A blockchain-enabled 5G authentication scheme against DoS attacks

A blockchain-enabled 5G authentication scheme against DoS attacks

Man Chun Chow[1] and Maode Ma[1]

[1]School of Electrical and Electronic Engineering, Nanyang Technological University, Singapore

E-mail: EMDMa@ntu.edu.sg

Abstract. The 5G network is designed to accommodate the increasing number of connected devices with higher speed, lower latency, and better security. To improve the network security and robustness, the Third Generation Partnership Project (3GPP) has standardized the 5G-AKA protocol for mutually authenticating user equipment (UE), base stations, and the core network. However, since there will be more devices in the 5G network, the denial-of-service (DoS) attacks are much easier to be carried out than ever before. In this paper, we address the security issues in the 5G network by introducing the blockchain-enabled authentication and key agreement scheme for the 5G Networks. Our proposed scheme uses private blockchain to provide a distributed database to store all authentication records. Also, we exploit the trapdoor collision property of the chameleon hash function, such that the entries in the blockchain can verify the incoming authentication requests. Furthermore, a subscription concealed identifier (SUCI) is mandatory in our protocol to protect the anonymity of the devices, and ECDH is employed to generate a session key. We use Scyther Tool to verify our protocol formally, and the security analysis shows that our protocol can achieve perfect forward secrecy, device anonymity, and mutual authentication and key agreement. Our scheme is also resistant to replay attacks, and most importantly, the DoS attacks. Finally, performance evaluation shows that our scheme is efficient for both UEs and base stations with rational computational costs.

1. Introduction

In recent years, the continuous growth of mobile subscribers and connected devices has fostered an active development of the fifth-generation cellular network (5G). 5G networks aim to simultaneously connect as many devices as possible in the same network. From mobile phones, connected vehicles to the internet of things (IoT) devices, all equipment can now connect to the same 5G network with high performance. While the 5G network is providing many benefits, it also poses more security challenges. Given the potential security risks, the Third Generation Partnership Project (3GPP) has standardized an authentication and key agreement procedure 5G-AKA in TS 33.501 [1]. 5G-AKA designs to mutually authenticate and create secure connections between user equipment (UE), base stations, and the 5G core networks. However, as some of the authentication procedures in 5G-AKA are inherited from the 4G EPS-AKA [2], security issues such as desynchronization attacks, DoS attacks and perfect session key forward secrecy [3] remain unsolved. In view of these security threats, in this paper, we propose a secure blockchain-enabled authentication and key agreement scheme (SBCA) for the 5G networks. Our proposed protocol offers the following advantages: First, it prevents not only the typical network attacks such as eavesdropping, man-in-the-middle attacks, replay attacks, and IMSI-catching attacks, but also effectively prevents the DoS attacks and provides perfect session key forward and

backward secrecy. Second, it provides a distributed and safe way to store all subscription information privately using blockchain and chameleon hash function. Third, it introduces only a tiny amount of performance overheads during the UE authentication. Therefore, it is suitable for most mobile devices and base stations to enhance the security without noticeable performance degradation.

2. System and security model

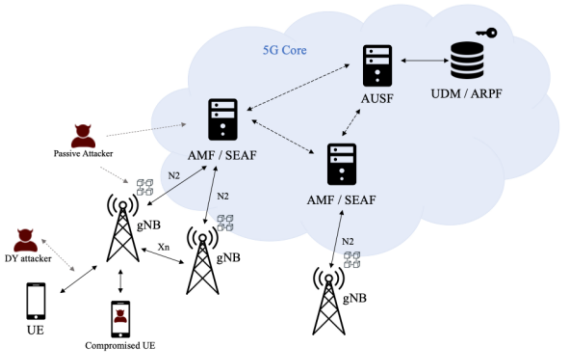

Figure 1. System and security model.

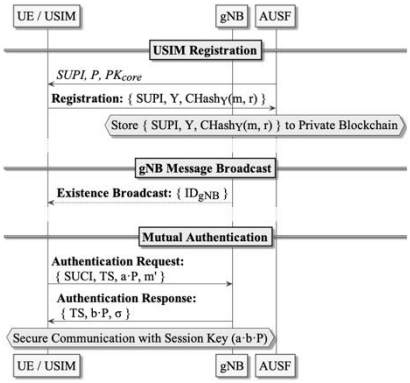

Figure 2. Sequence diagram of SBCA.

