Simplification Issues of An Authentication and Key Agreement Scheme for Smart Grid

Zhengjun Cao, Lihua Liu

Abstract. Key agreement and public key encryption are two elementary cryptographic primitives, suitable for different scenarios. But their differences are still not familiar to some researchers. In this note, we show that the Safkhani et al.'s key agreement scheme [Peer-to-Peer Netw. Appl. 15(3), 1595-1616, 2022] is a public key encryption in disguise. We stress that the ultimate use of key agreement is to establish a shared key for some symmetric key encryption. We also present a simplification of the scheme by removing some repetitive computations. To the best of our knowledge, it is the first time to clarify the fundamental differences between the two primitives. The techniques developed in this note will be helpful for the future works on designing such schemes.
Keywords: Authentication; Key agreement; Public key encryption; Weak reliance; Strong reliance.

1 Introduction

The smart grid moves the energy industry into a new era of reliability, availability, and efficiency [4, 6, 10]. Its benefits include: more efficient transmission of electricity, quicker restoration of electricity after power disturbances, reduced operations and management costs for utilities, ultimately lower power costs for consumers [2], reduced peak demand, improved security [1, 9], etc. Recently, Safkhani *et al.* [8] have presented a key agreement scheme for smart grid. It is designed to meet many security requirements, such as mutual authentication, session key agreement, traceability and anonymity, message confidentiality with compromised edge device, user to user privacy, resistent to replay attack impersonation attack, secret disclosure attack, desynchronization attack, insider attack, password guessing attack, and man-in-the-middle attack.

The scheme needs to use different tools, including Public Key Infrastructure (PKI), hash function, physically unclonable functions, elliptic curve, symmetric key encryption, especially, the presence of a trust authority. We find it is a hybrid scheme, not a key agreement, as claimed. We find it has confused public key encryption and key agreement. The negligence results in many repetitive computations. Besides, we find it is a public key encryption in disguise. To the best of our knowledge, it is the first time to find such flaws in the similar literatures.

Z. Cao, Department of Mathematics, Shanghai University, Shanghai, 200444, China.
L. Liu, Department of Mathematics, Shanghai Maritime University, Shanghai, 201306, China.
Email: liulh@shmtu.edu.cn

2 Preliminaries

Key agreement, key distribution, key exchange, and key transfer [5], are often confused, but their common target is to establish a shared key between users. The resulting key in key agreement is not preexisting. However, the resulting key in key transfer is preexisting, which should be recovered intactly.

The difference between key agreement and key transfer seems unfamiliar to some researchers. For illustration purposes, we now review Diffie-Hellman key exchange [3] and RSA [7] (see Table 1).

Table 1: Diffie-Hellman key exchange versus RSA

Diffie-Hellman key exchange	RSA
Setup. A prime p, a generator $g \in \mathbb{F}_p^*$.	*Setup.* Alice picks two big primes p, q, computes $n = pq$. Pick e and compute d such that $ed \equiv 1 \bmod \phi(n)$. Set the public key as (n, e), the private key as d.
$A \rightarrow B$. Alice picks an integer x_A to compute $y_A \equiv g^{x_A} \bmod p$. Send y_A to Bob.	
$A \leftarrow B$. Bob picks an integer x_B to compute the key $k \equiv y_A^{x_B} \bmod p$, and $y_B \equiv g^{x_B} \bmod p$. Send y_B to Alice.	$A \leftarrow B$. For $m \in \mathbb{Z}_n^*$, Bob checks the certification of public key (n, e), and computes $c \equiv m^e \bmod n$. Send c to Alice.
$A \downarrow$. Alice computes the key $k \equiv y_B^{x_A} \bmod p$.	$A \downarrow$. Alice computes $m \equiv c^d \bmod n$. (Usually, m is a session key, not a raw message)
System parameters p, g should be authentic. — weak reliance, unable to determine the other party's identity	Query PKI to authenticate the public key (n, e). — strong reliance, able to determine the target receiver's identity, but unable to determine the sender's identity

Notice that RSA requires a complex system setup, which relies on Public Key Infrastructure (PKI) to enable Bob to invoke Alice's true public key. Its authentication originates directly from the reliance on PKI. Such strong reliance could be unavailable in some scenarios. Whereas, a lightweight key agreement scheme is more applicable to some weak scenarios. It is worth noting that the usual size of RSA modulus is not less than 2048 bits. Such modular exponentiation is too expensive for some devices. So, RSA is used to transfer session keys, instead of raw data.

3 Review of the Safkhani et al.'s scheme

Let $\mathcal{E}$ be an elliptic curve over the finite field F_q. $\mathbb{G}$ is an elliptic curve group with a generator P. $h(\cdot)$ is a hash function. $ES_K(\cdot), DS_K(\cdot)$ are symmetric key encryption/decryption algorithms with key K, respectively. $PUF(\cdot)$ is a physically unclonable function. The scheme [8] is performed between two users, through a trusted authority (TA). It consists of four phases: initialization phase, registration phase, login and key agreement phase, and password change phase.

