SECURITY ANALYSIS OF SEAGKAP

1. INTRODUCTION

In this section, we prove the correctness of the protocol and provide a theoretical security proof in the Random Oracle Model. Authentication is proven secure under the EUF-CMA, and confidentiality is proven secure under the ROM-CPA. The formal analysis is further complemented using both BAN logic and the Scyther verification tool. In addition, we conduct an informal security analysis on replay resistance, forward and backward secrecy, and key consistency.

2. THEORETICAL PROOF

2.1. Authentication Proof

Assumptions. We work over a cyclic group G of prime order q with generator G. All secret keys $sk_i \in \mathbb{Z}_q^*$ (hence invertible). The hash $hash : \{0,1\}^* \to \mathbb{Z}_q$ is modeled as a random oracle. The reduction has access to a *single-signature Schnorr* signing oracle for the chosen target identity.

Honest Non-Colluding Model. We assume that all participating entities are honest and non-colluding. In particular, each participant follows the protocol specification exactly and generates its authentication messages according to the Schnorr signing algorithm using fresh randomness and its own secret key. No participant deviates from the prescribed message generation procedure, and no subset of participants colludes to forge or manipulate authentication messages. Private keys are never revealed to other participants or to the adversary.

Lemma 1. Equivalence of adjacent double-signature verification and sequential verification.

Proof. (1) Equation simplification.

Fix an honest participant P_i, with neighbors P_{i-1} and P_{i+1}. Upon receiving the first-round messages from neighbors P_{i-1} and P_{i+1}, participant P_i computes

$$S_i = s_{i+1} - s_{i-1},$$

$$X_i = r_i\big(R_{i+1} - R_{i-1}\big),$$

$$Y_i = r_i\big(h_{i+1}pub_{i+1} - h_{i-1}pub_{i-1}\big).$$

Substituting these into the verification equation

$$r_i S_i G \stackrel{?}{=} Y_i + X_i, \tag{1}$$

we obtain

$$r_i(s_{i+1}G - s_{i-1}G) \stackrel{?}{=} r_i(h_{i+1}pub_{i+1} - h_{i-1}pub_{i-1} + R_{i+1} - R_{i-1}).$$

Since $r_i \in \mathbb{Z}_q^*$ is invertible, applying the scalar multiplication r_i^{-1} to both sides yields

$$s_{i+1}G - s_{i-1}G = (R_{i+1} + h_{i+1}pub_{i+1}) - (R_{i-1} + h_{i-1}pub_{i-1}).$$

(2) Preliminaries and Notation

Let $\mathcal{T}$ denote the set of all possible protocol execution transcripts.

Let $\mathcal{T}_{honest} \subseteq \mathcal{T}$ denote the subset of transcripts generated under the honest non-colluding model, transcripts in which every participant follows the protocol specification exactly, generates authentication messages using the Schnorr signing algorithm with fresh randomness, does not collude with other participants, and never reveals its secret key.

(3) Verification Predicates

Sequential verification predicate.

Define a predicate $V_{seq} : \mathcal{T} \to \{0,1\}$ as

$$V_{seq}(\tau) = 1 \iff \big(s_{i+1}G = R_{i+1} + h_{i+1} \cdot pub_{i+1} \wedge s_{i-1}G = R_{i-1} + h_{i-1} \cdot pub_{i-1}\big),$$

Adjacent double-signature verification predicate.

Define a predicate $V_{\text{adj}} : \mathcal{T} \to \{0,1\}$ as

$$V_{adj}(\tau) = 1 \iff (s_{i+1} - s_{i-1})\,G = (R_{i+1} - R_{i-1}) + \big(h_{i+1}pub_{i+1} - h_{i-1}pub_{i-1}\big),$$

(4) Set-Theoretic Formulation

Define the following subsets of transcripts

$$A := \{\tau \in \mathcal{T} \mid V_{adj}(\tau) = 1\},$$

$$B := \{\tau \in \mathcal{T} \mid V_{seq}(\tau) = 1\}.$$

Lemma 1 claims

$$A \cap \mathcal{T}_{honest} = B \cap \mathcal{T}_{honest}.$$

Equivalently,

$$\forall \tau \in \mathcal{T}_{honest} : \quad V_{adj}(\tau) = 1 \iff V_{seq}(\tau) = 1.$$

(5) Sufficiency proof.

To prove the sufficiency, we show that $B \cap \mathcal{T}_{honest} \subseteq A \cap \mathcal{T}_{honest}$.

