).

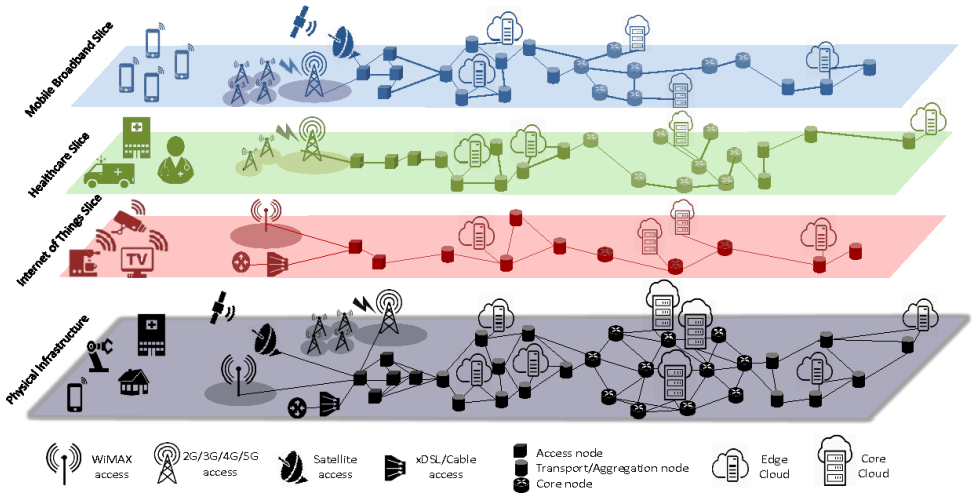


Figure 1.4 – Illustration of 5G network slices operating over a shared multi-vendor, multi-access infrastructure (Ordonez-Lucena et al. 2017)

1.5 Summary

In conclusion, this chapter provided a comprehensive overview of 5G technology, exploring the generational evolution of mobile networks, the new generations' capabilities, use cases, and enabling technologies such as virtualization, software-defined network, and network function virtualization.

The chapter also explained the architectural components of 5G, including radio access and core networks, and the core network functions with an overview of network slicing.

Chapter 2

Roaming in 5G Networks

2.1 Introduction

Roaming extends the coverage of a home operator's services, allowing its mobile users to use those services within another operator's network, which may be in another country (international roaming) or in the same country (national roaming). (Keller et al. 2021)

When it comes to roaming, the introduction of 5G brings both new opportunities and challenges. Recent reports indicate that by 2027, there will be a total of 526 million 5G roaming connections, a significant increase from 53 million in 2023 (ETTelecom 2023). Mobile network operators expect that roaming services will continue to function smoothly, whether their partners are using the Evolved Packet System (EPS), LTE, 5G, or a combination of these technologies (Keller et al. 2021). While 5G roaming shares many similarities with previous generations, there are still some important differences that need to be addressed.

Technically, the subscriber's registered network is known as the Home Public Land Mobile Network (HPLMN). Contrarily, the Visited Public Land Mobile Network (VPLMN) is the network that a subscriber briefly roams when beyond the HPLMN's limits.

In this chapter, we begin by describing the evolution of roaming from 2G to 5G mobile networks, highlighting the technological shifts that have shaped its development. We then examine the different types of roaming and the various approaches employed to enable it. Finally, the discussion focuses on the core 5G network functions involved in the roaming process, underscoring their critical role in delivering seamless connectivity across operators and borders.

2.2 Roaming evolution

Roaming has changed a lot over the years, evolving from its beginnings in 1G to the advanced features we see in today's 5G networks. In the early days of cellular networks, roaming was a new and limited function. It was only possible between networks that used the same analog technology, which was the case with 1G networks. Because of this, only users who needed to roam could do so, and it was often difficult and expensive. Additionally, since roaming agreements between different networks were not standardized, it was hard for consumers to know if they could use a particular network while roaming.

With the introduction of 2G networks in the 1990s, roaming became easier and more reliable. 2G networks used digital signaling instead of analog, which improved the quality and dependability of roaming. There was also a move towards standardizing roaming agreements, making it simpler for users to understand where they could roam.

When 3G networks launched in the early 2000s, roaming continued to improve. 3G offered higher data rates and more advanced roaming features, such as the ability to use multiple frequencies and technologies. This made roaming more useful for travelers, allowing them to stay connected on a wider range of networks and in more locations.

In the 2010s, 4G networks were rolled out, providing even faster internet speeds and more complex roaming capabilities. As long as a user's device was compatible with the local network, 4G made it easy to roam across different networks and countries. This

meant that people could stay connected while traveling abroad without worrying as much about high roaming fees.

Now, with the advent of 5G networks, roaming has become much more sophisticated. 5G offers even faster internet speeds and lower latency, making it easier for users to stay connected on the go. Additionally, 5G introduces features like network slicing, which allows users to create virtual networks tailored for specific uses, such as IoT devices or mission-critical applications. These advancements make roaming with 5G networks more efficient and versatile, enhancing the overall mobile experience for users worldwide.

2.3 Types of Roaming

There are currently two types of roaming, depending on the user's mobility (Keller et al. 2021).

2.3.1 National/Internal Roaming

National/Internal Roaming refers to the capability of mobile users to move seamlessly between different regions within a country while maintaining continuous network coverage and service quality. This form of mobility is particularly relevant in countries with vast geographical expanses, such as the United States, where multiple regional carriers operate independently. Initially, mobile operators may have introduced region-specific business offerings to cater to local markets, leveraging regional infrastructure to optimize service delivery. However, with the widespread adoption of the Global System for Mobile Communications (GSM) and the resulting reduction in operational costs, regional roaming has become less prevalent in many areas. The decline is further driven by the consolidation of network operators and the push towards nationwide coverage. Nevertheless, in countries where geographical barriers still pose significant challenges to network uniformity, regional roaming remains a critical feature.

2.3.2 International Roaming

International Roaming allows mobile subscribers to use their home network services while traveling outside their country of origin by connecting to foreign service providers' networks. This type of roaming is particularly beneficial for international travelers, including tourists, business professionals, and expatriates, who require reliable and continuous mobile connectivity across different countries. International roaming leverages standardized communication protocols, with the GSM being the most widely adopted standard globally. GSM's prevalence ensures compatibility with over 80% of mobile carriers worldwide, simplifying the process of establishing roaming connections. Users can make and receive calls, send text messages, and access data services abroad, although typically at higher costs compared to domestic usage.

2.4 Involved 5G Core Network Functions in Roaming

In the 5G core network, various network functions collaborate to facilitate seamless roaming experiences. Each NF plays a specific role in managing authentication, data sessions, policy enforcement, and overall network efficiency for roaming users. Below is a detailed overview of each NF's role in the roaming process (Association 2024):

- **AMF** The AMF handles the registration and connection management of roaming User Equipment (UE) when they connect to a visited network. It ensures that the UE maintains a stable connection as it moves across different cells or regions. During roaming, the AMF coordinates with other NFs to establish and manage data sessions.
- **UDM** The UDM manages subscriber data and provides the necessary authentication credentials for roaming users. It retrieves and supplies subscription information to other NFs, ensuring that only authorized users can access network services while roaming. The UDM also works with policy enforcement mechanisms to apply appropriate service entitlements and usage limits based on the user's subscription agreements.
- **AUSF** The AUSF is responsible for authenticating roaming UEs, ensuring that only legitimate users can access the network services. It collaborates with the UDM to verify user credentials and generate security keys that protect data transmissions.
- **SMF** The SMF manages the establishment, maintenance, and termination of data sessions for roaming UEs. It ensures that Protocol Data Unit sessions are correctly routed and that Quality of Service parameters are maintained. The SMF also interacts with the PCF to enforce data usage policies and optimize network resource allocation, minimizing latency and enhancing the user experience during roaming.
- **PCF** The PCF governs the application of network policies related to data usage, access rights, and charging for roaming services. It determines and enforces policies based on roaming agreements between Mobile Network Operators (MNOs), ensuring that data usage is appropriately managed and billed. The PCF also collaborates with the SMF to apply access restrictions and service prioritization, maintaining network efficiency and fairness among roaming users.
- **UPF** The UPF handles the routing and forwarding of user data packets between the roaming UE and external data networks. In roaming scenarios, it ensures that data traffic is efficiently managed, whether processed by the home or visited network's UPF.
- **NRF** The NRF maintains a registry of all available NFs within the 5G core network, facilitating their discovery and interaction during roaming. It ensures that the necessary NFs are accessible and can communicate effectively, enabling seamless integration between home and visited networks.

2.5 Roaming Approaches

As in the previous generation of cellular networks (4G/LTE), 5G supports the same roaming models, with some architectural differences. As of today, two types of roaming models are supported in 5G according to 3GPP TS 23.501 (GSMA 2020)

2.5.1 Home-Routed (HR)

In the home-routed roaming architecture, the home network is responsible for assigning the IP address to roaming subscribers. This ensures that all user-plane traffic from the roaming UE is anchored in the HPMN, giving the home operator full control over the user's data sessions. As illustrated in Fig. 2.1, the UE first connects to the Radio Access Network of the Visited Public Mobile Network (VPMN), which then forwards control signaling to the Access and Mobility Management Function. The AMF, along with the visited Session Management Function (vSMF), manages initial session establishment on behalf of the HPMN. On the data path, however, the architecture ensures that the UPF of the HPMN remains the central anchor. Packets from the visited UPF (vUPF) are tunneled across the inter-operator interface (N9) to the home UPF, which then forwards them to the Data Network (DN). The home SMF (hSMF) in the HPMN is directly connected to the UDM, from which it retrieves subscription and policy data.

The advantage of this model lies in the HPMN's ability to retain end-to-end visibility, billing control, and service policy enforcement for its subscribers. This makes it particularly suitable in scenarios where full trust between the home and visited operators is not established. However, this design introduces higher latency since every data packet must be routed back to the HPMN before reaching its destination. This additional round-trip overhead can negatively impact latency-sensitive applications, such as real-time communications or low-latency services, but remains acceptable for use cases prioritizing trust and control over performance.

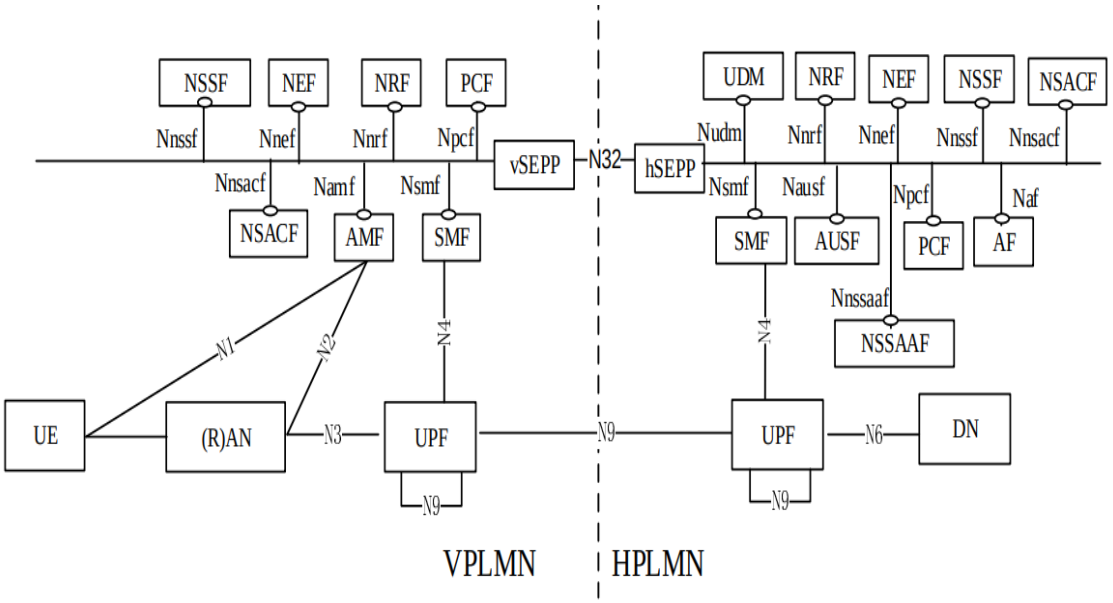


Figure 2.1 – Home-Routed architecture

2.5.2 Local-Breakout (LBO)

To address the latency limitations inherent in the home-routed roaming model, 5G introduces the Local Breakout (LBO) roaming architecture, illustrated in Fig. 2.2. In this design, the user plane traffic of a roaming UE is offloaded directly within the VPLMN, while the HPLMN remains responsible for authentication and subscription management.

As depicted in 2.2, the UE connects to the RAN of the visited network, which forwards signaling to the Access and Mobility Management Function and the Session Management Function. Here, the vSMF and visited vPCF manage session establishment and enforce policy charging based on roaming agreements. The UPF located in the visited network acts as the local anchor for the data session, forwarding traffic directly to the Data Network over interface N6. This design eliminates the need to tunnel user data back to the HPLMN, thereby reducing end-to-end latency.

Only the control-plane signaling traverses toward the HPLMN. The hSMF, connected to the UDM and AUSF, ensures that authentication, subscription validation, and essential policy information remain under the home operator's control.

From a Quality of Service perspective, LBO is often considered the most efficient roaming model, as it minimizes latency and optimizes resource usage in the visited network. However, this efficiency comes at the cost of reduced control for the HPLMN over user-plane traffic and IP address allocation, which is now managed by the visited operator. Additionally, the architecture introduces complexities in billing and settlement between operators, as intermediaries may be required to handle charging reconciliation.

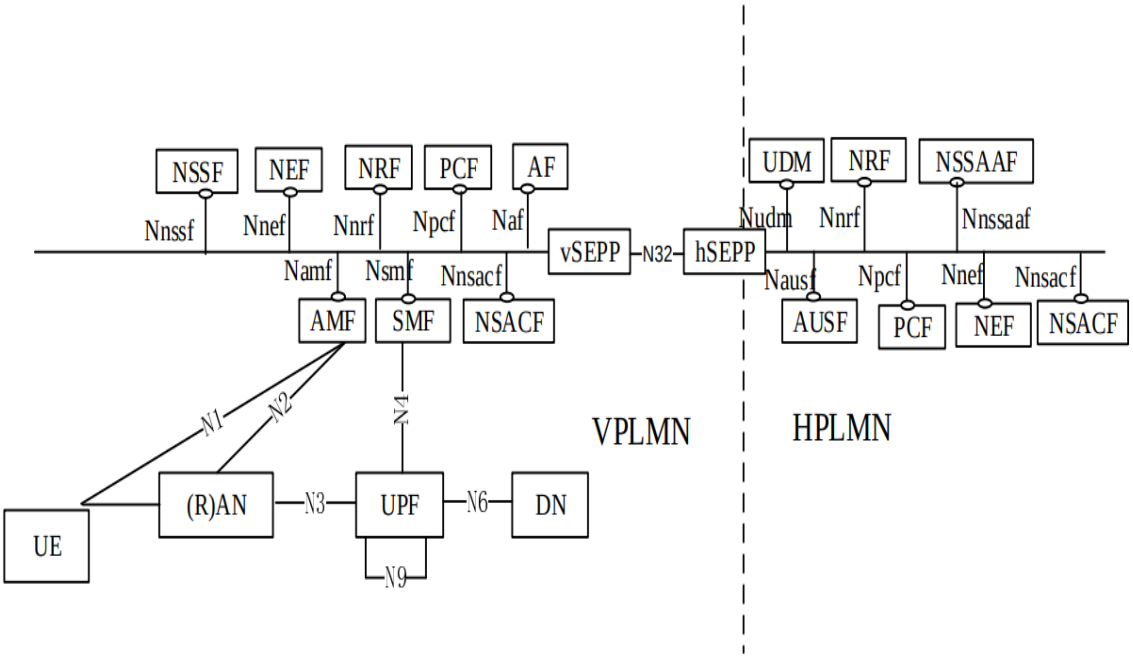


Figure 2.2 – Local-Breakout architecture

In both Local Breakout and Home-Routed architectures, communication between the VPMN and the HPMN is facilitated through the Security Edge Protection Proxy (SEPP) as shown in Fig. 2.1 and Fig. 2.2. The SEPP acts as a secure intermediary that handles signaling messages between the two networks. By routing communication through the SEPPs on both sides, the system ensures secure and standardized interconnectivity over the IPX network (Interconnect Packet Exchange). This design helps to safeguard sensitive data while enabling seamless interaction between the networks for efficient roaming operations.

2.6 Roaming Agreements and Charging

In order for mobile operators to provide roaming services to their subscribers, formal agreements must be established between different operators. According to Syniverse Technologies Inc.(Inc. 2002), there are some possibilities for how the roaming agreements can be formulated:

- **Direct Roaming Agreements**: Direct roaming agreements are established individually between two providers. Every time a provider wants to add a new roaming partner, they must sign a separate agreement with that partner.
- **Piggyback Roaming Agreements**: In piggyback roaming agreements, two providers sign an agreement where each agrees to expand their roaming areas not only in their regions but also in the regions of all their existing roaming partners. This means that when a provider joins with a new partner, they automatically gain access to the roaming areas of that partner's existing agreements.
- **Consolidator Roaming Agreements**: Consolidator agreements involve multiple providers signing a main agreement managed by a single operator. All providers

that join this master agreement agree to the same roaming conditions. This means that any provider in the consolidator can use the roaming areas of all other providers in the agreement.

- **Alliance (Consortium) Roaming Agreements**: In alliance or consortium roaming agreements, an intermediary manages and facilitates multiple roaming agreements. When a new provider joins the alliance, the intermediary establishes a roaming agreement between the new provider and all existing members of the alliance.
- **Roaming Broker Agreements**: Roaming broker agreements are considered one of the most efficient types of roaming agreements today. A roaming broker acts on behalf of multiple roaming partners, managing the agreements for them. Instead of each provider signing numerous bilateral agreements, they sign one agreement with the roaming broker. This single agreement replaces the need for hundreds of individual contracts, making it much simpler to expand roaming areas.

In the LBO architecture, the HMNO does not have access to the subscriber's roaming information, and the VMNO does not have the subscriber's charging details. Because of this, mobile network operators need to manage multiple partnerships, connect with networks around the world, and handle complex financial transactions. This complexity ensures that users can roam seamlessly, but it also requires significant coordination and cooperation between different MNOs globally (Huawei Technologies Co., Ltd. 2021).

To provide customers with international roaming services, providers need to establish roaming agreements with operators around the globe. For example, if there are 500 operators, the number of possible bilateral roaming agreements would be 125,000:

$$\binom{n}{2} = \frac{n!}{(n-2)! \cdot 2!} = 125000.$$

Managing such a vast number of agreements is highly complex and challenging. This complexity is further increased in countries like the United States, Russia, India, and Brazil, where network coverage is often regional rather than nationwide. In these cases, a provider might need to sign multiple agreements with different operators within the same country to ensure comprehensive coverage for their customers (Inc. 2002).

Additionally, providers face other obstacles like differing time zones, which can complicate coordination and support, as well as variations in billing record formats and settlement cycles. These differences require robust processes for exchanging billing records between roaming partners to accurately charge customers for their usage abroad. Efficiently managing these multiple relationships and handling the intricate financial transactions is essential for delivering seamless international roaming services, but it also demands significant resources and coordination from the provider.

Because of the complexities involved in managing international roaming agreements, the Roaming Broker Agreement has become the most widely used model. In this approach, MNOs rely on third-party entities known as Data Clearing Houses (DCHs) to facilitate seamless connectivity and simplify the roaming process. The DCH acts as an intermediary, handling the exchange of roaming billing records, managing financial settlements, and ensuring compliance with standardized roaming agreements.

Both the Home Session Management Function (H-SMF) and the Visited Session Management Function (V-SMF) must support charging mechanisms to ensure proper billing

and financial settlements between operators. The H-SMF communicates with the Home 5G Charging Function (H-CHF) in the HPLMN, which provides an integrated charging system supporting both prepaid and postpaid billing. Similarly, the V-SMF interacts with the Visited Charging Function (V-CHF) in the VPLMN to manage charging processes within the visited network (GSMA 2020).

For inbound roaming traffic (when a subscriber enters a visited network), the V-CHF generates Call Detail Records (CDRs), while for outgoing roaming traffic (when a subscriber returns to their home network), the H-CHF is responsible for creating CDRs. In the LBO roaming model, charging policies are enforced by the V-Policy Control Function (V-PCF) and V-CHF based on predefined roaming agreements, allowing for localized billing and reduced latency (GSMA 2020). On the other hand, in the HR roaming model, the H-PLMN directly manages the charging process, centralizing billing control but potentially increasing latency due to traffic routing back to the home network.

2.7 Summary

This chapter has examined the evolution of roaming from 2G to 5G, outlining the fundamental principles, types, and enabling architectures that support seamless mobility across heterogeneous networks. We first distinguished between different types of roaming. We then discussed the main architectural models for roaming in 5G networks, namely the *home-routed* and *local breakout* approaches. The home-routed model emphasizes centralized control by anchoring user-plane traffic in the home network, thus ensuring stronger policy enforcement and subscriber visibility at the cost of increased latency. By contrast, the local breakout model prioritizes performance by allowing user-plane traffic to be managed locally in the visited network, reducing latency but limiting the home operator's control.

Overall, roaming in 5G introduces a more complex ecosystem than in previous generations, as it must balance control, trust, latency, and quality of service across operators. The choice of roaming model depends on regulatory constraints, business agreements, and the application requirements of end-users. These considerations highlight the critical role of roaming architecture design in enabling secure, efficient, and scalable global mobility in the 5G era.

Chapter 3

Blockchain Technology

3.1 Introduction

Blockchain technology has risen as a revolutionary innovation capable of fundamentally transforming a wide range of industries. Its unique capacity to establish secure, transparent, and decentralized systems has drawn significant interest from businesses, governments, and individuals around the world. Blockchain's decentralized nature eliminates the need for intermediaries, which can reduce costs and increase efficiency in processes like financial transactions, supply chain management, voting systems, and many other industries.

In this chapter, we will explore the foundational principles of blockchain technology, examining its structural design and essential elements that make it impactful.

3.2 Definition

Blockchain is a decentralized and distributed digital ledger that records transactions and stores data in a transparent, secure, and tamper-resistant way. As shown in Figure 3.1, the architecture of blockchain functions as an ever-growing chain of linked blocks. Each block contains a cryptographic hash of the preceding block, a timestamp, and a comprehensive log of various transactions. The use of cryptography guarantees the integrity and security of each block, making it virtually impossible to alter any recorded information after it has been added to the chain (Zhai et al. 2019).

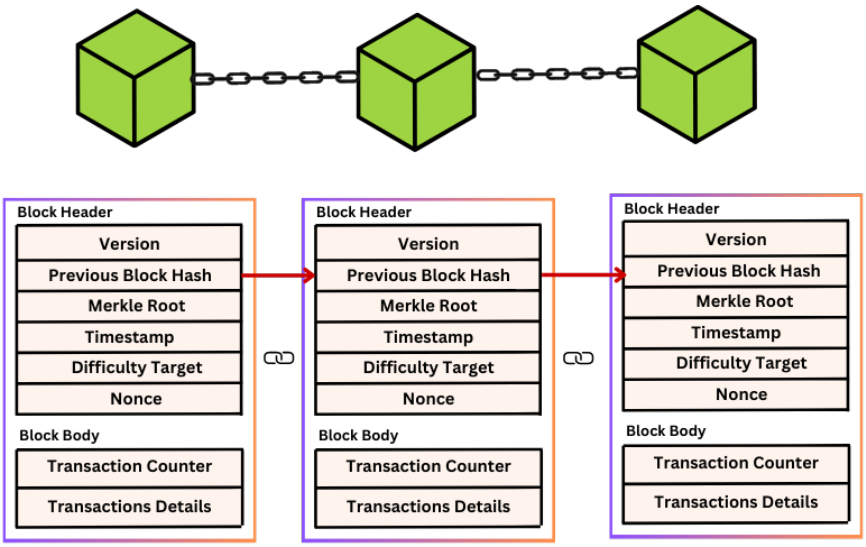


Figure 3.1 – Blockchain Architecture

A specialized data structure, such as a Merkle tree (also known as a hash tree), is utilized to efficiently and securely store all confirmed transaction records within the block body of a blockchain. A Merkle tree, as depicted in Figure 3.2, works by recursively hashing pairs of transactions until a single hash, known as the Merkle root, is generated.