TA selects $sk_{TA} \in Z_q^*$ as its secret key. The registration phase, login and key agreement phase can be described as follows (see Table 2). Since the ultimate use of the agreed key is to transfer raw data, we make up the whole procedure for comparison.

Table 2: The Safkhani et al.'s key agreement scheme

U_i: $\{ID_i, PW_i\}$	Registration	TA: $\{sk_{TA}\}$
Input identity ID_i, password PW_i.		
Pick a nonce $s_i \in Z_q^*$ to compute		Check the timestamp TS_i.
$A_i = PUF(PW_i \| s_i \| ID_i)$, $X_i = A_i \cdot P$.	$\xrightarrow[\text{[secure channel]}]{\{ID_i, X_i, TS_i\}}$	Pick $r_{TA} \in Z_q^*$ to compute
		$B_i = h(X_i \| r_{TA} \| ID_i)$,
	$\xleftarrow{\{PK_i, B_i\}}$	$Y_i = B_i \cdot P, PK_i = X_i + Y_i$.
Compute $sk_i = A_i + B_i$.		Store and distribute PK_i.
Check that $PK_i = sk_i \cdot P$.		
Compute $D_i = h(A_i \| s_i \| B_i)$.		
Store $\{s_i, B_i, D_i, PK_i\}$.		
U_i: $\{ID_i, PW_i, s_i, B_i, D_i, PK_i\}$	Key agreement & data transfer	U_j: $\{ID_j, PW_j, S_j, W_j, PK_j\}$
Compute $A_i = PUF(PW_i \| s_i \| ID_i)$.		
Check $D_i = h(A_i \| s_i \| B_i)$. If so,		
pick $r_i \in Z_q^*$ to compute		
$Z_i = r_i \cdot PK_i$, $sk_i = A_i + B_i$,		Check the timestamp TS_i.
$W_i = r_i \cdot sk_i \cdot PK_j$, $K_{ij} = h(W_i \| TS_i)$,		Compute $W_i = sk_j \cdot Z_i$,
$E_i = ES_{K_{ij}}(ID_i \| ID_j \| r_i)$,	$\xrightarrow[\text{[open channel]}]{M_1=\{L_i, E_i, Z_i, TS_i\}}$	$K_{ij} = h(W_i \| TS_i)$,
$L_i = h(Z_i \| E_i \| K_{ij} \| ID_i \| ID_j \| TS_i)$.		$DS_{K_{ij}}(E_i) = (ID_i \| ID_j \| r_i)$.
		Check $W_i = r_i \cdot sk_j \cdot PK_i$,
		and the validity of ID_j. Check if
		$L_i = h(Z_i \| E_i \| K_{ij} \| ID_i \| ID_j \| TS_i)$.
		If so, pick $r_j \in Z_q^*$ to compute
Check the validity of timestamp TS_j.	$\xleftarrow{M_2=\{L_j, Z_j, TS_j\}}$	$Z_j = r_j \cdot PK_j$, $W_j = r_j \cdot W_i$,
If so, compute $W_j = r_i \cdot sk_i \cdot Z_j$.		$L_j = h(Z_j \| W_j \| ID_j \| ID_i \| TS_j)$,
Check $L_j = h(Z_j \| W_j \| ID_j \| ID_i \| TS_j)$.		$SK_{ji} = h(ID_i \| ID_j \| W_i \| W_j \| TS_i \| TS_j)$.
If so, compute the session key		
$SK_{ij} = h(ID_i \| ID_j \| W_i \| W_j \| TS_i \| TS_j)$.		
Given the raw data $DATA$, compute		Compute the plaintext
$CT = ES_{SK_{ij}}(ID_i \| ID_j \| DATA)$.	$\xrightarrow{CT}$	$DS_{SK_{ji}}(CT) = ID_i \| ID_j \| DATA$.

Table 3: The revision of Safkhani et al.'s scheme

U_i: $\{ID_i, PW_i, s_i, B_i, D_i, PK_i\}$	Data transfer	U_j: $\{ID_j, PW_j, S_j, W_j, PK_j\}$
Input the raw data $DATA$.		
Compute $A_i = PUF(PW_i \| s_i \| ID_i)$.		
Check $D_i = h(A_i \| s_i \| B_i)$. If so,		
pick $r_i \in Z_q^*$ to compute		Check the timestamp TS_i.
$Z_i = r_i \cdot PK_i$, $sk_i = A_i + B_i$,		Compute $W_i = sk_j \cdot Z_i$,
$W_i = r_i \cdot sk_i \cdot PK_j$, $K_{ij} = h(W_i \| TS_i)$,		$K_{ij} = h(W_i \| TS_i)$,
$E_i = ES_{K_{ij}}(ID_i \| ID_j \| r_i \| DATA)$,	$\xrightarrow[\text{[open channel]}]{\{E_i, Z_i, TS_i\}}$	$DS_{K_{ij}}(E_i) = (ID_i \| ID_j \| r_i \| DATA)$.
		Retrieve PK_i with the identity ID_i.
		Check $W_i = r_i \cdot sk_j \cdot PK_i$.