Let $\tau \in B \cap \mathcal{T}_{honest}$ be arbitrary. By definition of B, both following individual Schnorr verification equations hold.

$$s_{i+1}G = R_{i+1} + h_{i+1}pub_{i+1}$$

$$s_{i-1}G = R_{i-1} + h_{i-1}pub_{i-1}$$

Subtracting the second equation from the first , we obtain

$$(s_{i+1}-s_{i-1})G = (R_{i+1}-R_{i-1})+(h_{i+1}pub_{i+1}-h_{i-1}pub_{i-1}),$$

which is exactly the adjacent double-signature verification equation. Hence $V_{adj}(\tau) = 1, \tau \in A$. Since $\tau \in \mathcal{T}_{honest}$ already, we conclude $\tau \in A \cap \mathcal{T}_{honest}$.

(6) Necessity proof.

To prove the necessity, we show that $A \cap \mathcal{T}_{honest} \subseteq B \cap \mathcal{T}_{honest}$.

Let $\tau \in A \cap \mathcal{T}_{honest}$ be arbitrary. Note that under the honest non-colluding model, all first-round signatures are generated honestly. Since all first-round authentication messages are also generated according to the Schnorr signature algorithm in the honest non-colluding model, the individual Schnorr verification equations forP_{i-1}andP_{i+1}naturally hold, which is the core of$V_{seq}(\tau) = 1$.

Consequently, for any transcript $\tau \in \mathcal{T}_{honest}$, the sequential verification predicate $V_{seq}(\tau) = 1$ holds by construction. In particular, this also holds for any $A \cap \tau \in \mathcal{T}_{honest}$.

More concretely, since $\tau \in \mathcal{T}_{honest}$, by definition of the honest non-colluding model, the authentication messages of each neighbor P_j ($j \in \{i-1, i+1\}$) are generated according to the Schnorr signing algorithm. That is, for each such j, there exists $r_j \in \mathbb{Z}_q$ satisfying

$$R_j = r_j G,$$

$$h_j = hash(context_j),$$

$$s_j = r_j + h_j sk_j \pmod q,$$

$$pub_j = sk_j G.$$

Therefore, for each neighbor P_j, we have

$$\begin{aligned} s_j G &= (r_j + h_j sk_j)G \\ &= r_j G + h_j(sk_j G) \\ &= R_j + h_j pub_j. \end{aligned}$$

Hence, both individual Schnorr verification equations hold, implying $V_{seq}(\tau) = 1$, and thus $\tau \in B$. Since $\tau \in \mathcal{T}_{honest}$, we obtain $\tau \in B \cap \mathcal{T}_{honest}$.

(7) Conclusion

Combining sufficiency proof and necessity proof, we conclude

$$A \cap \mathcal{T}_{honest} = B \cap \mathcal{T}_{honest},$$

or equivalently,

$$\forall \tau \in \mathcal{T}_{honest}: \quad V_{adj}(\tau) = 1 \iff V_{seq}(\tau) = 1. \square$$

Definition 1. First-round message acceptability. If an honest party P_i, after checking Equation (1), proceeds to the second round without requesting retransmission, then P_i is said to *accept* its neighbors' first-round messages.

Theorem 1. EUF-CMA authenticity of first-round messages via reduction to *single* Schnorr. If a PPT adversary $\mathcal{A}$ forges a valid first-round message for some honest P_i with non-negligible probability, then there exists a reduction $\mathcal{F}$ that breaks the EUF-CMA security of the *single-signature* Schnorr scheme with non-negligible advantage.

Proof. Game 0. Real execution. We consider the real protocol execution, where $\mathcal{A}$ may intercept, modify, or inject first-round messages $m_j = (ID_j, R_j, s_j)$.

Game 1. Oracle setup and ROM programming. The reduction $\mathcal{F}$ samples a target identity P_{j^*} uniformly at random. All Schnorr signing queries for P_{j^*} are answered via the *single-signature* Schnorr signing oracle; all other parties are simulated honestly. Hash queries are answered using a ROM table.

Game 2. From adjacent acceptance to a fresh single-signature forgery. By Lemma 1, if $\mathcal{A}$ makes an honest neighbor P_{j+1} accept in the first round, then the two underlying *single* Schnorr verifications for the adjacent parties succeed. If *all* first-round signatures attributed to the target P_{j^*} had been obtained from the signing oracle, then $\mathcal{A}$ would not have produced any *new* valid signature for P_{j^*}. Therefore, successful acceptance implies the existence of at least one *fresh* component

$$m^*_{j^*} = (ID^*_{j^*}, R^*_{j^*}, s^*_{j^*})$$

for P_{j^*} that was never returned by the oracle. Hence, $\mathcal{F}$ outputs $(ID^*_{j^*}, R^*_{j^*}, s^*_{j^*})$ as a valid *single* Schnorr EUF-CMA forgery.

By a standard index-guessing argument and ROM programming bounds, we obtain

$$Adv^{\text{EUF-CMA}}_{\text{1st-auth}}(\mathcal{A}) \le n \cdot Adv^{\text{EUF-CMA}}_{\text{Schnorr}}(\mathcal{F}) + \frac{q_{hash}+1}{|G|} + negl(\lambda),$$

where n is the number of participants, q_H is the number of hash queries, and $|G|$ is the group order. Since the *single-signature* Schnorr scheme is EUF-CMA secure in the ROM, the success probability of $\mathcal{A}$ is negligible. $\square$

Lemma 2. Equivalence of aggregated one-shot verification and sequential verification.

