PGUS: Pretty Good User Security for Thick MVNOs *with* a Novel Sanitizable Blind Signature

Yang Yang*, Quan Shi*, Prosanta Gope†✉, Behzad Abdolmaleki†, Biplab Sikdar*
*National University of Singapore, {y.yang, shi.quan}@u.nus.edu, bsikdar@nus.edu.sg
†University of Sheffield, {p.gope, behzad.abdolmaleki}@sheffield.ac.uk

***Abstract*—The rise of 5G technology has highlighted the critical role of Thick Mobile Virtual Network Operators (MVNOs) in providing customized mobile services. However, security and privacy challenges specific to Thick MVNOs remain inadequately addressed. In this paper, we present PGUS (Pretty Good User Security) for Thick MVNOs. Our proposed PGUS framework introduces a new cryptographic primitive called the Sanitizable Blind Signature (SBS), along with a novel Authentication and Key Agreement protocol named PGUS-AKA. Additionally, we have developed a seamless handover protocol, PGUS-HO, which is designed to secure all communication within a Thick MVNO environment. Furthermore, we conduct a thorough formal security analysis within the Universal Composability (UC) framework to address key threats, providing a strong solution for securing next-generation mobile networks. We also provide the evaluations on a 5G testbed which demonstrate the effectiveness of PGUS.**

1. Introduction

The rapid deployment of 5G technology marks a transformative era in mobile communications, providing the backbone for a connected society that increasingly relies on high-speed, reliable internet access. By the end of 2018, more than 5 billion people were subscribed to mobile services, representing 67% of the global population [7]. This widespread adoption underscores the critical importance of 5G networks in supporting daily communication and sophisticated applications such as smart cities, advanced healthcare systems and automated industries. In this landscape, Mobile Virtual Network Operators (MVNOs) play a crucial role by leveraging existing infrastructure to provide mobile services, often at a lower cost and with differentiated offerings [43], [46]. This model enhances consumer choice and introduces competitive dynamics into the telecommunications market. The advent of 5G has further empowered MVNOs like O_2 and Google Fi, enabling them to offer enhanced services such as higher data throughput and expanded IoT connectivity [33]. The market's expansion is evidenced by its projected compound annual growth rate (CAGR) of 7.7% from 2023 to 2030, with the overall market expected to reach nearly USD 149.13 billion by 2030 [38], particularly for the thick MVNO systems like Lebara, Lycamobile, Google Fi and M1 in Singapore. This growth is driven by increasing consumer demand for cost-effective and flexible mobile services, which MVNOs are well-positioned to provide through innovative service models and strategic partnerships. Within this thriving landscape, thick MVNOs are emerging as a key trend for the future. Unlike skinny MVNO models, thick MVNOs have significant control over network infrastructure, allowing them to offer highly customized services and more competitive pricing as shown in Figure 1. This capability positions thick MVNOs to capitalize on the growing market demand, making them a crucial element in the next phase of MVNO evolution. As the market expands, thick MVNOs are expected to play a pivotal role in shaping the future of mobile services, particularly as consumers and businesses increasingly seek tailored and reliable connectivity solutions [23], [25].

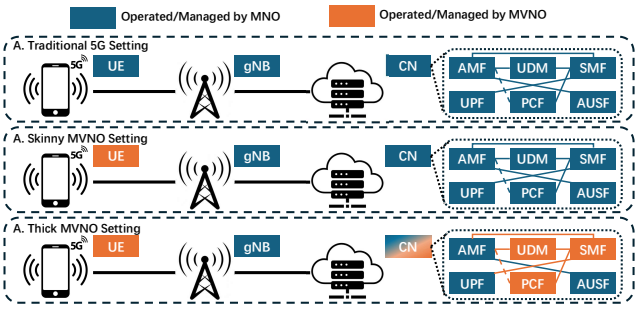

Figure 1. Roles of MNO and MVNO in Different Settings.

However, the interdependencies between MVNOs and Mobile Network Operators (MNOs) necessitate rigorous security measures to protect user data across these shared networks. To bolster these security measures, the 3GPP has introduced Technical Specification 33.501 [50], which outlines the protocols for the latest Authentication and Key Agreement (AKA) mechanisms specifically designed for 5G networks. However, the interdependencies between MVNOs and MNOs necessitate rigorous security measures to protect user data across these shared networks. In particular, the emergence of thick MVNO architectures exacerbates these challenges, as the division of control between MNOs (who manage the base stations) and MVNOs (who manage the core networks) introduces significant trust and account-

✉. Corresponding author.

ability issues. Unlike skinny MVNOs, where the entire infrastructure is controlled by the MNO, thick MVNOs operate with split ownership of critical network components, leading to potential security risks in the Authentication and Key Agreement process. For example, the gNodeB (gNB), under MNO control, might expose sensitive user location information [34], while the Core Network (CN), managed by the MVNO, could be vulnerable to attacks from Man-in-the-Middle (MitM) or Impersonation attacks. Therefore, in [34], the authors proposed an MVNO setting that considers base station anonymity with respect to CN. Furthermore, if we solve these problems and provide privacy for User Equipments (UEs) and gNB simultaneously, it will be easier for the MNO (gNB) to overbill the MVNO (CN) and overstate the number of UEs they serve, since the CN cannot know which UEs were serviced by the gNB and which gNB is claiming. This heightens the need for robust accountability mechanisms, ensuring that suspectable actions can be traced and attributed accurately. These privacy and security concerns necessitate the development of new cryptographic protocols and accountability frameworks tailored to address the unique challenges presented by thick MVNOs in 5G networks. However, our comprehensive literature review identifies a significant gap: no existing solution fully addresses the diverse security and privacy challenges inherent in 5G networks, particularly in the context of Thick MVNO settings, as outlined in §2. Current approaches fall short of providing a holistic framework that ensures both robust privacy protection and comprehensive security measures. This highlights the pressing need for a novel solution that effectively bridges the privacy and security gaps specific to Thick MVNO environments. Furthermore, any proposed solution must also incorporate strong accountability guarantees to mitigate trust issues between MNOs and MVNOs, ensuring transparency and traceability in the event of security breaches or malicious behaviour.

This intricate set of challenges, combined with the need to maintain seamless interoperability with current 5G infrastructure while respecting the resource-constrained nature of User Equipment, forms the foundation of our investigation. The emergence of Thick MVNOs introduces trust and privacy concerns that existing protocols cannot adequately address. Thus, our research is driven by the necessity to develop a solution that not only fills these critical security and privacy gaps but also ensures accountability, all while minimizing the impact on performance and computational overhead within the 5G ecosystem. In order to address all the above issues, here we introduce the PGUS (Pretty Good User Security) for Thick MVNO settings. The PGUS includes a novel cryptographic primitive termed *Sanitizable Blind Signature*, which is the underlying foundation of our proposed system. Building upon this foundation, we introduce a new AKA protocol named PGUS-AKA and a seamless handover (HO) protocol PGUS-HO, which are innovative schemes specifically designed to mitigate the security and privacy vulnerabilities present in current 5G-AKA and 5G-HO protocols. Our *key* contributions to this paper can be summarized as follows:

- We propose a *novel* sanitizable blind signature (SBS) scheme designed to protect messages with sensitive information in high-security-demand environments. This scheme allows the signer to generate a valid sanitizable signature without seeing the plaintext and introduces an innovative Trace mechanism that enhances accountability and enables double-spending checks, supporting several advanced security properties essential in sensitive applications. We provide a generic construction of SBS based on Structure-Preserving Signatures on Equivalence classes (SPSEQ) and Traceable Ring Signature (TRS). Our proposed SBS scheme not only strengthens privacy in secure communication channels but also offers broad applicability across various use cases within the domain of communication security.
- We proposed a *novel* secure privacy-preserving authenticated key-agreement scheme (PGUS-AKA) and a seamless handover (PGUS-HO) protocol which leverages the security properties of SBS. Meanwhile, our proposed PGUS also support international roaming with minor changes to the protocol. To the best of our knowledge, this is the *first* comprehensive privacy-preserving AKA and handover protocol, which has been designed by considering all the possible security and privacy issues (as discussed above) and fills the security and privacy gap under Thick MVNO settings. Especially, PGUS protocols ensure Base Station Anonymity and Trace Mechanism and prevent the MNO from overclaiming the number of UEs it serves.
- We provide a formal security modeling and analysis of our proposed PGUS protocols in the Universal Composability (UC) framework. We also give a formal proof of all the security properties of our proposed sanitizable blind signature (SBS) scheme. The proofs show that SBS achieves all necessary security properties.
- We have implemented the our proposed sanitizable blind signature (SBS) as an integral component of our system. Meanwhile, we evaluated the proposed PGUS scheme against the conventional 5G protocol using the widely adopted open-source 5G testbed, OpenAirInterface. The evaluation focuses on the end-to-end latency of the Authentication and Key Agreement (AKA) procedure, as well as the mobility performance, specifically the handover delay. Furthermore, we conducted real-world device experiments using a commercial Android smartphone to assess the practical feasibility of our approach. We make all source code publicly available to the academic community through a GitHub repository: {https://github.com/YYangNUS/PGUS}.

1.1. Paper Organization

The rest of this paper is structured as follows. Section 2 reviews the current state-of-the-art research and outlines the motivation behind our work. In Section 3, we present the necessary background information and cryptographic foundations essential for understanding our approach. Section 4 describes our threat model and outlines the design

Notation	Description
UE	User Equipment
CN	Core Network
pid	User Pseudo-identity
gid	Base Station Pseudo-identity
$\mathsf{pk}_{\mathsf{sig}}, \mathsf{sk}_{\mathsf{sig}}$	keys for SBS Signer.
spksig, ssksig	keys generated by SPSEQ for Signer.
tpksig, tsksig	keys generated by TRS for Signer.
$\mathsf{pk}_{\mathsf{san}}, \mathsf{sk}_{\mathsf{san}}$	keys for SBS Sanitizer.
spksan, ssksan	keys generated by SPSEQ for Sanitizer.
tpksan, tsksan	keys generated by TRS for Sanitizer.
$pk_{sig}^{CN}, sk_{sig}^{CN}$	keys when CN runs SBS.
$pk_{san}^{gNB}, sk_{san}^{gNB}$	keys when gNB runs SBS.
ADM	index of the admissible modifications
MOD	modification
dt	data sent by User to Signer
st	a state stored by User

TABLE 1. NOTATION USED IN OUR PROPOSED SBS SCHEME AND PGUS PROTOCOL

goals for the system. Our proposed scheme is detailed in Section 5. Following this, Section 6 presents our UC security framework and conducts a comprehensive security analysis. In Section 7, we discuss our implementation and evaluate its performance on the testbed. Finally, Section 8 offers the conclusion. Table 1 introduces the notations used throughout the paper; the full notations and abbreviations about 5G can be found in supplementary material [57].

2. Related Works and Motivation

A comprehensive analysis of 5G authentication and handover protocols, particularly the 5G-AKA protocol specified in 3GPP TS 33.501 [50], has uncovered critical security and privacy vulnerabilities. These include traceability attacks where adversaries can track users [9], impersonation risks due to identity misbinding [17], and logical flaws compromising sequence number confidentiality [11]. The lack of authentication for initial base station broadcasts exposes the system to fake base station attacks [29], [30], while transmitting user identities in plaintext over the air enables adversaries to extract sensitive information [18]. Even in standalone networks where identities are encrypted, the decryption by MNOs raises privacy concerns, allowing user activity monitoring. Furthermore, inherent flaws in the protocol enable attackers to link and trace users' activities across sessions [9], [29] or even reveal the user's encrypted identity (SUCI) [15], [28]. To address these issues, Wang et al. proposed 5G-AKA' [54] scheme that utilizes symmetric-key-based solutions to mitigate privacy concerns and linkability attacks. Their approach achieves this by encrypting the 128-bit RAND (defined in TS 24.501 [49]) in the challenge response phase. However, their solution fails to provide comprehensive privacy protection, leaving certain vulnerabilities unaddressed. Schmitt and Raghavan [44] introduced Pretty Good Phone Privacy (PGPP), a framework designed to enhance user identity and location privacy in cellular networks by decoupling authentication and billing from network connectivity. Central to their approach is the PGPP Gateway (PGPP-GW), which anonymizes user identifiers to prevent tracking while maintaining compatibility with existing 5G infrastructure. Although PGPP effectively mitigates bulk and targeted surveillance, it does not fully address protocol-level security challenges (does not provide a concrete security solution), instead calling for substantial infrastructural changes within 5G networks to achieve comprehensive privacy protection. Yu et al. [58] introduced AAKA, an AKA protocol that enhances user privacy by allowing anonymous authentication and protecting against tracking by MNOs. The protocol uses anonymous credentials and zero-knowledge proofs to enable secure access to cellular networks without revealing users' permanent identifiers, while also allowing lawful de-anonymization when necessary. However, AAKA does not address privacy concerns during handovers, especially for MVNO users, leaving them vulnerable to linkability and fake base station attacks. Additionally, AAKA faces challenges in adapting to Thick MVNO environments, as its two-party design between the UEs and the CN does not account for the additional involvement of the gNB, which complicates its application in more complex network settings. Hussain et al. [28] proposes a PKI-based authentication mechanism that allows a cellular device to authenticate a base station. [42] introduces the Broadcast But Verify scheme, a method for verifying gNB messages. And [47] presents a new entity called Core-PKG to facilitate authentication. While these works effectively address gNB-UE authentication, none of the above solutions can ensure the BSA and BSU, which are essential for protecting users' privacy in thick MVNOs. Rabiah et al. [4] proposed a privacy-preserving and UC-secure AKA protocol for MVNO, which offers several advantages. It ensures Mutual Authentication and provides strong privacy-preserving features for MVNO users, effectively protecting them against identity exposure and linkability attacks. The protocol also leverages advanced cryptographic techniques, such as zero-knowledge proofs and sanitizable signatures, to enable secure authentication while maintaining anonymity. Furthermore, it supports seamless handover across networks, preserving user privacy and unlinkability during transitions, and has demonstrated low overhead in a 5G testbed implementation. However, their scheme must assume mutual trust between gNB and CN before the AKA protocol, which is not compatible with Thick MVNOs.