2.1. System model

Our system model follows the 3GPP 5G TS 23.501 [4] and TS 33.501 [1], with some new features added to the functional entities. In Figure 1, a private blockchain network is running across different entities in the 3GPP 5G Core (5GC) network. First, Next Generation Node B (gNB) is the base station that provides data transmission between UEs and the core network. All gNBs in the 5G network are also the blockchain nodes storing subscribers' information issued by the network operator. Second, Access and Mobility Function (AMF) / Security Anchor Function (SEAF) is the server that verifies and forwards private key related messages from the Authentication Server Function (AUSF) to gNBs. AUSF is the authentication server that manages the access rights and distributes the secret keys for gNBs to access the private blockchain. Third, Unified Data Management (UDM) / Authentication credential Repository and Processing Function (ARPF) is the server that stores private keys of the blockchain network and all other sensitive information. In the 5GC, one AUSF serves multiple AMFs across the core network, and each AMF serves multiple neighboring gNBs. For all UEs under the coverage of 3GPP access (i.e. under the signal coverage of gNBs), they can access the 5GC via the nearby gNBs. A UE uses the private information stored in their Universal Subscriber Identity Module (USIM) to initiate an authentication request. Then, the gNB compares it with the chameleon hash records saved in the private blockchain to grant or deny UE's access to the network immediately.

2.2. Security model

Our security model is also shown in Figure 1. Our proposed scheme assumes the wireless channels in the 5G environment follows the Dolev-Yao model: there could be many active and passive attackers in the surroundings. Moreover, although we assume the connections within 5GC are always trustworthy, some accidents could happen in the future: the databases stored within the functional entities in 5GC could expose to the public in some rare situations. Finally, there could also be a lot of compromised botnet UEs that could start DoS attacks at any time in the 5G network. By having all the assumptions above, our proposed protocol should achieve *mutual authentication*, *key agreement*, *session key perfect forward secrecy*, and *device anonymity*. Also, it should prevent active attacks such as rogue base station attacks, replay attacks, man-in-the-middle (MITM) attacks, and DoS attacks. Passive attacks including eavesdropping and location tracking attacks should also be prevented.

Table 1. Notations.

Notation	Description
SUPI	Subscription Permanent Identifier of the UE
ID_{gNB}	Permanent Identifier of gNB
P	Elliptic Curve Generator
Y	Chameleon Hash Key in $\mathbb{G}$
$CHash_Y$ (m, r)	Chameleon Hash based on Hash Key Y
H(M)	Cryptographic Hash Function
HMAC(M, K)	Keyed-hash Message Authentication Code
σ	HMAC code
PK_{core}	Public Key of 5GC
SK_{core}	Private Key of 5GC
TS	Timestamp
K_{hmac}	HMAC Symmetric Key
E_{PKCore} (M)	Message M encrypted with Public Key of 5GC

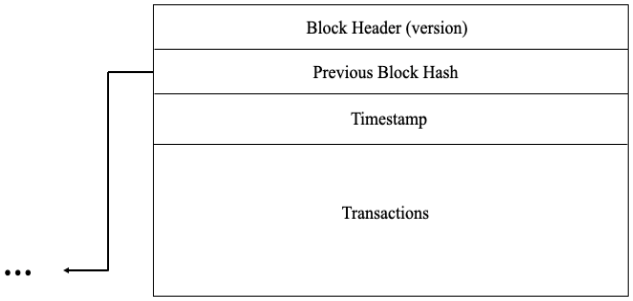

Figure 3. Structure of a block.

3. Our proposed SBCA scheme

This section discusses our proposed SBCA scheme in details. SBCA combines chameleon hash, Elliptic-curve Integrated Encryption Scheme (ECIES), Elliptic-curve Diffie-Hellman (ECDH), and keyed-hash message authentication code (HMAC) in different stages for UEs and gNBs to authenticate each other. Also, it employs a private blockchain as a distributed subscription repository. Therefore, all gNBs can approve authentication requests from UEs without the direct involvement of AUSF and ARPF. The notations used in this scheme are listed in Table 1, and Figure 2 shows the workflow with a sequence diagram. Specifically, SBCA consists of the following four phases:

Phase 1 – System Initialization: in this phase, let p be the modulus, $E(\mathbb{F}_p)$ be the elliptic curve over the finite field $\mathbb{F}_p$, P be the generator point on $E(\mathbb{F}_p)$ with order n, and $\mathbb{G}$ be the subgroup generated by multiplying the generator point P. We also let the cryptographic hash function be $H \subseteq \mathbb{Z}_n^*$. Then, AUSF and gNBs initialize following the procedures below:

1) AUSF generates an ECIES private key $SK_{core} = k$ and a public key $PK_{core} = k \cdot P$.
2) Whenever a new gNB attaches to the 5GC, AUSF takes a copy of ECIES private key SK_{core} from ARPF and install it into the secure enclave of the authenticated gNB.
3) AUSF also let the authenticated gNB to become one of the nodes in the private blockchain

For the private blockchain managed by AUSF, Figure 3 illustrates the structure of one block. For each block, it contains a header, a previous block hash, a timestamp, and multiple transactions.