Proof. (1) Equation simplification.

After receiving the second-round messages, participant P_i computes

$$T_{-i} = \sum_{j \neq i} T_j, \quad R_{-i} = \sum_{j \neq i} R_j, \quad W_{-i} = \sum_{j \neq i} H_j \cdot pub_j.$$

Then P_i checks the aggregated one-shot equation

$$T_{-i} \cdot pub_i \stackrel{?}{=} sk_i \cdot (R_{-i} + W_{-i}).$$

Since $pub_i = sk_i \cdot G$ and $sk_i \in \mathbb{Z}_q^*$ is invertible, applying the scalar multiplication sk_i^{-1} to both sides yields

$$T_{-i} \cdot G \stackrel{?}{=} R_{-i} + W_{-i}. \tag{2}$$

(2) Preliminaries and Notation.

Let $\mathcal{T}$ denote the set of all possible protocol execution transcripts.

Let $\mathcal{T}_{honest} \subseteq \mathcal{T}$ denote the subset of transcripts generated under the honest non-colluding model, where every participant follows the protocol specification exactly and generates its second-round authentication message (ID_j, X_j, R_j, t_j) according to the Schnorr signing algorithm.

(3) Verification Predicates.

Sequential verification predicate.

Define a predicate $V_{\text{seq2}} : \mathcal{T} \to \{0, 1\}$ as

$$V_{\text{seq2}}(\tau) = 1 \iff \bigwedge_{j \neq i} \left(t_j G = R_j + H_j \cdot pub_j\right),$$

all individual second-round Schnorr verification equations hold for all $j \neq i$.

Aggregated one-shot verification predicate.

Define a predicate $V_{\text{agg}} : \mathcal{T} \to \{0, 1\}$ as

$$V_{\text{agg}}(\tau) = 1 \iff T_{-i}G = R_{-i} + W_{-i},$$

the simplified aggregated equation Eq. (2) holds.

(4) Set-Theoretic Formulation.

Define the following subsets of transcripts

$$A := \{\tau \in \mathcal{T} \mid V_{\text{agg}}(\tau) = 1\},$$

$$B := \{\tau \in \mathcal{T} \mid V_{\text{seq2}}(\tau) = 1\}.$$

Lemma 2 claims

$$A \cap \mathcal{T}_{\text{honest}} = B \cap \mathcal{T}_{\text{honest}}.$$

Equivalently,

$$\forall \tau \in T_{\text{honest}} : \quad V_{\text{agg}}(\tau) = 1 \iff V_{\text{seq2}}(\tau) = 1.$$

(5) Sufficiency proof.

To prove the sufficiency, we show that $B \cap \mathcal{T}_{\text{honest}} \subseteq A \cap \mathcal{T}_{\text{honest}}$.

Let $\tau \in B \cap \mathcal{T}_{\text{honest}}$ be arbitrary. By definition of B, for every $j \neq i$, we have

$$t_j G = R_j + H_j \cdot pub_j.$$

Summing the above equalities over all $j \neq i$ and using the linearity of scalar multiplication in $\mathbb{G}$, we obtain

$$\sum_{j \neq i} t_j G = \sum_{j \neq i} R_j + \sum_{j \neq i} H_j \cdot pub_j. \tag{3}$$

According to the definitions

$$T_{-i} = \sum_{j \neq i} t_j, R_{-i} = \sum_{j \neq i} R_j, W_{-i} = \sum_{j \neq i} H_j \cdot pub_j,$$

Eq. (3) becomes $T_{-i}G = R_{-i} + W_{-i}$, which is exactly one-shot verification equation. Hence $V_{adj}(\tau) = 1, \tau \in A$. Since $\tau \in \mathcal{T}_{honest}$ already, we conclude $\tau \in A \cap \mathcal{T}_{honest}$.

(6) Necessity proof.

To prove the necessity, we show that $A \cap T_{\text{honest}} \subseteq B \cap T_{\text{honest}}$.

Let $\tau \in A \cap T_{\text{honest}}$ be arbitrary. Note that under the honest non-colluding model, all first-round signatures are generated honestly. Since all second-round authentication messages are also generated according to the Schnorr signature algorithm in the honest non-colluding model, the individual Schnorr verification equations for$P_j (j \neq i)$naturally hold, which is the core of$V_{seq}(\tau) = 1$.

Consequently, for any transcript $\tau \in \mathcal{T}_{honest}$, the sequential verification predicate $V_{seq}(\tau) = 1$ holds by construction. In particular, this also holds for any $A \cap \tau \in \mathcal{T}_{honest}$.