On the other hand, although some previous security protocols have utilized sanitizable signatures [8] to ensure seamless handover, where [2], [3] are based on the conventional 5G setting, and [4] has been designed for the skinny MVNO setting. However, this cryptographic primitive is not suitable for the Thick MVNO setting, as its signing algorithm and accountability mechanism require the CN to know the gNB's identity, which is considered to potentially leak user information [34]. All the protocols above have certain limitations. None of the previous protocols can be directly applied to Thick MVNO settings where additional security and privacy issues are required to be taken into account . Moreover, these protocols can not achieve key privacy properties such as Base Station Anonymity (BSA), Base Station Unlinkability (BSU), and Global Traceability (GT) (As discussed in §4), which are crucial for more com-

Schemes / Features	5G-AKA [49]	5G-AKA' [54]	PGPP [44]	AAKA [58]	UC-AKA [4]	Proposed Scheme
SMCT	5G	5G	Skinny MVNO	4G, 5G	Skinny MVNO	Thick+Skinny MVNO
UE Anonymity (UA)	YES	YES	YES	YES	YES	YES
UE Unlinkability (UEU)	NO	YES	YES	YES	YES	YES
Base Station Anonymity (BSA)	NO	NO	NO	NO	NO	YES
Base Station Unlinkability (BSU)	NO	NO	NO	NO	NO	YES
Mutual Authentication* (MA*)	NO	NO	NO	NO	NO	YES
Global Traceability (GT)	NO	NO	NO	NO	NO	YES

Footnote: **MA***: Mutual Authentication Under Thick MVNO Setting. **SMCT**: Supported Mobile Communication Type.

TABLE 2. COMPARISON WITH THE STATE-OF-THE-ART SCHEMES

prehensive privacy protection in mobile network ecosystems (§4.2). Table 2 shows the details of the comparison. These limitations suggest that further enhancements are needed to address the full range of privacy and security challenges faced by Thick MVNOs.

3. Background and Cryptographic Primitives

In this section, we briefly introduce the related background about MVNOs and some cryptographic primitives we used in our protocol.

3.1. The Cellular Ecosystem: MNOs and MVNOs

In 5G networks, Mobile Network Operators (MNOs) and Mobile Virtual Network Operators (MVNOs) operate in various configurations that affect both service delivery and security. While MNOs control the entire network infrastructure, MVNOs lease network resources from MNOs, with levels of dependency classified as Skinny, Thin, Light, Thick, and Full. Among these, Thick MVNOs stand out by owning significant network components, such as the Core Network (CN), providing them with greater control and flexibility compared to skinny MVNOs, which primarily depend on MNOs for network infrastructure. In a traditional 5G setup, the MNO typically controls both the base stations (gNB) and the CN, ensuring unified trust and security management. However, in a Thick MVNO model, the MVNO controls the CN while the MNO retains ownership of the gNB. This separation introduces potential trust and privacy challenges, as the gNB and CN, owned by different entities, must interact securely. A notable concern in Thick MVNOs is the potential for MNOs, via their gNBs, to access sensitive user data controlled by the MVNO's CN, leading to heightened privacy risks compared to traditional 5G setups or skinny MVNO structures. This paper focuses on these privacy issues within the Thick MVNO framework, particularly in the context of the 5G Authentication and Key Agreement (AKA) process.

3.2. Sanitizable Signature

The sanitizable signature scheme introduced in [8], [12], [13], [19] provides a mechanism for selective message modification while maintaining signature validity. SS achieves this by allowing a designated sanitiser to alter predefined parts of a signed message without invalidating the original signature, ensuring controlled flexibility. This is accomplished through a combination of traditional digital signatures and a sanitization algorithm, which enables both the message's integrity and authenticity to be preserved, even after selective modification. The scheme is particularly useful in contexts where sensitive information must be redacted or updated post-signing while still guaranteeing security properties like unforgeability and accountability.

Definition 1 (Sanitizable Signature). *An Sanitizable Signature (*SS*) scheme is defined as a tuple of six algorithms :*$\{\mathsf{KGen}, \mathsf{Sign}, \mathsf{Sanit}, \mathsf{Verify}, \mathsf{Prove}, \mathsf{Judge}\}$*. Specifically, a Sanitizable Signature scheme comprises the following probabilistic polynomial-time (PPT) algorithms:*

- $(\mathsf{pk_{sig}}, \mathsf{sk_{sig}}), (\mathsf{pk_{san}}, \mathsf{sk_{san}}), \leftarrow \mathsf{KGen}(1^\lambda, 1^\ell)$: The key generation algorithm inputs the security parameter λ and the upper limit of the message's blocks ℓ, and outputs the public and private key pairs of the signing and sanitizing steps respectively.
- $\sigma \leftarrow \mathsf{Sign}(m, \mathsf{ADM}, \mathsf{sk_{sig}}, \mathsf{pk_{san}})$: The signing algorithm inputs the message, the index of admissible modification ADM, the signing private key $\mathsf{sk_{sig}}$ and the sanitizing public key $\mathsf{pk_{san}}$, and outputs a signature σ.
- $(m', \sigma') \leftarrow \mathsf{Sanit}(m, \mathsf{MOD}, \sigma, \mathsf{pk_{sig}}, \mathsf{sk_{san}})$: The Sanitization algorithm inputs a message $m \in \{0,1\}^*$, a modification instruction $\mathsf{MOD} \subset \mathbb{N} \times \{0,1\}^\ell$ of the modifications to m, a signature σ, a public signing key $\mathsf{pk_{sig}}$ and a private sanitizing key $\mathsf{sk_{san}}$, then outputs a sanitized message m' and a corresponding sanitized signature σ'.
- $b \leftarrow \mathsf{Verify}(\mathsf{pk_{sig}}, \mathsf{pk_{san}}, m, \sigma)$: The verification algorithm inputs a message m, a signature σ, a signing public key $\mathsf{pk_{sig}}$, a sanitizing public key $\mathsf{pk_{san}}$, then outputs a bit $b \in \{0,1\}$, outputs 1 if validity holds, and 0 otherwise.
- $\pi \leftarrow \mathsf{Prove}(m, \sigma, \mathsf{sk_{sig}}, \mathsf{pk_{san}})$: The proof algorithm takes as input a message m, a signature σ, a signer private key $\mathsf{sk_{sig}}$, and a sanitizing public key $\mathsf{sk_{sig}}$, then outputs a proof π.
- $d \leftarrow \mathsf{Judge}(m, \sigma, \mathsf{pk_{sig}}, \mathsf{pk_{san}})$: The judge algorithm takes as input a message m, a signature σ, signing and sanitizing public keys $\mathsf{pk_{sig}}, \mathsf{pk_{san}}$, and proof π, then outputs a decision $d \in \{\mathsf{Sig}, \mathsf{San}\}$ indicating whether the message-signature pair was created by the signer

or the sanitizer.

A sanitizable signature allows specific modifications to the message even after the signing phase. It is especially suitable for scenarios where the content containing sensitive information needs to be processed flexibly while protecting the integrity of the signature. However, to generate a valid signature, the signer must have access to the plaintext message. This makes it impossible to use the conventional sanitizable signature in scenarios where the signer is not supposed to see the sensitive information of the message. Meanwhile, all the existing constructions of sanitizable signatures [12], [13], [19] assume that the signer has access to the plaintext message m. This access is necessary to define the permissible modifications (ADM) and share the plaintext between the signer and verifier to ensure accountability. However, these requirements conflict with the privacy-preserving properties intended to protect both the signer and the plaintext.

To address the aforementioned issues, here we will introduce a *novel* primitive SBS (**S**anitizable **B**lind **S**ignature), which is inspired by blind signature and sanitizable signature. Figure 2 shows the difference between SS and SBS. Compared to traditional definitions of SS, the SBS scheme introduces three additional algorithms, Extract, Derive and Trace, while removing the Prove and Judge algorithms. The roles of user and signer must be assigned to different parties during instantiation because an interactive protocol is executed similarly to blind signatures. The user provides the plaintext message and public key, while the signer signs using their private key without knowing any information about the plaintext. Additionally, SBS doesn't require the signer to generate a proof to trigger the accountability mechanism; instead, SBS collects the signatures to provide global traceability (GT), which includes a variant of accountability and double-spending check mechanism.

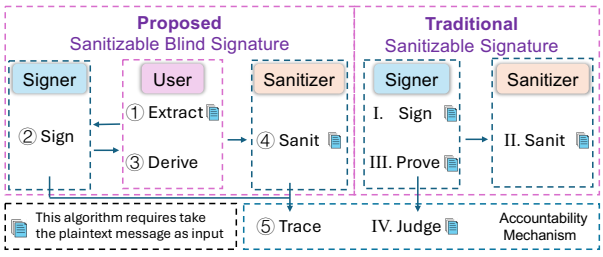

Figure 2. An overview of Sanitizable Signature and our proposed SBS

3.3. Proposed Sanitizable Blind Signature (SBS)

In this paper, we propose a novel sanitizable blind signature scheme SBS. The SBS scheme is the core of PGUS-AKA and PGUS-HO, enabling privacy-preserving measures in AKA to be compatible with Thick MVNOs. Our proposed SBS allows the signer to generate a signature without accessing m and ADM, while maintaining the core functionality and security properties of sanitizable signatures, thereby ensuring MA, UA, UEU, BSA, BSU (See §4.2). We also introduce an enhanced version of accountability through the Trace mechanism, which replaces the Prove and Judge algorithms of SS, thereby ensuring GT. We use blue color to represent the new algorithm introduced by SBS.

$(\mathsf{pk}_{\mathsf{sig}}, \mathsf{sk}_{\mathsf{sig}}) \leftarrow \mathsf{KGen}_{\mathsf{sig}}(1^\lambda, 1^l)$
Compute $\mathsf{BG} \leftarrow \mathsf{BGGen}(1^\lambda)$,
$\mathsf{BG} = (\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, G_1, G_2, G_T, e, q)$,
$(\mathsf{spksig}, \mathsf{sskeysig}) \leftarrow \mathsf{SPSEQ.KGen}(\mathsf{BG}, 1^l)$,
$(\mathsf{tpksig}, \mathsf{tsksig}) \leftarrow \mathsf{TRS.KGen}(1^\lambda)$,
$(\mathsf{pk}_{\mathsf{sig}}, \mathsf{sk}_{\mathsf{sig}}) := ((\mathsf{spksig}, \mathsf{tpksig}), ((\mathsf{sskeysig}, \mathsf{tsksig}))$
return $(\mathsf{pk}_{\mathsf{sig}}, \mathsf{sk}_{\mathsf{sig}})$.
$(\mathsf{pk}_{\mathsf{san}}, \mathsf{sk}_{\mathsf{san}}) \leftarrow \mathsf{KGen}_{\mathsf{san}}(1^\lambda, 1^l)$:
$(\mathsf{spksan}, \mathsf{sskeysan}) \leftarrow \mathsf{PKE.KGen}(1^\lambda)$,
$(\mathsf{tpksan}, \mathsf{tsksan}) \leftarrow \mathsf{TRS.KGen}(1^\lambda)$,
$(\mathsf{pk}_{\mathsf{san}}, \mathsf{sk}_{\mathsf{san}}) := ((\mathsf{spksan}, \mathsf{tpksan}), ((\mathsf{sskeysan}, \mathsf{tsksan}))$
return $(\mathsf{pk}_{\mathsf{san}}, \mathsf{sk}_{\mathsf{san}})$.
$(\mathsf{dt}, \mathsf{st}) \leftarrow \mathsf{Extract}(\mathsf{ADM}, m, \mathsf{pk}_{\mathsf{sig}}, \mathsf{pk}_{\mathsf{san}})$:
for each $i \in [\ell]$ **do**
Randomly choose $x_i, y_i \leftarrow \mathbb{Z}_q^*$,
$X_i := G_1^{x_i}, Y_i := X_i^{y_i}$,
$h_i := \mathsf{H}(i \| [\mathsf{pk}_{\mathsf{san}} \| m_i])^{y_i}$,
$\zeta_i := \begin{cases} y_i & i \in \mathsf{ADM} \\ 0 & \text{otherwise} \end{cases}$
endfor
$c \leftarrow \mathsf{PKE.Enc}(\mathsf{spksan}, \{\zeta_i\}_{i\in[\ell]})$
$\mathsf{dt} := (\{X_i\}_{i\in[\ell]}, \{Y_i\}_{i\in[\ell]})$
$\mathsf{st} := (c, \mathsf{dt}, \{h_i\}_{i\in[\ell]})$
$\sigma_{inner} \leftarrow \mathsf{Sign}(\mathsf{sk}_{\mathsf{sig}}, \mathsf{pk}_{\mathsf{san}}, \mathsf{dt})$:
Parse dt as $(\{X_i\}_{i\in[\ell]}, \{Y_i\}_{i\in[\ell]})$,
$\mu \leftarrow \mathsf{SPSEQ.Sign}(\mathsf{sskeysig}, (X_1, \ldots, X_\ell))$
$\eta \leftarrow \mathsf{SPSEQ.Sign}(\mathsf{sskeysig}, (Y_1, \ldots, Y_\ell))$
$\pi_{SS} := (\mu, \eta)$
$\pi_{TRS} := \mathsf{pk}_{\mathsf{sig}} \| \mathsf{pk}_{\mathsf{san}} \| \pi_{SS}$
$\sigma_{TRS} := \mathsf{TRS.Sign}(\mathsf{tsksig}, (\mathsf{tpksig}, \mathsf{tpksan}), \pi_{TRS})$
$\sigma_{inner} := (\pi_{SS}, \sigma_{TRS})$
$\sigma \leftarrow \mathsf{Derive}(\mathsf{st}, \sigma_{inner})$:
Parse σ_{inner} as (π_{SS}, σ_{TRS})
Parse st as $(c, \{X_i, Y_i\}_{i\in[\ell]}, \{h_i\}_{i\in[\ell]})$
$\sigma_{SS} := (\pi_{SS}, \{h_i, X_i, Y_i\}_{i\in[\ell]}, c)$
$\sigma := (\sigma_{SS}, \sigma_{TRS})$
$(m', \sigma') \leftarrow \mathsf{Sanit}(m, \mathsf{MOD}, \sigma, \mathsf{pk}_{\mathsf{sig}}, \mathsf{sk}_{\mathsf{san}})$:
Parse σ as $((\pi_{SS}, \{h_i, X_i, Y_i\}_{i\in[\ell]}, c), \sigma_{TRS})$
Parse π_{SS} as (μ, η)
$(\{\zeta_i\}_{i\in[\ell]}) \leftarrow \mathsf{PKE.Dec}(\mathsf{sskeysan}, c)$
$m' := \mathsf{MOD}(m)$
Randomly choose $r, s \leftarrow \mathbb{Z}_q^*$
$(X'_1, \ldots, X'_\ell) := (X_1, \ldots, X_\ell)^r$
$(Y'_1, \ldots, Y'_\ell) := (Y_1, \ldots, Y_\ell)^{r\cdot s}$
$\mu' \leftarrow \mathsf{SPSEQ.ChgRep}(\mathsf{spksig}, \{X_i\}_{i\in[\ell]}, \mu, r)$
$\eta' \leftarrow \mathsf{SPSEQ.ChgRep}(\mathsf{spksig}, \{Y_i\}_{i\in[\ell]}, \eta, r \cdot s)$
$\pi'_{SS} = (\mu', \eta')$
for each $i \in [\ell]$ **do**
$\zeta'_i := s \cdot \zeta_i$
$h'_i := \begin{cases} \mathsf{H}(i \| (\mathsf{pk}_{\mathsf{san}} \| m'_i))^{\zeta'_i} & i \in \mathsf{ADM} \\ h_i^s & \text{otherwise} \end{cases}$
endfor
$c' \leftarrow \mathsf{PKE.Enc}(\mathsf{spksan}, \{\zeta'_i\}_{i\in[\ell]})$
$\sigma'_{SS} := (\pi'_{SS}, \{h'_i, X'_i, Y'_i\}_{i=1}^{\ell}, c')$
$\pi'_{TRS} := \mathsf{pk}_{\mathsf{sig}} \| \mathsf{pk}_{\mathsf{san}} \| \pi'_{SS}$
$\sigma'_{TRS} \leftarrow \mathsf{TRS.Sign}(\mathsf{tsksan}, \{\mathsf{tpksig}, \mathsf{tpksan}\}, \pi'_{\mathsf{TRS}})$
$\sigma' := (\sigma'_{SS}, \sigma'_{TRS})$