Phase 2 – USIM Registration: during the production of USIM, the operator should install the chameleon hash trapdoor (k, x) and the ECIES public key of the 5GC PK_{core} to the USIM, and the chameleon hash digests to the private blockchain. In this way, UE can prove to the 5GC that it is the legitimate user by producing collisions with the trapdoor. The following shows the procedures:

1) The operator generates three random variables $m, r, n \subseteq \mathbb{Z}_n^*$. For x, it is used for generating $Y = x \cdot P$. For (m, r), it is the message for calculating a chameleon hash, i.e. $CHash_Y(m, r) = m \cdot P + r \cdot Y$. Then, the chameleon hash trapdoor (k, x) can be calculated by $k = m + rx$.
2) USIM stores the SUPI, ECC parameters, chameleon trapdoor (k, x) and PK_{core} permanently.
3) AUSF creates a new transaction including the SUPI, Y, $CHash_Y(m, r)$, timestamp, and a status code. The status code is a customizable number representing the activation status of the SUPI.

Phase 3 – gNB Broadcast: gNBs broadcast their own identities ID_{gNB} over the air freely.

Phase 4 – Mutual Authentication: UE runs the following steps to start authentication:

(1) UE → gNB: UE sends an authentication request to a gNB: **1)** Generate a random HMAC key K_{hmac}, random ECDH public key $a \cdot P$, and timestamp TS. **2)** Encrypt {SUPI, K_{hmac}} to generate a SUCI. **3)** Hash the message r' = H ($\{ID_{gNB}, SUPI, K_{hmac}, TS, a \cdot P\}$) **4)** Calculate new $m' = k - r'x$. **5)** Send the authentication request = {SUCI, TS, $a \cdot P$, m'} to gNB.

(2) gNB verifies the received authentication request with the following steps: **1)** Check the validity of the timestamp and decrypt SUCI into {SUPI, K_{hmac}} using the private key SK_{core} **2)** Calculate the r' = H ($\{ID_{gNB}, SUPI, K_{hmac}, TS, a \cdot P\}$) **3)** Search the latest transaction record of the SUPI in the private blockchain. Then, calculate $CHash_Y(m', r')' = m' \cdot P + r' \cdot Y$ **4)** Compare the calculated $CHash_Y(m', r')'$ with $CHash_Y(m, r)$ stored in the transaction. If they are equal, continue to send an authentication response. Otherwise, abort the protocol.

(3) gNB → UE: gNB sends an authentication response to UE with the following steps: **1)** Generate a random ECDH public key $b \cdot P$ **2)** Calculate σ = HMAC ({SUPI, TS, $b \cdot P$}, K_{hmac}), where TS is the received timestamp. **3)** Send the authentication response = {TS, $b \cdot P$, σ}

(4) UE verifies the received authentication response with the following steps: **1)** Calculate σ' = HMAC ({SUPI, TS, $b \cdot P$}, K_{hmac}). If σ equals to σ', calculate the ECDH session key using formula $a \cdot b \cdot P$. **2)** Similarly, gNB also calculates the ECDH session key using formula $b \cdot a \cdot P$. Since $a \cdot b \cdot P = b \cdot a \cdot P$, a session key is derived and both parties are mutually authenticated.

4. Security evaluation

4.1. Scyther tool

Scyther Tool [5] is an automatic formal verification tool for analyzing security protocols based on perfect cryptography assumption. In this section, we model our protocol with the Security Protocol Description Language (SPDL), and our formal verification result with Scyther Tool is shown in Figure 4. Our SPDL model has two roles, the gNB, and the UE. We firstly checked the secrecy of ECDH private keys using the claims of Secret a and Secret b. Then, we emulated the ECDH common key using g2(beta, a) (i.e. a · b · G) and g2(alpha, b) (i.e. b · a · G) and checked its secrecy using *SKR* claims. Moreover, to make sure that only gNB knows the randomly generated K_{hmac} from UE, we checked it with Secret Khmac claims at both parties. On the other hand, to make sure adversaries have no way to reveal the SUPI of the UE, we check the secrecy of SUPI using Secret MSIN claim. MSIN is the Mobile Subscriber Identification Number. It is the unique identifier of the USIM in SUPI. Finally, by testing all security claims including *Aliveness*, *Niagree*, *Nisynch,* and *Weakagree* in both parties, it shows that our protocol guarantees injective agreement with active initiator and responder. In other words, we can conclude that there is no attack found in our protocol.