More concretely, since $\tau \in \mathcal{T}_{honest}$, by definition of the honest non-colluding model, each participant $P_j (j \neq i)$ generates its second-round authentication message according to the Schnorr signing algorithm. That is, for each such j, there exists $r_j \in \mathbb{Z}_q$ satisfying

$$R_j = r_j G,$$

$$h_j = hash(context_j),$$

$$s_j = r_j + h_j sk_j \pmod q,$$

$$pub_j = sk_j G.$$

Therefore, for each $j \neq i$, we have

$$\begin{aligned} t_j G =& \left(r_j + H_j sk_j\right) G \\ =& r_j G + H_j \left(sk_j G\right) \\ =& R_j + H_j pub_j, \end{aligned}$$

so all individual second-round Schnorr verification equations hold. Hence $V_{\text{seq2}}(\tau) = 1, \tau \in B$. Since $\tau \in T_{\text{honest}}$, we obtain $\tau \in B \cap T_{\text{honest}}$.

(7) Conclusion Combining Sufficiency proof and Necessity proof, we conclude

$$A \cap \mathcal{T}_{honest} = B \cap \mathcal{T}_{honest},$$

or equivalently,

$$\forall \tau \in \mathcal{T}_{honest} : \quad V_{agg}(\tau) = 1 \iff V_{seq2}(\tau) = 1. \square$$

Theorem 2. EUF-CMA authenticity of second-round messages. If a PPT adversary $\mathcal{A}$ makes an honest P_i accept the second-round aggregate verification with non-negligible probability, then there exists a reduction $\mathcal{F}$ that breaks the EUF-CMA security of the *single-signature* Schnorr scheme with non-negligible advantage.

Proof. Game 0 Real execution. $\mathcal{A}$ interacts with honest parties and may intercept, modify, and inject second-round messages M_j.

Game 1. Oracle setup and ROM programming. The reduction $\mathcal{F}$ picks a target identity P_{j^*} uniformly at random. All Schnorr signing queries for P_{j^*} are answered via the *single-signature* Schnorr signing oracle; other parties are simulated honestly. Hash queries are answered using a ROM table.

Game 2. From aggregate acceptance to a fresh single-signature forgery. Suppose $\mathcal{A}$ succeeds in making some honest P_i accept the aggregated check Eq. (2). By Lemma 2, this implies that, for every included index $j \neq i$, the single-signature check $t_j \cdot G \stackrel{?}{=} R_j + H_j \cdot pub_j$ holds. If all such signatures for the target P_{j^*} had been obtained from the signing oracle, then $\mathcal{A}$ would not have produced any *new* valid signature for P_{j^*}. Therefore, successful aggregate acceptance implies the existence of at least one *fresh* component

$$(ID^*_{j^*}, X^*_{j^*}, R^*_{j^*}, t^*_{j^*})$$

for P_{j^*} that was never returned by the signing oracle. Thus, $\mathcal{F}$ outputs $(ID^*_{j^*}, X^*_{j^*}, R^*_{j^*}, t^*_{j^*})$ as a valid *single* Schnorr EUF-CMA forgery.

By a standard index-guessing argument and ROM programming bounds, we obtain

$$Adv^{\text{EUF-CMA}}_{\text{2nd-auth}}(\mathcal{A}) \;\leq\; n \cdot Adv^{\text{EUF-CMA}}_{\text{Schnorr}}(\mathcal{F}) + \frac{q_{Hash}+1}{|G|} + negl(\lambda),$$

where n is the number of participants, q_H is the number of hash queries, and $|G|$ is the group order. Since the single-signature Schnorr scheme is EUF-CMA secure in the ROM, the success probability of $\mathcal{A}$ is negligible. □

2.2. Confidentiality Proof

Theorem 3. Session-key confidentiality under the CDH assumption in the ROM-CPA model. Under the Computational Diffie–Hellman (CDH) assumption, the proposed protocol achieves indistinguishability of the session key in the ROM-CPA model. Formally, for any PPT adversary $\mathcal{A}$,

$$\left|\Pr[\mathcal{A}^{\mathsf{Real}} = 1] - \Pr[\mathcal{A}^{\mathsf{Random}} = 1]\right| \leq \mathsf{negl}(\lambda).$$

CDH Assumption. Let G be an elliptic-curve group of prime order q with generator G. Given $(G, a \cdot G, b \cdot G)$ for unknown $a, b \in \mathbb{Z}^*_q$, no PPT adversary can compute $ab \cdot G$ with non-negligible probability. That is,

$$Adv^{\mathsf{CDH}}_{\mathcal{A}}(\lambda) = \Pr\big[\mathcal{A}(G, a \cdot G, b \cdot G) = ab \cdot G\big] \leq \mathsf{negl}(\lambda).$$

Proof. We prove by a sequence of games $G0$–$G3$.