4 A simplification of the Safkhani et al.'s scheme

4.1 An observation

The scheme uses symmetric key encryption/decryption $ES_{K_{ij}}(\cdot), DS_{K_{ji}}(\cdot)$ to securely transfer ID_i, ID_j and the nonce r_i, in order to protect the identities. To this end, the user U_i invokes the secret key sk_i and the other party's public key PK_j to compute

$$W_i = r_i \cdot sk_i \cdot PK_j, \quad K_{ij} = h(W_i \| TS_i)$$

where TS_i is the timestamp. The authenticity of PK_j is ensured by the TA's distribution, as the usual public key certificate issued by some authority. The heavy requirement is hardly met for many applications. Naturally speaking, the scheme is a public key encryption in disguise, not a general key agreement.

4.2 A revision

Let $DATA$ be the raw data. The whole process can be simplified (see Table 3 for the revision). It is worth noting that the revision is not a usual public key encryption, in which the sender's identity ID_i should be anonymously authenticated. This is done by retrieving the public key PK_i corresponding to the recovered identity ID_i and checking if

$$W_i = r_i \cdot sk_j \cdot PK_i,$$

due to that

$$W_i = r_i \cdot sk_j \cdot PK_i = r_i \cdot sk_i \cdot PK_j$$

The consistency-checking ensures that the sender is legitimate and authenticated by some authority.

4.3 A clarification

In a common key transfer via public key encryption, the target recipient's public key must be invoked, but the sender's public key is not involved. Therefore, the recipient cannot determine who is the true sender. Of course, an adversary who has captured the ciphertext via open channels, cannot determine both sender's and recipient's identities.

4.4 A bug

The authority TA's secret key sk_{TA} is not invoked in the registration phase at all. So does the public key PK_{TA}. It is irrational to set a public key which will never be invoked. This redundant setting is a clear bug.

5 Conclusion

In this note, we show that the Safkhani *et al.*'s key agreement scheme is flawed because it is a public key encryption in disguise. We also clarify the difference between key agreement and key transfer.

The findings could be helpful for the future works on designing such schemes.

References

[1] K. S. Adewole and V. Torra. Dftmicroagg: a dual-level anonymization algorithm for smart grid data. *Int. J. Inf. Sec.*, 21(6):1299–1321, 2022.

[2] R. Cardenas, P. Arroba, J. L. Risco-Martin, and J. M. Moya. Modeling and simulation of smart grid-aware edge computing federations. *Clust. Comput.*, 26(1):719–743, 2023.

[3] W. Diffie and M. E. Hellman. New directions in cryptography. *IEEE Trans. Inf. Theory*, 22(6):644–654, 1976.

[4] T. Docquier, Y. Song, V. Chevrier, L. Pontnau, and A. A. Nacer. Performance evaluation methodologies for smart grid substation communication networks: A survey. *Comput. Commun.*, 198:228–246, 2023.

[5] A. Menezes, P. C. Oorschot, and S. A. Vanstone. *Handbook of Applied Cryptography*. CRC Press, 1996.

[6] M. N. Nafees, N. Saxena, A. A. Cardenas, S. Grijalva, and P. Burnap. Smart grid cyber-physical situational awareness of complex operational technology attacks: A review. *ACM Comput. Surv.*, 55(10):215:1–215:36, 2023.

[7] R. L. Rivest, A. Shamir, and L. M. Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Commun. ACM*, 21(2):120–126, 1978.

[8] M. Safkhani, S. Kumari, M. Shojafar, and S. Kumar. An authentication and key agreement scheme for smart grid. *Peer-to-Peer Netw. Appl.*, 15(3):1595–1616, 2022.

[9] K. Sarieddine, M. A. Sayed, D. Jafarigiv, R. Atallah, M. Debbabi, and C. Assi. A real-time cosimulation testbed for electric vehicle charging and smart grid security. *IEEE Secur. Priv.*, 21(4):74–83, 2023.

[10] S. Vahidi, M. Ghafouri, M. Au, M. Kassouf, A. Mohammadi, and M. Debbabi. Security of wide-area monitoring, protection, and control (WAMPAC) systems of the smart grid: A survey on challenges and opportunities. *IEEE Commun. Surv. Tutorials*, 25(2):1294–1335, 2023.