This root hash is then stored in the block header, serving as a cryptographic summary of all the transactions in the block (Zhai et al. 2019).

Each leaf node of the Merkle tree represents the hash of an individual transaction, while non-leaf nodes contain the hash of their child nodes. This hierarchical structure allows for quick and secure verification of transactions without needing to process the entire block.

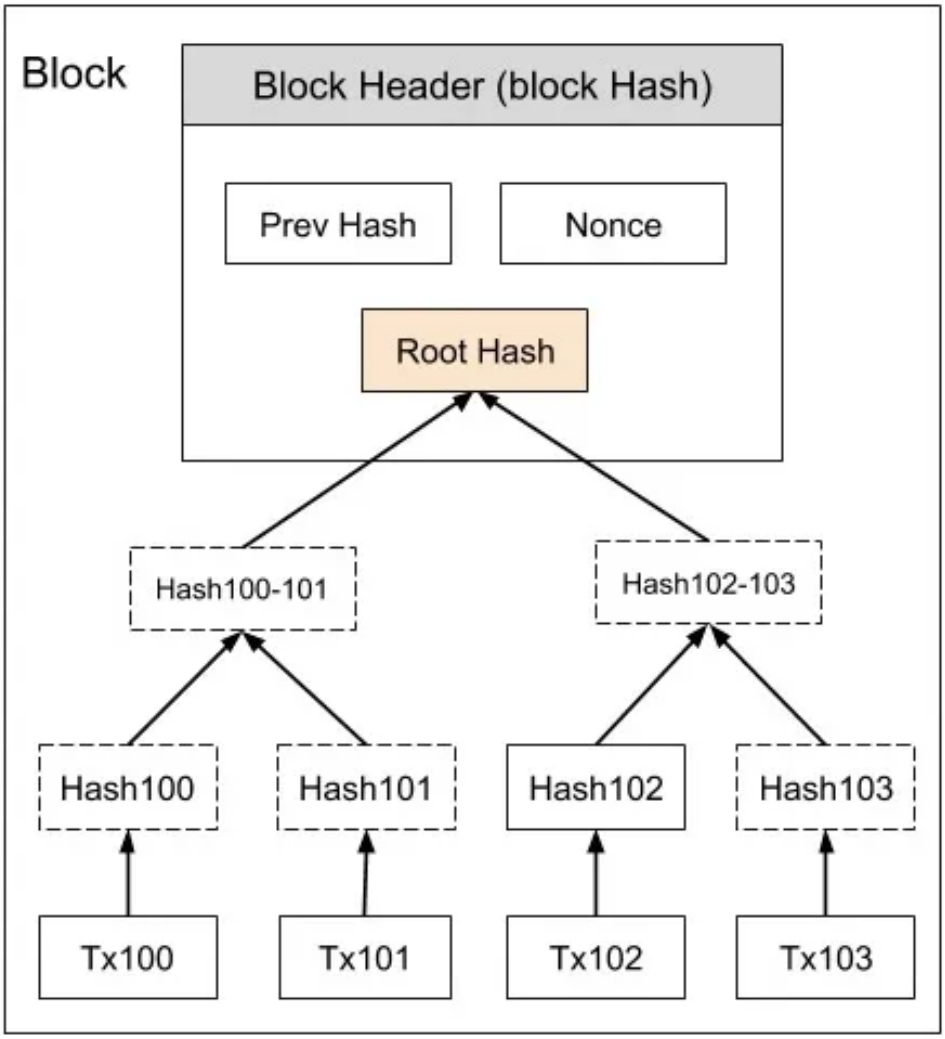


Figure 3.2 – Merkle Tree Structure

In a blockchain network, multiple nodes work together though peer-to-peer interaction to uphold the integrity and security of the distributed ledger. Transactions are verified and validated using a consensus mechanism, such as Proof-of-Work (PoW) or Proof-of-Stake (PoS), which ensures that each transaction is genuine and complies with the network's predefined rules (Zheng et al. 2018). This decentralized approach to validation promotes trust and transparency among participants, as it removes the reliance on a central authority to oversee or approve transactions.

3.3 Blockchain Key Elements

Blockchain technology is built on several foundational components that work together to create a secure, transparent, and decentralized system. These key elements form the backbone of blockchain networks, enabling them to function efficiently and reliably. Below, we explore each of these elements in detail.

3.3.1 Transaction

A transaction represents the transfer of data or value between participants within a blockchain network. Each transaction is cryptographically signed to ensure authenticity and integrity. Once initiated, transactions are grouped together and validated by network nodes before being added to a block. Transactions are immutable once recorded, meaning modifying or reversing a transaction would necessitate altering every subsequent block in the chain (Zheng et al. 2018). Given the properties of cryptographic hashing and the decentralized nature of the network, this process is extremely resource-intensive and practically unfeasible. However, it's worth noting that blockchain transactions are not entirely irreversible. In real-world scenarios, blockchain networks can encounter forks, where transactions on the shorter chain might be rolled back. Transaction finality is probabilistic in nature: as more blocks are added atop the block containing a transaction, its security increases, making the chance of reversal increasingly unlikely.

3.3.2 Block

A block is a container data structure that holds a collection of validated transactions. Each block includes a cryptographic hash of the previous block, creating a chain of interconnected blocks-hence the term “blockchain.” Blocks also contain a timestamp and a unique identifier called a nonce (in PoW systems). Once a block is added to the blockchain, its data becomes immutable, providing a secure and transparent record of all transactions (Zheng et al. 2018).

The size of blocks can vary across different blockchain systems (Göbel et al. 2017). Some blockchains employ fixed block sizes, ensuring uniformity and predictability, while others adopt dynamic block sizes, allowing flexibility to accommodate varying transaction volumes.

3.3.3 Distributed Ledger

The distributed ledger is a decentralized database shared across multiple nodes in the blockchain network (Zheng et al. 2018). Unlike traditional centralized ledgers, this ledger is maintained collectively by all participants, ensuring transparency and reducing the risk of single points of failure. Every node in the network holds an identical copy of the ledger, which is updated through consensus, ensuring consistency and accuracy across the system.

3.3.4 Consensus Protocol

A consensus protocol is a set of rules that allows a decentralized network of computers to agree on the state of a shared ledger, such as a blockchain. In blockchain systems, consensus protocols are responsible for validating and securely adding new transactions to the ledger. These protocols address consistency challenges inherent in distributed systems, ensuring that all nodes in the network reach agreement on proposed changes. The consensus process typically resolves two key questions: which node in the network proposes a new block, and whether the other nodes accept or reject that proposal.

To ensure the integrity and security of a blockchain, all nodes in the network must reach an agreement on the state of the ledger (Xu et al. 2023). Consensus protocols play a crucial role in achieving this by validating transactions, verifying their authenticity, and establishing a unified order and content for all transactions. However, challenges such as network disruptions (e.g., nodes going offline) or malicious activities can interfere with the consensus process. As a result, a robust consensus protocol must be designed to tolerate such issues and minimize their impact, ensuring that the final consensus outcome remains unaffected. Furthermore, the chosen consensus protocol must align with the specific requirements and constraints of the application scenario, balancing factors like scalability, security, and energy efficiency.

Consensus algorithms can be categorized primarily into two categories, as depicted in Fig 3.3.

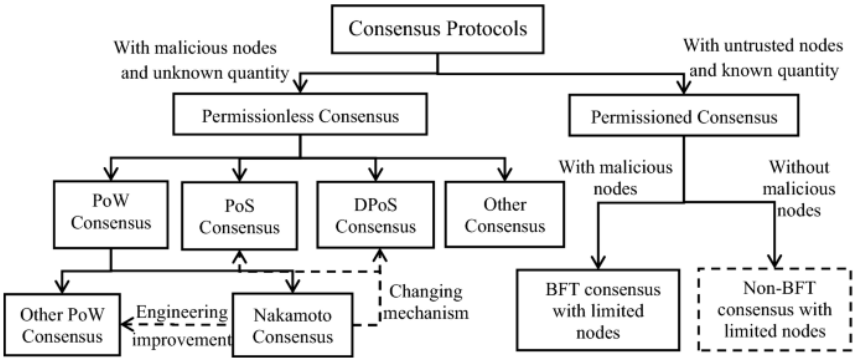


Figure 3.3 – Blockchain Consensus Protocols Categorization

3.3.4.1 Permissionless Consensus

This category includes consensus mechanisms used in public blockchains, where any node can join the network without prior approval. The main challenge here is dealing with malicious nodes while ensuring security and decentralization. Several consensus algorithms are created under this category; the most common ones are presented below:

- **Proof-of-Work (PoW)**: PoW requires network participants, known as miners, to solve computationally expensive cryptographic puzzles to validate transactions and propose new blocks. The first miner to find a valid solution gets rewarded, ensuring security by making attacks prohibitively expensive. However, PoW is criticized for its high energy consumption and scalability limitations due to slow transaction finality (Xiao et al. 2020).

- **Proof-of-Stake (PoS)**: PoS replaces computational work with economic incentives, where validators are chosen based on the number of tokens they stake. This dramatically reduces energy consumption compared to PoW, increases transaction throughput, and introduces finality guarantees. However, PoS faces risks such as the "nothing at stake" problem, mitigated using mechanisms like slashing penalties (Xiao et al. 2020).
- **Delegated Proof-of-Stake (DPoS)**: DPoS optimizes PoS by allowing token holders to elect a small number of delegates who validate transactions on their behalf. This increases efficiency, achieving high transaction throughput and near-instant block finality, but sacrifices a bit of decentralization since only a few nodes participate in consensus (Xiao et al. 2020).
- **Proof-of-Authority (PoA)**: Proof of Authority (PoA) is a specialized variant of Proof of Stake, where validators stake their identity and reputation instead of monetary tokens. To become a PoA validator within the consensus group, a participant must undergo a rigorous certification process to establish their authority. This process typically includes verifying the validator's unique identity, demonstrating their ability to consistently contribute to the consensus process, and making all certification details publicly accessible. The consensus group in a PoA system is designed to be small, stable, and subject to public scrutiny, ensuring users can trust the group for reliable transaction processing and blockchain maintenance. If a validator fails to perform their duties competently or engages in misconduct, they risk being discredited by both users and fellow validators, thereby losing their authority and position in the network (Xiao et al. 2020).

3.3.4.2 Permissioned Consensus

This category applies to private or consortium blockchains, where all participants are known entities. These networks prioritize efficiency, scalability, and lower resource consumption, often at the expense of decentralization.

- **Byzantine Fault Tolerant (BFT)**: BFT-based consensus mechanisms ensure network security even when a fraction of nodes act maliciously. They rely on message-passing and voting among a predefined set of participants. Practical Byzantine Fault Tolerance (PBFT) is widely used as it provides fast finality but requires high communication overhead (Chaudhry et al. 2018). Variants like Tendermint BFT improve efficiency by reducing the number of consensus rounds.
- **Non-Byzantine Fault Tolerant (BFT)**: For blockchains operating in fully trusted environments, simpler consensus mechanisms without Byzantine fault tolerance suffice. This mechanism optimizes performance and latency in private ledgers where malicious actors are not a concern, making them suitable for banking and enterprise applications (Chaudhry et al. 2018).

3.4 Blockchain Properties

Blockchain offers several properties that make it a powerful tool for enhancing security, transparency, and efficiency across various applications. These key properties include (Ali et al. 2021):

3.4.1 Decentralization

Blockchain operates in a decentralized manner, meaning no single entity has full control over the network. Instead, it relies on a distributed ledger maintained by multiple nodes, each holding a complete copy of the blockchain. This structure eliminates the need for intermediaries, reducing the risk of fraud, unauthorized changes, and single points of failure. Additionally, consensus mechanisms ensure that transactions are validated without reliance on third-party authorities, strengthening network security and autonomy.

3.4.2 Immutability

Once data is recorded on the blockchain, it cannot be altered or deleted. Each block contains a cryptographic hash of the previous block, forming a chain that makes tampering exceptionally difficult. Any attempt to modify historical data would require altering all subsequent blocks, which is computationally infeasible due to the consensus mechanism. This property ensures data integrity and trustworthiness.

3.4.3 Transparency

Blockchain provides a transparent and verifiable ledger where every transaction is recorded and visible to all participants. This openness enhances trust among users, as transactions can be independently audited without requiring a central authority. Since blockchain transactions require peer approval and are cryptographically secured, the system minimizes susceptibility to fraud or unauthorized modifications.

3.4.4 Security and Fault Tolerance

Blockchain is designed to be highly secure through the use of cryptographic techniques and consensus algorithms. Every transaction is verified and validated before being added to the ledger, making the network resistant to cyberattacks. Additionally, blockchain's fault tolerance ensures that the network remains operational even if some nodes fail or are compromised. Since transactions are distributed across multiple nodes, the system does not rely on a single point of failure, making it highly resilient against disruptions and a single point of failure.

3.5 Categories of Blockchain

Generally, blockchain is classified into three main categories (Ali et al. 2021):

- **Public blockchain**: A public blockchain is an open and permissionless network where anyone can participate, validate transactions, and contribute to the consensus process. These networks are fully decentralized and rely on cryptographic algorithms and permissionless consensus mechanisms such as PoW or PoS to ensure security and trust. Since public blockchains operate on a trustless model, they do not require intermediaries to verify transactions. Bitcoin and Ethereum are prime examples

of public blockchains, enabling global peer-to-peer transactions without a central authority.

- **Private blockchain**: A private blockchain is a closed, permissioned network where only authorized entities can access, validate transactions, and participate in the consensus process. These blockchains are commonly used in enterprise settings for internal business processes, allowing organizations to maintain control over who can join the network and access transaction data.
- **Consortium blockchain**: A consortium blockchain is a hybrid system that combines elements of both public and private blockchains. It is governed by a group of organizations that collectively manage the network, restricting participation to approved members. Unlike fully private blockchains, a consortium blockchain distributes control among multiple entities, reducing the risk of a single point of failure while still maintaining a level of privacy.

3.6 Smart Contracts

A smart contract is a self-executing program that runs on a blockchain, containing predefined rules and conditions encoded in high-level programming languages such as Solidity (Ethereum). These contracts automatically execute and enforce agreements between parties when specific conditions are met, eliminating the need for intermediaries(Zheng et al. 2020).

Smart contracts are deployed on the blockchain through transactions, which initiate their execution and ensure that the contract operates according to its encoded logic. Once deployed, a smart contract becomes an immutable part of the blockchain, meaning its code cannot be altered or tampered with, ensuring trust and security in digital agreements. This immutability guarantees that once a contract is set and deployed, it will execute exactly as written, without interference from third parties or external manipulation.

Many blockchain platforms support smart contract functionality, with Ethereum being the most widely recognized due to its Turing-complete virtual machine EVM (Rosa-Bilbao et al. 2023), which allows for the execution of complex decentralized applications (dApps). Another notable platform is Hyperledger Fabric, which is more enterprise-focused and designed for private and permissioned blockchain networks.

3.7 Limitations and Challenges

Despite its numerous advantages, blockchain technology encounters several challenges that impact its widespread adoption and scalability (Ali et al. 2021)

- **Scalability**: One of the most significant challenges in blockchain networks is scalability, or the ability to handle a high number of transactions efficiently. As the number of users and transactions increases, many blockchain networks experience congestion, leading to slower transaction processing times and higher fees. This issue is particularly evident in public blockchains such as Bitcoin and Ethereum, where each transaction must be validated by the network before being added to the

ledger. For blockchains that use PoW consensus mechanisms, like Bitcoin, transaction verification can become resource-intensive, limiting throughput to just a few transactions per second (TPS). Solutions such as layer-2 scaling (e.g., Channels, Rollups), sharding, and improved consensus mechanisms (e.g., Proof-of-Stake) are being implemented to overcome these limitations.

- **Interoperability**: Blockchain networks often operate in isolation, meaning that they cannot easily communicate or share data with other blockchains. This lack of interoperability creates fragmentation in the ecosystem, making it difficult for applications to function across multiple blockchain platforms. For example, assets or information stored on Ethereum cannot be directly accessed by Bitcoin's network without using bridging solutions or wrapped tokens. This limitation reduces the potential of blockchain to support multi-chain large-scale applications. Various projects are working on interoperability solutions, including cross-chain bridges, interoperability protocols like Chainlink Cross-Chain Interoperability Protocol (Hasret Ozan Sevim 2024), or even native blockchains that support interoperability like Polkadot and Cosmos (Belchior et al. 2021).
- **Lack of real-world data integration**: One fundamental limitation of blockchain technology is its inability to directly fetch data from the real world. Blockchains operate as isolated, self-contained ledgers, meaning they cannot natively access external data sources such as weather reports, stock prices, IoT device readings, or identity verification systems. This limitation restricts blockchain's applicability in industries where real-time, off-chain data is crucial. However, several solutions are being developed to mitigate this limitation, like Oracle. Oracles act as data bridges between blockchains and the external world that provide secure and decentralized data feeds to enable smart contracts to interact with real-world information, like Chainlink.

3.8 Summary

Blockchain technology provides a promising foundation for transforming various industries through its decentralized, transparent, and immutable properties. By eliminating intermediaries and ensuring secure, tamper-proof transactions, blockchain has the potential to enhance efficiency, trust, and automation across multiple sectors.

However, despite its advantages, blockchain is not without limitations. Key challenges include scalability constraints and the complexity of implementation. These issues must be carefully considered and addressed to unlock blockchain's full potential and ensure its practical applicability in large-scale systems.

This chapter offered a comprehensive introduction to blockchain technology, exploring its underlying structure, functionality, and practical applications. Given the challenges associated with 5G roaming models discussed in previous chapters, blockchain presents a viable solution for improving security, interoperability, and trust in next-generation communication networks. In the next chapter, we will explore and analyze existing works of integrating blockchain into 5G roaming.

Chapter 4

Review of Blockchain-based 5G Roaming Contributions

4.1 Introduction

This chapter focuses on analyzing and evaluating blockchain-based solutions for 5G roaming, introducing a specific threat model tailored to this environment. The goal is to examine how well these solutions handle potential security and privacy issues.

To begin, the chapter presents a threat model that reflects the unique aspects of blockchain-based 5G roaming systems. This model identifies the possible threats that could affect the system's integrity, confidentiality, and availability. Next, the chapter reviews and analyzes current research related to these systems.

In the final part, the chapter assesses the security of the proposed solutions by testing them against the defined threat model. The discussion ends by pointing out key challenges, current limitations, and open research questions in the field of blockchain-based 5G roaming.

4.2 Threat Model

In this section, we introduce a threat model that categorizes the common threats faced by blockchain-based 5G roaming frameworks. These threats generally arise from two types of participants: semi-honest and malicious. Semi-honest adversaries, also known as honest-but-curious, follow the protocol correctly but try to learn private information about other users-for example, attempting to discover a specific user's roaming activity. On the other hand, malicious or active adversaries not only seek private data but also intentionally break protocol rules to do so.

This threat model provides a foundation for objectively evaluating the frameworks discussed later in this section. It also helps in identifying their current limitations and challenges.

- **Lazy participants**: When there are no strong incentives or enforcement mechanisms in place, 5G roaming participants-especially MNOs-may behave in a selfish or passive way to reduce their resource usage. For example, a VMNO might delay reporting a user's roaming activity, or the HMNO might avoid paying for the roaming usage altogether.

- **Sybil attacks**: A Sybil attack is a network attack and is well-known in the context of P2P networks, where an attacker can forge or create numerous fake identities to gain a considerable influence on the network. Eventually, the attacker uses Sybil nodes and the attack method to trigger various threats that damage the reputation system of a P2P network (Iqbal et al. 2021).

- **Poisoning attacks**: In a poisoning attack, a VMNO intentionally modifies or fabricates the roaming usage data of a user to financially exploit the HMNO. This can involve inflating the amount of data used, extending call durations, or falsely reporting activity to make it appear that the UE consumed more resources than it actually did. As a result, the HMNO is overcharged for services that were never

fully consumed by the UE. These attacks are especially dangerous in decentralized systems where data is not verified by a trusted third party.

- **On-chain privacy leakage attacks**: In decentralized 5G roaming systems, billing and settlement processes are often handled through smart contracts on the blockchain. This requires some form of CDR-which includes sensitive information like time, duration, and location of roaming usage-to be exchanged or referenced on-chain. While blockchain brings transparency and automation, it also exposes metadata that can be observed by anyone with access to the network. This transparency creates a privacy risk, as unauthorized parties, such as external observers or competing MNOs, could monitor the blockchain to infer sensitive details about a user's roaming activity.

4.3 Literature Review

In this section, we present the relevant past contributions to 5G roaming settlements using Blockchain technology, highlighting the aspects of each solution and its limitations.

(Mafakheri et al. 2021) propose a new roaming network architecture for 5G based on a permissioned blockchain platform integrated with smart contracts, specifically utilizing Hyperledger technology. The solution aims to enhance visibility for MNOs regarding their subscribers' activities in visited networks, enable quick payment reconciliation, and reduce fraudulent transactions. Their model replaces traditional intermediaries like clearinghouses by implementing peer-to-peer transactions directly between MNOs through blockchain-enabled smart contracts. This blockchain framework supports subscriber discovery, identity management, and automated billing settlements, providing transparency, security, and privacy through the consortium blockchain model. Hyperledger's private channels are leveraged to maintain confidentiality and protect subscriber information, while offloading large-scale data storage to secure off-chain databases. However, the paper does not detail the mechanisms of how the solution specifically achieves subscriber discovery, identity management, and billing settlement. Additionally, it leaves open questions regarding accurate advance payment calculations between MNOs, the potential for consensus manipulation due to relying solely on the two directly involved MNOs, and lacks clarity on precisely what data types should remain off-chain. Moreover, the approaches suggested to enhance privacy are not extensively detailed.

(Gong et al. 2025) propose a Blockchain-Enhanced Core Network Architecture (BECNA) along with a Secure Decentralized Identity Authentication Scheme (SDIDAS) to overcome some of the weaknesses found in traditional centralized architectures of 5G core networks. BECNA uses blockchain to decentralize data storage, aiming to improve network security, stability, and reliability by removing Single Points of Failure (SPoF). Blockchain nodes are deployed near base stations to reduce data transmission delays and boost network responsiveness. However, the use of blockchain to replicate certain 5G core functions-such as the UDR and the PCF-raises concerns. This replication seems unnecessary and costly, as the issue of data redundancy can already be addressed using traditional methods like database replication and backups. Additionally, the UDR contains highly sensitive user information related to authentication and subscriptions. The proposed approach does not clearly explain how this sensitive data can be securely stored on a blockchain, given that

all nodes maintain a full copy of the distributed ledger, potentially increasing the risk of data exposure. To address identity management, the authors introduce SDIDAS, which uses Decentralized Identity (DID) technology to store identity information locally on the User Equipment. This approach helps reduce the risk of data leaks during cross-network roaming. Authentication is handled through cryptographic signatures verified by smart contracts, providing a secure and decentralized way to manage user credentials. While simulation results show that SDIDAS can be efficient in terms of gas consumption, the blockchain layer still adds some latency to the overall process.