Game $G0$. Real execution. All parties $\{P_1, P_2, \ldots, P_n\}$ execute the protocol honestly and derive the real group session key K. In the challenge phase, $\mathcal{A}$ is given either the real K or a random key. Let $p_0 = \Pr[b' = 1|S_0]$ denote the probability that $\mathcal{A}$ guesses correctly in $G0$.

Game $G1$. Replacing two ephemeral public keys. Replace the first-round messages R_{i-1}, R_i of parties P_{i-1} and P_i by two random group elements $R'_{i-1}, R'_i \in G$. Implicitly there exist unique $r'_{i-1}, r'_i \in \mathbb{Z}^*_q$ such that $R'_{i-1} = r'_{i-1} \cdot G$ and $R'_i = r'_i \cdot G$. Since R_{i-1} and R_i are uniformly distributed group elements, replacing them with independently sampled random elements does not change the adversary's view. Thus, $|p_0 - p_1| = 0$, where $p_1 = \Pr[b' = 1|S_1]$.

Game $G2$. Replacing the CDH term. In the shared secret K, the contribution of P_{i-1} and P_i involves a CDH term of the form $ab \cdot G$ (where $R'_{i-1} = b \cdot G$, $R'_i = a \cdot G$). The session key can be written as

$$K_i = nab \cdot G + \Theta,$$

$$\Theta = (n+1) \cdot X_i + (n+2) \cdot X_{i+1} + ... + X_{i-2},$$

where Θ collects all remaining terms independent of $ab \cdot G$. Since $ab \cdot G$ cannot be computed under the CDH assumption, we replace $nab \cdot G$ with a random group element $T \in G$, yielding

$$K'_i = T + \Theta.$$

If $\mathcal{A}$ can distinguish between K_i and K'_i, then the challenger can extract $ab \cdot G = n^{-1}T$, contradicting the CDH assumption. Hence, $|p_1 - p_2| \leq \mathsf{negl}(\lambda)$, where $p_2 = \Pr[b' = 1|S_2]$.

Game $G3$. All ephemeral keys replaced. Replace all parties' ephemeral public keys R_j by independent random elements $R'_j \in G$, without storing the corresponding secrets r_j. The group session key now becomes a uniformly random string K_{rand}. Since (R'_j) are identically distributed as $(r_j \cdot G)$ and the key is already replaced by random, we have $|p_2 - p_3| = 0$, where $p_3 = \Pr[b' = 1|S_3] = 1/2$.

Conclusion. By the triangle inequality,

$$|p_0 - 1/2| \leq |p_0 - p_1| + |p_1 - p_2| + |p_2 - p_3| \leq \mathsf{negl}(\lambda).$$

Therefore, under the CDH assumption, the proposed protocol achieves session-key confidentiality in the ROM-CPA. □

3. FORMAL SECURITY ANALYSIS

3.1. BAN Logic Analysis

BAN logic is employed as a formal deductive framework to analyze the authentication and key agreement properties of the proposed protocol. The analysis proceeds in four stages. First, the security goals are explicitly defined according to the

security properties claimed by the protocol. Second, an idealized model of the protocol is constructed by abstracting the protocol messages into BAN logic expressions while omitting low-level cryptographic operations. Third, a set of initial assumptions is established based on the protocol design, system model, and trust relationships among the participating entities. Finally, logical inference rules of BAN logic are applied to derive the specified goals. If a goal can be successfully derived, the corresponding security property is considered to be satisfied.

It is worth noting that BAN logic focuses on belief establishment and authentication reasoning, rather than algebraic correctness or bitwise key equality. Cryptographic operations such as Schnorr signatures are abstracted as ideal authentication primitives.

The following logical rules existed in BAN logic are used in the proof.

Jurisdiction rule:

$$\frac{P \mid\equiv Q \rightarrow X,\ P \mid\equiv Q \mid\equiv X}{P \mid\equiv X} \tag{4}$$

Message-Meaning rule:

$$\frac{P \mid\equiv \overset{K}{\longleftrightarrow} P,\ P \triangleleft \{X\}_K}{P \mid\equiv Q \mid\sim X} \tag{5}$$

Nonce Verification rule:

$$\frac{P \mid\equiv \sharp(X),\ P \mid\equiv Q \mid\sim X}{P \mid\equiv Q \mid\equiv X} \tag{6}$$

The proof process and result of SEAGKAP is as follow.

Formal Proof Goal. The goal of SEAGKAP is for all legitimate participants $P_1, ..., P_n$ to agree on the same session key after two rounds of interaction, and to achieve mutual authentication. This formal analysis aims to demonstrate the security and authenticity of SEAGKAP. To this end, we design four proof goals, presented as BAN logic definitions.

Let P_i and P_j $(i \neq j)$ be any two distinct members of the group.