Figure 3. Main block of SBS construction. Blue highlight represents the new algorithm SBS executes.

Definition 2 (Sanitizable Blind Signature). *An Sanitizable Blind Signature (*SBS*) scheme is defined as a tuple of seven algorithms:* {KGen, Extract, Sign, Derive, Sanit, Verify, Trace}. *Specifically, a Sanitizable Blind Signature scheme comprises the following probabilistic*

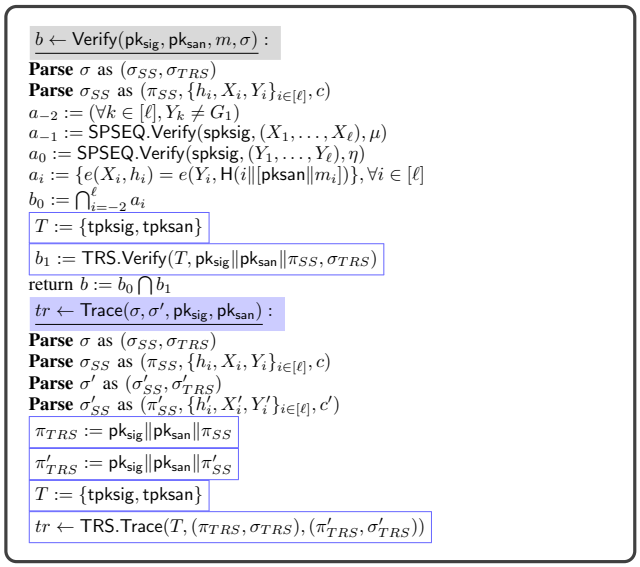

Figure 4. Verification and Trace mechanism of SBS.

polynomial-time (PPT) algorithms:

- $(\mathsf{pk_{sig}}, \mathsf{sk_{sig}}), (\mathsf{pk_{san}}, \mathsf{sk_{san}}), \leftarrow \mathsf{KGen}(1^\lambda, 1^\ell)$: The key generation algorithm inputs the security parameter λ and the upper limit of the message's blocks ℓ, and output the public and private key pairs of the signing and sanitizing steps respectively.
- $(\mathsf{dt}, \mathsf{st}) \leftarrow \mathsf{Extract}(\mathsf{ADM}, m, \mathsf{pk_{sig}}, \mathsf{pk_{san}})$: The extracting algorithm inputs the message $m \in \{0,1\}^*$ and the signing public key $\mathsf{pk_{sig}}$, and the sanitizing public key $\mathsf{pk_{san}}$, as well as $\mathsf{ADM} \in \mathbb{N} \times 2^{\mathbb{N}}$, a description of the admissible modifications to message by the sanitizer. Then the extracting algorithm outputs a data dt and a state st. The user keep the state st to itself, sending the data dt to signer.
- $\sigma_{inner} \leftarrow \mathsf{Sign}(\mathsf{sk_{sig}}, \mathsf{pk_{san}}, \mathsf{dt})$: The signing algorithm inputs data dt, and a signing private key $\mathsf{sk_{sig}}$, a sanitizing public key $\mathsf{pk_{san}}$, and outputs a signature σ_{inner}.
- $\sigma \leftarrow \mathsf{Derive}(\mathsf{st}, \sigma_{inner})$: The deriving algorithm inputs a state st and a inner signature σ_{inner}, then outputs a derived signature σ.
- $(m', \sigma') \leftarrow \mathsf{Sanit}(m, \mathsf{MOD}, \sigma, \mathsf{pk_{sig}}, \mathsf{sk_{san}})$: The Sanitization algorithm inputs a message $m \in \{0,1\}^*$, a modification instruction $\mathsf{MOD} \subset \mathbb{N} \times \{0,1\}^\ell$ of the modifications to m, a signature σ, a public signing key $\mathsf{pk_{sig}}$ and a private sanitizing key $\mathsf{sk_{san}}$, then outputs a sanitized message m' and a corresponding sanitized signature σ'.
- $b \leftarrow \mathsf{Verify}(\mathsf{pk_{sig}}, \mathsf{pk_{san}}, m, \sigma)$: The verification algorithm inputs a message m, a signature σ, a signing public key $\mathsf{pk_{sig}}$, a sanitizing public key $\mathsf{pk_{san}}$, then outputs a bit $b \in \{0,1\}$, outputs 1 if validity holds, and 0 otherwise.
- $tr \leftarrow \mathsf{Trace}(\sigma, \sigma', \mathsf{pk_{sig}}, \mathsf{pk_{san}})$: The Trace algorithm inputs two signatures σ, σ', as well as a signing public key $\mathsf{pk_{sig}}$ and a saniziting public key $\mathsf{pk_{san}}$. Then, the algorithm outputs one of the following strings: `"indep"`, or pk_i, $i \in \{\mathsf{sig}, \mathsf{san}\}$. This algorithm is highly flexible in its application. If any member whose public key is included in the ring T wants to prove that a signature σ has been or has not been generated by himself, he can use the same private key to generate another different signature σ^*, and submits (σ, σ^*) to Trace algorithm. If the output is `"indep"`, it proves that the signature does not belong to itself; if the output is the public key pk_i, it proves that the signature is generated by $i \in \{\mathsf{sig}, \mathsf{san}\}$ itself.

Remark

Our proposed SBS combines the strengths of sanitizable signature and blind signature, providing enhanced security, flexible sanitization, and global traceability across various application scenarios. In **Thick MVNO PGUS** (§5), SBS can establish mutual trust between gNB and CN while hiding gNB's identity from CN. In the Trace mechanism, the MVNO can control the number of public keys in ring T the MNO from overclaiming the number of users it serves. In addition, in the case of **medical data handling system** [48], SBS allows a hospital administrator to sign a patient's medical record without viewing sensitive details, while the patient can later modify certain information for insurance application purposes without revisiting the hospital. The SBS is also adaptable for secure routing, electronic contracts, and legal document signing, especially in cases where the signer may be offline or where communication overhead is high.

3.4. Generic Construction of SBS

3.4.1. Overview of Our Technique. We start from a generic construction of a "weak" sanitizable signature scheme in [13], it serves as the foundation for our SBS scheme's construction for the following reasons: (1) It offers both unlinkability and invisibility, addressing a more comprehensive set of security requirements. This makes it particularly attractive for applications where the message contains sensitive content. (2) The scheme employs a method similar to BLS signatures, enabling the private key $\mathsf{sk_{sig}}$ and the plaintext message m to be input at different stages. It helps us to decouple the original Signing algorithm.

Traceable Ring Signatures (TRS) provide a balance between anonymity and traceability without requiring a group manager [21], [22]. They support accountability by allowing ring members to voluntarily prove if they generated a signature and enabling "double-spending traceability", which can revoke anonymity for overbilling actors (sanitizing key reuses). We propose this traceability as an alternative to accountability, using TRS to trace dishonest participants' public keys while keeping honest ones anonymous, so traceability supports both active and passive accountability.

3.4.2. Decoupling signing and plaintext message. We give the first part of construction Π_{SS}. In the User-Signer

interaction protocol, the User runs the Extract and Derive algorithms, while the Signer only runs the Sign algorithm. The construction of interaction is shown in Figure 3 and explained as follows: (1) In the Π_{SS}.Extract algorithm, the User randomly selects $x_i, y_i \in \mathbb{Z}_q^*$, and then generating $X_i = G_1^{x_i}$ and $Y_i = X_i^{y_i}$. The User then packages all (X_i, Y_i) into dt and sends it to the Signer. Next, the User runs a hash function to convert the message m into h_i. Then, the User also runs a PKE.Enc to encrypt the ADM's corresponding y_i, then outputting a ciphertext c. We emphasize that this step provides invisibility [10]. Finally, the User packages $c, \mathsf{dt}, \{h_i\}_{i\in[\ell]}$ into st, storing for the subsequent Derive algorithm. (2) In the Π_{SS}.Sign algorithm, the Signer respectively signs two vectors $\{X_i\}_{i=1}^{l}$ and $\{Y_i\}_{i=1}^{l}$ using SPSEQ.Sign, packaging the result into π_{SS}, and return it to User. (3) In the Π_{SS}.Derive algorithm, the User combines st and π_{SS}, and reconstruct a valid signature $\sigma = \sigma_{SS}$.

In the Π_{SS}.Sanit algorithm (Figure 3), the sanitizer uses PKE.Dec to decrypt the ciphertext c and gets the information of y_i corresponding to admissible blocks. It rerandomizes the two vectors $\{X_i\}_{i=1}^{l}$ and $\{Y_i\}_{i=1}^{l}$ based on two scalars $r, s \in \mathbb{Z}_q^*$, then uses SPSEQ.ChgRep to achieve *perfect signature adaption*, which relies on the properties of SPSEQ [26]. Finally, it generates σ'_{SS} by using a similar approach to the signing part and returns $\sigma' = \sigma'_{SS}$.

Finally, in the Π_{SS}.Verify algorithm (Figure 4), the verifier checks the validity of SPSEQ, and also checks the relevance between message-signature pairs and two vector parameters, which represent correctness of both sanitizable signature and interaction between User and Signer.

3.4.3. Transformation for Traceability. We integrate the TRS into the Π_{SS} construction, creating Π_{SBS}. Both parties generate TRS key pairs, with the Signer including a combined public key and SPSEQ signature in the tag. The resulting signature σ is a pair $(\sigma_{SS}, \sigma_{TRS})$. During sanitization, the sanitizer follows the same approach, adjusting representations in SPSEQ. We note that tr will reject "`linked`" from TRS.Trace, converting it into $\perp$, while accepting "`indep`" or pk_i outputs. This approach guarantees the EUF-CMA of SBS [57].

The Π_{SBS}.Trace algorithm (Figure 4) allows ring members to verify signature authorship by generating a new signature with the same ring T. If it returns "`indep`", they did not create the signature; if it returns pk_i, it identifies them as the signer. Additionally, authorities can detect "double-spending" by linking suspect signatures without requiring voluntary evidence submission or Prove/Judge algorithms.

4. Threat Model and Design Goals

In this section, we first present our threat model. After that, we introduce the design goals to establish the foundational assumption and security objectives.

4.1. Threat Model

To comprehensively analyze potential security threats, we categorize them hierarchically into three types. We first consider the widely used Dolev-Yao threat model [18] as a foundational approach. In addition, we investigate threat scenarios specific to the Thick MVNO system architecture, addressing the unique privacy vulnerabilities. Finally, a UC (Universally Composable) security-based threat is incorporated to ensure robust security guarantees across different adversarial settings. We define our threat model as follows:

- $\mathcal{A}_1$: The Type 1 adversary utilizes a threat model commonly used in privacy-preserving systems to protect against information leaks between participants (UE, gNB and CN). By incorporating standards from the 3GPP group and insights from recent research on 5G authentication and handover, Thick MVNO assumes that all user-to-network communications are conducted over public channels, which an adversary can fully monitor and manipulate. This is also known as the Dolev-Yao model [18].
- $\mathcal{A}_2$: The Type 2 adversary aimed to analyse the Thick MVNO attacker since our protocol is specifically designed for Thick MVNO architectures. We assume that the MNO and MVNO are semi-honest (honest but curious) and non-colluding. While the user connects to these service providers, they may collect the user's sensitive information from the protocol execution, such as identity and location information [34], [44]. Meanwhile, the linkability attack [11] tries to link UE's information, such as 5G AKA sessions, or attempts to trace users' footprints or link the certificate sanitized by gNB, thereby impersonating an honest gNB. We also assume that MNOs may attempt to overbill MVNOs by lying about their traffic [34].
- $\mathcal{A}_3$: The Type 3 adversary is the UC-based adversary, which operates within the Universal Composability (UC) framework [14] to exploit the complex protocol compositions in systems like 5G and MVNO networks. This adversary can effectively simulate both the $\mathcal{A}_1$ and $\mathcal{A}_2$ internally, and can also engage with multiple protocols in real and ideal worlds, attempting to find weaknesses in their interactions. It can statically corrupt parties within the system and control the environment $\mathcal{Z}$ to manipulate inputs and observe outputs. While it monitors and manipulates message flows and accesses functionalities, it is constrained by the UC framework's simulator ($\mathcal{S}$), which ensures that real-world execution remains indistinguishable from ideal-world execution. Furthermore, the UC framework is crucial for 5G systems as it allows for secure protocol composition, ensuring that multiple cryptographic protocols work together seamlessly [4].

