A Note on "Secure Multifactor Authenticated Key Agreement Scheme for Industrial IoT"

Zhengjun Cao[1], Lihua Liu[2]

Abstract. We remark that the key agreement scheme [IEEE Internet Things J., 8(5), 2021, 3801–3811] is flawed. (1) It is insecure against internal attack, because any unauthorized sensing device (not revoked) can retrieve the final session key. (2) It could be insecure against external attack.
Keywords: Key agreement, secret sharing, internal attack, external attack.

1 Introduction

Recently, Vinoth *et al.* [1] have presented a key agreement scheme for industrial Internet of Things. The scheme makes use of password, biometrics, and smart card to identify the user, and utilizes the secret-sharing technology to construct a session key among the user and authorized sensing devices. In the proposed scenario, there are many entities: a user, the Gateway Node (GWN), n sensing devices. Its security goals include entity authentication, data confidentiality, and user anonymity. In this note, we remark that the scheme is flawed.

2 It is insecure against internal attack

To make it easier to follow the below discussion, we now depict the scheme as follows (see Table 1, or Fig.2, [1]). By the description of devices registration (see §V.B, [1]), we know, GWN will register the devices using secret-sharing technology and Chinese remainder theorem. GWN picks a unique identity ID_{SD_j} for each device SD_j, and pairwise coprime positive integers $k_1, \cdots, k_n$, where $j = 1, 2, \cdots, n$. GWN computes $\text{Mul} = \prod_{j=1}^{n} k_j, \text{Mul}_j = \text{Mul}/k_j$ and Nonce_j, s.t., $\text{Mul}_j \times \text{Nonce}_j \equiv 1 \bmod k_j$. Set

$$\gamma = \sum_{j=1}^{n} \text{Mul}_j \times \text{Nonce}_j \tag{1}$$

Note that γ *is set for the whole group of* n *devices, not for any authorized set of* $l\,(< n)$ *devices.* We find the secret γ and shares $k_j, j = 1, \cdots, n$, are not harmonically invoked. Concretely, GWN invokes γ to hide the nonce r_{GWN} as

$$M_4 = r_{GWN} \times \gamma, \tag{2}$$

[1]Department of Mathematics, Shanghai University, Shanghai, 200444, China
[2]Department of Mathematics, Shanghai Maritime University, Shanghai, 201306, China.
Email: liulh@shmtu.edu.cn