Goal1: $P_i| \equiv P_i \overset{Key}{\longleftrightarrow} P_j$

Goal2: $P_j| \equiv P_i \overset{Key}{\longleftrightarrow} P_j$

Goal3: $P_i| \equiv P_j| \equiv P_i \overset{Key}{\longleftrightarrow} P_j$

Goal4: $P_j| \equiv P_i| \equiv P_i \overset{Key}{\longleftrightarrow} P_j$

These goals collectively capture mutual authentication and shared-session-key belief among group members.

Idealization Model. An essential step in the formal analysis is the idealization of the protocol. This standardized description allows BAN logic to recognize the protocol, forming the goal derivation. BAN logic simplifies the process by focusing on variable transmission between entities and ignoring internal parameter conversion. The modeling results of SEAGKAP are as follows:

Let $\{\cdot\}_{Pub_i}$ denote a signature-authenticated message that can be verified using the certificate public key of participant P_i. The timestamp T, derived from the protocol-defined $Time_i$, is used to ensure message freshness. The protocol messages are idealized as follows:

Message 1. $P_i \rightarrow P_{i\pm1}$: m_1 is formalized as
$P_{i\pm1} \triangleleft \{\mathrm{ID}_i, R_i, T\}_{Pub_i}$

Message 2. $P_i \rightarrow P_j((\forall j \neq i))$: m_1 is formalized as
$P_j \triangleleft \{\mathrm{ID}_i, X_i, R_i, T\}_{Pub_i} \quad (\forall j \neq i)$

Formal Assumption. According to the protocol description, the following security assumptions are made for our proposed scheme before the execution of the protocol.

A1: $P_i \mid\equiv CA \Rightarrow (pub_k \mapsto ID_k)$
A2: $P_i \mid\equiv CA \mid\equiv (pub_k \mapsto ID_k)$
A3: $P_i \mid\equiv \#(T)$
A4: $P_i \mid\equiv \#(R_k)$
A5: $P_i \mid\equiv \#(X_k)$
A6: $P_i \mid\equiv P_k \Rightarrow (R_k, X_k)$
A7: $P_i \mid\equiv Key = \mathsf{KDF}(f(R_1, \ldots, R_n, X_1, \ldots, X_n))$
A8: $P_i \mid\equiv \#(T) \ \Rightarrow\ P_i \mid\equiv \#(Key)$

The freshness of protocol inputs is guaranteed by timestamp-bound Schnorr signatures, which ensures that accepted messages correspond to the current protocol session.

Derivation Proof. After completing the steps of goal formulation, idealized modeling, and state assumption, we proceed with the formal proof of the protocol. This involves the comprehensive use of BAN logic, and the specific derivation process is detailed below.

For any $i \neq j$, combining Assumptions A1 and A2 and applying the jurisdiction rule Eq. (4), participant P_j believes that the public key of P_i correctly binds to its identity

$$P_j \mid\equiv (\mathrm{pub}_i \leftrightarrow \mathrm{ID}_i)\,.$$

Upon receiving Message 2 and applying the Message-Meaning rule Eq. (5), P_j obtains

$$P_j \mid\equiv P_i \mid\sim (\mathrm{ID}_i, X_i, R_i, T)\,.$$

Given the freshness of T in the Assumptions A3, applying the Nonce Verification rule Eq. (6) yields

$$P_j \mid\equiv P_i \mid\equiv (\mathrm{ID}_i, X_i, R_i)\,. \tag{7}$$

Since Eq. (7) holds for all group members, and all participants derive the session key from the same authenticated and fresh parameter set $\{(\mathrm{ID}_k, X_k, R_k, T)\}_{k=1}^{n}$ and believe in the correctness of the key derivation function in the Assumptions A7. The Goal1

$$P_i| \equiv P_i \overset{Key}{\longleftrightarrow} P_j$$

and the Goal2

$$P_j| \equiv P_i \overset{Key}{\longleftrightarrow} P_j$$

are satisfied.

From Eq. (1), we have

$$P_j \mid\equiv P_i \mid\equiv (\mathrm{ID}_i, X_i, R_i)$$

Symmetrically, participant P_i can also obtain the following belief from Message 2 sent by P_j, P_i derives

$$P_i \mid\equiv P_j \mid\equiv (\mathrm{ID}_j, X_j, R_j)\,. \tag{8}$$

Because all participants successfully verify the second-round messages and derive the session key from the same authenticated set $\{(\mathrm{ID}_k, X_k, R_k, T)\}_{k=1}^{n}$, From Eq. (8), P_i believes that P_j derives the same session key based on the same set of fresh parameters,

$$P_i| \equiv P_j| \equiv P_i \overset{Key}{\longleftrightarrow} P_j,$$

which establishes Goal3. By symmetry, we can similarly obtain

$$P_j| \equiv P_i| \equiv P_i \overset{Key}{\longleftrightarrow} P_j$$

which establishes Goal4.