In summary, there is no conflict among the three types of threats, as their goals and attack scenarios are different. However, when considering them within the UC framework, $\mathcal{A}_3$ is already the strongest adversarial model, encompassing the attack behaviours of both $\mathcal{A}_1$ and $\mathcal{A}_2$. Therefore, our security analysis can be simplified by using only the UC model, as it can effectively simulate the behaviours of $\mathcal{A}_1$ and $\mathcal{A}_2$. Furthermore, under the UC framework, all attack

behaviours are constrained by the simulator. Note that a fully compromised core network (CN) inherently cannot provide complete user anonymity, as certain network components must necessarily identify users. Such a scenario, however, falls outside the threat model considered in this work. Specifically, the Universally Composable (UC) framework presented here assumes only partial corruption—namely, the compromise of the Home Subscriber Server (HSS)—while the components responsible for generating commitment keys (ck) and the Common Reference String (CRS) remain honest. Importantly, UC security does not imply protection against total system compromise; rather, it captures adversaries corrupting clearly defined subsets of entities. In our setting, only the CN elements involved in communication and authentication are considered vulnerable, not the entire CN infrastructure. This assumption will be explicitly clarified in the revised manuscript.

4.2. Design Goals

To ensure comprehensive security and privacy in 5G networks and **Thick MVNO** (Thick Mobile Virtual Network Operator) environments, a set of rigorous design principles is required. Some previous research [4], [9], [44], [54], [58] discussed the critical security requirements for the seamless integration of 5G networks and MVNO architectures, emphasizing the need for robust authentication, data protection, and subscriber privacy mechanisms. Moreover, as security demands intensify within 5G and Thick MVNO infrastructures, it becomes imperative to define additional design goals tailored to these advanced needs. Following are the design goals that we considered:

- **Mutual Authentication (MA*):** In Thick MVNO environments, Mutual Authentication (MA*) mandates that gNB, CN, and UE verify each other's identities before service provision to prevent threats like MitM attacks. Unlike in traditional 5G, where trust is assumed between gNB and CN, Thick MVNOs lack this trust, making existing AKA and handover protocols non-compliant with 5G's security standard for serving network authorization by the home network (TS 33.501) [50]. MA* introduces privacy-preserving authentication among gNB-CN, UE-gNB, and UE-CN, adding a requirement that gNB and CN establish mutual trust before the UE joins, while UE-gNB and UE-CN authentication follows [4]. This ensures weak agreement, confirming interaction without exposing identities.
- **User Anonymity (UA)** UA ensures that no *active* attacker can deduce a user's identity or track their activity. Maintaining the secrecy of the SUPI has been a standard in 5G [50]. While 3GPP primarily addresses passive attackers, an active attacker could still trace a UE if the sequence numbers from either the UE or CN are exposed, leading to a linkability attack [11]. This vulnerability could be exploited by external adversaries like the aforementioned $\mathcal{A}_1$, or by semi-honest entities like $\mathcal{A}_2$. For comprehensive UA, active attackers must be taken into account. Further, we will explore UE indistinguishability as part of *UE Unlinkability* in the following part.
- **Base Station Anonymity (BSA):** BSA is a critical security requirement for Thick MVNOs, designed to prevent the gNB from revealing its identity when communicating with the CN, thereby safeguarding the UE's location information and footprint. While skinny MVNOs [4] and conventional 5G [2], [3]— directly send the gNB's certificate to the CN, in Thick MVNO, BSA requires base station to conceal the certificate towards CN. According to [34], failing to enforce BSA allows semi-honest CNs ($\mathcal{A}_2$) to track and extract UE trajectories. Since gNB locations are public [32], knowing the gNB's position can further narrow down the UE's location, increasing privacy risks through trajectory leaks. While BSA provides protocol-layer protection for user location and trajectory [34], it is not sufficient against privacy leaks from application-layer or physical-layer [24]. However, it can coexist with other layer solutions for enhanced privacy in Thick MVNO architectures [52], [53].
- **Global Unlinkability (GU):** To address linkability attacks in practical scenarios, we adopt an approach distinct from the conventional notion of unlinkability typically associated with such attacks [31]. Instead, we extend the concept of "UE indistinguishability" as outlined in prior works [54]. For Thick MVNOs, the GU is characterized by two essential properties: UE Unlinkability (UEU) and Base Station Unlinkability (BSU). Recent findings [11] indicate that even with obfuscated long-term identifiers, active attackers may exploit certain vulnerabilities, such as failure messages, to correlate user sessions within the 5G AKA protocol. In response, previous work [4], [54] has proposed AKA protocols with the UEU property. As Thick MVNOs are also required to protect base station identities, we define BSU to capture the resilience against active attackers specifically targeting this dimension.
  UE Unlinkability (UEU) UEU is an extension of User Anonymity (UA) designed to address likability attacks. [31], [54] It requires that, given two user entities, UE_1 and UE_2, and any AKA or Handover session involving either UE_1 or UE_2, an active attacker cannot distinguish which of the two entities it is interacting with. [it helps to perverse the user footprint]
  Base Station Unlinkability (BSU) Similar to UEU, BSU is an extension of the Base Station Anonymity (BSA), which aims to address the linkability attacks [34] for the base station. It requires that, given two base station entities, gNB_1 and gNB_2, and any AKA or Handover session involving either gNB_1 or gNB_2, an active attacker cannot distinguish which of the two entities it is interacting with. Our protocol enforces BSU during the *System Registration* phase of Thick MVNOs, where the gNB and Core Network establish mutual trust and generate a sanitizable blind signature.
- **Global Traceability (GT)** Global Traceability (GT) in Thick MVNOs incorporates two essential properties:

Double-Spending Traceability (DST) and Responsibility & Exculpability (R&E). GT extends traditional accountability concepts in sanitizable signatures [8], [12] to enforce rigorous traceability.

Double-Spending Traceability (DST) DST ensures that both the Core Network and gNB maintain anonymity for a single operation under a specific set of asymmetric sanitizing keys. However, if a party performs multiple operations using the same private sanitizing key, their identity will be traced. This mechanism prevents key abuse between the CN and gNB, enabling the MVNO to implement effective traffic control. Notably, DST passively reveals the identity of violators without requiring them to actively provide accountability proof, offering protection against potential gNB overbilling in Thick MVNO environments [34].

Responsibility and Exculpability (R&E) R&E ensures that a party cannot deny actions it has legitimately performed, nor can an adversary falsely attribute actions to an uninvolved party. This property is important as gNB privacy introduces potential disputes between entities over signature sources. Achieving both responsibility and exculpability encapsulates a comprehensive form of accountability, aligning with established sanitizable signature accountability models [12].

Remark

One might wonder why MNOs and MVNOs would be interested in adopting the aforementioned changes and design goals, or why the CN (MVNO) would be willing to relinquish the user data it currently collects. [27], [37]. The incentives are outlined as follows: (1) A PGUS-based MVNO would likely attract more users in a competitive market, as users are increasingly concerned about information security [35], [55]. (2) Regulations related to user privacy and security are expected to become more stringent [16], and the PGUS scheme can help reduce potential financial losses from regulatory violations (such as GDPR fines [56]). (3) The MVNO (CN) can utilize the GT (Global Traceability) mechanism to preemptively control the user traffic served by the MNO (gNB), which guarantees the benefits of the MVNO and prevents selfish MNO from overbilling [34]. In summary, these three advantages are expected to outweigh the benefits the MVNO might derive from collecting or selling user information.

5. The Proposed Scheme

In this section, we present the PGUS scheme, which comprises two key components: the Authentication and Key Agreement (AKA) protocol PGUS-AKA §5.2 and the Handover Protocol PGUS-HO §5.3. The cryptographic primitives Non-Interactive Zero-Knowledge Proof ZK, Commitment Com, Public Key Encryption PKE are introduced in supplementary material [57].

5.1. System Setup

In the setup phase, essential parameters and cryptographic keys are generated to ensure the security and integrity of subsequent processes. The MVNO company is responsible for managing user registration and collaboration with the MNO. During this phase, it generates the common reference string crs, trapdoor td, commitment key ck, and the user's pseudo-identifier pid. Additionally, it creates the digital signature key pairs (spk_u, ssk_u) and securely distributes these values to the User Equipments. Following this, it computes the hash of the pid H_{pid} and stores it in a list $List_u$ for future reference. Meanwhile, It generates the SBS signing key pair $(\mathsf{pk_{sig}}, \mathsf{sk_{sig}}) \leftarrow \mathsf{SBS.KGen_{sig}}$ and the digital signature key pairs (spk_{CN}, ssk_{CN}), which it securely transmits to the CN for establishing trusted communications. The MNO entity is responsible for setting up the gNB. During this process, it generates the common reference string crs', trapdoor td', commitment key ck', and the gNB's pseudo-identifier gid. It also creates the sanitizing key pairs $(\mathsf{pk_{san}}, \mathsf{sk_{san}}) \leftarrow \mathsf{SBS.KGen_{san}}$ and digital signature key pairs (spk_g, ssk_g), securely distributing these values to the gNBs. Subsequently, it computes the hash of gid H_{gid} and stores it in a list $List_g$. The MVNO and MNO collaborate with each other and exchange essential information, including the parameters and public keys above. They also agree on the number of sanitizing key pairs, and share all these valid public sanitizing keys in ring T. (§3.3)

5.2. System Registration and Initial Authentication (AKA) Protocol

We divided PGUS-AKA protocol into three phases: System Registration, Initial Authentication, and Trace mechanism. The steps are also shown in Figure 5. The System Registration phase involves gNB and CN, at the end of Phase 1, gNB should have mutual authentication with CN, and get a SBS signature on his certificate. In Phase 2, new UE joins the execution and CN generates credentials for this UE.

Phase 1: System Registration. All gNBs that connect to the CN need to be registered and verified.

Step 1: $\mathbf{M}_1^{SR} : \mathsf{dt} \| com_{gNB} \| \pi_{ZK_{gNB}}$

The initialization of the proposed system registration begins with generating a unique certificate for the gNB. In this process, certificate generation leverages the concatenation of the gNB's identifier id_{gNB}, with a timestamp τ, thereby embedding temporal validity into the certificate. In the next step, let $(\mathsf{dt}, \mathsf{st}) \leftarrow \mathsf{SBS.Extract}(\mathsf{ADM}, C_{gNB}, pk_{sig}^{CN})$, where the gNB interacts with the SBS.Extract algorithm, taking as inputs the ADM, the gNB's certificate (C_{gNB}), and the CN's signing public key pk_{sig}^{CN}. It then selects a random value $u \in \{0,1\}^n$ and computes a commitment of their pseudo-identifier gid and a zero-knowledge proof $\pi_{ZK_{gNB}}$ for the $\mathcal{L}_{ZK_{gNB}} = \{(com_{gNB}, List_g) \mid \exists w_g := (gid, r)$ s.t. $com_{gNB} = \mathsf{Com}_{ck'}(gid, u) \wedge H_{gid} \in List_g\}$. The resulting message $\mathbf{M}_1^{SR}$ is then formulated as: $\mathbf{M}_1^{SR} : \mathsf{dt} \| com_{gNB} \| \pi_{ZK_{gNB}}$. This message, containing dt,

com_{gNB}, and $\pi_{ZK_{gNB}}$, is subsequently sent from the gNB to the CN as part of the registration phase.

Step 2: $\mathbf{M}_2^{SR} : \sigma_{inner}$

Upon CN receiving the message $\mathbf{M}_1^{SR}$, it begins by parsing the message components $(\mathsf{dt}, com_{gNB}, \pi_{ZK_{gNB}})$. The CN then performs a NIZK verification to check the validity of the anonymous source of dt, aborting if the zero-knowledge proof does not validate, i.e., if $\mathsf{ZK.V}(crs, (\mathsf{dt}, com_{gNB}), list_g, \pi_{ZK_{gNB}}) \neq 1$. When the proof is confirmed valid, the CN generates an inner signature $\sigma_{inner} \leftarrow \mathsf{SBS.Sign}(sk_{sig}^{CN}, pk_{san}^{gNB}, \mathsf{dt})$ by signing the data dt with its signing key sk_{sig}^{CN} and the gNB's public key pk_{san}^{gNB}. Finally, the CN forwards this inner signature as $\mathbf{M}_2^{SR}$ to the gNB. In this phase, CN does not get any information about gid or C_{gNB}.

Step 3: *Derive and Verification*

After the gNB receives the message $\mathbf{M}_2^{SR}$, containing σ_{inner}, the gNB initiates a derivation step to transform this inner signature. Specifically, the gNB computes a fixed signature $\sigma_{fix} \leftarrow \mathsf{SBS.Derive}(\mathsf{st}, \sigma_{inner})$ by applying the derivation function with the current state st and σ_{inner}. Following this, the gNB verifies the fixed signature to confirm its validity, aborting if $\mathsf{SBS.Verify}(C_{gNB}, \sigma_{fix}, pk_{sig}^{CN}, pk_{san}^{gNB}) \neq 1$. This verification ensures that the fixed signature has not been tampered with.

Phase 2: UE Initial Registration. All UE that connect to the gNB need to be registered and verified by CN.

Step 1: $\mathbf{M}_3^{IA} : C_{MOD}^{gNB}, \sigma_{MOD}^{gNB}$

The gNB updates the modified certificate component as $C_{MOD}^{gNB} \leftarrow id_{gNB} \| \tau_1$, incorporating its identifier and timestamp for temporal validity. Subsequently, the gNB performs a sanitization step, deriving a modified signature $\sigma_{MOD}^{gNB} \leftarrow \mathsf{SBS.Sanit}(\sigma_{fix}, C_{MOD}^{gNB}, pk_{sig}^{CN}, sk_{san}^{gNB})$ using the fixed signature σ_{fix}, the modified certificate C_{MOD}^{gNB}, CN's signing public key pk_{sig}^{CN}, and the gNB's sanitization private key sk_{san}^{gNB}. This sanitized message, $\mathbf{M}_3^{IA}$, containing $(C_{MOD}^{gNB}, \sigma_{MOD}^{gNB})$, is then broadcast to the UE.

