SUPPLEMENTAL MATERIALS FOR BLIDEHS

1 PRELIMINARY

An aggregated signature is a variant signature scheme used to aggregate any number of signatures into a single signature, thus requiring only one verification. BLS [1] is a typical aggregated signature scheme that uses curve pairing to combine the signatures of multiple senders from multiple messages into a single signature. However, BLS signature verification takes more time than conventional signature computation [2]. Recently, a short certificate-based signature scheme SCBS [3] was proposed as an efficient signature scheme with no pairing operation, short length signature and support for aggregation. The security of SCBS is based on the elliptic curve discrete logarithm problem (ECDLP).

Definition 4 (Elliptic Curves and ECDLP). Let F_p be a finite field defined over a prime p. Define E/F_p denotes the set of all points (x, y) on F_p that satisfy $y^2 = x^3 + ax = b$, where $a, b \in F_q$ and $4a^3 + 27b^2 \neq 0$. Let $\mathbb{G}$ be a cyclic subgroup of E/F_q, and g be a random generator of $\mathbb{G}$. For $\chi \in \mathbb{Z}_p^*$, χg can be caculated by $g + g + ... + g$ (χ times), which is called scalar multiplication. ECDLP means that it is infeasible to recover χ when g and χg are known.

The SCBS scheme can be defined as a seven-tuple ($Setup$, $UserKeyGen$, $UserCGen$, $Sign$, $SignVer$, $SignAgg$, $SignAggVer$) respectively representing system parameter initialization, public and private key generation of the signer, certificate generation, signing, signature verification of signer's signature by the receiver, aggregation of multiple signatures and verification.

We simplify the application of SCBS by defining the verification of the aggregate signatures of health monitoring devices (HDs) in BlideHS as a quadruplet ($Setup, Register, Sign, AggVer$) that represents the initialization of system parameters; the generation of public and private keys and certificates for HDs; the signing of the generated data by HDs; and the patient's verification of the received data and signatures.

2 THE PROPOSED SCHEME FOR BLIDEHS

2.1 System Initialization

Each PnF initializes the system parameters within the organization and registers the PT and PF identities on the PB, and then the PF is responsible for checking the reliability of the HDs and distributing certificates for them.

1) PB initialization and registration of members and HDs within the PnF: The manager Mg is elected within the PnF based on reputation or self-agreed, and is responsible for initializing the internal system of the PnF.

- $Setup_{PnF}(1^\theta) \rightarrow (pp, msk)$. Input the security parameter θ to obtain the elliptic curve parameters $(E/F_p, p, \mathbb{G})$. Mg randomly selects $sk_{Mg} \in \mathbb{Z}_p^*$ as private key, and then selects $P \in \mathbb{G}$ and computes $pk_{Mg} = sk_{Mg}P$ as public key. Mg also defines two hash functions: $Ha_0 : \{0,1\}^* \times \mathbb{G} \rightarrow \mathbb{Z}_p^*$ and $Ha_1 : \{0,1\}^* \times \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{Z}_p^*$. In PnF, the public parameter is $pp = (P, pk_{Mg}, Ha_0, Ha_1)$, and the master secret key is $msk = (sk_{Mg})$. The pp is embedded in the first block of the PB, i.e., the genesis block, as public information.
- $Deploy\&UpIndentity_{on}(pk_{Mg}, sk_{Mg}) \rightarrow Sc_{PnF}$. The pk_{Mg}, sk_{Mg} is converted by algorithms such as hash to obtain a unique address string ID_{Mg} that represents the Mg's PB account. The Mg uses this ID_{Mg} to deploy a designed smart contract on the PB, which becomes the first transaction on the blockchain. The deployed smart contract gets a unique address Sc_{PnF} on the PB. Subsequent transactions on the PB will be performed by loading the contract address Sc_{PnF}. Mg then creates on-chain accounts or updates account identities (i.e. PT or PF) for members of the organization through Sc_{PnF}'s registration and update algorithm $\texttt{UpIndentity}_{\texttt{on}}$.
- $Register_{HD}(pp, msk, HD_i, pk_{HD_i}, Sc_{PnF}) \rightarrow Cert_i$. PT's health monitoring device HD_i randomly selects $sk_{HD_i} \in \mathbb{Z}_p^*$ as the private key, calculates $pk_{HD_i} = sk_{HD_i}P$ as the public key, and then makes pk_{HD_i} public. Mg randomly selects $\gamma_i \in \mathbb{Z}_p^*$ to generate a certificate $Cert_i = (R_i, T_i)$ for HD_i by