So far, we have completed the formal analysis of SEAGKAP, and all the goals have been proved. Through the above BAN logic analysis, it is shown that SEAGKAP achieves group-wide mutual authentication and shared session-key belief among all legitimate participants. Each participant believes that the established session key is fresh and shared exclusively with authenticated group members.

3.2. Formal Analysis Tool

Scyther is employed to perform symbolic verification of SEAGKAP under the Dolev–Yao adversary model.In the Scyther specification, Schnorr signatures are abstracted as ideal digital signatures $\{m\}_{\mathrm{sk}}$ that are publicly verifiable via the corresponding public key, captured by the relation $\mathrm{inversekeys}(\mathrm{pk}, \mathrm{sk})$. The session identifier sid, which (abstracts the timestamp/session context in the protocol, is included in the signed payload to enforce session binding and to prevent replay across sessions.

We specify standard authentication claims Alive, Weakagree, Nisagree, Nisyncn for each role and a secrecy claim on an abstract session key derived from the authenticated broadcast parameters of both rounds. Fig. 1 illustrates the Scyther-based formal modeling of the SEAGKAP protocol, and the results show that all specified security claims are successfully satisfied within the given bounded analysis scope, as illustrated in Fig. 2.The authentication- and synchronization-related claims, including Alive, Weakagree, Niagree, and Nisynch, are all marked as Ok, indicating that the protocol is resilient to replay, impersonation, and desynchronization attacks within the current model and search bounds.

```
hashfunction H;
const pkA, pkB, pkC: Function;
secret skA, skB, skC: Function;
inversekeys (pkA,skA);
inversekeys (pkB,skB);
inversekeys (pkC,skC);

protocol SEAGKAP(A, B, C)
{
  role A
  {
    fresh rA: Nonce;
    fresh RA: Nonce;
    var RB: Nonce;
    var RC: Nonce;
    fresh sid: Nonce;
    send_1(A, B, A, RA, sid, {A,RA,sid}skA);
    send_2(A, C, A, RA, sid, {A,RA,sid}skA);
    recv_3(B, A, B, RB, sid, {B,RB,sid}skB);
    recv_5(C, A, C, RC, sid, {C,RC,sid}skC);

    fresh XA: Nonce;
    var XB: Nonce;
    var XC: Nonce;
    send_7(A, B, A, XA, sid, {A,XA,sid}skA);
    send_8(A, C, A, XA, sid, {A,XA,sid}skA);
    recv_9(B, A, B, XB, sid, {B,XB,sid}skB);
    recv_11(C, A, C, XC, sid, {C,XC,sid}skC);

    claim(A, Secret, H(rA, RC, XA, XB));
    claim(A, Alive);
    claim(A, Weakagree);
    claim(A, Niagree);
    claim(A, Nisynch);
  }

  role B
  {
    fresh RB: Nonce;
    fresh rB: Nonce;
    var RA: Nonce;
    var RC: Nonce;
    var sid: Nonce;
    recv_1(A, B, A, RA, sid, {A,RA,sid}skA);
    send_3(B, A, B, RB, sid, {B,RB,sid}skB);
    send_4(B, C, B, RB, sid, {B,RB,sid}skB);
    recv_6(C, B, C, RC, sid, {C,RC,sid}skC);

    fresh XB: Nonce;
    var XA: Nonce;
    var XC: Nonce;
    recv_7(A, B, A, XA, sid, {A,XA,sid}skA);
    send_9(B, A, B, XB, sid, {B,XB,sid}skB);
    send_10(B, C, B, XB, sid, {B,XB,sid}skB);
    recv_12(C, B, C, XC, sid, {C,XC,sid}skC);

    claim(B, Secret, H(rB, RA, XB, XC));
    claim(B, Alive);
    claim(B, Weakagree);
    claim(B, Niagree);
    claim(B, Nisynch);
  }
  role C
  {
    fresh rC: Nonce;
    fresh RC: Nonce;
    var RA: Nonce;
    var RB: Nonce;
    var sid: Nonce;
    recv_2(A, C, A, RA, sid, {A,RA,sid}skA);
    recv_4(B, C, B, RB, sid, {B,RB,sid}skB);
    send_5(C, A, C, RC, sid, {C,RC,sid}skC);
    send_6(C, B, C, RC, sid, {C,RC,sid}skC);

    fresh XC: Nonce;
    var XA: Nonce;
    var XB: Nonce;
    recv_8(A, C, A, XA, sid, {A,XA,sid}skA);
    recv_10(B, C, B, XB, sid, {B,XB,sid}skB);
    send_11(C, A, C, XC, sid, {C,XC,sid}skC);
    send_12(C, B, C, XC, sid, {C,XC,sid}skC);
    claim(C, Secret, H(rC, RB, XC, XA));
    claim(C, Alive);
    claim(C, Weakagree);
    claim(C, Niagree);
    claim(C, Nisynch);
  }
}
```