Table 1: The Vinoth *et al.*'s key agreement scheme

User U_i	Gateway Node (GWN)	Sensing Device(SD_j)
$Gen(\cdot)$, $Rep(\cdot)$ are generation and reproduction algorithms of fuzzy extractor, respectively, and $h(\cdot)$ is a hash function.	For the dealer P_0 and n devices $P_1, \cdots, P_n$, compute $x_i = \varphi(P_i), i = 0, \cdots, n$. Pick n-dimensional $Vector_1, Vector_2$, and a secret value S, s.t., $S = Vector_1 \cdot x_0, S^2 = Vector_2 \cdot x_0$. Pick $\mathrm{ID}_{\mathrm{SD}_j}$, compute $s_j = Vector_1 \cdot x_j$, $f_j = Vector_2 \cdot x_j$. Pick pairwise coprime positive integers $k_1, \cdots, k_n$. Compute $\mathrm{Mul}_j = \prod_{t=1}^{n} k_t/k_j$, Nonce_j, s.t., $\mathrm{Mul}_j \times \mathrm{Nonce}_j \equiv 1 \bmod k_j$. Set $\gamma = \sum_{j=1}^{n} \mathrm{Mul}_j \times \mathrm{Nonce}_j$. $\xRightarrow[\text{(secure channel)}]{\mathrm{ID}_{\mathrm{SD}_j}, s_j, f_j, k_j}$	
Choose ID_i, PW_i, imprint biometrics B_i. Compute $(BK_i, \tau_i) = Gen(B_i)$. Pick a nonce a, compute $TPW_i = h(\mathrm{ID}_i \Vert PW_i \Vert BK_i) \oplus a$. $\xRightarrow{\mathrm{ID}_i, TPW_i}$ Compute $RPW_i = h(\mathrm{ID}_i \Vert PW_i \Vert BK_i)$, $A'_i = A_i \oplus TPW_i \oplus RPW_i$, $C'_i = C_i \oplus TPW_i \oplus h(\mathrm{ID}_i \Vert BK_i)$, $D_i = a \oplus h(\mathrm{ID}_i \Vert BK_i)$, $V_i = h(RPW_i \Vert A_i \Vert a \Vert h(\mathrm{ID}_i \Vert BK_i)) \bmod \omega$, where ω is a medium integer to thwart online guessing attack. Store $\{TID_i, A'_i, C'_i, D_i, V_i, \tau_i, \omega\}$.	Generate the key $\mathrm{KEY}_{\mathrm{GWN}}$. Set $KEY_{GWN-U_i} = h(\mathrm{ID}_i \Vert \mathrm{KEY}_{\mathrm{GWN}})$, $A_i = KEY_{GWN-U_i} \oplus TPW_i$ $C_i = \mathrm{ID}_{\mathrm{GWN}} \oplus TPW_i$. Pick a 128-bit temporary identity TID_i. $\xLeftarrow{\{TID_i, A_i, C_i\}}$	
Pick a nonce r_i and timestamp TS_1, compute $BK_i = Rep(B_i, \tau_i)$, $RPW_i = h(ID_i \Vert PW_i \Vert BK_i)$, $ID_{GWN} = C'_i \oplus h(ID_i \Vert BK_i)$, $M_1 = A_i \oplus RPW_i \oplus r_i$, $M_2 = h(TID_i \Vert M_1 \Vert ID_{GWN} \Vert r_i \Vert TS_1)$. $\xrightarrow[\text{open channel}]{TID_i, M_1, M_2, TS_1}$ Check $\vert TS_4 - TS'_4 \vert \leq \triangle TS$. $Dec_{KEY_{GWN-U_i}}(M_{12}) = (r_{GWN}, r_i, M_9)$. Check $M_{14} = h(M_{12} \Vert M_9 \Vert r_i)$. Compute $SK = h\left(ID_i \Vert ID_{GWN} \Vert r_{GWN} \Vert r_i \Vert M_9 \Vert KEY_{KEY-U_i}\right)$ Check $M_{16} = h(SK \Vert ID_{GWN} \Vert ID_i)$. Set $TID_i^{new} = h(ID_i \Vert KEY_{GWN-U_i} \Vert TS_4) \oplus M_{13}$ Update TID_i with TID_i^{new}.	Check $\vert TS_1 - TS'_1 \vert \leq \triangle TS$. Use TID_i to look up ID_i, KEY_{GWN-U_i}, and compute $r_i = M_1 \oplus KEY_{GWN-U_i}$. Check $M_2 = h(TID_i \Vert M_1 \Vert ID_{GWN} \Vert r_i \Vert TS_1)$. If so, pick r_{GWN} and TS_2 to compute $M_4 = r_{GWN} \times \gamma, M_5 = Enc_{r_{GWN}}\left(ID_i, ID_{GWN}, r_i, r_{GWN} \oplus KEY_{GWN-U_i}\right)$, $M_6 = h\left(ID_i \Vert ID_{GWN} \Vert r_i \Vert M_4 \Vert KEY_{GWN-U_i} \Vert TS_2\right)$. $\xrightarrow{M_4, M_5, M_6, TS_2}$ Check $\vert TS_3 - TS'_3 \vert \leq \triangle TS$. Compute $Dec_{r_{GWN}}(M_8) = (s_j, f_j, ID_{SD_j})$, $\theta_1 = \sum_{t=1}^{l} \lambda_t s_t, \theta_2 = \sum_{t=1}^{l} \lambda_t f_t$. Check $\theta_2 = \theta_1^2$. Set $S = \theta_1$. $M_9 = h(S \Vert r_{GWN}), M_{10} = M_9 \times \gamma$, $M_{11} = h(M_9 \Vert M_{10})$. Generate TID_i^{new}, TS_4. $\xrightarrow{M_{10}, M_{11}}$ Compute $M_{12} = Enc_{KEY_{GWN-U_i}}(r_{GWN}, r_i, M_9)$, $M_{13} = h(ID_i \Vert KEY_{GWN-U_i} \Vert TS_4) \oplus TID_i^{new}$, $M_{14} = h(M_{12} \Vert M_9 \Vert r_i)$. $\xleftarrow{M_{12}, M_{13}, M_{14}, TS_4}$ $\xleftarrow{M_{16}}$	Check $\vert TS_2 - TS'_2 \vert \leq \triangle TS$. Compute $r_{GWN} = M_4 \bmod k_j$, $Dec_{r_{GWN}}(M_5) = \left(ID_i, ID_{GWN}, r_i, r_{GWN} \oplus KEY_{GWN-U_i}\right)$, check $M_6 = h\left(ID_i \Vert ID_{GWN} \Vert r_i \Vert M_4 \Vert r_{GWN} \oplus KEY_{GWN-U_i} \oplus r_{GWN} \Vert TS_2\right)$. If so, generate TS_3, compute $M_8 = Enc_{r_{GWN}}(s_j, f_j, ID_{SD_j})$ $\xleftarrow{M_8, TS_3}$ $\xleftarrow{\{M_8^{(SD_i)}, TS_3^{(SD_i)}\}_{SD_i \text{is in the authorized set}}}$ Compute $M_9 = M_{10} \bmod k_j$. Check $M_{11} = h(M_9 \Vert M_{10})$. Compute $SK = h\left(ID_i \Vert ID_{GWN} \Vert r_{GWN} \Vert r_i \Vert M_9 \Vert KEY_{GWN-U_i}\right)$, $M_{16} = h(SK \Vert ID_{GWN} \Vert ID_i)$