4.2. Security analysis

Our protocol provides the following security properties:

Mutual Authentication and Key Agreement: In the SBCA, UE uses its chameleon hash trapdoor (k, x) to calculate a new message m'. Since finding a collision of a hash function without knowing the trapdoor is a computationally infeasible discrete log problem, UE effectively proves to gNB that it is a legitimate sender of the authentication request. Besides, since only the legitimate gNBs having the private key SK_{core} in their secure enclaves can read the encrypted HMAC key, gNB also proves to UE

that it is a genuine base station by issuing the authentication response using the HMAC code σ. Thus, both parties can be mutually authenticated upon the completion of the protocol.

Session Key Perfect Forward Secrecy: in SBCA, the session key is generated with the randomly generated ECDH public keys. Hence, even if all permanent keys are stolen in the future, adversaries still cannot recover the session key. Also, since the session key is totally new for every session, compromising the current session key will not affect the secrecy of the previous or future sessions.

Device Anonymity: since the SUCI is made mandatory in SBCA, the permanent identifier of requesting UEs are always concealed with ECIES encryption. Hence, eavesdroppers cannot identify a specific device or use the error messages to trace a device.

DoS Semantic Attack Prevention: semantic attacks exploiting the weaknesses of the protocol are prevented in SBCA, because the authentication request contains a timestamp and a chameleon hash message m'. By doing so, adversaries cannot hold extra sessions by replaying old authentication messages. Also, since the ECIES and chameleon hash used in authentication requests are computationally inexpensive, adversaries could not exhaust the gNBs with dummy requests easily.

DoS Flooding Attack Prevention: SBCA alleviates the effects of authentic requests flooding by decentralizing the authentication work from the AUSF/ARPF to all gNBs. As the number of gNB deployment is increasing rapidly, the total computational power of gNBs is growing, making it more difficult for adversaries to flood and paralyze the whole 5G network. The performance evaluation in the next section also shows that SBCA accommodates much more incoming requests than the conventional centralized schemes.

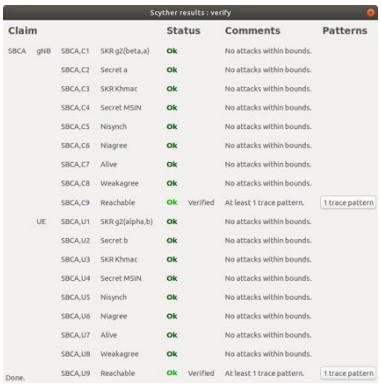

Figure 4. Formal verification with scyther tool.

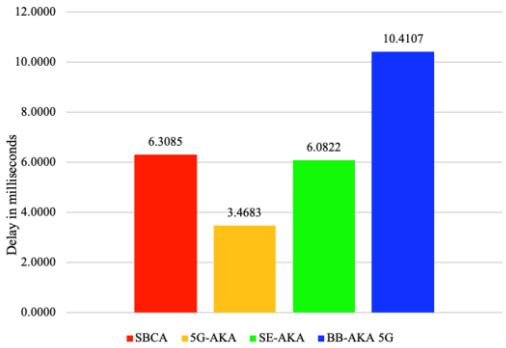

Figure 5. Experimental total computational delays.

5. Performance evaluation

In this section, 5G-AKA [2], SE-AKA [6] and blockchain-based BB-AKA 5G [7] are included to show that SBCA achieves the best in terms of balancing security and performance.

5.1. Computational overhead

All simulations are conducted on a computer with Intel® Core™ i5-3210M CPU @ 2.5 GHz CPU and 16GB of RAMs. We use Charm Crypto v0.5 [8] with Python 3.7.5 to write the simulation code, and we follow the NIST recommendation [9] to use 256-bit equivalent key strength throughout all simulations. Finally, Figure 5 shows the experimental computational overheads of different schemes. It shows that SBCA has unavoidably introduced a tiny amount of additional computational overhead (2.8402 ms more than 5G-AKA). However, with this unnoticeable delay, SBCA also provides perfect forward secrecy and DoS attack prevention that the existing 5G-AKA cannot provide.

5.2. Communication overhead

In communication overheads, we follow the 5G dense urban service requirement in TS 22.261 [10] on the wireless connection, and 1Gbps optical fiber for wired connections within 5GC. The distance

between gNB and the nearest functional entity is assumed to be 1 km, and the distance between two functional entities within the 5GC is 50 meters. We also assume the blockchain nodes can synchronize their database parallelly without significant effects on their transmission speed. Finally, the sums of propagation and transmission delays are derived in Figure 6. it shows that SBCA achieves the lowest total communication delays from 160-bit to the 384-bit key length. Although the total delay of SBCA is slightly higher than 5G-AKA by 3.43 us in the 512-bit, it is still much faster than other schemes.