Fig. 1. Scyther-based formal model of SEAGKAP

Scyther results : verify

Claim				Status	Comments
SEAGKAP	A	SEAGKAP,A1	Secret H(rA,RC,XA,XB)	Ok	No attacks within bounds.
		SEAGKAP,A2	Alive	Ok	No attacks within bounds.
		SEAGKAP,A3	Weakagree	Ok	No attacks within bounds.
		SEAGKAP,A4	Niagree	Ok	No attacks within bounds.
		SEAGKAP,A5	Nisynch	Ok	No attacks within bounds.
	B	SEAGKAP,B1	Secret H(rB,RA,XB,XC)	Ok	No attacks within bounds.
		SEAGKAP,B2	Alive	Ok	No attacks within bounds.
		SEAGKAP,B3	Weakagree	Ok	No attacks within bounds.
		SEAGKAP,B4	Niagree	Ok	No attacks within bounds.
		SEAGKAP,B5	Nisynch	Ok	No attacks within bounds.
	C	SEAGKAP,C1	Secret H(rC,RB,XC,XA)	Ok	No attacks within bounds.
		SEAGKAP,C2	Alive	Ok	No attacks within bounds.
		SEAGKAP,C3	Weakagree	Ok	No attacks within bounds.
		SEAGKAP,C4	Niagree	Ok	No attacks within bounds.
		SEAGKAP,C5	Nisynch	Ok	No attacks within bounds.

Done.

Fig. 2. Scyther-based formal analysis result of SEAGKAP

4. INFORMAL SECURITY ANALYSIS

4.1. Replay Resistance

Each round message embeds a timestamp as a core input, enforcing a strict validity window. If the timestamp deviates from the verifier's clock, verification fails. Moreover, because timestamps are bound to Schnorr signatures, only entities holding valid private keys can generate updated valid signatures with fresh timestamps. Since adversaries cannot alter signatures or generate new valid ones, replay attempts are effectively prevented.

4.2. Forward and Backward Secrecy

Session keys incorporate fresh randomness generated inside each party's TEE at every invocation. Due to the hardware isolation of TEEs, adversaries cannot extract or predict these random values. Even if one session key is compromised, past keys remain unrecoverable (forward secrecy) and future keys remain unpredictable (backward secrecy), ensuring lifecycle security of key agreement.

4.3. key consistency

In the proposed protocol, each participant derives the final session key locally using broadcast parameters and its own private randomness, eliminating the need for centralized coordination or a trusted third party. Signature verification in both rounds guarantees that all parties maintain a consistent view of R_i and X_i, while any forged or inconsistent intermediate values are detected and discarded during the verification phase. Using the following key computation formula, it is straightforward to prove that the protocol ensures correctness. Consequently, under honest execution, all honest parties compute the same session key, which guarantees both correctness and key consistency.

$$
\begin{aligned}
K_i =& nr_i \cdot R_{i-1} + (n-1)\cdot X_i + (n-2)\cdot X_{i+1} \\
&+ \cdots + X_{i-2} \\
=& r_i \cdot R_{i-1} + (r_i \cdot R_{i-1} + X_i) \\
&+ (r_i \cdot R_{i-1} + X_i + X_{i+1}) + \cdots \\
&+ (r_i \cdot R_{i-1} + X_i + \cdots + X_{i-2}) \\
=& r_i \cdot R_{i-1} + (r_i \cdot R_{i-1} + r_i \cdot R_{i+1} - r_i \cdot R_{i-1}) \\
&+ (r_i \cdot R_{i-1} + r_i \cdot R_{i+1} - r_i \cdot R_{i-1} \\
&+ r_{i+1} \cdot R_{i+2} - r_{i+1} \cdot R_i) + \cdots \\
&+ (r_i \cdot R_{i-1} + r_i \cdot R_{i+1} - r_i \cdot R_{i-1} + \cdots \\
&+ r_{i-2} \cdot R_{i-1} - r_{i-2} \cdot R_{i-3}) \\
=& r_i \cdot R_{i-1} + r_i \cdot R_{i+1} + r_{i+1} \cdot R_{i+2} \\
&+ \cdots + r_{i-2} \cdot R_{i-1} \\
=& r_i \cdot R_{i+1} + r_{i+1} \cdot R_{i+2} + \cdots \\
&+ r_{i-2} \cdot R_{i-1} + r_{i-1} \cdot R_i
\end{aligned}
$$

where $X_i = r_i \cdot (R_{i+1} - R_{i-1})$. The right-hand side is a cyclically invariant symmetric sum, and therefore it follows that $K_i = K_j$ for any i, j. By the deterministic nature of the key derivation function, it follows that $Key_i = Key_j$, that is, all honest parties derive the same session key.