Step 2: $\mathbf{M}_4^{IA} : \pi_{ZK_{UE}}, com_{UE}, PK_u, \tau_2, \sigma^{UE}$

Upon receiving the broadcast message from the gNB, the UE first verifies the gNB's identifier id_{gNB} and timestamp τ. Then, it proceeds with a verification step, aborting if $\mathsf{SBS.Verify}(C_{MOD}^{gNB}, \sigma_{MOD}^{gNB}, pk_{sig}^{CN}, pk_{san}^{gNB}) \neq 1$, to ensure the integrity and authenticity of the modified certificate C_{MOD}^{gNB} and the sanitized signature σ_{MOD}^{gNB}. From this step, PGUS seamlessly integrates with the AKA in skinny MVNOs [4]. The UE generates a key pair $(PK_u, SK_u) \leftarrow \mathsf{PKE.KGen}(\lambda)$, where λ is the security parameter. It then selects a random value $r \in \{0,1\}^n$ and computes a commitment of their pseudo-identifier pid and a zero-knowledge proof $\pi_{ZK_{UE}}$ for the $\mathcal{L}_{ZK_{UE}} = \{(com_{UE}, List_u) \mid \exists w_u := (pid, r)$ s.t. $com_{UE} = \mathsf{Com}_{ck}(pid, r) \wedge H_{pid} \in List_u\}$. Finally, the user signs all the previous computations and sends them to gNB. Finally, the UE signs the concatenated message $\pi_{ZK_{UE}} \| com_{UE} \| PK_u \| \tau_2$ with its signing secret key ssk_u, resulting in the signature σ, and sends $\mathbf{M}_2^{IA} : (\pi_{ZK_{UE}}, com_{UE}, PK_u, \tau_2, \sigma^{UE})$ to the gNB. The signature Sig here could be any commom schemes which satisfy EUF-CMA security. (e.g. Schnorr Signature [45])

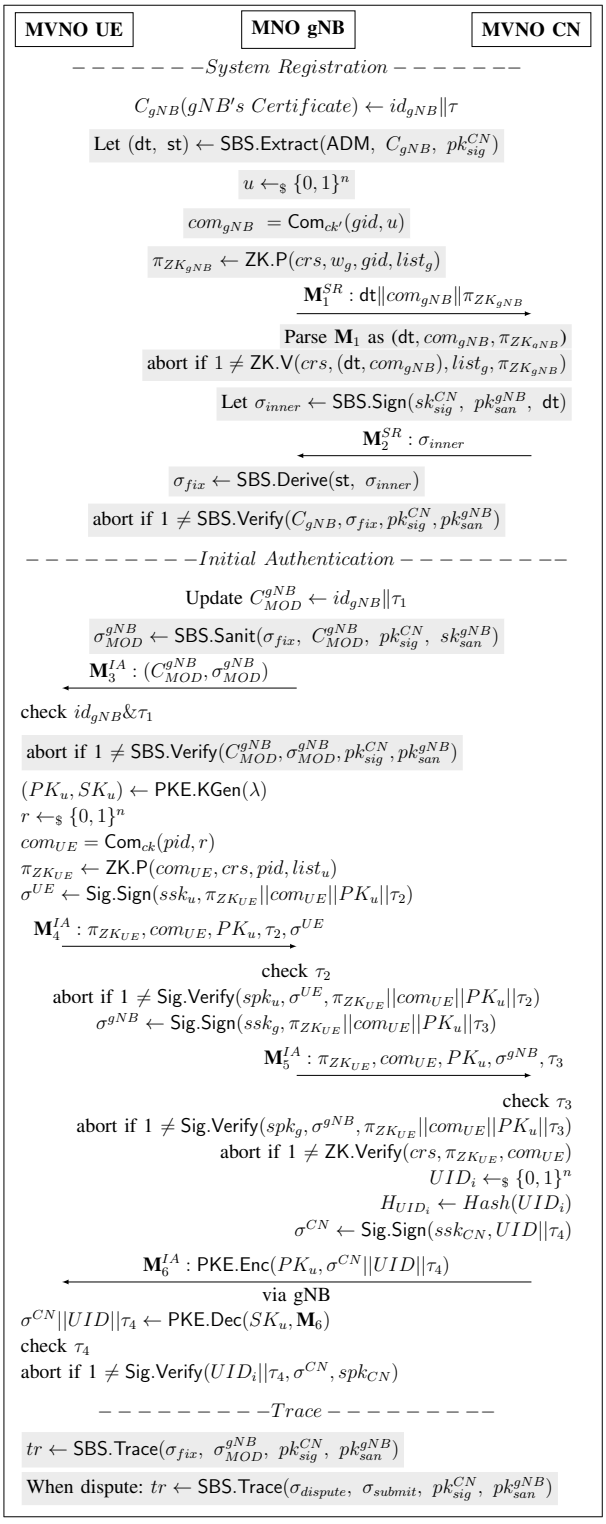

Figure 5. PGUS Authentication and Key Agreement Protocol

Step 3: $\mathbf{M}_5^{IA} : \pi_{ZK_{UE}}, com_{UE}, PK_u, \sigma^{gNB}, \tau_3$

Upon receiving the message from the UE, the gNB first checks the timestamp τ_2. It then verifies the in-

tegrity of the received message by checking if $1 = \mathsf{Sig.Verify}(spk_u, \sigma^{UE}, \pi_{ZK_{UE}}||com_{UE}||PK_u||\tau_2)$, aborting if the verification fails. Once the verification is successful, the gNB generates a new signature $\sigma^{gNB} \leftarrow \mathsf{Sig.Sign}(ssk_g, \pi_{ZK}||com_{UE}||PK_u||\tau_3)$ using its own signing secret key ssk_g and a new timestamp τ_3. Finally, the gNB sends $\mathbf{M}_5^{IA} : (\pi_{ZK_{UE}}, com_{UE}, PK_u, \sigma^{gNB}, \tau_3)$ to the CN.

Step4: $\mathbf{M}_6^{IA} : \mathsf{PKE.Enc}(PK_u, \sigma^{CN}||UID||\tau_4)$

Upon receiving the message from the gNB, the CN first checks the validity of the timestamp τ_3. It then verifies the signature σ^{gNB} using spk_g, ensuring that $1 = \mathsf{Sig.Verify}(spk_g, \sigma^{gNB}, \pi_{ZK_{UE}}||com_{UE}||PK_u||\tau_3)$; the process aborts if verification fails. Following this, the CN confirms the integrity of the zero-knowledge proof by verifying that $1 = \mathsf{ZK.Verify}(crs, \pi_{ZK_{UE}}, com_{UE})$. Once all checks and verifications are successful, the CN newly and randomly generates a universal identifier UID_i, which will be the user's identifier during handover phase. Then, the CN stores the UID_i and computes its hash $H_{UID_i} \leftarrow \mathsf{Hash}(UID_i)$. Subsequently, the CN creates a signature $\sigma_U \leftarrow \mathsf{Sig.Sign}(ssk_{CN}, UID_i||\tau_4)$, where τ_4 is a new timestamp. The CN then sends $\mathbf{M}_6^{IA} : \mathsf{PKE.Enc}(PK_u, \sigma_U||UID_i||\tau_4)$, encrypted with the UE's public key PK_u, to the UE via the gNB.

Step 5: *UE Finalize Initial Authentication*

Upon receiving $\mathbf{M}_6^{IA}$ from the gNB, the UE first decrypts the message $\mathbf{M}_6^{IA}$ using its private key SK_u to retrieve $\sigma^{CN}||UID_i||\tau_4$. It then checks the validity of the timestamp τ_4. Following this, the UE verifies the signature σ_U by ensuring that $1 = \mathsf{Sig.Verify}(UID_i||\tau_4, \sigma^{CN}, spk_{CN})$, using CN's public signing key spk_{CN}. The process aborts if the verification fails, thereby ensuring the integrity and authenticity of the received message.

Phase 3: Trace phase in PGUS-AKA. In this phase, the UE initiates a tracing process to establish accountability, aimed at determining the origin of certain actions or messages. Initially, a trace identifier tr is generated by applying the tracing function $\mathsf{SBS.Trace}$ to the fixed signature σ_{fix} and the modified signature σ_{MOD}^{gNB}, alongside CN's public signing key pk_{sig}^{CN} and gNB's sanitization public key pk_{san}^{gNB}. This trace allows the UE to link specific actions to their originating entities, ensuring traceability within the system. In the event of a dispute, the UE recalculates the trace using the disputed signature $\sigma_{dispute}$ and the submitted signature σ_{submit} through $\mathsf{SBS.Trace}$, combined again with the relevant public keys pk_{sig}^{CN} and pk_{san}^{gNB}. This process provides a reliable mechanism for verifying the authenticity and origin of potentially contentious actions, thereby reinforcing accountability in interactions. All the fixed signatures σ_{fix} are collected by gNB for the subsequent "double-spending Trace" (in §5.3).

5.3. Handover Protocol

Our PGUS-Handover protocol also embeds the Trace mechanism. The steps are also shown in Figure 6. The

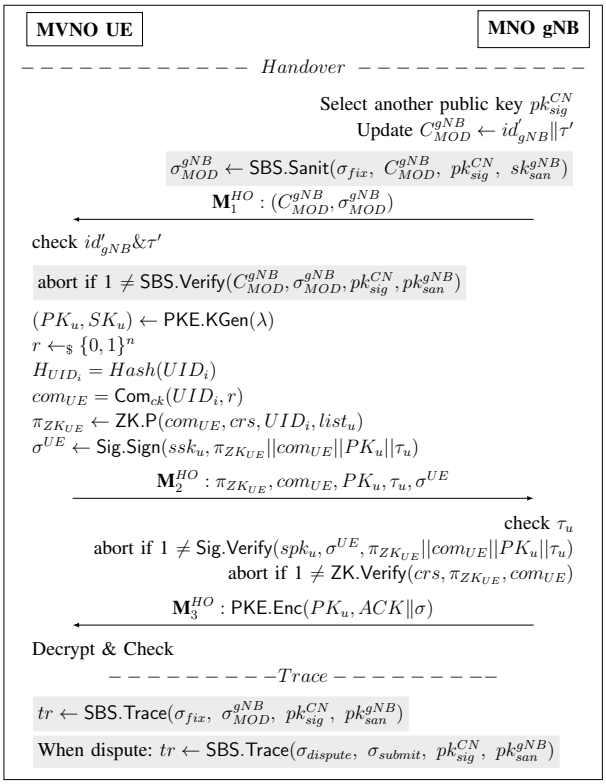

Figure 6. PGUS Handover Protocol

UEs do not need to execute another initial authentication, instead, PGUS-HO provides seamless handover because of the $\mathsf{SBS.Sanit}$ algorithm.

Step 1: $\mathbf{M}_1^{HO} : (C_{MOD}^{gNB}, \sigma_{MOD}^{gNB})$

Similar to the initial authentication, the gNB undertakes the same preparation steps. The gNB selects a new public key, denoted as pk_{sig}^{CN}, to prepare for the upcoming communication. The gNB updates the modified certificate component to $C_{MOD}^{gNB} \leftarrow id'_{gNB}\|\tau'$, incorporating its identifier and a timestamp for temporal validity. The gNB generates a sanitized signature $\sigma_{MOD}^{gNB} \leftarrow \mathsf{SBS.Sanit}(\sigma_{fix}, C_{MOD}^{gNB}, pk_{sig}^{CN}, sk_{san}^{gNB})$ using the fixed signature σ_{fix}, the modified certificate C_{MOD}^{gNB}, the CN's signing public key, and the gNB's sanitization private key. The gNB sends $\mathbf{M}_1^{HO} : (C_{MOD}^{gNB}, \sigma_{MOD}^{gNB})$ to the UE for validation.

Step 2: $\mathbf{M}_2^{HO} : \pi_{ZK_{gNB}}, com_{UE}, PK_u, \tau_u, \sigma^{UE}$

The UE checks the identifier id'_{gNB} and timestamp τ'. It aborts if the validation fails with $\mathsf{SBS.Verify}(C_{MOD}^{gNB}, \sigma_{MOD}^{gNB}, pk_{sig}^{CN}, pk_{san}^{gNB}) \neq 1$. From this step, PGUS seamlessly integrates with the AKA in skinny MVNOs [4]. The UE generates its own key pair $(PK_u, SK_u) \leftarrow \mathsf{PKE.KGen}(\lambda)$ based on the security parameter λ. It then selects a random value $r \in \{0,1\}^n$ and computes a commitment of their universal-identifier UID_i and a zero-knowledge proof $\pi_{ZK_{UE}}$ for the $\mathcal{L}_{ZK_{UE}} = \{(com_{UE}, List_u) \mid \exists w_u := (UID_i, r)$ s.t. $com_{UE} = \mathsf{Com}_{ck}(UID_i, r) \wedge H_{UID_i} \in List_u\}$. Then UE creates a signature $\sigma \leftarrow \mathsf{Sig.Sign}(ssk_u, \pi_{ZK_{UE}}||com_{UE}||PK_u||\tau_u)$

over the combined data to ensure integrity. The UE sends $\mathbf{M}_2^{HO} : (\pi_{ZK_{UE}}, com_{UE}, PK_u, \tau_u, \sigma^{UE})$ to the gNB.

Step 3: $\mathbf{M}_3^{HO} : \mathsf{PKE.Enc}(PK_u, ACK \| \sigma)$

The gNB verifies the timestamp τ_u in the received message from the UE, aborting if it is not valid. The gNB checks the signature by ensuring $1 = \mathsf{Sig.Verify}(spk_u, \sigma^{UE}, \pi_{ZK_{UE}} || com_{UE} || PK_u || \tau_u)$ and verifies the zero-knowledge proof with $1 = \mathsf{ZK.Verify}(crs, \pi_{ZK_{UE}}, com_{UE})$, aborting if any check fails. Upon successful verification, the gNB sends an acknowledgement ACK to the UE, encrypted with the UE's public key PK_u.

Step 4: $Decrypt\ \&\ Check$

The UE decrypts the received acknowledgement and verifies its contents, ensuring the message's integrity.

> Remark
>
> Although our proposed PGUS protocol has been designed for Thick MVNOs, the proposed protocols are also compatible with skinny MVNOs. Precisely, in the case of the Thick MVNO model, the gNB is operated by the MNO, while the CN is under the control of the MVNO. However, if we remove this assumption and consider a scenario where both the gNB and CN are managed by the MNO (i.e., Skinny MVNO), the proposed solution remains functional. In that case, as compared to the existing security solution of the skinny MVNO setting [4], our proposed PGUS-AKA and PGUS HO will provide additional security properties such as BSA, BSU, and GT.