(Refaey et al. 2019) propose a blockchain-based policy and charging control framework aimed at enhancing the efficiency, transparency, and trust in cellular network roaming. Their solution replaces the traditional, centralized roaming model with a decentralized architecture in which smart contracts manage roaming agreements and charging rules between MNOs and users. The framework relies on three key smart contracts: the Mobile Network Operator Contract (MNOC), which enables VMNOs to securely query user data from HMNOs; the Cursory Contract (CC), which logs users' roaming history and notifies them of updates; and the Registrar Contract, which links user and operator identities to blockchain addresses. These contracts support flexible roaming with minimal changes to existing infrastructure. Through a case study focused on the European Union, the authors assess the system's financial and performance outcomes. Findings indicate that the blockchain model can boost operator revenues-particularly from international users-and improve user satisfaction by enabling real-time and transparent roaming decisions. The architecture also incorporates off-chain data access to maintain privacy and supports scalability and interoperability across different operators. Although the blockchain implementation improves security, trust, and user satisfaction compared to traditional systems, the proposed solution lacks a comprehensive empirical evaluation regarding transaction latency and throughput. Also, some aspects remain unclear, particularly how billing settlement is handled. The process by which the VMNO exchanges CDRs with the smart contract, and how actual billing is performed, is not fully addressed.

(Weerasinghe et al. 2021) present a blockchain-based service platform tailored for Local 5G Operators (L5GOs) to support efficient roaming and offloading services. The proposed solution introduces several innovative features, including a reputation management system to ensure quality of service and a fraud prevention mechanism based on smart contracts. In this framework, stakeholders-such as subscribers, MNOs, and L5GOs-register on the blockchain, enabling dynamic roaming and offloading agreements. These agreements are influenced by calculated scores that factor in signal strength, network capacity, service cost, and reputation. These scores are updated after each roaming or offloading session to reflect the most recent performance. The proposal lacks clarity on certain operational aspects. Notably, it does not explain who is responsible for registering stakeholders, especially MNOs, into the blockchain system, which opens risks for spamming and phishing attacks. Furthermore, it does not address how roaming settlement is handled between the VMNO and the HMNO, which is a crucial component of roaming services. The implementation was tested on Ethereum's Rinkeby Testnet, which introduced a latency around 15 times higher than the maximum delay recommended by the 3GPP standard. Privacy concerns also remain unresolved, particularly regarding the protection of sensitive subscription data on-chain.

(Nguyen et al. 2021) propose BlockRoam, a blockchain-based roaming management system primarily designed to mitigate roaming fraud by significantly reducing data ex-

change latency between Home and Visitor Public Mobile Networks (HPMN and VPMN). Unlike traditional systems, where fraud detection can take several hours due to delayed CDR transmission, BlockRoam introduces a blockchain solution that reduces this delay to under three minutes, enabling near real-time fraud detection and response. This latency reduction-not automation of billing or roaming agreements the central contribution of the paper. BlockRoam's blockchain infrastructure is built on a customized Proof-of-Stake (PoS) consensus model employing dynamically selected committees. Each epoch, the committee members are chosen based on stake distribution, and the leader selection is performed using the Follow-The-Satoshi (FTS) algorithm. This design trades off block confirmation finality in favor of lower block times, making it well-suited for high-frequency roaming data updates. Compared to traditional PoW systems like Bitcoin or delayed-finality PoS systems like Cardano, BlockRoam achieves substantially faster confirmation times even under varying adversarial conditions. The authors demonstrate that this setup maintains the blockchain's persistence and liveness properties, aligning with the security guarantees of the Ouroboros protocol. To sustain network security and participation, BlockRoam incorporates a Stackelberg game-theoretic economic model that balances the profits of stake pools and stakeholders. Stakeholders can allocate their resources either to self-mining or to stake pools, and their expected rewards are calculated accordingly, incentivizing cooperation and wider decentralization. While BlockRoam successfully eliminates the need for Data Clearing Houses and enhances fraud mitigation, the approach still does not address trustless roaming settlement and raises privacy concerns, particularly regarding the public exposure of CDR data and how they are handled on-chain.

4.4 Security Analysis

In this section, we evaluate the presented contributions using the threat model introduced earlier. The goal is to assess how effectively each solution addresses potential threats such as lazy participant behavior, poisoning attacks, Sybil identities, and privacy leakage. By analyzing these aspects, we can better understand the security robustness and practical limitations of the proposed architectures. Table 4.4 summarizes the analysis.

In the work presented by (Mafakheri et al. 2021), the use of a permissioned ledger partly discourages lazy behavior, as each MNO must be identified and must endorse transactions to proceed. This setup makes participants more accountable. To further encourage cooperation, the authors introduce an incentive mechanism where roaming funds are locked in a smart contract before the session begins. This creates a financial motivation for both the HMNO and the VMNO to complete the transaction and settle promptly. Since participation is restricted to verified entities-typically just the two MNOs involved in the roaming- the risk of Sybil attacks is reduced. The consortium model ensures that only recognized MNOs can join the network. However, the solution makes a strong trust assumption: that neither MNOs behaves maliciously. If one of the two participating nodes acts dishonestly, the system cannot reach consensus or finalize the transaction state, due to the limited number of validators in the private channel. Additionally, the framework does not address poisoning attacks. The VMNO could falsify CDR data to overcharge the HMNO, and the system lacks a built-in mechanism to verify the correctness of submitted records. On the privacy side, the use of Hyperledger Fabric's private channels enables selective sharing of transaction data. This helps limit exposure of sensitive information

and reduces the chances of on-chain privacy leakage.

The work in (Gong et al. 2025) presents a blockchain-based solution focused primarily on decentralized identity management rather than on the financial or operational aspects of roaming settlement. As a result, evaluating the framework against lazy participation threats is not applicable. The proposed architecture leverages the Ethereum public blockchain, where Sybil attacks are inherently mitigated due to the high cost of participation-launching a large-scale identity attack would require significant financial resources. Each entity on the network is associated with a cryptographically verifiable Decentralized Identifier (DID), presumably issued or approved by participating operators. However, the authors do not detail how the system ensures the integrity of user activity data or how manipulations by the VMNO, such as inflating usage records to overcharge the HMNO, could be detected or prevented. While the use of DIDs helps improve privacy by allowing users to store sensitive information locally and only publish verifiable proofs (e.g., signatures) on-chain, some privacy risks remain. Notably, the solution suggests that critical 5G core functions such as UDR and PCF-which may include sensitive subscription and authentication data-are stored on-chain. However, the paper does not explain how this sensitive data is protected in a public blockchain environment, leaving unresolved concerns about potential data exposure and privacy breaches.

In the work presented by (Refaey et al. 2019), the policy and charging control framework uses smart contracts to automate payment reconciliation between MNOs. This automation aims to reduce settlement delays by handling agreement execution on-chain. However, the framework does not clearly explain how roaming revenue is actually transferred from the HMNO to the VMNO after the roaming session ends. As a result, an HMNO could still intentionally delay the payment without facing immediate enforcement. The solution combines Ethereum's public blockchain and off-chain private databases. At the time of publication, Ethereum operated under a PoW consensus mechanism, which inherently mitigated Sybil attacks due to the high computational cost and energy required to carry them out. Sensitive information, including user data, is stored in private databases rather than on-chain, minimizing risks of data exposure and further reducing the threat of Sybil attacks targeting personal records. Despite these design choices, the authors do not address the problem of data integrity regarding roaming activity. Specifically, the framework lacks a mechanism to verify whether the VMNO has submitted accurate CDR. As such, the VMNO could potentially manipulate usage data to inflate charges, and the system provides no safeguards to detect or prevent such behavior.

In the work proposed by (Weerasinghe et al. 2021), the system employs a score-based selection mechanism that ranks operators after each session, effectively incentivizing MNOs to maintain high service quality in order to be selected for future roaming or offload opportunities. At the time of publication, the system relied on the Ethereum blockchain, which used PoW consensus mechanism. This inherently limits Sybil attacks due to the high energy and computational costs required to compromise the network. However, while the use of performance-based scores adds value to the selection process, the framework lacks transparency about who has the authority to update these scores. This introduces a centralization risk-if a single party or untrusted entity is responsible for score updates, it opens the door to manipulation through falsified metrics. Moreover, the paper does not address how roaming settlement is conducted between the HMNO and the VMNO. In particular, the risk of poisoning attacks remains, as the VMNO could generate fake CDRs to inflate charges.

The framework proposed by (Nguyen et al. 2021) incorporates token-based incentives, where participants stake tokens and risk slashing if they behave maliciously or fail to participate, thereby discouraging inactivity. While this mechanism reduces laziness in confirming transactions, it does not fully prevent an MNO from avoiding the initiation of a transaction; for instance, by withholding the submission of CDR necessary for settling a roaming session. BlockRoam relies on PoS with dynamically selected committees. To mitigate Sybil attacks, the system requires operators to lock tokens and join through a deposit-based committee structure. The FTS algorithm is used to randomly select committee members in proportion to their stake, making it financially costly for a Sybil attacker to influence the network at scale. Since the solution does not handle roaming settlement directly, poisoning attacks are outside of the scope. However, because usage events are recorded on-chain in near real-time, the system introduces a risk of privacy leakage. Even partial transaction data may reveal user roaming patterns. The authors acknowledge this issue and clarify that the short settlement cycle is aimed more at fraud prevention than privacy protection. To address this limitation, they suggest the use of optional privacy layers or partial commit schemes in future enhancements.

	Lazy Participants	Sybil Attacks	Poisoning Attacks	Privacy Leakage
Mafakheri et al. 2021	Solved	Reduced	Not solved	Mitigated
Gong et al. 2025	Not applicable (focus on identity)	Mitigated	Not solved	Not solved
Refaey et al. 2019	Not solved (HMNO can delay payment)	Mitigated	Not solved	Mitigated
Weerasinghe et al. 2021	Solved	Mitigated	Not solved	Not addressed
Nguyen et al. 2021	Partially mitigated	Mitigated	Not applicable	Not solved

Table 4.1 – Security analysis of existing frameworks against the proposed threat model.

4.5 Challenges and Limitations

Following the analysis of the current state-of-the-art in blockchain-based 5G roaming frameworks, several limitations emerge that remain unaddressed and warrant further investigation:

- **Privacy of CDRs**: A common goal across many proposed solutions is to eliminate reliance on centralized intermediaries, such as the Data Clearing House, by automating the roaming settlement process through smart contracts. These contracts require access to usage data in the form of CDRs to initiate payment transfers from the HMNO to the VMNO. However, CDRs contain highly sensitive information that, if exposed, could compromise user privacy. Despite this, none of the works that adopt public blockchains (e.g., Ethereum) provide a concrete method to protect the privacy of CDRs once they are referenced or stored on-chain. This creates a significant privacy risk, as public ledgers are inherently transparent and immutable.

- **Strong Trust Assumptions and Lack of Dispute Mechanisms**: All reviewed solutions implicitly assume that the VMNO acts honestly when reporting the roaming activity of the user. This includes generating the CDR and submitting it to the blockchain accurately. However, in a real-world adversarial setting, a VMNO could behave maliciously and generate inflated or falsified CDRs to unjustly overcharge the HMNO. Current solutions lack built-in verification mechanisms to validate the authenticity of CDRs or to allow the HMNO to challenge potentially fraudulent records. The absence of such dispute resolution processes leaves the system vulnerable to abuse and undermines trust between roaming partners.

- **Scalability Challenges**: Another issue is scalability. Many proposed frameworks deploy their smart contracts on public blockchain networks-particularly Ethereum, although decentralized and secure, suffer from limited throughput and high latency. These networks do not yet support the scale and responsiveness required by a global, high-volume system such as 5G roaming.

The presented limitations highlight the ongoing need for more efficient, secure, and privacy-aware mechanisms in blockchain-based 5G roaming systems. Despite promising advancements, there remains a significant research gap in achieving true end-to-end verifiable and privacy-preserving blockchain-based 5G roaming settlement.

4.6 Conclusion

This chapter has explored the integration of blockchain technology into 5G roaming frameworks and assessed the security and privacy implications of recent proposals. A threat model tailored to the unique challenges of decentralized 5G roaming was introduced, and five notable blockchain-based solutions were evaluated accordingly, based on the literature available at the time of writing. Our analysis revealed that, while blockchain introduces valuable properties such as transparency, immutability, and automated transaction execution, several critical vulnerabilities remain. These include the lack of end-to-end privacy preservation for CDR data, insufficient mechanisms to verify the authenticity of CDR generation, and significant scalability limitations when applied to high-throughput, latency-sensitive 5G environments.

Despite these gaps, the surveyed approaches offer clear advantages over traditional systems. They reduce reliance on third-party clearinghouses, enable partial decentralization of roaming operations, and introduce mechanisms for earlier fraud detection. Notably, the use of real-time or near real-time ledger updates has the potential to greatly reduce delays in reporting roaming activity, which in conventional systems can take up to 18 hours for settlement processing (Starhome Mach 2018). However, most of the reviewed solutions still require further refinement, particularly in areas involving off-chain data verification and advanced privacy-preserving techniques.

In conclusion, to move blockchain-based 5G roaming solutions from experimental concepts to practical deployments, future research must confront the persistent challenges of privacy, data integrity, and network scalability. The following chapters of this work will build upon these findings and propose new methodologies that combine advanced

cryptographic tools with novel verification mechanisms. The goal is to design verifiable, privacy-preserving, and scalable blockchain-based roaming architectures that can support the full operational complexity of global 5G roaming.

Part II

Contribution

Chapter 5

Contribution

5.1 Introduction

In the preceding chapter, we thoroughly analyzed existing blockchain-based roaming solutions and identified several critical gaps. Specifically, we highlighted significant issues related to user privacy, assumptions of inherent trust between mobile operators, and substantial scalability bottlenecks. Existing frameworks often either compromise user confidentiality by exposing sensitive usage data on-chain or rely extensively on trust between operators, thus limiting their robustness and practicality in real-world scenarios. Additionally, current blockchain implementations struggle to efficiently handle the high transaction throughput necessary for 5G roaming services.

To address these limitations, this chapter introduces B5GRoam, a novel framework explicitly designed to ensure secure, private, and efficient roaming settlements in 5G networks. At a high level, our solution establishes secure roaming agreements between HMNO and VMNO through smart contracts that remove the necessity for mutual trust. These contracts act as automated, verifiable agreements governing roaming terms and rates. Further, our solution integrates Trusted Execution Environments (TEEs) and non-interactive zero-knowledge proofs (zkSNARKs) to provide robust privacy guarantees. VMNOs utilize zkSNARK proofs to cryptographically prove billing amounts without exposing sensitive usage metrics. To our knowledge, this is the first attempt to utilize zero-knowledge cryptography to secure 5G roaming's privacy.

To overcome scalability issues inherent in blockchain systems, B5GRoam employs a Layer-2 zkRollup approach, significantly enhancing transaction throughput and cost efficiency. This combination of smart contract automation, TEE-enhanced privacy, cryptographic verification via zkSNARKs, and zkRollup-based scalability collectively positions B5GRoam as a comprehensive and advanced solution capable of effectively meeting the rigorous demands of 5G roaming.

In the following sections, we will incrementally detail these innovative aspects of our framework, thoroughly demonstrating how each component contributes to overcoming previously identified gaps and collectively ensures secure, scalable, and privacy-preserving roaming settlements.

5.2 Design Objectives and Requirements

In this section, we present the design objectives our solution attempts to achieve and the requirements for that.

5.2.1 Design Objectives

Based on the limitations derived from our comprehensive analysis in Chapter 4, we identify three main objectives that our solution aims to satisfy:

- **Privacy**: Chapter 4 revealed that existing blockchain-based solutions often expose sensitive CDRs directly on-chain or require intermediaries to process user data, significantly undermining user confidentiality. This exposure can lead to potential on-chain privacy leakage attacks, enabling unauthorized observers or competitors to infer sensitive user behavior and consumption patterns. Hence, our design must explicitly address this critical vulnerability by employing cryptographic commitments, ensuring that raw usage data remains confidential and protected at all times while still enabling the VMNO to provide accurate billing costs.
- **Trustlessness**: Our threat model clearly indicated that relying on trust between operators could result in serious risks, such as data poisoning attacks, where visited operators can maliciously manipulate usage data to unfairly charge the home operators. Thus, our solution must eliminate this trust assumption and enforce roaming agreements automatically. Moreover, the system should be resistant even if one of the MNOs acts maliciously.
- **Scalability**: The detailed analysis presented in Chapter 4 demonstrated the severe limitations of blockchain scalability, particularly within Layer-1 networks such as Ethereum, which exhibit low transaction throughput, high latency, and significant transaction costs. Given that roaming settlements in 5G networks necessitate processing numerous transactions rapidly and economically, our solution must resolve the scalability issue, and the system must keep performing very well even under high-pressure periods.

To effectively meet our previously outlined design objectives, we detail the key functional and non-functional requirements of our proposed B5GRoam framework. These requirements form the baseline for our subsequent design and implementation phases.

5.2.2 Functional Requirements

The functional requirements define the core capabilities that B5GRoam must provide to ensure secure, trustless, and privacy-preserving 5G roaming settlements:

- **Establishing Roaming Agreements:** The system must allow HMNOs and VMNOs to securely agree on roaming rates and terms using verifiable cryptographic signatures.
- **Smart Contract Automation:** The solution should automatically enforce the terms encoded within roaming agreements when settling roaming sessions via blockchain-based smart contracts, ensuring transparent and automated settlements without manual intervention.

- **Secure Session Initialization:** B5GRoam must support a secure and authenticated roaming session initialization, triggered by authenticated signatures from the HMNO.
- **Proving cost calculation using zkSNARKs:** VMNOs must generate zero-knowledge proofs to cryptographically prove billing calculations, ensuring accuracy without exposing roaming usage data.
- **Privacy-Preserving CDR Submission:** The system should verify that the CDR used to calculate the total cost correctly matches the actual usage from the User Equipment (UE), using cryptographic commitments calculated within a Trusted Execution Environment (TEE).
- **Automatic Trustless Settlement:** Smart contracts should verify zkSNARK proofs and commitments and automatically execute the settlement payments to VMNOs upon successful verification.

5.2.3 Non-Functional Requirements

Non-functional requirements define the operational attributes and quality characteristics the B5GRoam system must exhibit to provide a viable, secure, and scalable solution:

- **Privacy and Confidentiality:** Sensitive user information, particularly CDRs, must remain confidential and should never be exposed.
- **Security:** The solution must be robust against the threats identified in 4.2, including Sybil attacks, poisoning attacks, and lazy participant behaviors, maintaining the integrity and authenticity of all transactions.
- **Scalability and Performance:** The framework should support high transaction throughput suitable for extensive 5G roaming activities, ensuring minimal latency and acceptable transaction costs.
- **Accurate Billing**: The system must enforce the VMNO to generate accurate total cost and allow HMNO to dispute the settlement in case of incorrect billing is detected.
- **Transparency and Auditability:** While maintaining data privacy, the system must allow sufficient transparency for auditing purposes, enabling authorized entities to verify compliance and correctness of settlements easily.
- **Interoperability:** B5GRoam must be designed with interoperability considerations, ensuring seamless integration with existing 5G infrastructure components, roaming protocols.

5.3 System Design

At the core of our approach lies a modular and robust architecture carefully designed to meet the key objectives outlined in Section 5.2. This section provides a comprehensive breakdown of the system's structural components and operational logic. We begin by describing the different architectural layers that compose our solution. We then introduce the key actors involved in the framework and explain their respective responsibilities. Following that, we detail the major modules that make up the B5GRoam system and

explain how they interact. Finally, we present a high-level system overview along with the main operational flows, illustrating how the components work together to enable secure and efficient 5G roaming settlements.

5.3.1 System Actors

The B5GRoam framework involves three main actors, each playing a critical role in ensuring the correct functioning of the roaming authentication, usage tracking, and trustless settlement processes. These actors include the UE, the HMNO, and the VMNO. Their interactions are governed by predefined roaming agreements enforced via smart contracts as detailed in Section 5.4.

5.3.1.1 User Equipment (UE)

Also referred to as the subscriber, the UE represents the mobile device owned and operated by the end user. In the B5GRoam framework, we assume that the UE is equipped with a Trusted Execution Environment (TEE), which is used to handle sensitive data and compute cryptographic commitments over usage records. This assumption is practical and well-founded, as most modern smartphones rely on ARM-based processors that support ARM TrustZone, a hardware-based security extension that enables isolated execution environments (Arm Ltd. 2020).

Additionally, major smartphone manufacturers provide their own TEE implementations on top of TrustZone. For instance, Samsung's *TEEGRIS* is widely used across its devices to execute sensitive operations such as biometric verification, digital rights management, and cryptographic computations (Samsung Electronics 2025). In our framework, the TEE is responsible for securely computing a commitment over CDR using a privacy-preserving hash function and submitting this commitment to the blockchain. This process ensures that the UE's raw usage data remains confidential, even as it participates in a verifiable billing and settlement protocol.

5.3.1.2 Home Mobile Network Operator (HMNO)

The Home Mobile Network Operator is the original service provider with which the UE holds a subscription. In traditional 5G roaming scenarios, the HMNO plays a central role in authenticating its subscribers and coordinating access to foreign networks. In B5GRoam, the HMNO is responsible for issuing a digitally signed authorization that attests to the authenticity of the UE when roaming is initiated. This signature is later consumed by the blockchain smart contract to authorize and open a roaming session.

From a settlement perspective, the HMNO acts as the payer. It locks the estimated cost of the session into an on-chain escrow contract at the beginning of the roaming session. This role ensures that the VMNO is compensated fairly and without delay, once the session concludes and the billing proof is verified. Importantly, the HMNO does not need to be continuously involved in the session; its role is confined to issuing the initial signature and funding the escrow.

5.3.1.3 Visited Mobile Network Operator (VMNO)

The Visited Mobile Network Operator provides connectivity and services to the UE while it is roaming outside its home network. This includes handling SMS, voice calls, and data usage. The VMNO is responsible for enforcing network policies, recording usage information, and initiating the settlement process once the session concludes.

Upon receiving the HMNO's signed authorization and successfully authenticating the UE via standard 5G procedures, the VMNO interacts with the smart contract to open the roaming session. At the end of the session, it collects usage metrics, calculates the total billing amount based on the roaming agreement, and generates a zero-knowledge proof demonstrating the correctness of the billing. This proof is then submitted to the on-chain verifier, which validates it against the UE's previously submitted commitment. If the verification succeeds, the locked funds are released directly to the VMNO's address. This role positions the VMNO as the primary initiator of the trustless settlement mechanism in B5GRoam.

5.3.2 System Layers

Our solution is composed of four main layers, each serving a distinct function and working together to meet the design objectives of privacy, trustlessness, and scalability. These layers are depicted in Fig. 5.1 and described in detail below.

5.3.2.1 5G Core Layer

The 5G Core Layer represents the foundational telecommunications infrastructure responsible for managing user connectivity, mobility, and session orchestration. It includes critical components such as the Access and Mobility Management Function (AMF), Session Management Function (SMF), and User Plane Function (UPF). This layer handles core network responsibilities, including user authentication and session establishment, for both domestic and roaming subscribers. In our architecture, the 5G Core Layer remains unchanged and operates independently of blockchain logic. However, it serves as the initial trigger point for roaming events, feeding authenticated session data into the next layer for secure, decentralized processing.

5.3.2.2 5G Roaming Interaction Layer

The 5G Roaming Interaction Layer acts as the bridge between traditional 5G roaming procedures and the decentralized settlement mechanisms introduced in our framework. This layer is activated during the establishment of a new roaming session. Upon successful authentication of the UE, the HMNO generates a digital signature asserting the legitimacy of the user. The VMNO then interacts with a smart contract to register the new roaming session and to lock an estimated cost in the contract. This integration ensures accountability and auditability without introducing trust assumptions between operators.