  $$R_i = \gamma_i P, T_i = \gamma_i + sk_{Mg} Ha_0(HD_i, pk_{HD_i}).$$

  Mg then sends the $Cert_i$ to the HD_i. Mg also uploads or updates all HD_i numbers and trust profiles to the PB in batches using the algorithm $\texttt{UpHDs}_{\texttt{on}}$. This transaction can be read by the members on the PB.

2.2 Multi-source Data Aggregation Authentication

The data generated by the HDs need to be signed and authenticated in order to be recognized as valid data and thus generate a trusted EHR. In this phase, each HD_i signs the generated data using the latest certificate $Cert_i$ and its own private key sk_{HD_i}. The PT can efficiently aggregate and verify all the data in a portable device such as a mobile phone by using only the IDs and public keys of the HDs. The PT also uploads the data hash and signature trustworthiness generated by the HDs to the PB for publication.

1) HDs generate data with signatures.

- $Sign_{HD}(pp, m_i, sk_{HD_i}, pk_{HD_i}, Cert_i) \rightarrow \sigma_i$. For the generated health monitoring message $m_i \in \{0,1\}^*$, HD_i randomly selects $l_i \in \mathbb{Z}_p^*$ and then obtains the message signature $\sigma = (Z_i, U_i)$ as

  $$U_i = l_i P + R_i, Z_i = l_i + T_i + sk_{HD_i} Ha_1(m_i, pk_{HD_i}, U_i).$$

2) PT verifies aggregated data and signature of HDs.

- $AggVer_{PT}(pp, \{HD_i, pk_{HD_i}, m_i, \sigma_i\}_{i\in[n]}) \rightarrow 1/0$. For n messages $\{m_i\}_{i\in[n]}$ and signatures $\{\sigma_i\}_{i\in[n]}$ generated by HDs, PT aggregates $\{Z_i\}_{i\in[n]}$ to obtain $Z = \sum_{i=1}^{n} Z_i$. Then based on $\{HD_i\}_{i\in[n]}$ and $\{pk_{HD_i}\}_{i\in[n]}$, the algorithm verifies whether the following equation holds

  $$ZP = \sum_{i=1}^{n} U_i + (\sum_{i=1}^{n} Ha_0(HD_i, pk_{HD_i}))pk_{Mg} + \sum_{i=1}^{n} Ha_1(m_i, pk_{HD_i}, U_i)pk_{HD_i}.$$

  If the equation does not hold, it outputs 0 to indicate that the aggregated signature is incorrect, otherwise,

it outputs 1. Alternatively, it can be determined whether a device has generated unauthenticated data by verifying the following equation

$$Z_iP = U_i + Ha_0(HD_i, pk_{HD_i})pk_{Mg} + Ha_1(m_i, pk_{HD_i}, U_i)pk_{HD_i}.$$

3) PT reports the results of aggregation verification.

- $Report_{PT}(H(msg), Blist, Sc_{PnF})$. PT updates the trustworthiness of HDs at off-chain, and if the trustless number of a device reaches the upper limit, it will be blacklisted and temporarily unable to upload data to PT. PT also stores the result of an aggregation verification on the chain through the $\texttt{Report}_{\texttt{on}}$ algorithm of Sc_{PnF}, which includes the hash value $H(msg)$ of the aggregated trusted data msg and the new list of trustless devices $Blist$. Mg within PnF strictly audits the identity and certificate of the defaulted devices, then updates the certificate and reports to PB via $\texttt{UpHDs}_{\texttt{on}}$.