and the device SD_j invokes k_j to recover the nonce

$$r_{_{GWN}} \equiv M_4 \bmod k_j \tag{3}$$

Clearly, a corrupted device SD_s (not revoked), even unauthorized for the current session, can also recover the same nonce by computing

$$r_{_{GWN}} \equiv M_4 \bmod k_s, \tag{3'}$$

because M_4 is transported via an open channel (see the blue-colored parts, Table 1).

Using $r_{_{GWN}}$, the corrupted device can compute

$$Dec_{r_{_{GWN}}}(M_5) = (ID_i, ID_{_{GWN}}, r_i, r_{_{GWN}} \oplus KEY_{_{GWN-U_i}})$$

where M_5 is also publicly accessible, and $Dec(\cdot)$ is a symmetric key decrypting algorithm. By the recovered nonce $r_{_{GWN}}$ and the component $r_{_{GWN}} \oplus KEY_{_{GWN-U_i}}$, it is easy to recover $KEY_{_{GWN-U_i}}$. Now, all components

$$ID_i, ID_{_{GWN}}, r_{_{GWN}}, r_i, KEY_{_{GWN-U_i}}, M_9 = M_{10} \bmod k_s$$

can be obtained by the adversary for computing the final session key

$$SK = h\left(ID_i \| ID_{_{GWN}} \| r_{_{GWN}} \| r_i \| M_9 \| KEY_{_{GWN-U_i}}\right) \tag{4}$$

We want to stress that in a secret sharing scheme [2], an owner of a share is not assumed to directly use it for transporting data. The below simple relation

$$M_4 = r_{_{GWN}} \times \gamma \quad \Longrightarrow \quad r_{_{GWN}} \equiv M_4 \bmod k_j$$

is insufficient to securely transfer the nonce $r_{_{GWN}}$.

3 It could be insecure against external attack

The calculations of $M_4 = r_{_{GWN}} \times \gamma$ and $M_{10} = M_9 \times \gamma$ are actually computed over the ring $\mathbb{Z}_k$, where $k = [k_1, k_2, \cdots, k_n]$ is the lowest common multiple. Since they are pairwise coprime, $k = k_1 \times \cdots \times k_n$. In view of that the residue $r_{_{GWN}}$ modulo k_j is used as the key for $Dec(\cdot)$, the bit-length of k_j is greater than 256. In general,

$$\text{BitLength}(r_{_{GWN}}) = \text{BitLength}(h(\cdot)) = 256,$$

and $\text{BitLength}(k) \geq 256n$, such as the popular SHA-256, and AES-256. By the equations

$$\gamma = \sum_{j=1}^{n} \text{Mul}_j \times \text{Nonce}_j \bmod k, \;\; M_9 = h(S \| r_{_{GWN}}),$$

it is very likely that $r_{_{GWN}} \times \gamma < k, M_9 \times \gamma < k$. So,

$$M_4 = r_{_{GWN}} \times \gamma, \;\; M_{10} = M_9 \times \gamma \tag{5}$$

are two common equalities. An external adversary can recover the common divisor γ from M_4 and M_{10}, both are transported via open channels. Thus, $r_{_{GWN}}, M_9$ can also be exposed. Now, the adversary can compute $Dec_{r_{_{GWN}}}(M_5)$ to obtain $ID_i, ID_{_{GWN}}, r_i, r_{_{GWN}} \oplus KEY_{_{GWN-U_i}}$, which means that all components for the final hashing (see Eq.(4)) can be successfully retrieved.

4 Conclusion

We show that the Vinoth *et al.*'s key agreement scheme is insecure. It is worth noting that a key agreement scheme being integrated with secret-sharing technology could be vulnerable to internal attack. One should carefully design such a scheme and balance its security goals.

References

[1] R. Vinoth, et al., "Secure multifactor authenticated key agreement scheme for industrial iot," IEEE Internet Things J., vol. 8, no. 5, pp. 3801-3811, 2021.

[2] A. Shamir, "How to share a secret," Commun. ACM, vol. 22, no. 11, pp. 612-613, 1979.