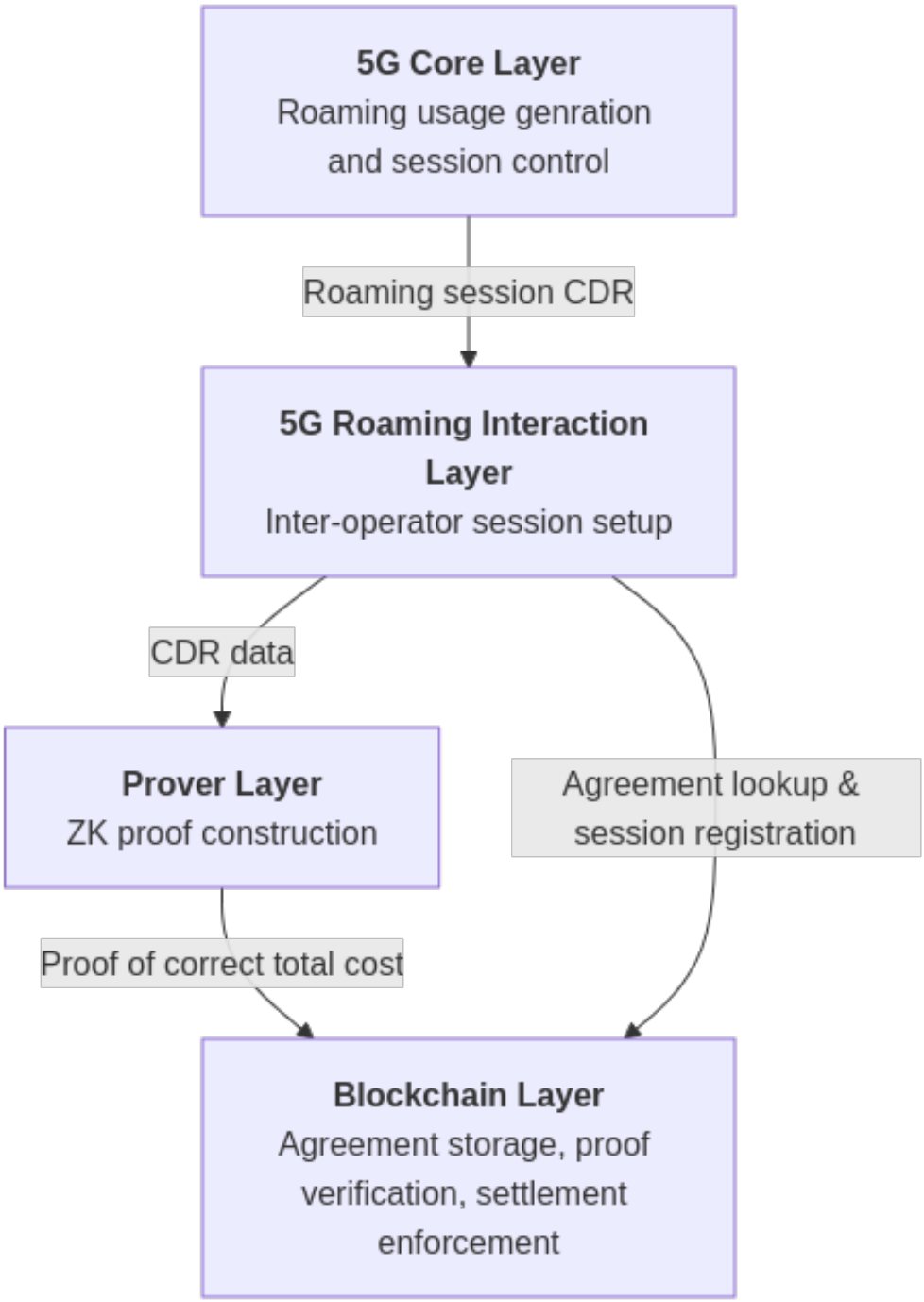


Figure 5.1 – System Layers

5.3.2.3 Prover Layer

The Prover Layer plays a critical role in preserving user privacy while ensuring billing correctness. This layer is invoked at the end of the roaming session and is utilized by both the UE and the VMNO. The UE uses a Trusted Execution Environment (TEE) to compute a secure commitment over its roaming usage, while the VMNO calculates the

total cost based on the agreed rates and generates a zero-knowledge proof (zkSNARK) attesting to the correctness of this computation. These proofs allow billing verification without revealing actual CDRs or usage details, providing strong cryptographic privacy guarantees.

5.3.2.4 Blockchain Layer

The Blockchain Layer orchestrates the entire trustless roaming settlement process. It hosts smart contracts that enforce roaming agreements, manage session states, verify zero-knowledge proofs, and facilitate automated payments. By verifying the zkSNARK proofs against the UE's commitment and the predefined agreement, the smart contract can securely release locked funds to the VMNO without relying on either party's honesty. This layer also incorporates a Layer-2 zkRollup mechanism to bundle multiple operations into a single proof, substantially improving throughput and reducing on-chain costs. It interfaces with both the Prover Layer and the 5G Roaming Interaction Layer, thereby completing the loop of a fully decentralized, secure, and efficient roaming settlement system.

5.3.3 Framework Modules

The B5GRoam framework is composed of several modular components that interact to achieve a secure, privacy-preserving, and trustless roaming settlement system. Figure 5.2 illustrates the architectural modules and their interconnections. Each module plays a specific role, and the interaction among them ensures that the roaming agreement is enforced on-chain, usage data is processed securely, and settlements are performed verifiably. Below, we describe each module in detail.

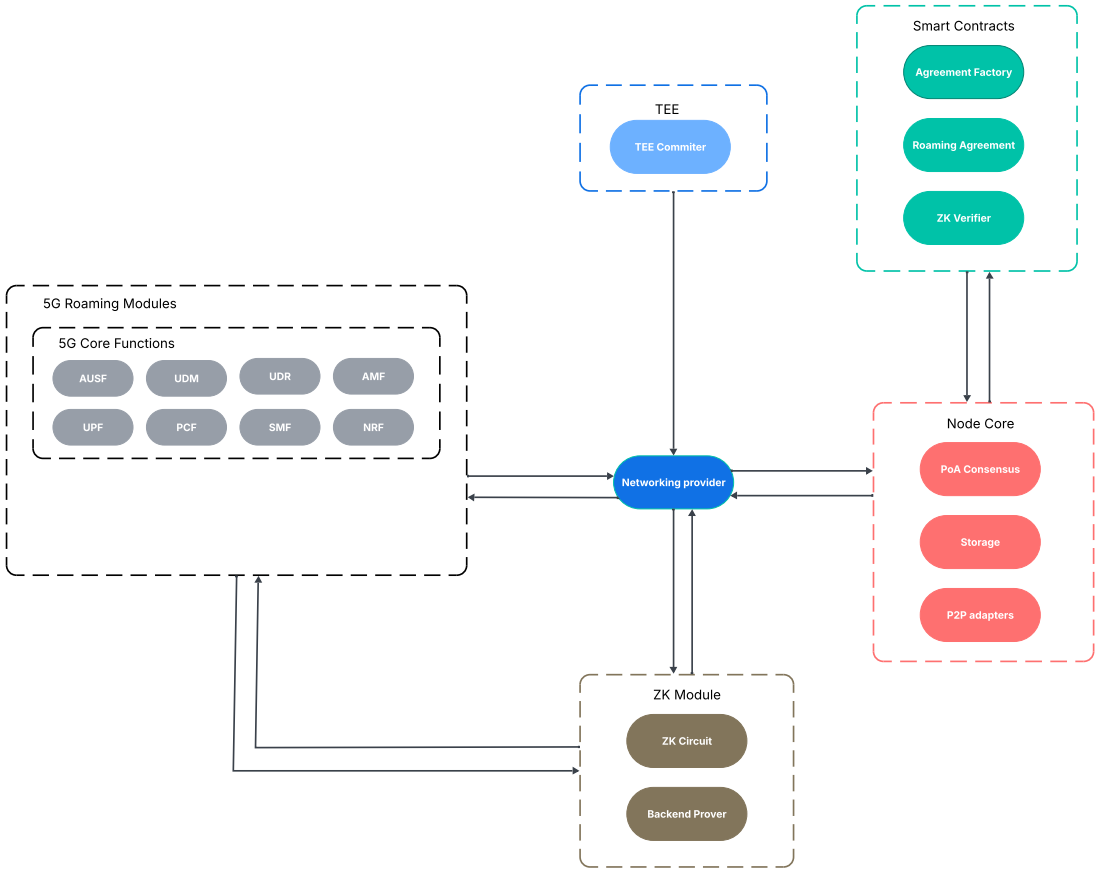


Figure 5.2 – Framework modules

5.3.3.1 5G Roaming Modules

This module represents the standard 5G Core architecture, composed of key network functions including AMF, SMF, UDM, PCF, UPF, AUSF, UDR, and NRF. These functions are responsible for user authentication, session management, policy control, and routing. This module operates independently of the blockchain layer and is kept unmodified to ensure compatibility with existing 5G infrastructure. However, it interfaces with the networking provider to trigger the initiation of roaming sessions and transmit relevant session data for blockchain interaction.

5.3.3.2 TEE Module

The Trusted Execution Environment (TEE) module contains a secure enclave (e.g., Intel SGX or ARM TrustZone) capable of securely processing sensitive data from the User Equipment (UE). Its main component, the *TEE Committer*, receives raw CDRs generated during a roaming session and produces a cryptographic commitment using a privacy-preserving hash function such as Poseidon and submits the commitment to on-chain contracts. This commitment ensures that the UE's actual usage remains private while still allowing the VMNO to later prove billing correctness via zero-knowledge proofs. The TEE module interfaces with the Networking Provider to transmit its output to the on-chain verification process.

5.3.3.3 ZK Module

The Zero-Knowledge (ZK) Module is responsible for generating cryptographic proofs that verify the correctness of roaming charges without exposing sensitive user data. It is composed of two main subcomponents:

- **ZK Circuit:** Encodes the arithmetic logic for billing computation. It defines the constraints required to prove that the total roaming cost is computed using agreed-upon rates.
- **Backend Prover:** It produces a zkSNARK proof attesting that the billing was done correctly according to the logic defined in the circuit and given a set of inputs (rates, usage data, and total cost),

5.3.3.4 Networking Provider

The Networking Provider module serves as the middleware that connects the 5G domain, the TEE, the ZK module, and the blockchain. It facilitates secure and reliable communication between these components and ensures the correct forwarding of session metadata, signatures, commitments, and proofs. This module abstracts protocol-specific concerns and enables interoperability between the off-chain and on-chain environments.

5.3.3.5 Smart Contracts Module

This module encapsulates the decentralized logic for roaming settlement deployed on the blockchain. It contains three key smart contracts:

- **Agreement Factory:** A contract that allows HMNOs and VMNOs to register their bilateral roaming agreement by verifying mutual digital signatures and deploying a dedicated Roaming Agreement contract instance.
- **Roaming Agreement:** Each agreement instance governs the lifecycle of roaming sessions under the agreed terms. It stores roaming session states, holds escrowed funds, and manages billing settlement upon proof verification.
- **ZK Verifier:** A smart contract that verifies zkSNARK proofs submitted by the VMNO to validate billing correctness. It ensures that the submitted proof matches the UE's previously committed usage and the agreed rates.

These contracts form the core of B5GRoam's trustless execution environment, guaranteeing transparency and automation.

5.3.3.6 Node Core

The Node Core represents the underlying blockchain infrastructure. It includes:

- **PoA Consensus:** A Proof-of-Authority consensus mechanism used for validating blocks in a permissioned network.
- **Storage:** Responsible for maintaining the state of smart contracts and storing event logs.

- **P2P Adapters:** Enables peer-to-peer networking between blockchain nodes to propagate transactions and blocks.

The Node Core executes the smart contracts and ensures that all participants observe the same canonical state.

5.4 System Flow

This section presents a detailed walkthrough of the B5GRoam protocol, illustrating the end-to-end lifecycle of a roaming session, from agreement registration to payment settlement. Each step highlights the interactions between the system's actors and components, as well as the data and trust transitions that occur across the 5G, cryptographic, and blockchain layers. The flow diagram (Fig 5.4) serves as a visual reference for understanding the sequence of operations

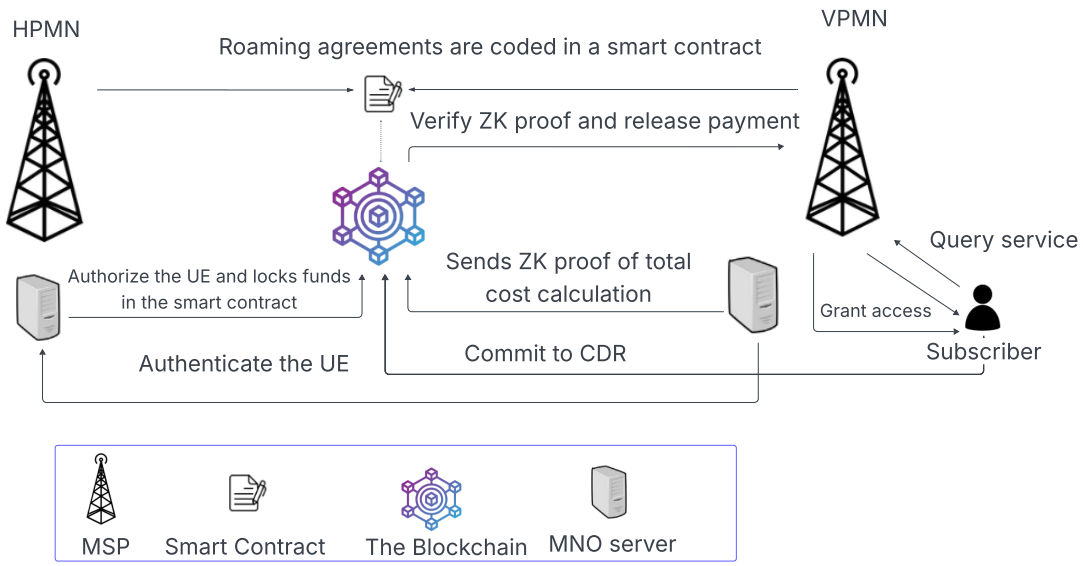


Figure 5.3 – B5GRoam flow

5.4.1 Roaming Agreements Creation

Before any roaming session can take place, the HMNO and the VMNO must first establish a bilateral agreement that defines the economic terms under which roaming services will be delivered and settled. In B5GRoam, this agreement is formalized and encoded as an on-chain smart contract, allowing it to be publicly auditable, cryptographically verifiable, and automatically enforceable.

Agreement Parameters. The roaming agreement encodes the following essential parameters:

- **Service rates:** Unit costs for SMS messages, voice call minutes, and data usage (per megabyte).
- **HMNO and VMNO identifiers:** Ethereum addresses that represent the two operators in the agreement.

- **Agreement metadata:** Expiration timestamp, versioning information, and a unique agreement ID.

To ensure that both the HMNO and VMNO consent to the agreement parameters, B5GRoam employs an off-chain ECDSA signature mechanism. Each operator signs the agreement hash using their private key, and these signatures are submitted to the blockchain for verification.

The raw agreement parameters-the HMNO and VMNO addresses, service rates (SMS, voice, data), and metadata (e.g., expiration timestamp, nonce)-are first ABI-encoded into a byte array. This array is then hashed using the Keccak-256 hash function, which produces a 32-byte digest:

```
message_hash = keccak256(abi.encodePacked(
    hmno_address,
    vmno_address,
    sms_rate,
    voice_rate,
    data_rate,
    expiry,
    nonce
));
```

Each operator then generates an ECDSA signature (Johnson et al. 2001) over the `message_hash` using their respective private keys. The ECDSA signature is a tuple (v, r, s), where:

- $r = (k \cdot G)_x \mod n$, where k is a random nonce and G is the secp256k1 generator point.
- $s = k^{-1}(H(m) + r \cdot d) \mod n$, where d is the private key and $H(m)$ is the hashed message.
- v is the recovery ID (27 or 28 on Ethereum) used to recover the signer's public key.

This produces the final ECDSA signature (v, r, s), which is sent along with the original agreement parameters to the `AgreementFactory` smart contract.

On-chain Verification. Upon receiving the agreement parameters and both ECDSA signatures, the `AgreementFactory` contract reconstructs the `message_hash` using the same hashing logic. It then uses the `ecrecover` precompiled contract in Ethereum to extract the signer's address from each signature:

```
signer = ecrecover(message_hash, v, r, s);
```

The contract ensures that:

- The recovered address from each signature matches the declared HMNO and VMNO addresses.
- The agreement hash has not already been used to create another contract (preventing replay).

Only when both signatures are valid and correctly match the operator identities does the contract proceed to deploy the corresponding `RoamingAgreement` instance. This verification process guarantees authenticity and mutual agreement while preserving decentralization and eliminating reliance on any centralized signature authority.

The `RoamingAgreement` instance is initialized with the agreed parameters and becomes the canonical reference for any roaming session between the two operators. Its address serves as a unique identifier for future interactions.

The `RoamingAgreement` contract exposes entry points for:

- Starting a roaming session by registering a new user session along with an HMNO authorization signature.
- Locking the estimated cost in the contract.
- Releasing funds to the VMNO upon successful zk proof verification.

Fig. 5.4 illustrates the sequence flow:

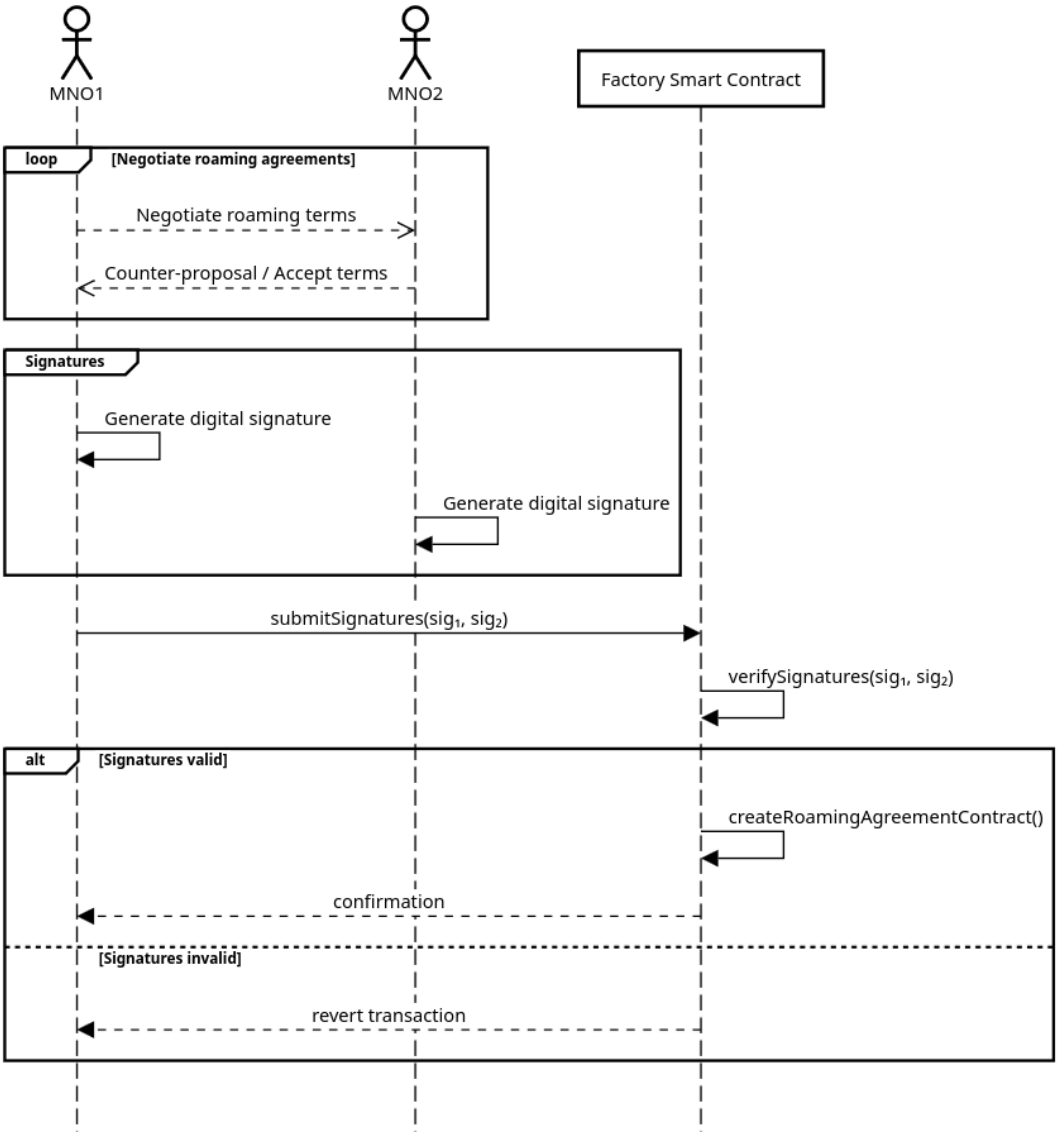


Figure 5.4 – Creating Agreement Sequence Flow

5.4.2 Subscriber Roaming Session Initiation and Authorization

Once a valid roaming agreement has been established and deployed on-chain, the next phase involves the initiation of an individual subscriber roaming session. This phase ensures that the User Equipment (UE) is authenticated by the HMNO and authorized to roam under the agreed terms. The session setup is secured using a digitally signed authorization and accompanied by an on-chain locking of the estimated session cost.

Service Query and Access Request. The process begins when a UE enters the VMNO's geographical coverage area and attempts to access network services. This is a standard 5G roaming behavior, where the UE issues a service query and performs a registration request (e.g., using the `N1/N2 interface`) to the local Visited Access and Mobility Management Function (AMF). The VMNO's AMF initiates the authentication procedure by forwarding the UE's credentials and identity to the HMNO. Once the HMNO receives the authentication request, it validates the UE's identity via the AUSF and UDM core functions and responds with an authentication vector. This completes the mutual authentication phase at the protocol level, enabling the VMNO to trust the subscriber's legitimacy.

Generation of the Roaming Session Authorization Signature. In B5GRoam, the classical authentication process is augmented with a blockchain-bound authorization. Upon successful UE authentication, the HMNO computes a digital signature over a structured session payload that includes:

- The UE's unique identifier, presented as a unique Ethereum address.
- The address of the `RoamingAgreement` smart contract.
- The estimated cost of the session
- A unique session nonce to prevent replay attacks.

This payload is ABI-encoded and hashed using `keccak256`. The HMNO signs this hash using ECDSA, producing a signature (v, r, s). This signature authorizes the `RoamingAgreement` contract to lock the estimated session funds on behalf of the HMNO, and is valid only for the specific UE and roaming session.

Session Registration on the Blockchain. The signed authorization is then forwarded to the VMNO, which is responsible for invoking the `startSession` function on the corresponding `RoamingAgreement` smart contract. This function takes the following parameters:

- The ABI-encoded session payload.
- The ECDSA signature (v, r, s) from the HMNO.

The contract performs the following logic:

1. Reconstruct the message hash and use `ecrecover` to extract the signer's address.
2. Check that the signer is indeed the HMNO registered in the agreement.
3. Validate that the nonce has not been used (anti-replay).
4. Transfer the estimated funds from the HMNO's on-chain balance into the contract.
5. Store the session metadata (UE address, VMNO address, session status, session start time, etc.) in the contract's state.

If all checks pass, the session is successfully registered, and the UE is considered authorized to roam. The locked funds are held until the session concludes and billing verification is performed.