3 Performance Evaluation

3.1 Theoretical Analysis

TABLE 1 and TABLE 2 compare the storage overhead and computation overhead of the aggregation authentication scheme of BlideHS with [1], [4]–[6].

Let $|pp|$, $|PK_{HD}|$, $|Cert|$, $|\sigma|$ be the size of the public parameters, HD public key, certificate and signature for the aggregation authentication phase, respectively. Let $|\mathbb{Z}_p|$ and $|\mathbb{G}|$ denote the length of an element on the groups $\mathbb{Z}_p$ and $\mathbb{G}$. Define t_{e1}, t_h, t_{sm}, and t_p to denote the time consumed by a modular exponentiation operation on group $\mathbb{G}$, a MaptoPoint hash operation, a scalar multiplication, and a bilinear pairing operation, respectively. These operations are much more expensive than other cryptographic operations and need to be prioritized.

1) Aggregation authentication phase. The SCBS scheme we adopt enjoys smaller storage overhead and minimal computation overhead. Specifically, the public parameter of SCBS requires $2|\mathbb{G}|$, which is slightly larger than [1] and [4]. The $Register_{HD}$ algorithm is divided into HD generating private and public keys ($KeyGen_{HD}$) and Mg generating certificates for HD ($Cert_{Mg}$), both of which require only one t_{sm} for SCBS, and storing a public key and certificate of HD require the size of $|\mathbb{G}|$ and $|\mathbb{Z}_p| + |\mathbb{G}|$ respectively. In the $KeyGen$ phase, the computation overhead and storage overhead of the other schemes are almost equal because a modular exponentiation operation t_e and a scalar multiplication t_{sm} are close in time. In the $CertGen$ phase, [1], [4] and [6] are certificate-free implementations. But [6] has a pseudo-identity / partial key extraction process, which is an identity generation process, swe classify it in the $CertGen$ step in TABLE 2. It can be seen that the identity generation process of [6] consumes one more t_{sm} than the certificate generation process of SCBS, and the certificate storage overhead consumes $|\mathbb{Z}_p| + |\mathbb{G}|$ more than SCBS. When HD signs the message ($Sign_{HD}$), [1] and [4], which require only one $|\mathbb{G}|$ to store the signature, require a larger computation overhead $t_{e1} + t_h$. [6] has the same overhead as SCBS in the signature phase, while [5] requires one more $|\mathbb{G}|$ storage overhead. Since the $AggVer_{PT}$ phase mainly verifies whether the equation holds, we do not consider the storage overhead for this part. The aggregation verification of SCBS is the same as [6], and consumes less time and energy than [1], [4], and [5].

3.2 Experimental Analysis

1) Experiment Setting.

We design comprehensive experiments to evaluate the performance of the proposed scheme for BlideHS, all of which are performed on a PC running Ubuntu 18.04 (Intel(R) Xeon(R) E3-1230 v5 CPU @3.40GHz; 16G RAM).

We use the A-type elliptic curve with 160-bit group order in JPBC for our experiments, which is also known as $E/F_p : y^2 = x^3 + x$. In this paper, we use the Solidity[1] language to write smart contracts. For PB within the PnF organization, we use Ganache[2], a personal Ethereum blockchain deployment platform that supports the Solidity language to test smart contract functionality within PnF without transaction fees.

2) Experiment Results.

a) Multi-source Data Aggregation Authentication Efficiency. To evaluate the efficiency of BlideHS in the multi-source data authentication phase of HDs, we compared the effectiveness of the adopted SCBS with [1], [4], [5], and [6] applied to BlideHS. As shown in Fig. 1(a)-1(b), as the number of HDs devices or messages increases, the total time consumption of $KeyGen$, $CertGen$, $Sign$, $AggVer$ and the total storage overhead of $|pp|$, $|PK_{HD}|$, $|Cert|$, $|\sigma|$ for each scheme increases. We observe that SCBS consumes the least amount of time overall, while the required storage space is larger than that of the [1] and [4] schemes, which is consistent with the results of the theoretical analysis. As can be seen from Fig. 1(c), SCBS takes less than 4s overall time when the number of HDs devices or messages $n = 100$ due to its small computation overhead at each step. In Fig. 1(d), the storage space required for $|pp|$ is very small and even seems to "disappear", because the public parameters are completed at initialization and its storage overhead is not affected by changes in the number of HDs or messages.