5.3. Average delays under request flooding

To illustrate the delays of protocols while having many authentic requests, we model all centralized protocols (i.e. 5G-AKA and SE-AKA) to be the M/M/1 queuing model, and our SBCA to be the M/M/c queuing model. For simplicity, we assume the arrivals of requests and the locations of UEs within the 1km^2 area are under Poisson distribution. We follow the system model listed in Section 3 and assume the server-side executions of the centralized protocols to be 80% faster than the delays listed in Figure 5. The ISD of gNBs is assumed to be 200m. Consequently, each server needs to handle the requests from $1km^2/\pi 0.1^2 = 31.8 \approx 32$ gNBs. Under all these assumptions, the expected average time of completing the protocol is shown in Figure 7. It shows that SBCA is more robust in terms of accommodating more requests, and thus, it should alleviate the effects of DoS attacks much better than the existing schemes.

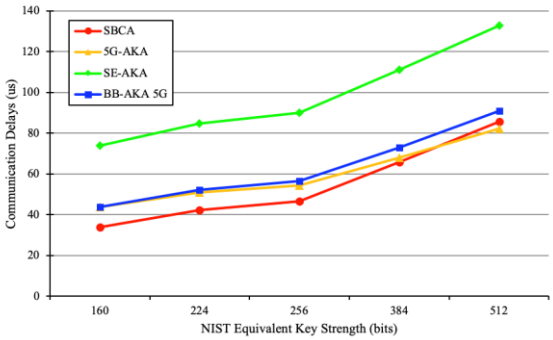

Figure 6. Total communication delays.

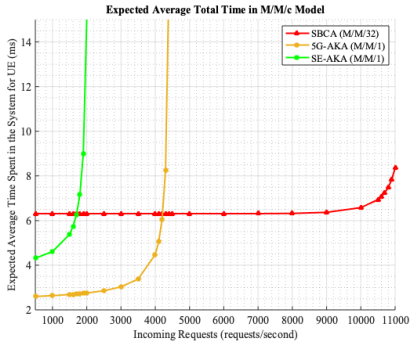

Figure 7. Average delays under request flooding.

6. Conclusion

In the 5G era, the explosive growth of mobile penetration makes DoS attacks more feasible. To secure 5G networks from DoS and other major network attacks, a Secure Blockchain-based 5G Authentication and Key Agreement (SBCA) protocol is proposed. The security evaluation has proved that SBCA is secure and resistant to various network attacks, and the performance evaluation has also shown that SBCA has improved the 5G authentication security with an optimal amount of overheads.

Acknowledgments

This work is supported by the MOE AcRF Tier 1 funding for the project of RG 26/18 by Ministry of Education, Singapore.

References

[1] 3GPP 2018 Security architecture and procedures for 5G System (Release 16.3.0); TS 33.501
[2] Basin D, Radomirovic S, Dreier J, Sasse R, Hirschi L and Stettler V 2018 A formal analysis of 5g authentication *Proc. ACM Conf. Comput. Commun. Secur.* 1383–96
[3] Cao J, Ma M, Li H, Ma R, Sun Y, Yu P and Xiong L 2020 A survey on security aspects for 3GPP 5G networks *IEEE Commun. Surv. Tutorials* **22** 170–95
[4] 3GPP 2019 System architecture for the 5G System (5GS) (Release 15.7.0); TS 23.501 353

[5] Cremers C 2020 The Scyther Tool *CISPA Helmholtz Cent. Inf. Secur.* 1
[6] Lai C, Li H, Lu R and Shen X 2013 SE-AKA: A secure and efficient group authentication and key agreement protocol for LTE networks *Comput. Networks* **57** 3492–510
[7] Haddad Z, Fouda M M, Mahmoud M and Abdallah M 2020 Blockchain-based Authentication for 5G Networks *2020 IEEE Int. Conf. Informatics, IoT, Enabling Technol. ICIoT 2020* 189–94
[8] Akinyele J A, Garman C, Miers I, Pagano M W, Rushanan M, Green M and Rubin A D 2013 Charm: a framework for rapidly prototyping cryptosystems *J. Cryptogr. Eng.* **3** 111–28
[9] Barker E 2016 Recommendation for Key Management – Part 1: General *NIST Spec. Publ. 800-57* 1–142
[10] 3GPP 2019 Technical Specification Group Services and System Aspects; Service requirements for the 5G system; Stage 1 (Release 17); TS 22.261