Fig5.5 illustrates the sequence flow:

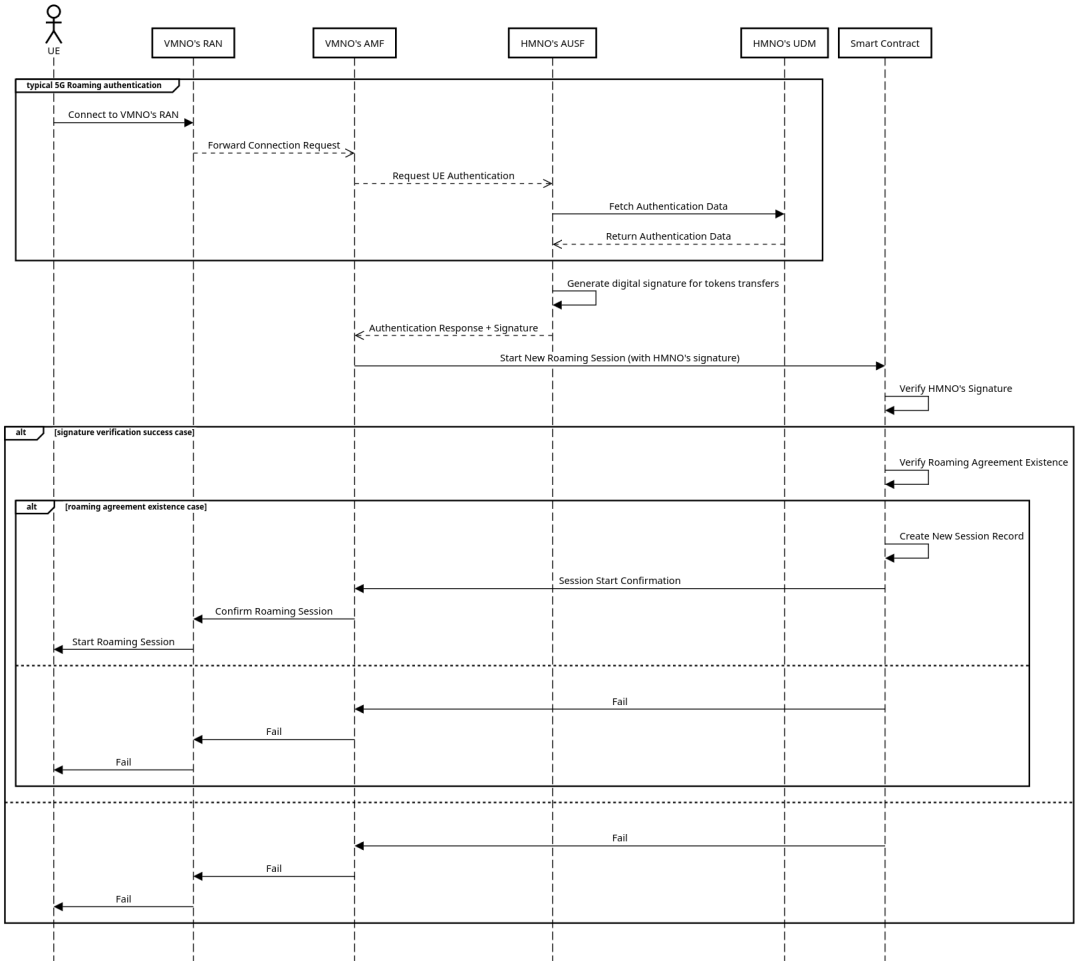


Figure 5.5 – Starting Roaming Session Sequence Flow

5.4.3 Usage Commitment via TEE

Once the subscriber's roaming session is active and the user begins consuming services, B5GRoam requires a mechanism for recording this usage in a way that enables verifiable settlement without compromising user privacy. Submitting raw usage data to the blockchain would violate confidentiality and expose sensitive behavioral patterns. To address this, the UE securely generates a cryptographic commitment to its CDR within a TEE and submits it on-chain, as depicted in Fig. 5.6.

The use of a TEE is critical in this process: it ensures that the commitment is computed over authentic, tamper-proof usage data directly sourced from the UE. Without this hardware-based isolation, a malicious subscriber could modify their CDR before hashing it, or even collude with the VMNO to overreport usage and increase settlement amounts. By anchoring the commitment computation inside a trusted, verifiable enclave (e.g., ARM TrustZone or Intel SGX), B5GRoam ensures that the resulting hash accurately represents

the true usage, independent of the user's intentions.

The TEE-generated commitment thus serves a dual purpose: it preserves privacy while simultaneously acting as a cryptographic anchor for trustless billing validation.

Throughout the roaming session, the UE logs its own activity locally, tracking the number of SMS sent, total voice call duration, and data usage in megabytes. These metrics are structured into a CDR tuple:

```
CDR = {
    sms_count: uint64,
    voice_minutes: uint64,
    data_megabytes: uint64
}
```

To prepare the CDR for commitment, the UE performs the following:

- Packs the CDR fields into a fixed-length binary encoding.
- Computes a cryptographic hash over this encoded data.

The hash computation is executed inside the TEE, which guarantees confidentiality and tamper-resistance. B5GRoam uses the Poseidon hash function due to its SNARK-friendly algebraic structure, allowing efficient use inside zkSNARK circuits.

$$h = \text{Poseidon}(sms\_count, voice\_minutes, data\_megabytes)$$

This commitment h serves as a cryptographic fingerprint of the CDR. The resulting hash is submitted to the blockchain. The TEE invokes the `commitCDR` function on the `RoamingAgreement` smart contract, submitting the roaming session ID and the calculated Poseidon hash h by signing a transaction using its private key.

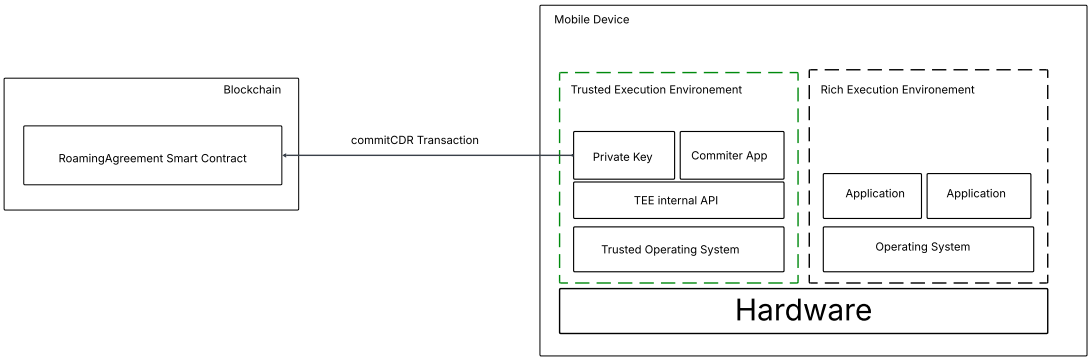


Figure 5.6 – UE Commitment Diagram

The `RoamingAgreement` smart contract first verifies that the UE's address making the call matches the one stored with the session metadata and then stores this commitment on-chain, indexed by the session ID. Only one commitment per session is allowed, ensuring immutability and preventing rollback attacks.

This step lays the cryptographic foundation for the VMNO to later prove billing correctness using a zkSNARK, while ensuring that the user retains full privacy over their detailed usage metrics.

5.4.4 Zero-Knowledge Billing and Settlement Verification

After the roaming session concludes and the UE has submitted its usage commitment on-chain, the final step in the B5GRoam protocol involves verifying that the billed amount is accurate and consistent with that commitment, without revealing the actual usage data. This process is initiated by the VMNO, who is responsible for proving the correctness of the billing calculation and triggering settlement on-chain.

To finalize payment, the VMNO must demonstrate that the total cost it claims corresponds to actual usage and adheres to the rates defined in the roaming agreement. This must be done in a verifiable manner, ensuring that the HMNO does not overpay and that the protocol remains trustless. One general-purpose approach to verifiable off-chain computation is the use of *dispute games*, where a prover submits a result and other parties are allowed to challenge it within a specified time window. However, dispute games rely on an important assumption: the challenger must have access to the underlying data to validate or challenge the claim.

In the case of roaming settlement, this assumption breaks down-only the VMNO and the UE have access to the subscriber's usage data, and that data is intentionally kept private. The HMNO (or any other party) has no way to verify the correctness of the claim without violating user privacy. As a result, dispute games are not applicable here, since no external challenger can meaningfully assess or dispute the VMNO's computation.

Instead, B5GRoam uses zero-knowledge proofs (ZKPs) to allow the VMNO to prove that:

- The billed cost was computed correctly based on the agreed rates stored in the `RoamingAgreement` smart contract.
- The usage values used in that calculation match the hash commitment previously submitted by the UE.

This proof is generated off-chain and submitted on-chain along with the total claimed cost and session identifier. The `RoamingAgreement` smart contract verifies the proof using a built-in verifier component, and if the proof is valid, it unlocks the locked funds to the VMNO.

If the proof verification fails, and given the properties of ZK proofs as will be discussed in 6.5, it only means that the VMNO is acting maliciously, either by using incorrect rates or using incorrect usage data. In such a case, the settlement is rejected, and the funds remain locked until valid proof is submitted. This ensures that no invalid or unverifiable billing can be enforced.

Through this mechanism, B5GRoam achieves trustless and private settlement, ensuring that roaming agreements are respected, users' data remains protected, and operators are compensated fairly without relying on centralized intermediaries and without the need for an interactive or dispute-driven process.

Fig 5.7 illustrates the sequence flow:

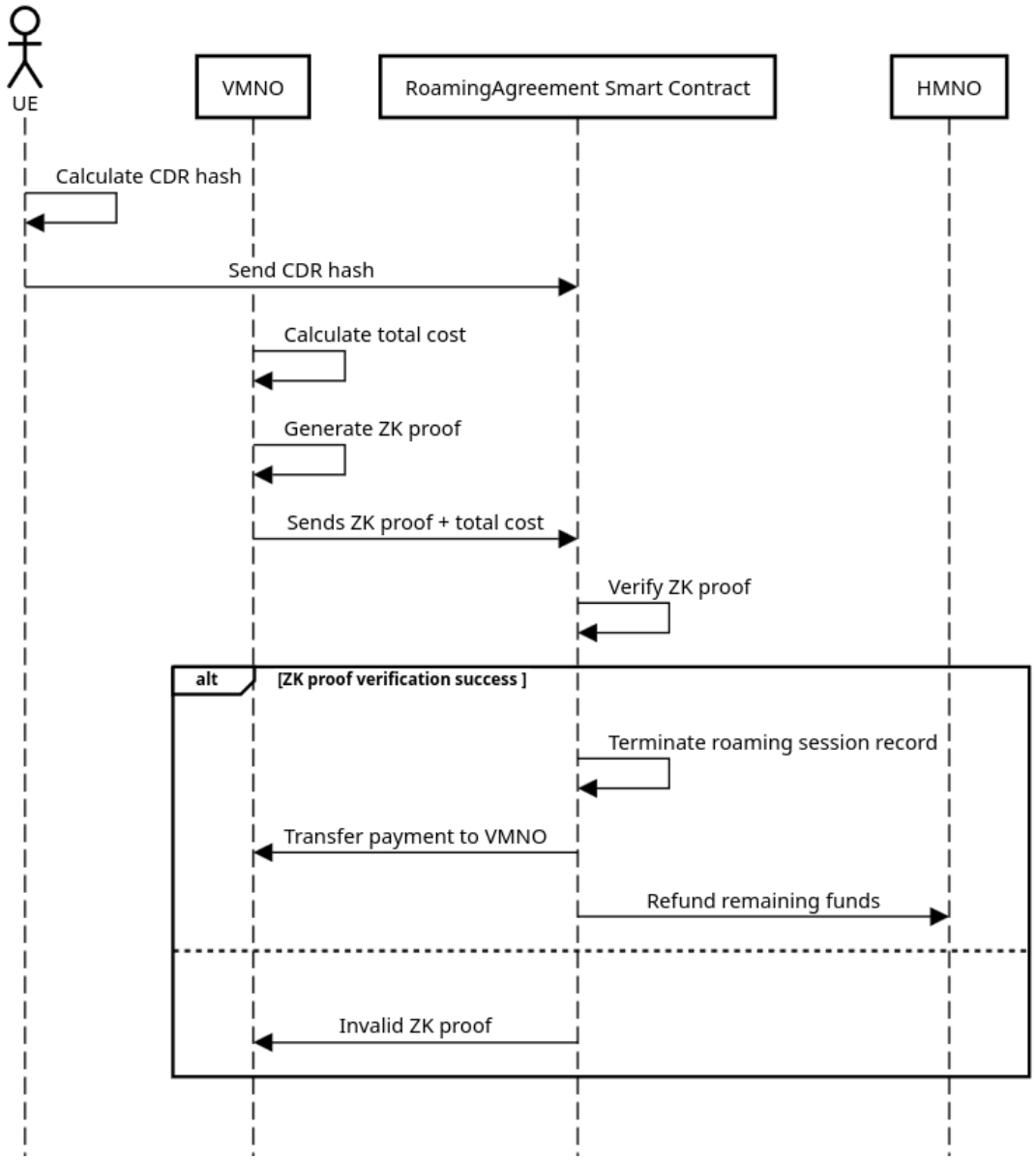


Figure 5.7 – Terminating Roaming Session Sequence Flow

5.4.5 ZK-Rollups

A crucial architectural decision in B5GRoam is the selection of the underlying blockchain infrastructure used for executing roaming settlement logic. After careful consideration, we opted for the Ethereum public blockchain as the settlement layer for our protocol. This decision is motivated by the unparalleled security guarantees offered by Ethereum's highly decentralized validator set and the substantial amount of value staked in its consensus mechanism. By anchoring our protocol on Ethereum, we inherit a mature, robust, and widely trusted security foundation, minimizing the risks associated with novel or less battle-tested networks. Additionally, since B5GRoam's design ensures that no sensitive or private data is ever directly stored on-chain, we do not face the privacy risks commonly associated with public blockchains.

To address the scalability and transaction cost limitations of using Ethereum L1 di-

rectly as discovered in 8.3.3, we further adopt **zk-rollups** as our execution environment for settlement transactions and proof verification. Zk-rollups combine the security of Ethereum with massive improvements in throughput and cost efficiency by offloading most computation from the main chain. While the experimental and benchmarking results in 8.3.3 will provide a quantitative justification for this choice, here we present a conceptual overview of how zk-rollups operate and their relevance to B5GRoam.

A zk-rollup is a layer-2 protocol that bundles ("rolls up") hundreds or thousands of transactions into a single batch, which is processed off-chain but anchored to Ethereum L1 through succinct cryptographic proofs. The workflow of a zk-rollup, as illustrated in Fig5.8, comprises three main stages:

1. **Off-chain Transaction Execution:** Users submit their transactions to the zk-rollup, where a specialized operator or sequencer executes these transactions in an off-chain environment. The state transitions–such as payments, settlements, or contract interactions–are computed and stored off L1, dramatically reducing the load on Ethereum's mainnet.
2. **Proof Generation:** For each batch of executed transactions, the zk-rollup sequencer generates a zero-knowledge proof (typically a zk-SNARK or zk-STARK). This proof attests that all transactions in the batch were processed correctly, that the resulting new state is valid, and that all protocol rules were enforced.
3. **On-chain Proof Verification and State Update:** The sequencer submits both the succinct proof and the updated state root to an on-chain rollup contract deployed on Ethereum L1. The contract verifies the proof's validity and, if correct, updates the canonical state accordingly. All settlement outcomes are thus cryptographically guaranteed and secured by Ethereum's consensus.

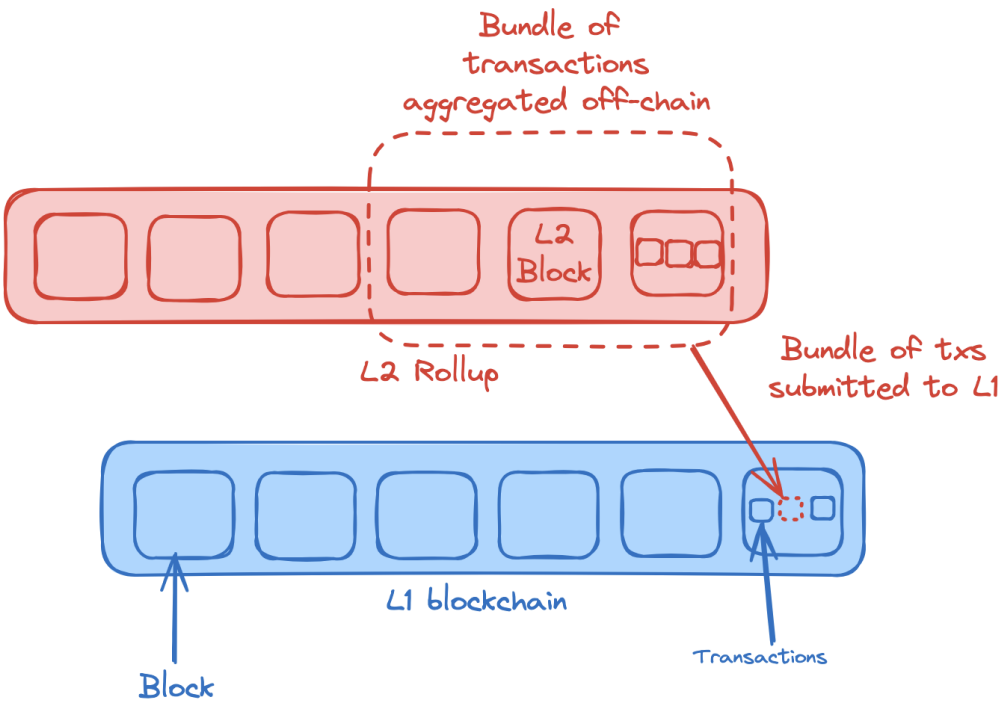


Figure 5.8 – ZK-Rollup Illustration

By adopting zk-rollups, B5GRoam achieves scalability and cost-effectiveness without sacrificing security or decentralization. The protocol benefits from rapid settlement finality, extremely low per-transaction fees, and the composability of Ethereum's smart contract ecosystem–all while retaining the strong privacy guarantees of zero-knowledge proofs.

Chapter 6

Zero Knowledge Proof Generation

6.1 Introduction

Zero-Knowledge Proofs (ZKPs) are a fundamental cryptographic primitive that enables one party (the prover) who possesses some secret knowledge w relating to a public statement x to convince another party(the verifier) that x is true, based on knowledge of w, without revealing any information about w itself. This powerful notion was first formalized by Goldwasser, Micali, and Rackoff in their seminal paper *"The Knowledge Complexity of Interactive Proof-Systems"* (Goldwasser et al. 2019), which introduced the concept of knowledge complexity and established the foundation for the zero-knowledge paradigm.

Since their introduction, ZKPs have evolved from interactive protocols (requiring multiple rounds of communication) to highly efficient non-interactive protocols. The latter were made practical by the Fiat-Shamir heuristic (Fiat et al. 1986), leading to Non-Interactive Zero-Knowledge proofs (NIZKs).

In the B5GRoam framework, ZKPs underpin the trustless settlement of roaming charges by allowing the VMNO to prove correct billing without revealing the underlying usage data–a capability that would be impossible with conventional cryptographic tools.

In this chapter, we will explore the different ZK constructions that can be used in our system, discuss the circuit that encodes the VMNO's proof, and the security properties the ZK proof guarantees.

6.2 ZKP Statement and Circuit Logic

At the core of B5GRoam's trustless settlement mechanism lies the zero-knowledge circuit, which formalizes the statement that the VMNO must prove to the smart contract verifier. The essence of the proof is twofold:

- The VMNO must demonstrate that the total cost billed for a roaming session was computed correctly according to the agreed-upon rates.
- The VMNO must prove that the CDR used to calculate the total cost is consistent with the hash commitment previously submitted by the UE.

In formal terms, the VMNO seeks to prove that he used private values `sms_count`, `voice_minutes`, and `data_megabytes`, such that when the public rates `rate_sms`, `rate_voice`, `rate_data` (specified in the on-chain agreement) are applied, the resulting cost `total_cost` matches the public input, and the Poseidon hash of the usage tuple matches the on-chain commitment submitted earlier by the UE.

Symbolically, the public inputs to the circuit are:

- `commit_hash`: The hash commitment posted by the UE.
- `rate_sms`, `rate_voice`, `rate_data`: The unit rates for SMS, voice minutes, and data, as defined in the agreement.
- `total_cost`: The calculated total cost the VMNO is entitled to.

The private inputs (witness) are:

- `sms_count`: Number of SMS messages sent.

- `voice_minutes`: Number of voice call minutes used.
- `data_megabytes`: Number of megabytes of data consumed.

6.2.1 Circuit Structure and Constraints

The zero-knowledge circuit is an arithmetic circuit, operating over a finite field. The circuit algorithm, depicted in Algorithm 1, enforces the following constraints:

1. The VMNO's calculated total amount matches the billing computation defined by the smart contract rates.
2. The VMNO's usage data matches the data usage committed by the subscriber's UE through the hash commitment.

The actual usage values remain secret; only the fact that their correctness and consistency are revealed to the verifier. The following pseudo-algorithm represents the circuit's logic:

Algorithm 1 Circuit Algorithm

Require: n_sms, n_mb, n_min –usage metrics
Require: r_sms, r_mb, r_voice –billing rates from smart contract
Require: h_cdr –CDR hash committed by UE
Require: $total$ –Total amount to charge
1: total_check $\leftarrow (n\_sms \times r\_sms) + (n\_mb \times r\_mb) + (n\_min \times r\_voice)$
2: h_check $\leftarrow$ Poseidon(n_sms, n_mb, n_min)
3: **assert** h_check == h_cdr
4: **assert** total_check == total
5: proof $\leftarrow$ zkSNARK.Prove(circuit, private_inputs = $\{n\_sms, n\_mb, n\_min\}$, public_inputs = $\{total, h\_cdr\}$)
6: return proof

By enforcing these conditions, the system ensures both accuracy in billing and privacy of user data, as the zkSNARK proof reveals nothing beyond the correctness of the computation.

6.3 Selection of Zero-Knowledge Proof Construction

The effectiveness of privacy-preserving, verifiable roaming settlement in B5GRoam depends crucially on the underlying ZKP system. Numerous ZK constructions have been developed, each with its own trade-offs in terms of prover efficiency, proof size, verification cost, and suitability for on-chain deployment. Choosing the right construction is a foundational decision that impacts the usability, scalability, and cost of the entire protocol.

6.3.1 Comparison of Modern ZK Constructions

Modern zero-knowledge proof systems can be broadly categorized into two main families: **zk-SNARKs** (Succinct Non-interactive Arguments of Knowledge) and **zk-STARKs** (Scalable Transparent Arguments of Knowledge). Both families enable succinct, non-interactive proofs, but their underlying cryptographic assumptions and system properties differ, leading to distinct trade-offs for blockchain-based applications.

Zk-STARKs are a newer class of proof systems that emphasize transparency (i.e., no trusted setup required) and post-quantum security, relying only on hash functions for their security. They are highly scalable and support the generation of proofs for large computations, making them attractive for high-throughput applications and layer-2 scaling solutions. However, STARK proofs are currently much larger in size-ranging from tens to hundreds of kilobytes-and on-chain verification remains significantly more computationally expensive compared to SNARKs. As a result, deploying STARK verifiers in EVM-compatible smart contracts is currently cost-prohibitive and impractical for resource-constrained settlement systems like B5GRoam.

zk-SNARKs represent the most mature and widely adopted family of succinct zero-knowledge proofs. SNARK systems achieve extremely short proof sizes (typically a few hundred bytes) and highly efficient verification, enabling fast, low-cost proof verification within smart contracts. Their main drawback is the need for a trusted setup in most practical schemes, and, in some cases, cryptographic assumptions based on elliptic curve pairings.

Within the SNARK family, several constructions have emerged as standards for blockchain applications:

- **Groth16** (Groth 2016): A pairing-based SNARK protocol that delivers near-optimal proof size (less than 300 bytes) and extremely fast verification. Groth16 proofs can be verified on-chain using Ethereum's precompiled `bn128` pairing operations, making them ideal for EVM environments. However, it requires a circuit-specific trusted setup for each new application.
- **PLONK** (Gabizon et al. 2019): An advanced SNARK that introduces a universal and updatable trusted setup, significantly reducing the setup ceremony's complexity. PLONK also provides efficient verification and supports more flexible circuit designs, though its proofs are slightly larger than Groth16. PLONK is increasingly adopted by next-generation privacy and scaling projects.
- **Bulletproofs** (Bünz et al. 2018): A non-pairing-based zero-knowledge proof protocol that does not require a trusted setup and produces logarithmic-sized proofs. Bulletproofs are particularly efficient for range proofs and confidential transactions, but their general proofs are larger than SNARKs, and on-chain verification is less efficient, making them less suitable for EVM-based protocols.