[1] and [4] can save storage space because there is no certificate generation process, which makes other schemes appear more expensive. However, normally the $KeyGen$, $CertGen$ and $|pp|$, $|Cert|$ are obtained at initialization and $|pp|$ is embedded in the first block of the PB. From the PT's point of view, only the final aggregation authentication work is required. In other words, the storage space required to verify the data reliability ($|pk|$ and $|\sigma|$) is only slightly larger than that of [1] and [4], which have minimal overhead, while our SCBS scheme ensures that the aggregation authentication ($AggVer$) of 100 HDs devices or messages is completed within 1s.

Therefore, the multi-source data authentication phase of BlideHS is lightweight that meets the requirements of practical applications. Also, since [7] uses BLS ([1] or [4]) scheme in this phase, the efficiency of our BlideHS aggregated authentication is higher than the BHDSF proposed in [7].

1. https://soliditylang.org/
2. https://trufflesuite.com/ganache/

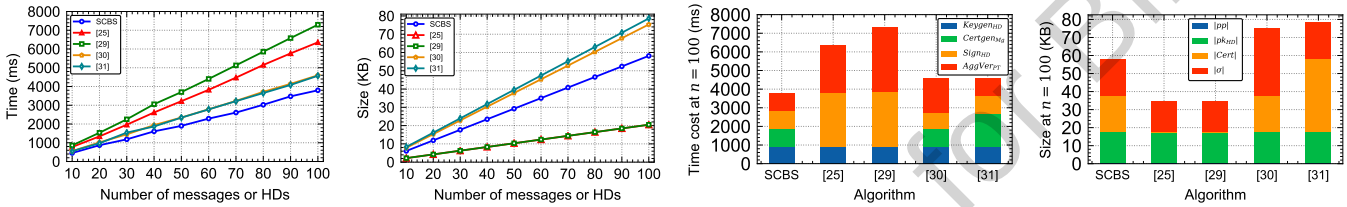
TABLE 1: Comparison of storage overhead

Phase	Type	[1]	[4]	[5]	[6]	**SCBS**
Aggregation Authentication	$\|pp\|$	$\|\mathbb{G}\|$	$\|\mathbb{G}\|$	$2\|\mathbb{G}\|$	$3\|\mathbb{G}\|$	$2\|\mathbb{G}\|$
	$\|pk_{HD}\|$	$\|\mathbb{G}\|$	$\|\mathbb{G}\|$	$\|\mathbb{G}\|$	$\|\mathbb{G}\|$	$\|\mathbb{G}\|$
	$\|Cert\|$	$\perp$	$\perp$	$\|\mathbb{Z}_p\|+\|\mathbb{G}\|$	$2(\|\mathbb{Z}_p\|+\|\mathbb{G}\|)$	$\|\mathbb{Z}_p\|+\|\mathbb{G}\|$
	$\|\sigma\|$	$\|\mathbb{G}\|$	$\|\mathbb{G}\|$	$\|\mathbb{Z}_p\|+2\|\mathbb{G}\|$	$\|\mathbb{Z}_p\|+\|\mathbb{G}\|$	$\|\mathbb{Z}_p\|+\|\mathbb{G}\|$

TABLE 2: Comparison of computation overhead

Phase	Algorithm	[1]	[4]	[5]	[6]	**SCBS**
Aggregation Authentication	$KeyGen_{HD}$	t_{e1}	t_{e1}	t_{sm}	t_{sm}	t_{sm}
	$CertGen_{Mg}$	$\perp$	$\perp$	t_{sm}	$2t_{sm}$	t_{sm}
	$Sign_{HD}$	$t_{e1}+t_h$	$t_{e1}+t_h$	t_{sm}	t_{sm}	t_{sm}
	$AggVer_{PT}$	$n(t_h+t_p)+t_p$	$n(t_{e1}+t_h+t_p)+t_p$	$(2n+2)t_{sm}$	$(n+2)t_{sm}$	$(n+2)t_{sm}$

n is the number of signed HDs or messages.