Trace phase in PGUS-HO. Similar to the initial authentication, there is also a trace phase following the handover. In this stage, the use of a new public key enables the initiation of an updated accountability phase, where any subsequent actions can be traced for verifiable attribution. The UE can use the same way as AKA Trace phase to generate the trace identifier tr, and all the fixed signatures σ_{fix} are collected by gNB for the subsequent "double-spending Trace".

Double-Spending Trace. When the MVNO (CN) pays the MNO (gNB) for services provided to this MVNO's users, there is a risk of overbilling the number of the users it serves, as gNB and CN are managed by two different companies, and gNB might exaggerate the number of UEs receiving its AKA and handover services. PGUS offers the MVNO an effective method to mitigate potential overbilling. During the setup phase, the MVNO and MNO have an agreement on the number of sanitizing key pairs and the content of public keys pk_{san}^{gNB}. In the settlement phase, gNB submits all generated σ_{fix} to the MVNO, which can then perform spot checks using the $\mathsf{SBS.Trace}$ algorithm. If gNB has misused keys, i.e., used the same sk_{san}^{gNB} to generate multiple signatures $\sigma_{fix_1}, \sigma_{fix_2}$, then $\mathsf{SBS.Trace}$ will reveal the corresponding $pk_{san}^{gNB} \leftarrow \mathsf{SBS.Trace}(\sigma_{fix_1}, \sigma_{fix_2}, pk_{sig}, pk_{san}...)$. Consequently, the MNO cannot overstate the number of UEs it claims to serve.

5.4. Roaming scenario

According to 3GPP Technical Specifications TS 133 501 [50] and TS 123 502 [51], our scheme can support roaming with minor modifications. In the following, we explain it in two key phases: registration phase and roaming phase.

Registration Phase: Before enabling roaming, two service providers need to establish a connection and complete system initialization. This could be initialized while signing the cooperation contract between the two service providers. During this phase, the Visiting Public Land Mobile Network (VPLMN) authenticates with the Home Public Land Mobile Network (HPLMN) using our proposed PGUS system registration protocol. During this process, the VPLMN generates an SBS signature that roaming UEs can later verify.

Roaming Phase: When a UE moves to the VPLMN, it will initiate the PGUS-AKA because the visiting gNB contains the SBS-generated signature in the registration phase, which is issued and certified by the HPLMN. This also indicates that this country or region supports the PGUS user. Since roaming UEs require authentication at the AUSF of the HPLMN, the VPLMN forwards the ZKP and other supporting data through the N32 tunnel to obtain the authentication response from the HPLMN. By doing this, our proposed PGUS can ensure secure authentication and seamless roaming.

5.5. Integration PGUS to 5G AKA

The following outlines the integration of our proposed PGUS protocol into the conventional 5G system [50]. During the NG Setup process, which manages the authentication of the gNB, we embed Message $\mathbf{M}_1^{SR}$ into the `NGSetupRequest`. This design allows the CN to authenticate the gNB without requiring prior knowledge of the gNB's specific identity, enabling a secure initial setup phase that minimizes identity exposure and reduces potential vulnerabilities associated with identity-based attacks. Upon receiving the `NGSetupRequest` with Message $\mathbf{M}_1^{SR}$, the CN performs authentication procedures and validates the legitimacy of the gNB. Following this, the CN responds with the `NGSetupResponse` containing Message $\mathbf{M}_2^{SR}$, providing the gNB with a confirmation of successful authentication. This response establishes a trusted communication channel between the CN and gNB, ensuring that subsequent interactions are conducted within a secure framework. Upon successful verification by the CN, the gNB obtains a certificate indicating support for the PGUS extension, thereby enabling it to serve PGUS users.

In subsequent initial authentication (PGUS-AKA) and handover (PGUS-HO) processes, the UE can receive information about the gNB from $\mathbf{M}_1^{IA/HO}$ embedded within the `RRCSetup` message, allowing it to verify the authenticity of the gNB. This integration provides the UE with critical authentication data at the beginning of each connection, ensuring that only legitimate gNBs are trusted. By incorporating gNB verification into the `RRCSetup` message, this approach strengthens network security by establishing

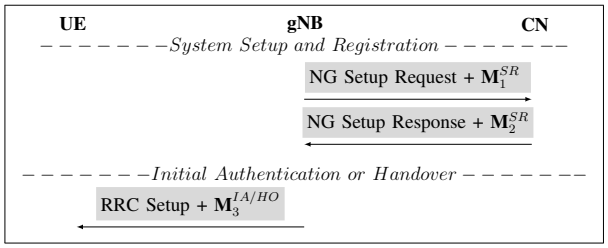

Figure 7. Integration of our PGUS AKA and HO protocols with 5G

mutual authentication early in both the initial connection setup and during handovers, effectively protecting against unauthorized entities and impersonation attacks. This integration enables authentication from UE to the gNB by embedding additional authentication-related fields into the existing `RRCSetup` message. Importantly, it maintains full compatibility with existing 5G standards by leveraging the standard-defined `nonCriticalExtension` field. Legacy devices that do not support the PGUS extension can seamlessly proceed with the standard 5G-AKA protocol. The overall integration design is illustrated in Figure 7.

Remark: Why in RRCSetup Message

One might ask why we integrate our PGUS extension into `RRCSetup` message. As defined in TS 133 501 Section 6 [50], while the standard 5G-AKA (NAS layer protocol) provides mutual authentication between the UE and CN, it fails to verify UE to the gNB in advance. To address this gap with minimal modifications to the standard 5G protocol, we integrate the PGUS extension message into the `RRCSetup` message. This design addressed security risks associated with Thick MVNO deployments while preserving the original structure of the Authentication Request/Response procedures. Specifically, in our testbed implementation, we utilize the `nonCriticalExtension` field of the `RRCSetup` message, ensuring compatibility with conventional 5G protocol specifications (more details can be found in Section 7). After RRC is completed, 5G-AKA can proceed, ensuring layer separation and 3GPP compliance. This design enhances security without modifying NAS-AKA, making PGUS scalable and compatible with existing 5G standards.

6. Security Framework And Analysis

In this section, we formalize the security of our proposed SBS and the PGUS-AKA protocol. We begin with a security proof for SBS, followed by an analysis of PGUS-AKA under the Universal Composability (UC) framework. Due to space limitations, we only provide sketches here. The experiment-based definitions of each property and full proofs of the following theorems will be available in the supplementary material [57].

6.1. Security of the SBS scheme

In this section, we will analyze the security properties of the proposed SBS scheme. The introduction of Structure-Preserving Signatures on Equivalence classes (SPSEQ), Traceable Ring Signature (TRS) and other primitives are postponed to Supplementary Material [57].

6.1.1. Security Properties of SBS. Here, we briefly recall the security properties of sanitizable signatures (SS) proposed by Bultel et al. [13]. ①*Immutability:* The sanitizer is restricted from modifying any part of the message outside the admissible blocks. ②*Transparency:* Sanitized signatures are indistinguishable from original signatures. ③*Invisibility*: The set of admissible modifications remains hidden from external observers. ④*Unlinkability:* Sanitized signatures cannot be traced back to their original sources. ⑤*Accountability*: The signer and sanitizer cannot falsely attribute signatures to one another. Our proposed SBS successfully preserves all of the above properties, while upgrading accountability to stronger notion, responsibility and exculpability.⑥*Responsibility:* The signer is accountable for their generated signatures and cannot deny authorship, and ⑦*Exculpability:* An honest party can refute false accusations of generating a signature they did not create. These two properties do not require plaintext messages to initiate accountability mechanisms. Instead, they take two signatures as input to trace suspicious parties. Finally, our SBS holds ⑧*EUF-CMA security*, it is computationally infeasible for an adversary to forge a valid signature on any new message, even after observing signatures on chosen messages. We provide formal definitions of all the properties in [57].

6.1.2. Security Result of SBS. Below, we provide the two theorems as security results of SBS, and sketch the proof of Theorem 1 and Theorem 2. For full and detailed proof, see the supplementary material [57].

Theorem 1 (Properties of SBS). *We assume that* Π_{SBS} *is in the random oracle model. If* SPSEQ *is EUF-CMA secure and perfectly adapts signatures, and* TRS *is anonymous, then the construction* Π_{SBS} *is transparent; if* PKE *is IND-CCA secure, then* Π_{SBS} *is invisible; if equivalence class relation* $\mathcal{R}$ *is class-hiding,* SPSEQ *perfectly adapt signatures, and* PKE *is correct and IND-CCA secure, and* TRS *is unforgeable, then* Π_{SBS} *is unlinkable; if* TRS *is tag-linkable, then* Π_{SBS} *is responsible; if* TRS *is exculpable, then* Π_{SBS} *is exculpable; if* SPSEQ *is EUF-CMA secure, then* Π_{SBS} *is immutable.*

Theorem 2 (EUF-CMA of SBS). *If* Π_{SBS} *holds all the security properties mentioned in Theorem 1, then it is a sanitizable signature scheme that holds EUF-CMA security.*

Sketch of Proof: We establish a reduction between the security properties of SBS and the SS construction in [13]. The proof for each property in Theorem 1 leverages a reduction to known security properties of the underlying cryptographic components used in SBS (such as SPSEQ and TRS). The proof of Theorem 2 constructs adversaries that

The Functionality of System Registration $\mathcal{F}_{Reg}$

The functionality $\mathcal{F}_{Reg}$ is parameterized by a message space $\mathcal{M}$, and it interacts with an ideal adversary $\mathcal{S}$ and parties $\mathcal{P}_1, ..., \mathcal{P}_n$ including a set of gNBs and a CN.

1) $\mathsf{message}_1$ called by $\mathcal{P}_i$

Upon receiving $(\mathsf{message}_1, sid, sgid, \mathcal{P}_i, \mathcal{P}_j, \mathbf{M}_1)$ from $\mathcal{P}_i$, it proceeds as follows:
If a tuple $(sid, sgid, \cdots)$ with the same $(sid, sgid)$ was previously recorded, do nothing.
Otherwise, record $(sid, sgid, \mathcal{P}_i, \mathcal{P}_j, \mathbf{M}_1)$ and send $(\mathsf{message}_1, sid, sgid, \mathcal{P}_i, \mathcal{P}_j)$ to $\mathcal{P}_j$ and Sim.

2) $\mathsf{message}_2$ called by $\mathcal{P}_i$

Upon receiving $(\mathsf{message}_2, sid, sgid, \mathcal{P}_i, \mathcal{P}_j, \mathbf{M}_2)$ from $\mathcal{P}_i$, it proceeds as follows:
If a tuple $(sid, sgid, \cdots)$ with the same $(sid, sgid)$ was previously recorded, record $(sid, sgid, \mathcal{P}_i, \mathcal{P}_j, \mathbf{M}_2)$ and send $(\mathsf{receipt}, sid, sgid, \mathcal{P}_i, \mathcal{P}_j)$ to $\mathcal{P}_j$ and Sim.
Otherwise, do nothing.

3) $\mathsf{corrupt}$ called by $\mathcal{S}$

Upon receiving $(\mathsf{corrupt}, sid, sgid)$ from $\mathcal{S}$, it proceeds as follows:
If there has already been an entry $(sid, sgid, \mathcal{P}_i, \mathcal{P}_j, \mathbf{M}_k)$, send $\mathbf{M}_k$ to $\mathcal{S}$. $(k \in \{1, 2\})$
If $\mathcal{S}$ provides some $\mathbf{M}_{k'}$ and $(\mathsf{receipt}, sid, sgid, \mathcal{P}_i, \mathcal{P}_j)$ has not yet been written on $\mathcal{P}_j$'s output tape, change the record to $(sid, sgid, \mathcal{P}_i, \mathcal{P}_j, \mathbf{M}_{k'})$. $(k' \in \{1, 2\})$

The Functionality of Authentication $\mathcal{F}_{Auth}$

The functionality $\mathcal{F}_{Auth}$ is parameterized by a message space $\mathcal{M}$, and it interacts with an ideal adversary $\mathcal{S}$ and parties $\mathcal{P}_1, ..., \mathcal{P}_n$ including multiple UEs, gNBs and a CN.

1) $\mathsf{message}_p$ called by $\mathcal{P}_i$, $p \in \{3, 6\}$

Upon receiving $(\mathsf{message}_p, sid, suid, \mathcal{P}_i, \mathcal{P}_j, \mathbf{M}_p)$ from $\mathcal{P}_i$, $p \in \{3, 6\}$, it proceeds as follows:
If a tuple $(sid, suid, \cdots)$ with the same $(sid, suid)$ was previously recorded, do nothing.
Otherwise, record $(sid, suid, \mathcal{P}_i, \mathcal{P}_j, \mathbf{M}_p)$ and send $(\mathsf{message}_p, sid, suid, \mathcal{P}_i, \mathcal{P}_j)$ to $\mathcal{P}_j$ and Sim.

2) $\mathsf{message}_q$ called by $\mathcal{P}_i$, $q \in \{4, 5\}$

Upon receiving $(\mathsf{message}_q, sid, suid, \mathcal{P}_i, \mathcal{P}_j, \mathbf{M}_q)$ from $\mathcal{P}_i$, it proceeds as follows:
If a tuple $(sid, suid, \cdots)$ with the same $(sid, suid)$ was previously recorded, record $(sid, suid, \mathcal{P}_i, \mathcal{P}_j, \mathbf{M}_q)$ and send $(\mathsf{receipt}, sid, suid, \mathcal{P}_i, \mathcal{P}_j)$ to $\mathcal{P}_j$ and Sim.
Otherwise, do nothing.