Given the current state of the art, SNARKs, particularly Groth16 and PLONK, offer the optimal balance of proof size, verification speed, and developer tooling for on-chain applications. STARKs, while promising for future scalability and transparency, remain unsuitable for EVM-based settlement systems due to their large proof size and high verification cost. As a result, our protocol focuses on SNARK-based constructions for verifiable, cost-efficient roaming settlement.

The following table summarizes existing SNARK proofs and their characteristics:

zk-SNARK	Setup	Prove time	Verify time	Proof size
Groth16	circuit dependant	$O(n)$	$O(1)$	Constant
Plonk	Universal Setup	$O(n \log(n))$	$O(\log(n))$	$O(\log(n))$
Bulletproofs	–	$O(n)$	$O(n)$	$O(\log(n))$

Table 6.1 – Summary of existing SNARK constructions

Given these considerations, and based on the benchmark results from 8.3.3, B5GRoam adopts the **Groth16 zk-SNARK** construction for its settlement circuit.

6.4 Proof Generation and Verification Workflow

The core of B5GRoam's privacy-preserving settlement protocol is the ZKP workflow that allows the VMNO to prove, in a trustless and privacy-preserving manner, that the billed total cost for a subscriber's roaming session is accurate and consistent with the usage commitment provided by the UE. This is achieved without revealing the sensitive underlying usage metrics to any other party, including the HMNO or the blockchain itself. More specifically, the VMNO proves that he knows all the necessary inputs and intermediate values during execution that lead to the total cost he claims.

Figure 6.1 illustrates the end-to-end ZK proof workflow, which involves the following high-level steps:

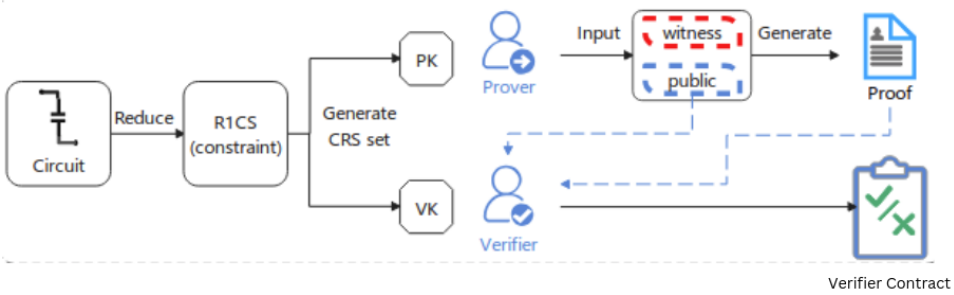


Figure 6.1 – ZK Proof Generation Flow

1. **Circuit Design:** The circuit design is carried out by formalizing the billing constraints logic as an arithmetic circuit. This circuit, as explained in Algorithm 1 encodes the two critical conditions that must be satisfied for a valid settlement: (1) the total cost must be computed correctly based on the agreed billing rates and the user's actual usage; and (2) the usage data must correspond exactly to the value committed by the UE, as enforced by a SNARK-friendly hash function such as Poseidon. All arithmetic and logical checks (including multiplications, additions, and hashing) are expressed as circuit constraints that must be satisfied by the prover's inputs.

2. **Constraint System Compilation:** Once the circuit is defined, the next step is constraint system compilation. The high-level arithmetic circuit is compiled into a constraint system, known as Rank-1 Constraint System (R1CS). In this phase, every computation within the circuit is broken down into a set of low-level constraints (quadratic equations) over a finite field. The result is a system of equations that must be satisfied for the proof to be considered valid. Circuit efficiency is a key consideration here; the use of SNARK-optimized hash functions and arithmetic reductions helps minimize the number of constraints and, consequently, the cost of proof generation and verification.
3. **Trusted Setup:** For SNARK-based protocols such as Groth16, the trusted setup phase is essential for generating the cryptographic material required by both the prover and verifier. This setup is performed only once for each specific arithmetic circuit and produces a Common Reference String (CRS) that is hardcoded into the system. The CRS consists of two parts: the Proving Key (PK), used by the prover to construct zero-knowledge proofs, and the Verifying Key (VK), deployed in the verifier smart contract to be used for proof verification. The soundness and security of Groth16 depend critically on the randomness used during this setup; the CRS must be generated with true, unpredictable entropy to ensure that no party can later create fraudulent proofs. Once generated, the setup remains valid for all subsequent proofs pertaining to the same circuit.
4. **Proof Generation (Prover Side):** With the circuit and the prover key in place, the proof generation step is performed by the VMNO, who acts as the prover. The VMNO prepares the public inputs, such as the total cost, the rates agreed upon in the contract, and the commitment hash from the UE, as well as the private witness, which consists of detailed usage data and all the intermediary values calculated during circuit execution. Using the circuit and the PK, the prover runs the Groth16 proof generation algorithm to generate a succinct zero-knowledge proof that attests, without revealing the underlying data, that the billing was computed correctly and is consistent with the UE's commitment.
5. **Proof Submission and On-Chain Verification:** After generating the proof, the VMNO proceeds to proof submission and on-chain verification. In this step, the VMNO submits only the succinct zero-knowledge proof and the claimed total cost to the blockchain by invoking the verifier smart contract. The contract, acting as the verifier, retrieves all remaining public inputs from its own storage: specifically, the agreed billing rates and the UE's commit hash, both of which were previously registered at the time of agreement and session initiation. The contract then calls the Groth16 verification algorithm, supplying the proof, the submitted total cost, and the stored rates and commitment as public inputs, together with the on-chain Verifying Key. The verifier contract checks the proof and if it succeeds, it means that (1) the proof is valid for the underlying circuit and key, (2) the total cost is consistent with the stored rates and the committed usage, and (3) the usage data used in the proof matches the UE's previously submitted commitment, without ever revealing it. If all checks pass, the proof is accepted and the protocol proceeds with settlement; otherwise, the transaction reverts and no funds are released.

This workflow enables B5GRoam to guarantee correctness and fairness in roaming settlements, while preserving user privacy and eliminating the need for trusted third parties.

6.5 ZK SNARKs Security Properties

In our B5GRoam settlement framework, the verification of roaming charges is expressed as a relation:

$$R(x, w)$$

where x denotes the public input and w denotes the private witness (e.g., detailed per-session roaming records). The relation R encodes the verification logic, ensuring that the publicly claimed total cost is consistent with the sum of all charges computed from the agreed rates. zk-SNARKs proofs have three security properties that our system relies on: *soundness*, *completeness*, and *zero-knowledge* (Groth 2016).

6.5.1 Soundness

The soundness property ensures that for any probabilistic polynomial-time adversary, the probability of producing a proof π such that the verifier accepts (x, π) while there exists no witness w satisfying $R(x, w) = 1$ is negligible:

$$\Pr\left[\text{Verify}(x, \pi) = 1 \ \wedge \ \nexists w : R(x, w) = 1\right] \leq \epsilon$$

where ϵ is a negligible function in the security parameter. In our roaming-settlement context, this guarantees that a dishonest operator cannot fabricate a proof for an incorrect total cost unless they can provide underlying session data and rate calculations consistent with the agreed pricing model encoded in R.

6.5.2 Completeness

Completeness guarantees that if the prover possesses a valid witness w such that $R(x, w) = 1$, then the verifier will accept the proof with overwhelming probability:

$$\forall (x, w) : R(x, w) = 1 \implies \Pr\left[\text{Verify}(x, \text{Prove}(x, w)) = 1\right] \approx 1$$

In our use case, this means that as long as the settlement calculation was done correctly, the proof verification will always succeed.

6.5.3 Zero-Knowledge

The zero-knowledge property ensures that the proof π reveals nothing about the witness w beyond the validity of the statement $R(x, w) = 1$. Formally, for every proof, there exists a simulator that can generate an indistinguishable proof without knowing w:

$$\{\pi \leftarrow \text{Prove}(x, w)\} \ \approx \ \{\pi' \leftarrow \text{Simulate}(x)\}$$

In the roaming settlement, this ensures that while the verifier learns the settlement is computed correctly according to the agreed rates, they gain no access to sensitive per-session data such as call durations or individual charges.

Chapter 7

Implementation and Proof of Concept

7.1 Introduction

This chapter presents the practical realization of the B5GRoam framework, translating the proposed architecture into a working prototype. We begin by outlining the frameworks, development tools, libraries, and programming languages employed in the implementation, emphasizing the rationale behind each technological choice. The chapter then delves into the construction of the SNARK proof circuit, detailing how the protocol's core business logic is encoded for zero-knowledge proof generation and verification. We also introduce the main business model underpinning the proof-of-concept, highlighting the interactions between system actors and the flow of settlement logic.

7.2 Used Frameworks and Tools

The implementation of B5GRoam leverages a modular development stack. Below, we summarize the principal frameworks and tools employed in building and evaluating the protocol:

- **Go Ethereum (geth):** An open-source Ethereum client written in Go, used to deploy, run, and interact with private or public Ethereum networks. Go Ethereum provides essential blockchain infrastructure for testing smart contracts, simulating network interactions, and validating protocol correctness under realistic conditions.
- **Foundry:** A fast, portable, and modular toolkit for Ethereum application development. Foundry supports efficient smart contract compilation, local node simulation, unit and integration testing, as well as automated deployment scripts, streamlining the entire development workflow for Solidity-based protocols.
- **Hyperledger Caliper:** A blockchain benchmarking framework that allows for the quantitative evaluation of blockchain performance. Hyperledger Caliper was used to design and execute benchmarks for key protocol operations, measuring transaction throughput, latency, and resource utilization on various network configurations.

- **Docker:** A containerization platform that enables consistent, isolated deployment of applications and services. Docker simplifies the setup of complex, multi-component environments, facilitating reproducibility and ease of deployment across different host systems.
- **Git, GitHub, and GitLab:** Distributed version control and collaborative development tools essential for managing source code, tracking changes, and supporting collaborative workflows. GitHub and GitLab additionally provide continuous integration, automated testing, and documentation hosting for the B5GRoam codebase.
- **zkSync:** A leading zk-rollup solution for scaling Ethereum with zero-knowledge proofs. zkSync enables secure, high-throughput settlement of protocol transactions while minimizing costs, and serves as the L2 execution environment for B5GRoam's.

7.3 Programming Languages

- **Solidity:** The primary language for writing smart contracts on the Ethereum platform. Solidity was used to implement the core protocol logic, including agreement management, on-chain verification, and settlement contracts. Its widespread adoption and compatibility with Ethereum's virtual machine made it the ideal choice for the protocol's on-chain components.
- **Circom:** A domain-specific language designed for creating arithmetic circuits used in zero-knowledge proof systems, particularly zk-SNARKs. Circom enables the precise and efficient definition of complex circuit logic, including hash checks and billing constraints, which are central to B5GRoam's privacy-preserving settlement flow.
- **Noir:** A modern, expressive language for developing zero-knowledge circuits, offering enhanced developer ergonomics and support for multiple proving systems. Noir was used to prototype and test alternative ZK circuit designs, contributing to both the flexibility and extensibility of the overall implementation.

7.4 Libraries

The B5GRoam implementation leverages two essential cryptographic libraries to ensure secure digital signatures and efficient zk proof generation:

- **OpenZeppelin ECDSA Library:** This widely adopted Solidity library provides robust and thoroughly audited implementations of elliptic curve digital signature algorithms. In B5GRoam, the OpenZeppelin ECDSA library is used within smart contracts to securely verify the authenticity of signatures generated by the HMNO and other authorized parties.
- **Snarkjs**: Snarkjs is a JavaScript library widely used for implementing and testing zero-knowledge proofs based on the Groth16 proving system. It provides a complete toolchain for generating trusted setup parameters, compiling arithmetic circuits into R1CS, producing and verifying proofs. In the context of B5GRoam, `snarkjs` is employed to simulate the end-to-end proof workflow: from circuit compilation to proof generation and verification.

7.5 SNARK Proofs

The implementation of the SNARK circuit in B5GRoam encapsulates both the billing logic and the integrity check of the user's committed data. At its core, the circuit accepts the CDR metrics-specifically, the number of SMS messages sent, the total bandwidth consumed (in megabytes), and the number of minutes spent on voice calls, as private inputs. These sensitive usage details remain hidden from the verifier and are never exposed publicly. In parallel, the circuit receives as public inputs the total claimed charge, the previously published Poseidon hash commitment of the CDR (generated and submitted by the UE), and the agreed-upon billing rates for each usage type (SMS, MB, and voice).

```
pragma circom 2.0.0;

include "./poseidon.circom";

template CDRCircuit() {
    // CDR details
    //  - n_sms: the number of SMS messages
    //  - n_mb: the total MB of bandwidth consumed
    //  - n_min: the total number of minutes made in a voice call
    signal input n_sms;
    signal input n_mb;
    signal input n_min;

    // Public Inputs:
    //  - T: the total charge computed
    //  - hashCDR: the hash commitment of the CDR (provided by the UE)
    //  - r_sms: charge rate per SMS
    //  - r_mb: charge rate per MB bandwidth
    //  - r_voice: charge rate per minute of foice call
    signal input T;
    signal input hashCDR;
    signal input r_sms;
    signal input r_mb;
    signal input r_voice ;


    signal smsTotal;
    signal mbTotal;
    signal voiceTotal;
    signal computedTotal;
```

Figure 7.1 – SNARK Circuit Inputs

Within the circuit, the logic proceeds by first computing the cost for each usage type: multiplying the private usage values by their corresponding public rates. The sum of these components yields the total computed charge, which is then strictly enforced to equal the public total submitted for settlement. This guarantees that the final billed amount aligns perfectly with the public rates and the private, user-specific consumption. In addition, the circuit instantiates a Poseidon hash component, using the private usage data as input, and enforces that the resulting hash matches the public commitment value. This cryptographic linkage ensures that the usage metrics applied in the proof are exactly those committed to by the user.

```
    smsTotal <== n_sms * r_sms;
    mbTotal <== n_mb * r_mb;
    voiceTotal <== n_min * r_voice;

    computedTotal <== smsTotal + mbTotal + voiceTotal;
    computedTotal === T;

    // Compute the Poseidon hash of the CDR data.
    component poseidonHash = Poseidon(3);
    poseidonHash.inputs[0] <== n_sms;
    poseidonHash.inputs[1] <== n_mb;
    poseidonHash.inputs[2] <== n_min;

    // Enforce that the computed hash equals the public hashCDR.
    poseidonHash.out === hashCDR;
}
```

Figure 7.2 – SNARK Circuit Logic

7.6 Roaming Sessions Management

The `AgreementFactory` contract manages the lifecycle of roaming agreements between mobile network operators. It maintains mappings to track deployed agreements and their metadata, including the contract address and expiry date, using a key derived from the participating operator addresses. The factory also stores references to the verifiers used for SNARK proof validation and to the payment token contract.

```
contract AgreementFactory {
    using ECDSA for bytes32;

    struct AgreementRecord {
        address contractAddress;
        uint256 expiry;
    }

    // Mapping key: keccak256(abi.encode(sortedMno1, sortedMno2))
    mapping(bytes32 => AgreementRecord) public agreements;
    mapping (bytes32 => uint) public numberOfAgreements;
    address public immutable usdc;
    address public ultraVerifier;
    address public groth16Verifier;
    address public plonkVerifier;
```

Figure 7.3 – AgreementFactory States

To initiate a new roaming agreement, the `createAgreement` function is invoked with the addresses of the two mobile network operators, the negotiated rates for SMS, data, and voice services, an expiry timestamp, and digital signatures from both parties. The contract first sorts the input addresses to ensure that agreement keys are order-independent, then constructs a unique key by hashing the sorted addresses. It checks that the operator addresses are distinct and nonzero, ensures that the expiry date is in the future, and verifies that no existing active agreement already exists for the same operator pair.

The signatures are validated to confirm mutual consent. If all validations pass, a new `Agreement` contract is deployed with the agreed rates, the expiry date, and references to the payment token and verifier contracts. The factory then records the new agreement's contract address and expiry date in its mapping, indexed by the unique agreement key.

```
    function createAgreement(
        address[2] calldata mnos,
        uint256[3] calldata rates,
        uint256 expiry,
        bytes[2] calldata signatures
    ) external returns(address){
        (address sortedMno1, address sortedMno2) = _sortAddresses(mnos[0], mnos[1]);
        bytes32 agreementKey = keccak256(abi.encode(sortedMno1, sortedMno2));

        _validateCreationParams(sortedMno1, sortedMno2, rates, expiry, agreementKey);
        _validateSignatures(sortedMno1, sortedMno2, rates, expiry, signatures, agreementKey);

        Agreement newAgreement = _deployAgreement(sortedMno1, sortedMno2, rates, expiry);
        _storeAgreement(agreementKey, address(newAgreement), expiry);

        return address(newAgreement);
    }
```

Figure 7.4 – Create Agreement Code

The `Agreement` contract acts as the central management hub for all roaming sessions established between two mobile network operators under a specific agreement. At its core, it maintains immutable parameters defining the terms of the agreement: the addresses of both operators, the agreed rates for SMS, voice, and data services, the payment token, and the expiry date. The contract tracks the full lifecycle of each roaming session using a session counter and mappings that index session records by unique identifiers. Each session record contains key details such as the subscriber's address, the home operator, estimated and actual costs, session start and end times, the commitment hash submitted by the UE, and the current session status. To ensure secure and reliable settlement, the contract also maintains nonces per user to prevent replay attacks, and holds references to multiple SNARK verifier contracts to enable on-chain validation of zero-knowledge proofs. This structure allows the Agreement contract to securely orchestrate all settlement activity between the two operators.

```
contract Agreement {
    using ECDSA for bytes32;
    using SafeERC20 for IERC20;
    UltraVerifier private ultraVerifier;
    Groth16Verifier private groth16Verifier;
    PlonkVerifier private plonkVerifier;

    enum SessionStatus { UNKOWN, PENDING, COMPLETED }

    struct Session {
        address ueAddress;
        address hmno;
        uint256 estimatedCost;
        uint256 actualCost;
        uint256 startTime;
        uint256 endTime;
        bytes32 ueCDRHash;
        SessionStatus status;
    }

    struct Rates {
        uint256 sms;
        uint256 voice;
        uint256 data;
    }

    // Immutable agreement parameters
    address public immutable mno1;
    address public immutable mno2;
    Rates public rates;
    IERC20 public paymentToken;
    uint256 public immutable expiry;

    // Session management
    uint256 private sessionCounter;
    mapping(uint256 => Session) public sessions;
    mapping(address => uint256) public nonces;
```

Figure 7.5 – Agreement Contract States

To initiate a new roaming session, the `startRoamingSession` function is called with the subscriber's address, an estimated session cost, a unique nonce, and a signature from the HMNO. The contract first verifies that the agreement has not expired, that the subscriber's address and estimated cost are valid, and reconstructs a message hash over the relevant session parameters. Using the provided signature, the contract recovers the HMNO's address and checks that it matches one of the two operators specified in the

agreement. To guard against replay attacks, it also verifies and updates the nonce associated with the HMNO. Upon successful validation, the contract transfers the estimated funds from the HMNO to itself, increments the session counter, and creates a new session record storing all pertinent session details in the contract's state. The new session is marked as `pending`, and an event is emitted for off-chain monitoring. This process ensures that only authorized operators can start sessions, that funds are locked up front, and that each session is uniquely recorded and traceable.

```
function startRoamingSession(
    address ueAddress,
    uint256 estimatedCost,
    uint256 nonce,
    bytes calldata signature
) external returns(uint){
    require(block.timestamp < expiry, "Agreement expired");
    require(ueAddress != address(0), "Invalid UE address");
    require(estimatedCost > 0, "Invalid estimated cost");

    bytes32 messageHash = keccak256(
        abi.encodePacked(
            address(this),
            ueAddress,
            estimatedCost,
            nonce
        )
    );
    // Determine HMNO address from signature
    address hmno = _recoverSigner(messageHash, signature);

    // Validate HMNO is part of this agreement
    require(hmno == mno1 || hmno == mno2, "Unauthorized HMNO");
    require(hmno != address(0), "Invalid HMNO address");

    address vmno = hmno == mno1 ? mno2 : mno1;

    // Check and update nonce
    require(nonces[hmno] == nonce, "Invalid nonce");
    nonces[hmno]++;

    // Transfer estimated funds
    _transferFunds(hmno, estimatedCost);

    // Create session record
    sessionCounter++;
    sessions[sessionCounter] = Session({
        ueAddress: ueAddress,
        hmno: hmno,
        estimatedCost: estimatedCost,
        actualCost: 0,
        startTime: block.timestamp,
        endTime: 0,
        ueCDRHash: bytes32(0),
        status: SessionStatus.PENDING
    });

    emit SessionStarted(sessionCounter, hmno, vmno, ueAddress, estimatedCost);

    return sessionCounter;
}
```

Figure 7.6 – Start Roaming Session Code

Once a roaming session has been initiated, only the UE associated with that session is authorized to submit the CDR hash using the `submitCDR` function. The contract enforces this by checking that the caller is indeed the original subscriber. Upon a valid submission, the provided CDR hash-computed inside the UE's trusted execution environment-is recorded in the session state. This committed hash serves as an immutable reference to the user's actual usage, anchoring it on-chain without exposing any sensitive details. Later, this value will be used as a public input in the zero-knowledge proof verification, ensuring that the settlement is based on the genuine, user-authorized data.

```
function submitCDR(uint sessionID, bytes32 cdrHash) external {
    Session storage session = sessions[sessionID];
    require(msg.sender == session.ueAddress, "Only UE can supply CDR hash o f his roaming session");
    require(session.ueCDRHash == bytes32(0), "CDR hash for this session is already set");
    session.ueCDRHash = cdrHash;
}
```

Figure 7.7 – Submit CDR Code

To finalize a roaming session, the VMNO calls the termination function, submitting the session ID, a Groth16 zkSNARK proof, and the computed total cost. The contract first checks that the session is pending and that the user's CDR commitment has already been submitted, rejecting any attempt to finalize an incomplete or invalid session. It verifies that the caller is the authorized VMNO for this agreement, then prepares the set of public inputs needed for proof verification-these include the total cost, the UE's committed CDR hash, and the agreed billing rates for SMS, data, and voice. The contract calls the on-chain Groth16 verifier, supplying the proof and public inputs; if the proof fails, the process halts. It then checks that the final total cost does not exceed the original estimate, safeguarding against overcharging. If all checks pass, the contract settles the payment between the HMNO and VMNO (refunding any excess funds if needed), updates the session state to completed, and emits a termination event for off-chain auditing and transparency.

```
function terminateRoamingSessionWithGroth16Verifier(
    uint256 sessionId,
    uint256[2] memory pointA_,
    uint256[2][2] memory pointB_,
    uint256[2] memory pointC_,
    uint256 totalCost
) external {
    Session storage session = sessions[sessionId];

    // Validate session state
    if(session.status != SessionStatus.PENDING) {
        revert InvalidSessionState();
    }
    if(session.ueCDRHash == bytes32(0)) {
        revert("CDR hash not commited by the UE");
    }

    // Verify caller is VMNO
    address vmno = session.hmno == mno1 ? mno2 : mno1;
    require(msg.sender == vmno, "Only VMNO can terminate");

    // Prepare public inputs for proof verification
    uint256[5] memory publicInputs;
    publicInputs[0] = totalCost;
    publicInputs[1] = uint256(session.ueCDRHash);
    publicInputs[2] = rates.sms;
    publicInputs[3] = rates.data;
    publicInputs[4] = rates.voice;

    // Verify ZK proof
    if(!groth16Verifier.verifyProof(pointA_, pointB_, pointC_, publicInputs)) {
        revert InvalidProof();
    }

    // Validate total cost consistency
    if(totalCost > session.estimatedCost) {
        revert CostMismatch();
    }

    // Calculate final settlement
    uint256 refundedAmount = _settlePayment(session.hmno, vmno, session.estimatedCost, totalCost);

    // Update session state
    session.actualCost = totalCost;
    session.endTime = block.timestamp;
    session.status = SessionStatus.COMPLETED;

    emit SessionTerminated(sessionId, vmno, totalCost, refundedAmount);
}
```

Figure 7.8 – Terminating Roaming Session Code

7.7 Conclusion

This chapter has detailed the practical realization of the B5GRoam protocol, from the selection of frameworks and programming tools to the implementation of its core smart contracts and zero-knowledge proof circuits. We demonstrated how the business logic and cryptographic assurances envisioned in the protocol's design have been translated into a working proof-of-concept.