(a) Total computation overhead (b) Total storage overhead (c) Computation overhead at $n = 100$ (d) Storage overhead at $n = 100$

Fig. 1. Multi-source data aggregation authentication overhead.

TABLE 3: Smart contract gas consumed in PnF.

Algorithm	`Deploy`	`UpIndentity`$_{on}$			`UpHDs`$_{on}$`_1`			`UpHDs`$_{on}$`_10`			`Report`$_{on}$`_1`	`Report`$_{on}$`_10`
		①	②	③	①	②	③	①	②	③		
Gas consumed	385131	43718	24506	14353	25318	44530	14759	45185	218105	43593	43645	234544

①: false to true or initialize to true, ②: initialize to false, ③: true to false.

b) Blockchain Transaction Efficiency.

In the trusted PB, on-chain transactions within the PnF simulated using Ganache can be processed instantly. We measure the consumption of gas in smart contracts Sc_{PnF} designed for BlideHS.

From TABLE 3, we can see that deploying the smart contract Sc_{PnF} (i.e., `Deploy` operation) of a PB consumes a relatively large amount of gas, but `Deploy` only needs to be called once at initialization. Since `UpIdentity`$_{on}$ and `UpHDs`$_{on}$ take bool type parameters, we discuss the consumption of gas in three cases, i.e., ①: false to true or initialize to true, ②: initialize to false, and ③: true to false. We observe that converting the state to true consumes the most gas. This is due to the default value of bool type as false in Solidity. `UpHDs`$_{on}$ and `Report`$_{on}$ take array parameters and it can be seen that the consumption of gas increases with the increase of the array length.

REFERENCES

[1] D. Boneh, C. Gentry, B. Lynn, and H. Shacham, "Aggregate and verifiably encrypted signatures from bilinear maps," in *International conference on the theory and applications of cryptographic techniques*. Springer, 2003, pp. 416–432.

[2] R. Blum and T. Bocek, "Superlight–a permissionless, light-client only blockchain with self-contained proofs and bls signatures," in *2019 IFIP/IEEE Symposium on Integrated Network and Service Management (IM)*. IEEE, 2019, pp. 36–41.

[3] G. K. Verma, N. Kumar, P. Gope, B. Singh, and H. Singh, "Scbs: a short certificate-based signature scheme with efficient aggregation for industrial-internet-of-things environment," *IEEE Internet of Things Journal*, vol. 8, no. 11, pp. 9305–9316, 2021.

[4] D. Boneh, M. Drijvers, and G. Neven, "Compact multi-signatures for smaller blockchains," in *International Conference on the Theory and Application of Cryptology and Information Security*. Springer, 2018, pp. 435–464.

[5] G. K. Verma, B. Singh, N. Kumar, O. Kaiwartya, and M. S. Obaidat, "Pfcbas: Pairing free and provable certificate-based aggregate signature scheme for the e-healthcare monitoring system," *IEEE Systems Journal*, vol. 14, no. 2, pp. 1704–1715, 2019.

[6] J. Cui, J. Zhang, H. Zhong, R. Shi, and Y. Xu, "An efficient certificateless aggregate signature without pairings for vehicular ad hoc networks," *Information Sciences*, vol. 451, pp. 1–15, 2018.

[7] J. Zhang, Y. Yang, X. Liu, and J. Ma, "An efficient blockchain-based hierarchical data sharing for healthcare internet of things," *IEEE Transactions on Industrial Informatics*, vol. 18, no. 10, pp. 7139–7150, 2022.

CONTACT INFORMATION:
Zhihuang Liu
Fuzhou University
herecristliu@gmail.com