3) $\mathsf{corrupt}$ called by $\mathcal{S}$

Upon receiving $(\mathsf{corrupt}, sid, suid)$ from $\mathcal{S}$, it proceeds as follows:
If there has already been an entry $(sid, suid, \mathcal{P}_i, \mathcal{P}_j, \mathbf{M}_k)$, send $\mathbf{M}_k$ to $\mathcal{S}$. $(k \in \{3, 4, 5, 6\})$
If $\mathcal{S}$ provides some $\mathbf{M}_{k'}$ and $(\mathsf{receipt}, sid, suid, \mathcal{P}_i, \mathcal{P}_j)$ has not yet been written on $\mathcal{P}_j$'s output tape, change the record to $(sid, suid, \mathcal{P}_i, \mathcal{P}_j, \mathbf{M}_{k'})$. $(k' \in \{3, 4, 5, 6\})$

Figure 8. The ideal functionality of System Registration, $\mathcal{F}_{REG}$, and Initial Authentication (or handover), $\mathcal{F}_{Auth}$

would break exculpability and responsibility properties of SBS if EUF-CMA were violated.

6.2. Security of the PGUS protocol

In this section, we model the security of our proposed PGUS protocol within the Universal Composability (UC) framework under the assumption of static corruptions. The UC framework provides strong composability guarantees, ensuring that our security proofs remain valid even when our protocol is composed concurrently with an unbounded number of arbitrary protocols. To capture the desired security properties, we construct two ideal functionalities, $\mathcal{F}_{Reg}$ and $\mathcal{F}_{Auth}$. We then demonstrate UC security by showing that no static adversary can distinguish between the ideal execution and the real-world protocol execution, which means that our proposed PGUS-AKA and PGUS-HO are UC-secure.

6.2.1. UC Framework. We briefly introduce the UC framework proposed by Canetti et al. [14]. In the ideal world, "dummy" parties, potentially controlled by an "ideal process adversary" (referred to as the simulator Sim), interact directly with the ideal functionality $\mathcal{F}$. In contrast, in the real world, the parties, which may be corrupted by a real-world adversary $\mathcal{A}$, execute a protocol Π designed to emulate the behaviour of the ideal functionality $\mathcal{F}$ through interactions among the parties themselves. If for any PPT adversary $\mathcal{A}$, there exists a Sim such that no environment $\mathcal{Z}$ can distinguish between the actual protocol executions in the real world and the hypothetical executions simulated in the ideal world, then the protocol Π is UC-secure.

In UC framework, $\mathcal{F}$ expects each incoming message to contain a session identifier sid. In our executions of PGUS-AKA protocol, besides the session identifier sid, the functionalities must take two extra unique identifiers $suid$

and $sgid$, used in *Initial Authentication and handover* phase and *System Registration* phase respectively, to capture a sender sends to the same receiver multiple times within a session. We assume that the combination of them is globally unique.

We consider a static corruption model, where the adversary may choose parties to corrupt dynamically during execution. Our framework also reflects partial corruption assumptions. For instance, while the adversary may corrupt certain components (e.g., the AUSF), other critical parts (such as those generating the commitment keys ck and CRS) are assumed to remain honest. This is consistent with our thick MVNO architecture, where only specific CN components handling communication and authentication are susceptible to corruption.

6.2.2. The Ideal Functionality of PGUS protocol. In the UC framework, the ideal functionality acts as an incorruptible trusted third party, which can execute the specific functions performed by the protocol. We present two ideal functionalities, modeling the *System Registration* phase protocol as $\mathcal{F}_{Reg}$, and modeling the *Initial Authentication* phase protocol as $\mathcal{F}_{Auth}$ in Figure 8. We note that $\mathcal{F}_{Auth}$ here is the same as $\mathcal{F}_{MVNO}$ in [4], while it captures the handover phase protocol in the meantime.

$\mathcal{F}_{Reg}$ and $\mathcal{F}_{Auth}$ in Figure 8 are designed to guarantee the following security properties against any static adversary as follow: (1) Correctness: Any entity possessing internal data that satisfies the relations will successfully establish mutual authentication unless the adversary stops the process. (2) Unforgeability: Any entity that does not possess the internal data satisfying the relations cannot successfully establish mutual authentication. This prevents MitM attacks or Impersonation attacks. (3) Anonymity: Achieving UA against other entities and achieving BSA against CN. (4) Unlinkability: Under the anonymity guarantees, an adversary cannot link multiple protocol sessions to the same entity.

6.2.3. Security Result of PGUS in UC Framework. We present the theorems demonstrating that our PGUS-AKA and PGUS-HO UC-securely realize the functionalities $\mathcal{F}_{Reg}$ and $\mathcal{F}_{Auth}$.

Theorem 3. (UC realization of $\mathcal{F}_{Reg}$) *The System Registration phase of PGUS-AKA Protocol in Fig.1 UC-realizes $\mathcal{F}_{Reg}$ in the hybrid model.*

Theorem 4. (UC realization of $\mathcal{F}_{Auth}$) *The Initial Authentication phase of PGUS-AKA Protocol in Fig.1 and the handover phase of PGUS-HO protocol in Fig.2 UC-realizes $\mathcal{F}_{Auth}$ in the hybrid model.*

Theorem 4 follows directly from [4]; the only difference is that we replace the Sanit and Verify of SS with our proposed SBS. Since SBS covers all the functionalities and security properties of SS, and SBS also holds EUF-CMA security (See Theorem 2), this replacement does not change the validity of the UC security proof of [4]. The full formal proof of Theorem 3 is in supplementary material [57]. We sketch the proof of Theorem 3 as follows:

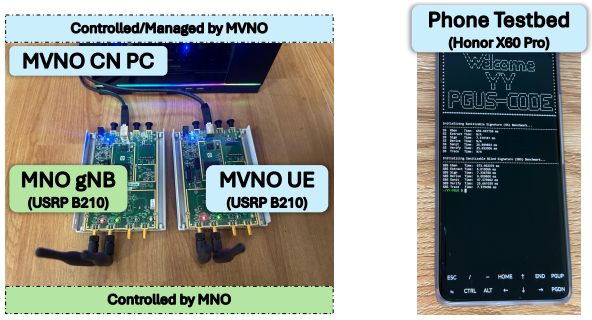

Figure 9. Testbed Setup

Sketch of Proof: In our proof, adversary $\mathcal{A}$ interacts with real parties $\mathcal{P}_i$, while the simulator Sim interfaces between $\mathcal{A}$ and the ideal functionality $\mathcal{F}_{Reg}$. A sequence of hybrid games (G_0, G_1, G_2, G_3) establishes indistinguishability between real and ideal worlds. In G_0, This game models the protocol's execution in the real world, the environment $\mathcal{Z}$ controls the $\mathcal{A}$ and see the interactions between each party $\mathcal{P}_i$, while observing adversary $\mathcal{A}$'s attempts to disrupt the protocol. In G_1, $\mathcal{A}$ controls the gNB, the Sim verifies the commitment-binding zero-knowledge proofs sent by the gNB and generates an internal signature σ_{inner} using the SBS. In G_2, $\mathcal{A}$ controls the CN, the Sim generates a commitment and simulates zero-knowledge proofs without knowing the gNB's pseudo-identifier gid. The final game G_3 is the ideal world in the CRS model, the Sim forwards messages to realize the protocol's functions, making G_3's execution indistinguishable from the ideal world model using $\mathcal{F}_{Reg}$.

7. Implementation and Evaluation

In this section, we first present the detailed setup of our testbed, which includes both an OpenAirInterface based simulation environment and a real-world evaluation using a commercial Android smartphone. We then describe the implementation of our proposed sanitizable blind signature SBS primitive and compare it with the conventional sanitizable signature scheme (SS). Following that, we present the results of our end-to-end latency and handover experiments conducted within the OpenAirInterface testbed. Finally, we demonstrate how our approach is evaluated on a commercial smartphone and the performance in the real-world phone environments.

7.1. Testbed Setup

In order to implement the PoC of the proposed system and also demonstrate its practicality here, we build a realistic testbed to test and evaluate our proposed protocol. Figure 9 shows the details of our testbed. To build the environment, we use a PC with an i9-13900k CPU to perform the experiment. we also use two USRP B210 software-defined radios (SDRs) connected to the PC running on an Ubuntu 24.04 desktop OS. To better simulate the 5G environment, we use a popular open-source 5G technique

stack called OpenAirInterface [36]. We simulate the UE, base station (gNB) and the core network. To integrate our proposed protocol into the standard 5G protocol, we modify the code of the core network and the gNB stack based on the OpenAirInterface. As shown in Figure 9, one of the USRP B210 acts as the UE and another USRP B210 acts as the 5G base station. To comprehensively evaluate the effectiveness of our proposed scheme, we implement and test the code of our proposed sanitizable blind signature SBS. In order to demonstrate the real-world effectiveness of our approach, we further include a commercial Android smartphone as part of our experimental setup. Specifically, we use a Honor X60 Pro running MagicOS 9 (Android 15), equipped with a Snapdragon 6 Gen 1 processor (4×A78 @ 2.2GHz + 4×A55 @ 1.8GHz), 12GB of RAM. Since direct modification of the baseband processor is highly restricted, we choose to use the Cortex-A78 [6] cores on the application processor to approximate the performance characteristics of a typical baseband environment. The Cortex-A78 is not the latest generation processor; it operates at a moderate frequency and does not leverage dedicated cryptographic accelerators, making it a suitable processor for evaluating our scheme under realistic, resource-constrained conditions.

	Conventional 5G	Proposed PGUS	Difference
End-To-End Cost	34.2ms	38.2ms	4ms
Handover Cost	1.4ms	3.2ms	1.8ms

Operation	KGen	Extract	Sign	Derive	Sanit	Verify	Trace
SS (ms)	150.312	-	3.471	-	8.510	8.714	-
SBS (ms)	162.951	2.371	1.474	0.053	23.123	8.903	1.027
Diff (ms)	+12.693	+2.371	-1.997	+0.053	+14.613	+0.189	+1.027

TABLE 3. EVALUATING END-TO-END AND HANDOVER COSTS IN 5G AND PGUS ON TESTBED SETUP AND TIME COMPARISON BETWEEN SANITIZABLE SIGNATURES (SS) AND OUR PROPOSED SBS.

7.2. End-to-End Cost with Crypto Evaluation

In this section, we provide a detailed evaluation of the end-to-end cost of our proposed scheme. According to 5G standards such as TS 38.413 [1], the gNB must register with the core network through the `NGSetupRequest` message, and the core network responds with the `NGSetupResponse` message. In our design, we integrate our messages $\mathbf{M}_1^{SR}$ and $\mathbf{M}_2^{SR}$ into the `NGSetup` messages. Since this procedure takes place before the gNB broadcasts the system information (SI) message, it does not affect the initial authentication process or any other real-time communication phases. To further streamline the process, we embed the certificate of the gNB and its associated signature into the `RRCSetup` message, as it can handle larger message sizes and is the last downlink message before the NAS-AKA protocol begins, making it an ideal point to introduce authentication of the gNB. Figure 7 illustrates how we incorporated our messages into the existing structure. Given that the `NGSetup` messages do not interfere with real-time communication, we measure the end-to-end time between the UE receiving the `SystemInformationBlockType1` (SIB1) message and sending the `RRCSetupComplete` message. As demonstrated in Table 3, our scheme incurs minimal overhead when compared to the conventional 5G protocol. It shows that the end-to-end time for our scheme is 38.2ms, while the conventional 5G protocol requires 34.2ms. This results in a marginal increase of 4 milliseconds, which is negligible considering the significant security enhancements and compatibility with Thick MVNOs provided by our scheme. Besides, Table 3 also provides a breakdown of the time required for each cryptographic operation. Our SBS scheme incurs an additional 12.693ms in the KGen operation and 14.613ms in the Sanit operation compared to traditional sanitizable signature schemes [13]. This is because our SBS invokes TRS twice during key generation, and it uses PKE to encrypt the index ADM in the sanitization. However, these overheads provide PGUS-AKA with Global Traceability (GT), Base Station Anonymity (BSA), and Global Unlinkability (GU), which are important security requirements for Thick MVNOs. These results highlight the low computational overhead of our scheme across all operations, making it highly suitable for integration into real-time 5G communication systems.

7.3. Handover Cost Evaluation

Here, we provide the details of our handover experiments. In this experiment, we aim to evaluate the performance difference between the conventional 5G protocol and the proposed PGUS scheme. To achieve this, we focus on the F1 handover procedure, as it is supported by the OpenAirInterface platform. We trigger the F1 handover by forcing the Distributed Unit (DU) gNB to send the `RRCReconfiguration` message. We embed the gNB's certificate and its associated signature into the `RRCReconfiguration` message, allowing the UE to verify the authenticity of the gNB during the handover process. We measure the time taken by the UE to process the `RRCReconfiguration` message, starting from the moment it is received. As shown in Table 3, our proposed PGUS scheme introduces a processing delay of 3.2ms, compared to 1.4ms in the conventional 5G protocol. This 1.8ms increase is a reasonable trade-off given the significantly enhanced security guarantees provided by PGUS.

Operation	KGen	Extract	Sign	Derive	Sanit	Verify	Trace
SS (ms)	650.037	-	7.310	-	26.899	25.653	-
SBS (ms)	673.902	6.010	7.336	0.001	97.570	25.661	7.579
Diff (ms)	+23.865	+6.010	+0.026	+0.001	+70.671	+0.008	+7.579

TABLE 4. TIME DIFFERENCE OF EACH OPERATION BETWEEN SANITIZABLE SIGNATURES (SS) AND OUR PROPOSED SBS ON COMMERCIAL PHONE.

7.4. Android Phone based Evaluation

In this section, we present our experiments conducted on a real-world device using an Android phone testbed. Due

to the restricted access to the baseband processor imposed by modern chipset vendors [40] [41], we follow a similar approach as prior state-of-the-art works [54] [58] [4] to perform our evaluation on the application processor of a commercial smartphone. Section 7.1 details the specifications of the phone, which is equipped with a Snapdragon 6 Gen 1 processor and runs MagicOS 9 (Android 15). To effectively evaluate the performance of our implementation, we use Termux [20] to set up a Linux environment on top of the phone's operating system. All supporting libraries are configured and compiled natively for the Cortex processor architecture. We compiled essential cryptographic libraries, such as RELIC [5] and PBC [39], on the phone to ensure compatibility and enable full functionality of our code. The implementation runs entirely within this environment without relying on external computation resources. Each experiment is repeated 200 times to obtain stable and statistically meaningful results. We report the average runtime and the final outcomes are summarized in Table 4.