Having established a functioning prototype, the next chapter will focus on a rigorous evaluation of the system. We will conduct comprehensive tests and performance analyses to assess the scalability, efficiency, and security of the proposed architecture under various real-world scenarios.

Chapter 8

Tests and Results

8.1 Introduction

In this chapter, we present the results of our experiments and tests conducted on the B5GRoam implementation. Our evaluation focuses on two main aspects: the performance of zero-knowledge proofs, both off-chain proof generation and on-chain proof verification, and the scalability of our system by measuring the throughput of different interactions within the blockchain, including agreement creation, session management, and settlement.

8.2 ZK Snarks Constructions Benchmark

To evaluate the performance characteristics of our protocol and choose the optimal construction, we benchmark three widely used SNARK constructions: `Groth16`, `Plonk`, and `UltraVerifier`. For each, we measure the memory consumption and the time required to generate a proof off-chain, as well as the gas cost incurred when verifying the proof on-chain. These metrics provide a comparative view of the computational and financial efficiency of each proof system, helping to inform the most practical choice for real-world deployment.

8.2.1 Proof Generation Overhead: Memory and Time Analysis

We evaluated the memory requirements and proving time for the three constructions. The results are illustrated in Fig 8.1 and Fig 8.2.

Groth16 and Plonk both exhibit high memory consumption during proof generation, with Plonk requiring slightly more memory than Groth16 (over 300,000 KB versus approximately 275,000 KB, respectively). In contrast, UltraHonk demonstrates a significant reduction in memory usage, consuming less than one-third of the resources required by the other two systems. This makes UltraHonk particularly attractive for deployment in resource-constrained environments.

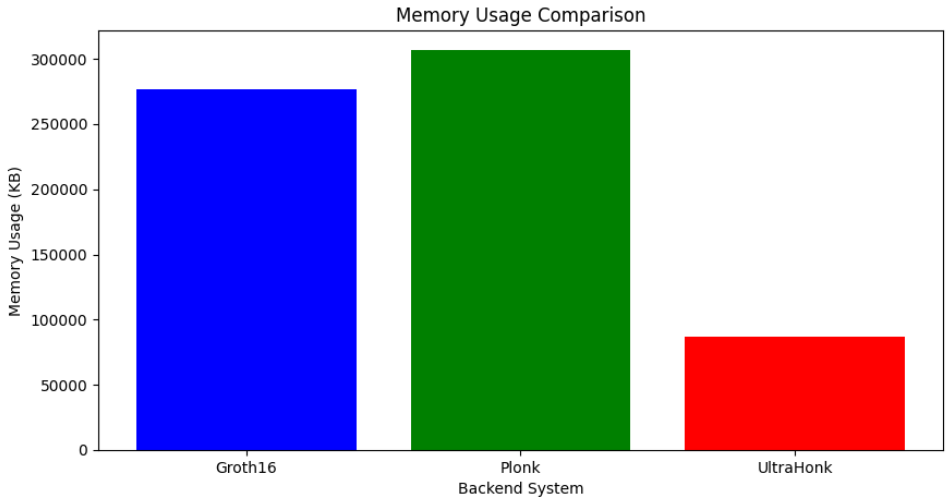


Figure 8.1 – ZK Proof Memory Consumption for each construction

The results for proving time follow a similar trend. Groth16 demonstrates moderate proof generation speed, averaging 0.4 seconds per proof, while Plonk proves to be the slowest, with an average time of around 0.75 seconds. UltraHonk again stands out, achieving the fastest proving time at just 0.16 seconds, more than twice as fast as Groth16 and nearly five times faster than Plonk.

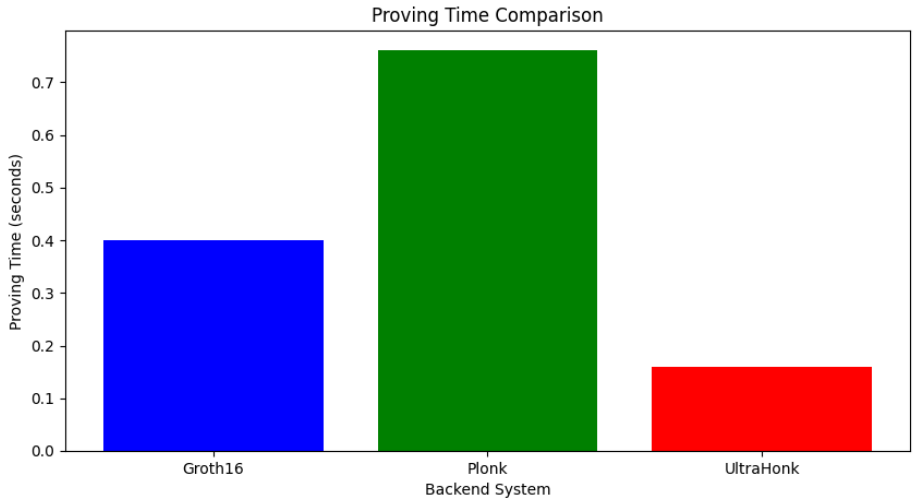


Figure 8.2 – ZK Proof time for each construction

These results indicate that UltraHonk offers substantial improvements in both memory efficiency and proof generation speed, making it an ideal choice for the off-chain proving side. However, since proofs are ultimately verified on-chain, it is essential to consider the gas consumption of each construction before making a final selection. The next subsection will analyze the on-chain verification costs associated with each SNARK system.

8.2.2 On-Chain Proofs Verification Overhead: Gas Consumption Analysis

In addition to off-chain proof generation performance, the cost of on-chain proof verification is a critical factor in selecting a SNARK construction for real-world blockchain deployments. Figure 8.3 presents the measured gas consumption for verifying a single proof with `Groth16`, `Plonk`, and `UltraHonk` verifiers.

Groth16 stands out as the most gas-efficient, consuming around 225,000 gas per verification. Plonk requires noticeably more gas, with a typical cost of approximately 270,000 gas per proof. UltraHonk, while efficient in proof generation, exhibits the highest on-chain overhead, consuming nearly 380,000 gas for verification. This difference is especially significant in public blockchains, where transaction fees are directly tied to gas usage and can impact both scalability and user experience.

These results highlight an important trade-off: while UltraHonk is ideal for environments where off-chain resource constraints and proving latency are the primary concerns, Groth16 remains the optimal choice for Ethereum-based deployments where on-chain verification cost is paramount. Therefore, when selecting a SNARK construction for B5GRoam or similar protocols, it is crucial to balance both off-chain and on-chain performance metrics. As a result, Groth16 is the best option for B5GRoam.

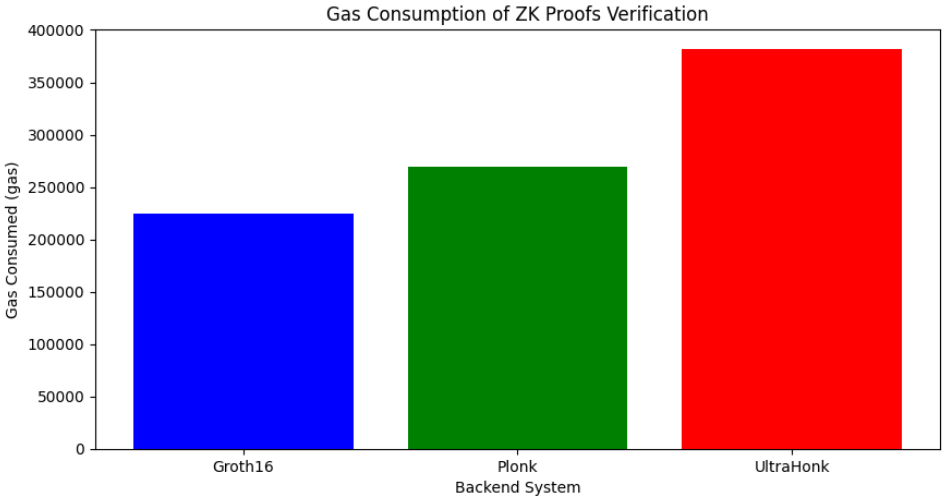


Figure 8.3 – ZK Proof Verification's Gas Consumption

8.3 B5GRoam Benchmark

8.3.1 Benchmark Environment

All benchmarks for the B5GRoam protocol were performed on a Dell XPS 15 9530 laptop equipped with a 12th-generation Intel Core i7 processor (14 cores) and 16GB of RAM. The system operates under Manjaro Arc, an Arch-based Linux distribution.

For blockchain emulation, we deployed a local Ethereum network using Go Ethereum (Geth) version 1.14.5. The configuration, as summarized in table 8.1, consists of four

nodes, each configured to participate in a Proof-of-Authority (PoA) consensus algorithm. Specifically, we used the Clique variant of PoA, with a block period of 5 seconds and an epoch length of 30,000 blocks. The network gas limit was set to 0x800,000,000 to accommodate the computational demands of zero-knowledge proof verification and other protocol-intensive transactions.

Consensus	PoA
Number of nodes	4
GasLimit	0x80000000
Period	5
Epochlength	30000

Table 8.1 – Blockchain Benchmark Environment

8.3.2 Benchmark Configuration

To systematically evaluate the performance of B5GRoam, we adopted a modular benchmarking approach using Hyperledger Caliper. The system was decomposed into three main modules: *creating roaming agreement*, *starting roaming session*, and *terminating roaming session*. By isolating each core flow, we were able to benchmark their performance individually, providing granular insights into transaction latency, throughput, and bottlenecks at each stage.

This modular methodology enables us to identify and optimize performance-critical segments of the protocol without interference from unrelated components. It also facilitates a more scalable approach to benchmarking, where improvements or changes in one module can be tested and tuned independently, ultimately resulting in a more robust and efficient overall system.

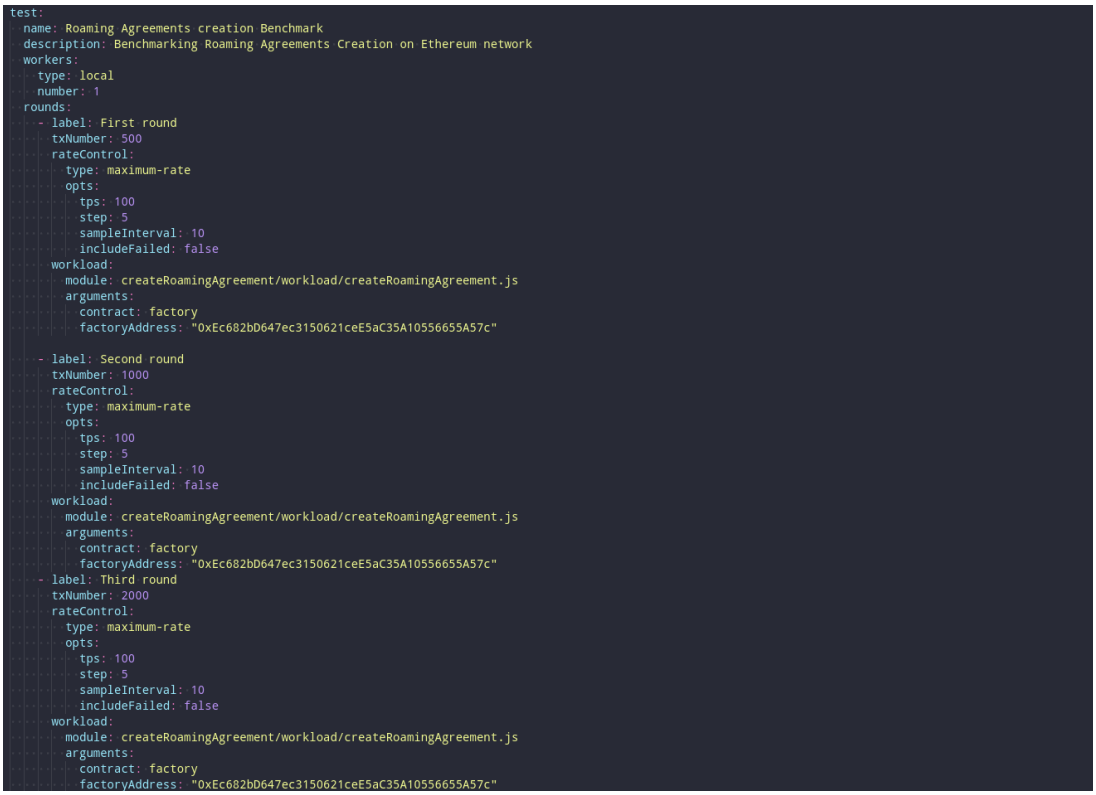
8.3.2.1 Caliper Network Configuration

Figure 8.4 shows an example of the network configuration. This configuration is crucial for running performance benchmarks with Hyperledger Caliper on an Ethereum-based network. It defines how Caliper connects to the blockchain, specifies the test network parameters, and maps smart contract deployments for use in test scenarios. The file includes the WebSocket endpoint for a locally running Ethereum node, the chain ID, and account credentials for submitting transactions. It also registers the deployed contract addresses, along with their ABI definitions, so Caliper can automatically interact with and benchmark these contracts.

Figure 8.4 – Caliper Network Configuration

8.3.2.2 Caliper Benchmark Configuration

Figure 8.5 shows an example of a benchmark configuration. This benchmark configuration file defines the testing workload for evaluating the module's performance in the B5GRoam system using Hyperledger Caliper. It specifies a local worker setup and orchestrates the benchmark into multiple rounds, each with an increasing number of transactions. For each round, the benchmark uses a maximum-rate controller to generate transactions at up to 100 TPS, collecting samples every 10 seconds. The workload section points to a custom JavaScript module that drives the necessary logic for running the benchmark.



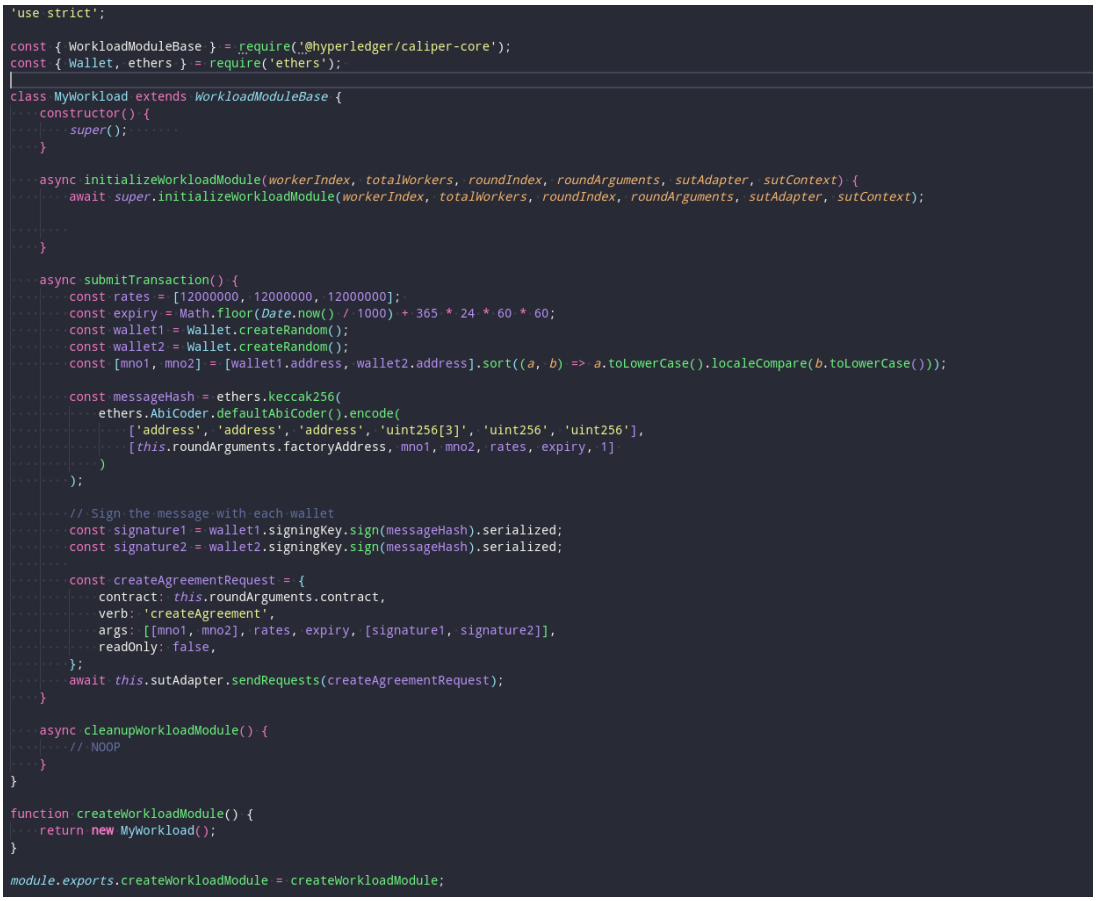
```yaml
test:
  name: Roaming Agreements creation Benchmark
  description: Benchmarking Roaming Agreements Creation on Ethereum network
  workers:
    type: local
    number: 1
  rounds:
    - label: First round
      txNumber: 500
      rateControl:
        type: maximum-rate
        opts:
          tps: 100
          step: 5
          sampleInterval: 10
          includeFailed: false
      workload:
        module: createRoamingAgreement/workload/createRoamingAgreement.js
        arguments:
          contract: factory
          factoryAddress: "0xEc682bD647ec3150621ceE5aC35A10556655A57c"

    - label: Second round
      txNumber: 1000
      rateControl:
        type: maximum-rate
        opts:
          tps: 100
          step: 5
          sampleInterval: 10
          includeFailed: false
      workload:
        module: createRoamingAgreement/workload/createRoamingAgreement.js
        arguments:
          contract: factory
          factoryAddress: "0xEc682bD647ec3150621ceE5aC35A10556655A57c"
    - label: Third round
      txNumber: 2000
      rateControl:
        type: maximum-rate
        opts:
          tps: 100
          step: 5
          sampleInterval: 10
          includeFailed: false
      workload:
        module: createRoamingAgreement/workload/createRoamingAgreement.js
        arguments:
          contract: factory
          factoryAddress: "0xEc682bD647ec3150621ceE5aC35A10556655A57c"
```

Figure 8.5 – Caliper Benchmark Configuration

8.3.2.3 Caliper Workload Configuration

Figure 8.6 illustrates an example of a workload file in Hyperledger Caliper. Workload modules are essential components of Caliper benchmarks, as they handle the construction and submission of transactions (TXs) to the blockchain. These modules encapsulate the core logic related to protocol operations, benchmarking strategies, or user behaviors, effectively acting as the "brain" of the simulated System Under Test (SUT) client. They dynamically determine which type of transaction should be issued at any point in time. Implemented as Node.js modules, workload modules expose a standardized API but offer significant flexibility in how the testing logic is defined. This allows integrating custom business rules or auxiliary functionality as needed.



```
'use strict';

const { WorkloadModuleBase } = require('@hyperledger/caliper-core');
const { Wallet, ethers } = require('ethers');

class MyWorkload extends WorkloadModuleBase {
    constructor() {
        super();
    }

    async initializeWorkloadModule(workerIndex, totalWorkers, roundIndex, roundArguments, sutAdapter, sutContext) {
        await super.initializeWorkloadModule(workerIndex, totalWorkers, roundIndex, roundArguments, sutAdapter, sutContext);


    }

    async submitTransaction() {
        const rates = [12000000, 12000000, 12000000];
        const expiry = Math.floor(Date.now() / 1000) + 365 * 24 * 60 * 60;
        const wallet1 = Wallet.createRandom();
        const wallet2 = Wallet.createRandom();
        const [mno1, mno2] = [wallet1.address, wallet2.address].sort((a, b) => a.toLowerCase().localeCompare(b.toLowerCase()));

        const messageHash = ethers.keccak256(
            ethers.AbiCoder.defaultAbiCoder().encode(
                ['address', 'address', 'address', 'uint256[3]', 'uint256', 'uint256'],
                [this.roundArguments.factoryAddress, mno1, mno2, rates, expiry, 1]
            )
        );

        // Sign the message with each wallet
        const signature1 = wallet1.signingKey.sign(messageHash).serialized;
        const signature2 = wallet2.signingKey.sign(messageHash).serialized;

        const createAgreementRequest = {
            contract: this.roundArguments.contract,
            verb: 'createAgreement',
            args: [[mno1, mno2], rates, expiry, [signature1, signature2]],
            readOnly: false,
        };
        await this.sutAdapter.sendRequests(createAgreementRequest);
    }

    async cleanupWorkloadModule() {
        // NOOP
    }
}

function createWorkloadModule() {
    return new MyWorkload();
}

module.exports.createWorkloadModule = createWorkloadModule;
```

Figure 8.6 – Caliper Workload Configuration

8.3.3 Results and Discussions

In this section, we provide and discuss the results of benchmarking the system's performance and scalability. Our evaluation is based on two key performance metrics:

- **Latency**: Latency measures the time elapsed from when a transaction is submitted to the blockchain until it is confirmed in a block, reflecting the responsiveness of the system.
- **Throughput**: Transactions per Second (TPS) represents the number of transactions successfully processed by the blockchain per second, indicating its scalability

The initial phase of our testing focuses on assessing the performance of a traditional Layer 1 blockchain to identify potential scalability bottlenecks. We benchmark three core operational flows of the B5GRoam system: creating a roaming agreement, which involves establishing a contractual relationship between two MNOs; starting a roaming session, where an authorized home MNO initiates a roaming service for a specific user; and terminating a roaming session, which finalizes the session by submitting the zk proof and settling the associated costs. The latter flow is benchmarked alone since there are multiple ZK Constructions to choose from. This phase allows us to understand the baseline limitations of an L1 setup before exploring optimizations.

Figure 8.7 illustrates the maximum latency observed for the two flows- `createRoamingAgreement` and `startRoamingSession`- across different transaction volumes, ranging from 500 to 5000 transactions.

For both operations, maximum latency remains relatively stable for lower transaction counts, fluctuating only slightly between 6.13s and 6.19s up to around 4000 transactions. This suggests that, under moderate loads, the underlying Layer 1 blockchain processes transactions consistently without significant queuing delays. However, an important difference emerges at the higher transaction range: while `startRoamingSession` maintains latency close to its baseline even at 5000 transactions (6.17s), `createRoamingAgreement` experiences a noticeable spike to 6.33s.

This increase in latency for `createRoamingAgreement` at higher loads likely reflects the additional computational and storage complexity involved in agreement creation compared to session initiation. Agreement creation requires persisting more contract state data, deploying a new Agreement contract, and performing additional validation checks, which can accumulate processing delays when the network is saturated.

Overall, these results indicate that while both flows handle moderate traffic well, `createRoamingAgreement` is more sensitive to high-throughput scenarios. This reinforces the need for potential optimizations or scaling strategies for agreement creation when deployed in high-demand environments, as it could become a latency bottleneck under heavy usage. Furthermore, a maximum latency of around 6 seconds is not ideal, particularly for `startRoamingSession`, where such a delay directly impacts the Quality of Service (QoS) by slowing down the initiation of new roaming sessions, potentially affecting user experience in real-world deployments.