8. Conclusion

In this article, we presented a novel cryptographic primitive SBS, which enables a sanitizable signature without revealing the plaintext message to the signer. Then, we introduced PGUS, which is composed of AKA and HO protocols for Thick MVNOs, where the security of these protocols is based on SBS. The proposed PGUS can also extended to the roaming scenarios. We provided formal proofs and implementation in a 5G environment and a real-world device. Our work is the first step towards a comprehensive privacy-preserving communication scheme in Thick MVNOs. In future, we will consider a broader range of threat models and extend the application of SBS to other scenarios that require privacy preservation and flexible information modification.

Acknowledgment

This research was supported by the National Research Foundation, Singapore and Infocomm Media Development Authority under its Future Communications Research Development Programme, under grant FCP-NUS-RG-2022-019. Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not reflect the views of National Research Foundation, Singapore and Infocomm Media Development Authority. This research is also supported by A*STAR, CISCO Systems (USA) Pte. Ltd and National University of Singapore under its Cisco-NUS Accelerated Digital Economy Corporate Laboratory (Award I21001E0002).

References

[1] 3GPP. 5g ; ng-ran; ng application protocol (ngap) (3gpp ts 38.413 version 16.10.0 release 16).

[2] Rabiah Alnashwan, Prosanta Gope, and Benjamin Dowling. Privacy-aware secure region-based handover for small cell networks in 5g-enabled mobile communication. *IEEE Transactions on Information Forensics and Security*, 18:1898–1913, 2023.

[3] Rabiah Alnashwan, Prosanta Gope, and Benjamin Dowling. Unihand: Privacy-preserving universal handover for small-cell networks in 5g-enabled mobile communication with kci resilience. In *2024 IEEE 37th Computer Security Foundations Symposium (CSF)*, pages 96–111. IEEE Computer Society, 2024.

[4] Rabiah Alnashwan, Yang Yang, Yilu Dong, Prosanta Gope, Behzad Abdolmaleki, and Syed Rafiul Hussain. Strong privacy-preserving universally composable aka protocol with seamless handover support for mobile virtual network operator. In *Proceedings of the 2024 on ACM SIGSAC Conference on Computer and Communications Security*, CCS '24, page 2057–2071, 2024.

[5] D. F. Aranha, C. P. L. Gouvêa, T. Markmann, R. S. Wahby, and K. Liao. RELIC is an Efficient LIbrary for Cryptography. https://github.com/relic-toolkit/relic.

[6] Arm Ltd. Cortex-A78: Fourth-Generation, High-Performance CPU Based on DynamIQ Technology, 2025. https://www.arm.com/products/silicon-ip-cpu/cortex-a/cortex-a78.

[7] GSM Association. The mobile economy 2022, 2022.

[8] Giuseppe Ateniese, Daniel H Chou, Breno De Medeiros, and Gene Tsudik. Sanitizable signatures. In *Computer Security–ESORICS 2005: 10th European Symposium on Research in Computer Security, Milan, Italy, September 12-14, 2005. Proceedings 10*, pages 159–177. Springer, 2005.

[9] David Basin, Jannik Dreier, Lucca Hirschi, Saša Radomirovic, Ralf Sasse, and Vincent Stettler. A formal analysis of 5g authentication. In *Proceedings of the 2018 ACM SIGSAC conference on computer and communications security*, pages 1383–1396, 2018.

[10] Michael Till Beck, Jan Camenisch, David Derler, Stephan Krenn, Henrich C Pöhls, Kai Samelin, and Daniel Slamanig. Practical strongly invisible and strongly accountable sanitizable signatures. *Cryptology ePrint Archive*, 2017.

[11] Ravishankar Borgaonkar, Lucca Hirschi, Shinjo Park, and Altaf Shaik. New privacy threat on 3g, 4g, and upcoming 5g aka protocols. *Proceedings on Privacy Enhancing Technologies*, 2019(3):108–127, 2019.

[12] Christina Brzuska, Marc Fischlin, Tobias Freudenreich, Anja Lehmann, Marcus Page, Jakob Schelbert, Dominique Schröder, and Florian Volk. Security of sanitizable signatures revisited. In *Public Key Cryptography–PKC 2009: 12th International Conference on Practice and Theory in Public Key Cryptography, Irvine, CA, USA, March 18-20, 2009. Proceedings 12*, pages 317–336. Springer, 2009.

[13] Xavier Bultel, Pascal Lafourcade, Russell WF Lai, Giulio Malavolta, Dominique Schröder, and Sri Aravinda Krishnan Thyagarajan. Efficient invisible and unlinkable sanitizable signatures. In *Public-Key Cryptography–PKC 2019: 22nd IACR International Conference on Practice and Theory of Public-Key Cryptography, Beijing, China, April 14-17, 2019, Proceedings, Part I 22*, pages 159–189. Springer, 2019.

[14] Ran Canetti. Universally composable security: A new paradigm for cryptographic protocols. In *Proceedings 42nd IEEE Symposium on Foundations of Computer Science*, pages 136–145. IEEE, 2001.

[15] Merlin Chlosta, David Rupprecht, Thorsten Holz, and Christina Popper. Lte security disabled: misconfiguration in commercial networks. In *Proceedings of the 12th conference on security and privacy in wireless and mobile networks*, pages 261–266, 2019.

[16] European Commission, Joint Research Centre, L Bargiotti, I Gielis, B Verdegem, P Smits, F Pignatelli, R Boguslawski, and D Keogh. *Guidelines for public administrations on location privacy – European Union location framework*. Publications Office, 2020.

[17] Cas Cremers and Martin Dehnel-Wild. Component-based formal analysis of 5g-aka: Channel assumptions and session confusion. In *Network and Distributed System Security Symposium (NDSS)*. Internet Society, 2019.

[18] Danny Dolev and Andrew Yao. On the security of public key protocols. *IEEE Transactions on information theory*, 29(2):198–208, 1983.

[19] Nils Fleischhacker, Johannes Krupp, Giulio Malavolta, Jonas Schneider, Dominique Schröder, and Mark Simkin. Efficient unlinkable sanitizable signatures from signatures with re-randomizable keys. *Cryptology ePrint Archive*, 2015.

[20] Fornwall and Termux contributors. Termux: Terminal emulator for android, 2025. https://github.com/termux/termux-app.

[21] Eiichiro Fujisaki. Sub-linear size traceable ring signatures without random oracles. *IEICE TRANSACTIONS on Fundamentals of Electronics, Communications and Computer Sciences*, 95(1):151–166, 2012.

[22] Eiichiro Fujisaki and Koutarou Suzuki. Traceable ring signature. In *International Workshop on Public Key Cryptography*, pages 181–200. Springer, 2007.

[23] Future Market Insights. Mobile virtual network operator (mvno) market forecast, 2034, 2024. Accessed: 2024-09-03.

[24] Hadi Givehchian, Nishant Bhaskar, Eliana Rodriguez Herrera, Héctor Rodrigo López Soto, Christian Dameff, Dinesh Bharadia, and Aaron Schulman. Evaluating physical-layer ble location tracking attacks on mobile devices. In *2022 IEEE symposium on security and privacy (SP)*, pages 1690–1704. IEEE, 2022.

[25] Grand View Research. Mobile virtual network operator (mvno) market share report, 2030, 2024. Accessed: 2024-09-03.

[26] Christian Hanser and Daniel Slamanig. Structure-preserving signatures on equivalence classes and their application to anonymous credentials. In *Advances in Cryptology–ASIACRYPT 2014: 20th International Conference on the Theory and Application of Cryptology and Information Security, Kaoshiung, Taiwan, ROC, December 7-11, 2014. Proceedings, Part I 20*, pages 491–511. Springer, 2014.

[27] Silke Holtmanns. How private is pretty good phone privacy?, 2022. https://commsrisk.com/how-private-is-pretty-good-phone-privacy/.

[28] Syed Rafiul Hussain, Mitziu Echeverria, Ankush Singla, Omar Chowdhury, and Elisa Bertino. Insecure connection bootstrapping in cellular networks: the root of all evil. In *Proceedings of the 12th conference on security and privacy in wireless and mobile networks*, pages 1–11, 2019.

[29] Abhishek Jain, Stephan Krenn, Krzysztof Pietrzak, and Aris Tentes. Commitments and efficient zero-knowledge proofs from learning parity with noise. In *International Conference on the Theory and Application of Cryptology and Information Security*, pages 663–680. Springer, 2012.

[30] Akinori Kawachi, Keisuke Tanaka, and Keita Xagawa. Concurrently secure identification schemes based on the worst-case hardness of lattice problems. *ASIACRYPT 2008*, page 372, 2008.

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

[32] Unwired Lab. Opencellid - open database of cell towers & geolocation, 2024. https://www.opencellid.org/.

[33] Telefónica UK Limited. O_2, 2024. https://www.o2.co.uk/.

[34] Z Luo, S Fu, N Crooks, S Hasan, C Maciocco, S Ratnasamy, and S Shenker. Loca: A location-oblivious cellular architecture. In *20th USENIX Symposium on Networked Systems Design and Implementation (NSDI 23)*. USENIX Association, 2023.

[35] Lily Hay Newman. A phone carrier that doesn't track your browsing or location, 2022. https://www.wired.com/story/pretty-good-phone-privacy-android/.

[36] OpenAirInterface, 2024. https://openairinterface.org/.

[37] Eric Priezkalns. Is it a good idea to offer to disguise imsis?, 2022. https://commsrisk.com/is-it-a-good-idea-to-offer-to-disguise-imsis/.

[38] PS Market Research. Mobile virtual network operator market size, share & demand forecasts, 2024-2030, 2024. Accessed: 2024-09-03.

[39] PyPBC. PyPBC is a Python wrapper for the PBC (Pairing-Based Cryptography) library. https://github.com/Jemtaly/pypbc.

[40] Inc. Qualcomm Technologies. Secure boot and image authentication. Technical report, Qualcomm Technologies, Inc., 2019. Accessed: 2025-04-16.

[41] Inc. Qualcomm Technologies. An introduction to access control on qualcomm snapdragon platforms. Technical report, Qualcomm Technologies, Inc., September 2020. Accessed: 2025-04-16.

[42] Alexander J Ross, Bradley Reaves, Yomna Nasser, Gil Cukierman, and Roger Piqueras Jover. Fixing insecure cellular system information broadcasts for good. In *Proceedings of the 27th International Symposium on Research in Attacks, Intrusions and Defenses*, pages 693–708, 2024.

[43] Erwin Jairo Sacoto Cabrera, Luis Guijarro, and Patrick Maillé. Game theoretical analysis of a multi-mno mvno business model in 5g networks. *Electronics*, 9(6):933, 2020.

[44] Paul Schmitt and Barath Raghavan. Pretty good phone privacy. In *30th USENIX Security Symposium (USENIX Security 21)*, pages 1737–1754, 2021.

[45] Claus-Peter Schnorr. Efficient signature generation by smart cards. *Journal of cryptology*, 4:161–174, 1991.

[46] Dong Hee Shin and Michael Bartolacci. A study of mvno diffusion and market structure in the eu, us, hong kong, and singapore. *Telematics and Informatics*, 24(2):86–100, 2007.

[47] Ankush Singla, Rouzbeh Behnia, Syed Rafiul Hussain, Attila Yavuz, and Elisa Bertino. Look before you leap: Secure connection bootstrapping for 5g networks to defend against fake base-stations. In *Proceedings of the 2021 ACM Asia Conference on Computer and Communications Security*, pages 501–515, 2021.

[48] sproof GmbH. The digital signature in hospitals, 2024. https://www.sproof.io/en/use-cases/industries/digital-signature-hospital-healthcare/.

[49] 3GPP TS 24.501 v16.13.0. Non-access-stratum (nas) protocol for 5g system (5gs), 2023.

[50] 3GPP TS 33.501 v16.9.0. Security architecture and procedures for 5g system, 2022.

[51] 3GPP TS 23.502 v17.7.0. Procedures for the 5g system (5gs), 2023.

[52] Dong Wang, Bo Bai, Wenbo Zhao, and Zhu Han. A survey of optimization approaches for wireless physical layer security. *IEEE Communications Surveys & Tutorials*, 21(2):1878–1911, 2018.

[53] Tao Wang, Xiang Cai, Rishab Nithyanand, Rob Johnson, and Ian Goldberg. Effective attacks and provable defenses for website fingerprinting. In *23rd USENIX Security Symposium (USENIX Security 14)*, pages 143–157, 2014.

[54] Yuchen Wang, Zhenfeng Zhang, and Yongquan Xie. {Privacy-Preserving} and {Standard-Compatible}{AKA} protocol for 5g. In *30th USENIX Security Symposium (USENIX Security 21)*, pages 3595–3612, 2021.

[55] Lance Whitney. Data privacy is a growing concern for more consumers, 2021. https://www.techrepublic.com/article/data-privacy-is-a-growing-concern-for-more-consumers/.

[56] Ben Wolford. What are the gdpr fines?, 2020. https://gdpr.eu/fines/.

[57] Y. Yang. Pgus code and supplementary material, 2025. https://github.com/YYangNUS/PGUS.

[58] Hexuan Yu, Changlai Du, Yang Xiao, Angelos Keromytis, Chonggang Wang, Robert Gazda, Y Thomas Hou, and Wenjing Lou. Aaka: An anti-tracking cellular authentication scheme leveraging anonymous credentials. In *Network and Distributed System Security (NDSS) Symposium 2024*. Internet Society, 2024.

Meta-Review

The following meta-review was prepared by the program committee for the 2025 IEEE Symposium on Security and Privacy (S&P) as part of the review process as detailed in the call for papers.

Summary

The paper proposes a sanitizable blind signature scheme designed to protect messages with sensitive information. Then, this scheme is used in designing a privacy-preserving authenticated key agreement and a handover procedure for 5G.

Scientific Contributions

- Provides a Valuable Step Forward in an Established Field.
- Addresses a Long-Known Issue.

Reasons for Acceptance

1) The paper proposes a novel blind signature scheme to protect messages with sensitive information in high-security-demand environments. Furthermore, a secure privacy-preserving authenticated key-agreement scheme and a seamless handover protocol are proposed.
2) The schemes are designed to fill the security and privacy gap under Thick MVNO 5G settings.
3) The paper provides formal security modeling and analysis of the proposed protocols in the Universal Composability (UC) framework and includes formal proof of all the security properties of the proposed signature schemes.

Noteworthy Concerns

1) The paper provides modifications and schemes to handle roaming, but they are not evaluated in implementations.