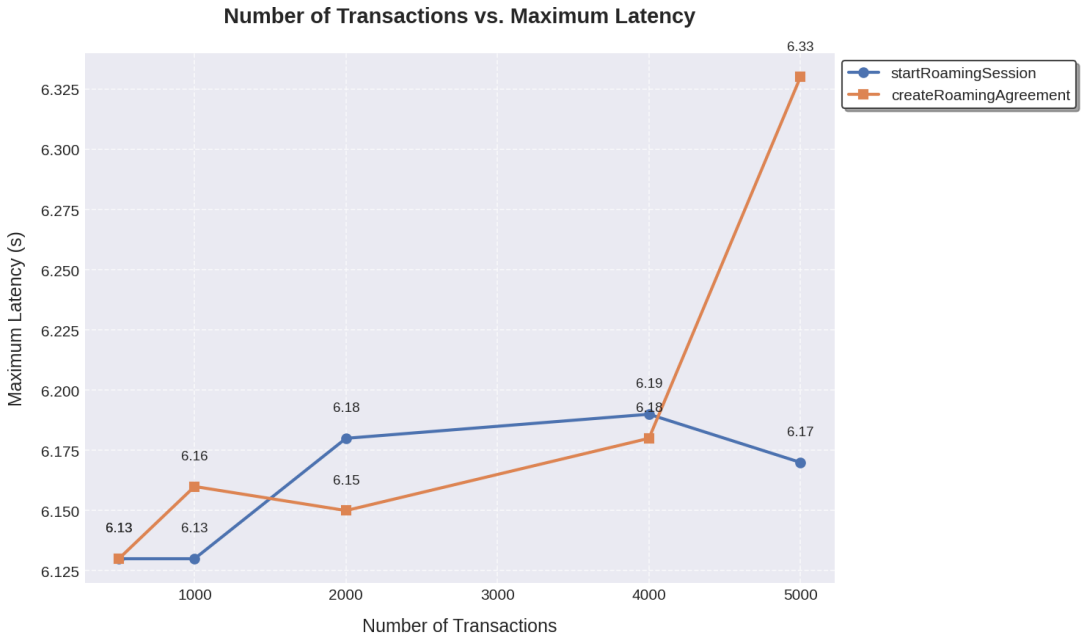


Figure 8.7 – L1 Latency under different transaction loads

The throughput comparison between the `startRoamingSession` and `createRoamingAgreement` flows, presented in figure 8.8 highlights notable differences in how each operation scales as the number of transactions increases. At lower transaction volumes (500 transactions), `startRoamingSession` achieves a significantly higher throughput of 141.9 TPS, compared to 87.4 TPS for `createRoamingAgreement`. This gap reflects the relative computational complexity and gas cost of the two flows - creating a roaming agreement involves more on-chain operations and data storage, inherently reducing the achievable TPS.

As the workload grows to 1,000 and 2,000 transactions, throughput for both flows decreases, with `startRoamingSession` dropping from 141.9 to 90.7 and then 76.8 TPS, while `createRoamingAgreement` declines from 87.4 to 69.1 and then 59.7 TPS. This drop is expected, as increased transaction volumes push the blockchain network closer to its processing limits, leading to longer block inclusion times and reduced sustained throughput.

Overall, these results indicate that `startRoamingSession` is the more scalable of the two flows in terms of throughput, consistently outperforming `createRoamingAgreement` across all tested transaction volumes. However, the overall declining trend in TPS as transaction count increases reaffirms the scalability constraints of a Layer 1 Ethereum setup. In real-world deployments, this throughput limitation could pose challenges in high-traffic environments, especially if multiple roaming agreements need to be created during peak operational periods.

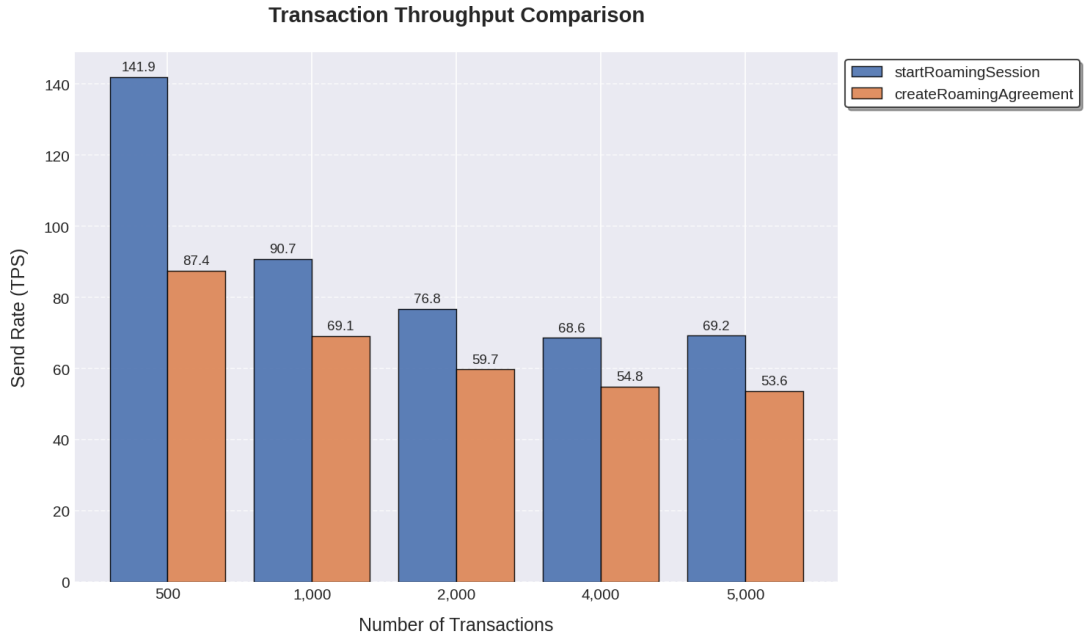


Figure 8.8 – L1 TPS under different transaction loads

The figure 8.9 compares the settlement flow when proofs are verified on-chain with two constructions: Groth16 exhibits textbook verification scalability. Across all tested loads, from 500 to 5,000 settlement transactions, latency remains essentially constant at 0.11 seconds, consistent with the theoretical O(1) complexity of its verification procedure. This is enabled by the EVM's efficient BN254 pairing precompiles, which execute the fixed set of elliptic curve operations required by Groth16 proofs in constant time. Throughput remains stable, between 98-100 TPS. UltraVerifier, by contrast, suffers under load. While initial throughput at 500 transactions is 56-58 TPS, it plateaus across all test cases. Latency, meanwhile, grows almost linearly-from 7 seconds at 500 transactions to 56 seconds at 5,000, indicating that each proof verification consumes a significant portion of the block's gas. As the transaction queue grows, confirmation times inflate, leading to poor performance under high concurrency.

Operationally, this tells us two things. First, for B5GRoam's on-chain settlement step, Groth16 is the right choice: it keeps verification latency tiny and predictable, which

prevents settlement from becoming a throughput or QoS bottleneck. Second, while UltraVerifier is attractive off-chain because of very fast proving, its current EVM verification path is not suitable when verifications must happen on-chain.

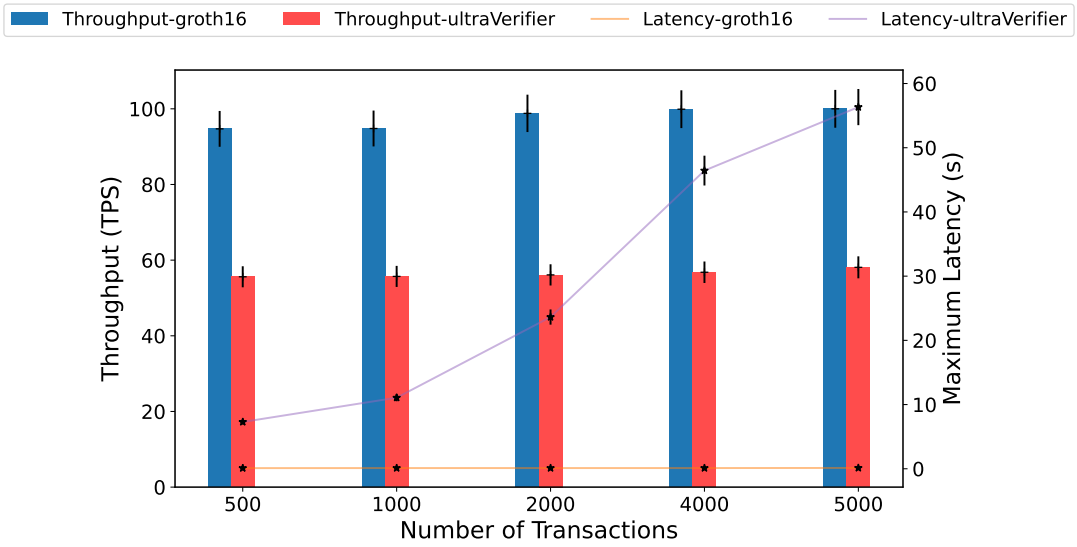


Figure 8.9 – L1 Latency and Throughput under different transaction loads for on-chain settlement

We omitted PLONK from the end-to-end settlement benchmark for one practical reason: PLONK sits between the two extremes we show here-proofs are larger and verification is costlier than Groth16, but far cheaper than UltraVerifier- so it doesn't change the design conclusion that Groth16 is preferable for on-chain verification.

The previous benchmark results reveal critical insight that the underlying Layer 1 (L1) EVM blockchain may suffice for small to medium-sized roaming workloads, where Groth16's low-latency verification ensures smooth operation. However, under B5GRoam's heavy traffic conditions, the L1 behaves slowly and is costly, resulting in severe network congestion and prohibitively high transaction fees. This underscores the necessity of adopting scalability enhancements as Layer 2 rollups or dedicated settlement chains, to meet the demands of large-scale deployments. To overcome these limitations, we opted for Layer 2 zk-Rollups, which are designed to handle a significantly higher volume of transactions, provide near-instant confirmations, and greatly reduce gas consumption, while maintaining security comparable to that of L1.

To evaluate the performance benefits of Layer 2 (L2) solutions compared to traditional Layer 1 (L1) execution, we conducted a benchmarking experiment focused on gas consumption of verifying Groth16 proofs on-chain. We used a ZK rollup solution, specifically zkSync [1], and set up a local zkSync environment [2] that included both L2 and L1 nodes. Table 8.2 presents the results for different transaction volumes. The results demonstrate a clear and consistent advantage in using L2. For example, processing 60 transactions on L1 consumes over 15 million gas units, while the same workload on L2 costs only around 388,749 units, representing a reduction of over 96%. The Layer-2 execution pipeline, comprising $commitBlocks$(batch formation), $proofBlocks$ (ZKP generation),

1. https://www.zksync.io/
2. https://github.com/matter-labs/local-setup

and *executeBlocks* (on-chain finalization), adds only modest overhead while delivering a significant cost reduction versus Layer 1. Even when the workload scales to 500 transactions, the total gas consumed on L2 remains far below the L1 baseline. This batching also magnifies throughput: coupling a 60-tx batch size with an underlying Layer 1 capacity of 120 tx/s yields an effective rate of 7,200 tx/s. Collectively, these findings confirm that zk-rollups not only cut fees but also unlock the transaction volume required for real-world deployment of B5GRoam.

Table 8.2 – Benchmark results comparing Dual Layer and Single Layer transaction costs. Batch size represents the number of L2 TXs within each batch.

Total TXs	**Dual Layer 2**						**Single Layer (L1)**
	Batches	**Batch size**	**Commit**	**Prove**	**Execute**	**Total**	
60	1	60	205,907	83,676	99,166	388,749	15,636,180
100	2	50	411,802	167,328	182,794	761,924	26,060,300
200	3	67	617,625	250,980	297,486	1,166,091	52,120,600
500	7	72	1,458,233	585,588	694,162	2,737,983	130,301,500

8.4 Conclusion

This chapter has systematically evaluated the performance of the proposed roaming framework, providing a comprehensive view of the system's scalability and efficiency. The initial L1 benchmarking revealed that while a traditional EVM-based blockchain can process small to medium roaming workloads with acceptable throughput and latency, it struggles significantly under heavy traffic. Throughput degradation under high load further underscored the risk of network congestion and elevated transaction fees, confirming that a purely L1-based approach would be insufficient for large-scale deployments of B5GRoam.

The L2 zk-rollup evaluation highlighted the transformative impact of batching and off-chain proof generation. Using zkSync in a dual-layer setup achieved over 96% gas cost reduction compared to L1 while sustaining throughput magnitudes higher than the baseline, enabling the system to handle thousands of transactions per second effectively. These findings confirm that an L1-L2 hybrid architecture not only preserves security but also ensures cost-efficiency and responsiveness under demanding operational conditions.

Subsequent analysis of cryptographic verification performance demonstrated the efficiency of Groth16 proofs, achieving near-constant latency and high throughput compared to other ZK Snarks constructions.

In summary, the benchmarking results confirm the efficiency and scalability of the proposed B5GRoam architecture. The evaluation demonstrated the practicality of in-

tegrating zero-knowledge proofs for secure and transparent roaming settlement, while highlighting the performance gains achieved through Layer 2 zk-rollups. These outcomes reflect a robust and forward-looking solution, capable of addressing the limitations of traditional blockchain-based settlement and paving the way for high-performance, trustless roaming in next-generation networks.

General Conclusion

The deployment of 5G networks represents a decisive step in the evolution of mobile communication, enabling ultra-low latency, enhanced capacity, and the support of a wide range of new applications. However, alongside these transformative opportunities, roaming remains a critical challenge. The sheer scale of anticipated roaming traffic, combined with the operational, financial, and security demands of inter-operator collaboration, makes traditional roaming frameworks increasingly inadequate. Delayed reconciliation, reliance on centralized intermediaries, and unresolved issues of fairness and privacy highlight the limitations of existing approaches.

This thesis has addressed these issues by investigating blockchain as an enabling technology for roaming settlement. Blockchain offers transparency, auditability, and trust in inherently adversarial or semi-trusted environments. Yet, as our analysis revealed, integrating blockchain into roaming systems introduces new challenges, particularly concerning scalability and the protection of sensitive user data in a transparent ledger environment.

In response to these complexities, we proposed and implemented B5GRoam, a blockchain-based and zero-trust-oriented framework designed to enable secure, transparent, and privacy-preserving roaming settlement for 5G networks. The framework formalizes roaming agreements as on-chain contracts, enforces billing consistency through cryptographic proofs, and eliminates the need for centralized clearinghouses. By combining cryptographic verification with decentralized automation, B5GRoam strengthens trust between operators while reducing operational overhead.

The proof-of-concept implementation and evaluation demonstrated that such a framework is not only feasible but also scalable under realistic workloads. The results confirmed that blockchain can serve as a practical foundation for inter-operator settlement in 5G.

While this work offers concrete progress, several open challenges remain. Future research should investigate further optimization of cryptographic protocols to reduce proof generation costs and batching proof techniques that enable proving multiple roaming sessions in a single proof. Addressing these areas will be crucial to ensuring that blockchain-based roaming settlement becomes a practical and widely adopted solution in real-world 5G deployments.

In conclusion, this thesis contributes to bridging the gap between the opportunities of 5G roaming and the limitations of traditional settlement systems. By leveraging decentralization and formalizing trust relationships through zero-knowledge cryptography, it lays the groundwork for a new generation of secure, transparent, and privacy-preserving roaming frameworks.

Bibliography

Ali, Omar, Ashraf Jaradat, Atik Kulakli, and Ahmed Abuhalimeh (Jan. 2021). "A Comparative Study: Blockchain Technology Utilization Benefits, Challenges and Functionalities". In: *IEEE Access* PP, pp. 1–1.

Association, GSM (2024). *5GS Roaming Guidelines Version 9.0*. Tech. rep. Version 9.0, February 2024. Next Generation Mobile Networks Alliance.

Belchior, Rafael, André Vasconcelos, Sérgio Guerreiro, and Miguel Correia (2021). "A survey on blockchain interoperability: Past, present, and future trends". In: *Acm Computing Surveys (CSUR)* 54.8, pp. 1–41.

Blanco, Bego, Jose Oscar Fajardo, Ioannis Giannoulakis, Emmanouil Kafetzakis, Shuping Peng, Jordi Pérez-Romero, Irena Trajkovska, Pouria S. Khodashenas, Leonardo Goratti, Michele Paolino, Evangelos Sfakianakis, Fidel Liberal, and George Xilouris (2017). "Technology pillars in the architecture of future 5G mobile networks: NFV, MEC and SDN". In: *Computer Standards & Interfaces* 54.4, pp. 216–228.

Bünz, Benedikt, Jonathan Bootle, Dan Boneh, Andrew Poelstra, Pieter Wuille, and Gregory Maxwell (2018). "Bulletproofs: Short Proofs for Confidential Transactions and More". In: *2018 IEEE Symposium on Security and Privacy (SP)*. IEEE, pp. 315–334.

Chaudhry, Natalia and Muhammad Murtaza Yousaf (2018). "Consensus algorithms in blockchain: Comparative analysis, challenges and opportunities". In: *2018 12th international conference on open source systems and technologies (ICOSST)*. IEEE, pp. 54–63.

Dangi, Ramraj et al. (2021). "Study and Investigation on 5G Technology: A Systematic Review". In: *Sensors (Basel, Switzerland)* 22.1, p. 26.

ETTelecom (Mar. 2023). "5G roaming connections to grow 900% to 526 million by 2027: Report". In: *ETTelecom (The Economic Times)*.

Fiat, Amos and Adi Shamir (1986). "How to prove yourself: Practical solutions to identification and signature problems". In: *Conference on the theory and application of cryptographic techniques*. Springer, pp. 186–194.

Göbel, J. and A. E. Krzesinski (2017). "Increased block size and Bitcoin blockchain dynamics". In: *2017 27th International Telecommunication Networks and Applications Conference (ITNAC)*, pp. 1–6.

Goldwasser, Shafi, Silvio Micali, and Chales Rackoff (2019). "The knowledge complexity of interactive proof-systems". In: *Providing sound foundations for cryptography: On the work of shafi goldwasser and silvio micali*, pp. 203–225.

Gong, Yi, Boyuan Yu, Lei Yang, Fanke Meng, Lei Liu, Xinjue Hu, and Zhan Xu (2025). "Toward next-generation networks: A blockchain-based approach for core network architecture and roaming identity verification". In: *Digital Communications and Networks* 11.2, pp. 326–336.

Groth, Jens (2016). "On the Size of Pairing-Based Non-Interactive Arguments". In: *Advances in Cryptology – EUROCRYPT 2016*. Springer, pp. 305–326.

GSMA (2020). *5GS Roaming Guidelines Version 2.0*. Tech. rep. Version 2.0, February 2020. Next Generation Mobile Networks Alliance.

Huawei Technologies Co., Ltd. (2021). *LTE International Roaming Whitepaper*. Tech. rep. Huawei Technologies Co., Ltd.

Inc., Syniverse Technologies (Jan. 2002). *International Roaming Guide 2.c*. Tech. rep. Syniverse Technologies Inc.

Iqbal, Mubashar and Raimundas Matulevičius (2021). "Exploring Sybil and Double-Spending Risks in Blockchain Systems". In: *IEEE Access* 9, pp. 76153–76177.

Johnson, Don, Alfred Menezes, and Scott Vanstone (2001). "The elliptic curve digital signature algorithm (ECDSA)". In: *International journal of information security* 1.1, pp. 36–63.

Keller, Ralf, David Castellanos, Anki Sander, Amarisa Robison, and Afshin Abtin (2021). "Roaming in the 5G System: The 5GS Roaming Architecture". In: *Ericsson Technology Review* 2021.6, pp. 2–11.

Mafakheri, Babak, Andreas Heider-Aviet, Roberto Riggio, and Leonardo Goratti (2021). "Smart contracts in the 5G roaming architecture: the fusion of blockchain with 5G networks". In: *IEEE Communications Magazine* 59.3, pp. 77–83.

Nguyen, Cong T, Diep N Nguyen, Dinh Thai Hoang, Hoang-Anh Pham, Nguyen Huynh Tuong, Yong Xiao, and Eryk Dutkiewicz (2021). "Blockroam: Blockchain-based roaming management system for future mobile networks". In: *IEEE Transactions on Mobile Computing* 21.11, pp. 3880–3894.

Ordonez-Lucena, Jose, Pablo Ameigeiras, Diego Lopez, Juan J. Ramos-Munoz, Javier Lorca, and Jesus Folgueira (2017). "Network Slicing for 5G with SDN/NFV: Concepts, Architectures, and Challenges". In: *IEEE Communications Magazine* 55.5, pp. 80–87.

Refaey, Ahmed, Karim Hammad, Sebastian Magierowski, and Ekram Hossain (2019). "A blockchain policy and charging control framework for roaming in cellular networks". In: *IEEE Network* 34.3, pp. 170–177.

Rosa-Bilbao, Jesús and Juan Boubeta-Puig (2023). "Ethereum blockchain platform". In: *Distributed Computing to Blockchain*. Elsevier, pp. 267–282.

Sharma, Sudhir, M Deivakani, K Srinivasa Reddy, AK Gnanasekar, and G Aparna (2021). "Key enabling technologies of 5G wireless mobile communication". In: *Journal of Physics: Conference Series*. Vol. 1817. 1. IOP Publishing, p. 012003.

Vora, Lopa J (2015). "Evolution of mobile generation technology: 1G to 5G and review of upcoming wireless technology 5G". In: *International Journal of modern trends in Engineering and research* 2.10, pp. 281–290.

Weerasinghe, Nisita, Tharaka Hewa, Maheshi Dissanayake, Mika Ylianttila, and Madhusanka Liyanage (2021). "Blockchain-based roaming and offload service platform for local 5G operators". In: *2021 IEEE 18th Annual Consumer Communications & Networking Conference (CCNC)*. IEEE, pp. 1–6.

Xiao, Yang, Ning Zhang, Wenjing Lou, and Y. Thomas Hou (2020). "A Survey of Distributed Consensus Protocols for Blockchain Networks". In: *IEEE Communications Surveys Tutorials* 22.2, pp. 1432–1465.

Xu, Jie, Cong Wang, and Xiaohua Jia (Jan. 2023). "A Survey of Blockchain Consensus Protocols". In: *ACM Computing Surveys* 55.

Zhai, Sheping, Yuanyuan Yang, Jing Li, Cheng Qiu, and Jiangming Zhao (Feb. 2019). "Research on the Application of Cryptography on the Blockchain". In: *Journal of Physics: Conference Series* 1168, p. 032077.

Zheng, Zibin, Shaoan Xie, Hong-Ning Dai, Weili Chen, Xiangping Chen, Jian Weng, and Muhammad Imran (2020). "An overview on smart contracts: Challenges, advances and platforms". In: *Future Generation Computer Systems* 105, pp. 475–491.

Zheng, Zibin, Shaoan Xie, Hong-Ning Dai, Xiangping Chen, and Huaimin Wang (2018). "Blockchain challenges and opportunities: A survey". In: *International journal of web and grid services* 14.4, pp. 352–375.

Webography

Arm Ltd. (2020). *ARM TrustZone Technology*. Online Documentation. URL: https://developer.arm.com/documentation/100690/0200/ARM-TrustZone-technology.

Gabizon, Ariel, Zachary J. Williamson, and Oana Ciobotaru (2019). *PLONK: Permutations over Lagrange-bases for Oecumenical Noninteractive arguments of Knowledge*. IACR Cryptology ePrint Archive. URL: https://eprint.iacr.org/2019/953.

Hasret Ozan Sevim (2024). *A Survey on Trustless Cross-chain Interoperability Solutions in On-chain Finance*. Whitepaper. URL: https://dlt2024.di.unito.it/wp-content/uploads/2024/05/DLT2024_paper_14.pdf.

Open5Gcore (2022). *5G Architecture*. URL: https://www.open5gcore.org/open5gcore/fundamental-core-functionality.

Samsung Electronics (2025). *TEEgris Overview*. Online Documentation. URL: https://developer.samsung.com/teegris/overview.html.

Starhome Mach (2018). *Operator's Roaming Fraud Losses Can Reach EUR 40,000 Per Hour*. PR Newswire. URL: https://www.prnewswire.com/news-releases/starhome-mach-operators-roaming-fraud-losses-can-reach-40000-per-hour-598836021.html.