Sketch of Security Proof for (n+1)Sec Protocol

The (n+1)Sec protocol is composed of following subprotocols:

1. **TDH**: Triple DH deniable Authentication
2. **FAGKE**: Flexible Authenticated Group Key Exchange protocol presented in [ACMP]
3. **SecCom**: Secure (authenticated confidential) Send and Receive.
4. **TCA**: Transcript Consistency Assurance.

The threat model for each of these protocols is described in Section VI. The security of FAGKE is proven in the presented threat model. The SecCom consists of conventional "sign" and "encrypt" functions and its security has been studied as a subprotocol to various protocols. We are not aware of any existing proof for the TDH and TCA subprotocols.

The sketch of the proof is structured as follows: Section 3 deals with security of TDH, namely its deniability. The authentication of TDH will be proven as a part of AKE security proof. We also prove the TDH protocol as a 2-party secure AKE in model presented in [ACMP]. Section 4 proves the security properties of the group key exchange protocol. In Section 5 we prove that $(n+1)$sec provides an authenticated secure channel and as such the SecComm part of $(n+1)$sec is as secure as intended. Section 6, demonestrates the security of $(n+1)$sec against message origin adversary. In Section 7, we give the proof of the security properties of TCA.

1 General Definition

In this section we introduce the ideas and definition we are using throughout the proof.

Definition 1. *Suppose $\mathbb{G}$ is a multiplicative group. Given arbitary $g, g^a, g^b \in \mathbb{G}$, the* ***Computational Diffie-Hellman (CDH) problem*** *is to compute g^{ab}.*

Definition 2. *Following the notation in Definition 1, given arbitary g, g^a, g^b, $g^c \in \mathbb{G}$, the* ***Decisional Diffie-Hellman (DDH) problem*** *is to determine if $g^c = g^{ab}$.*

Definition 3. *Following the notation in Definition 2,* ***Gap Diffie-Hellman problem*** *is to compute g^{ab} while having access to a DDH oracle. In other words,* ***GDH assumption*** *for group $\mathbb{G}$ asserts that even if DDH is easy in $\mathbb{G}$, computing g^{ab} is hard.*

Definition 4. *A* ***Gap Diffie-Hellman Solver*** *or a* ***GDH solver $\mathcal{S}$*** *for group $\mathbb{G}$ is a function $\mathcal{S}$ defined as*

$$\mathcal{S}: (g, g^a, g^b, \mathcal{O}_{\mathrm{DDH}}) \longmapsto g^{ab}$$

Where $\mathcal{O}_{\mathrm{DDH}}$ is a DDH oracle for group $\mathbb{G}$.

Definition 5. *We indicate* ***the set of all possible participants*** *(in the universe) by $\mathcal{U}$, such that $|\mathcal{U}| = N$, where each participant is represented by a unique identity U_i. Each U_i is verifiably identified by a long-term public key LPK_i for which it possesses its corresponding long-term private key LSK_i.*

In modelling the chat session, in terms of the adversarial models and protocol specifications, the notation of [ACMP] is followed. This notation is common to other publications on group key exchange such as [GBNM], and is adhered to for consistency.

Definition 6. *We indicate **the set of all possible participants** (in the universe) by $\mathcal{U}$, such that $|\mathcal{U}| = N$. Each participant is represented by a unique identity U_i. Each U_i is verifiably identified by a long-term public key LPK_i for which it possesses its corresponding long-term private key LSK_i.*

Definition 7. *A **multi-party chat session** is an ordered pair $\mathcal{S} := (S, \mathrm{sid}^{\mathcal{S}})$, in which $S \subseteq \mathcal{U}$ and sid is the unique session id computed as a function of the participants' id and their ephemeral keys. The **Ephemeral key** of participant U_i is the private and public key pair of $(x_i^{\mathcal{S}}, y_i^{\mathcal{S}})$ that is generated by a participant for the purpose of participating in $\mathcal{S}$ such that*

$$y_i^{\mathcal{S}} = x_i^{\mathcal{S}} g$$

in additive notation, where g is the generator of a group G with hard Discrete Logarithm Problem (DLP). We refer to either of x_i or y_i as the ephemeral key of user U_i when there is chance of ambiguity. Without loss of generality we assume:

$$S := \{U_1, ..., U_n\}$$

*The ordered **list of participants** and ordered list of their ephemeral keys is defined as:*

$$\mathrm{plist}^{\mathcal{S}} := (U_1, ..., U_n)$$

$$\mathrm{klist}^{\mathcal{S}} := (y_1, ..., y_n)$$

Accordingly, we denote the interviewing concatenation of these two lists as:

$$\mathrm{plist}^{\mathcal{S}} | \mathrm{klist}^{\mathcal{S}} := U_1 | y_1 | U_2 | y_2 | \cdots | U_n | y_n$$

The order is to be uniquely computable by all participants (lexicographically ordered using the long-term public key of participants, for example).

A subset of participants might want to start a session in which the remaining parties are excluded (for example when those parties leave the chatroom). The following definition formalizes such situation:

Definition 8. *For a session $\mathcal{S} = (S, \mathrm{sid}^{\mathcal{S}})$, $\mathcal{T} = (S, \mathrm{sid}_{\mathcal{T}})$ is called a **sub-session** of $\mathcal{S}$ if $T \subset S$ and all participants $U_i \in T$ use the same ephemeral key for both $\mathcal{S}$ and $\mathcal{T}$. In other words, the same ephemeral keys are used to compute $\mathrm{sid}_{\mathcal{T}}$. In such situation, we call $\mathcal{S}$ the super-session of $\mathcal{T}$.*

Definition 9. *An **authenticated group key exchange (AGKE)** is a protocol Π each participant executes in order to communicate (by means of sending, receiving or computing) a cryptographic secret - namely a key - among the other parties of a session. By $\Pi_i^{\mathcal{S}}$ we refer to the **instance of the protocol run by** U_i for session $\mathcal{S}$. The $\mathrm{sid}^{\mathcal{S}}$ computed by $\Pi_i^{\mathcal{S}}$ and denoted by $\mathrm{sid}_i^{\mathcal{S}}$ (or sid_i when there is no chance of confusion) is called the **session id observed by** U_i. Similarly, $\mathrm{plist}_i^{\mathcal{S}}$ (or plist_i) is the list of participants which U_i believes are participating in the attack and $\mathrm{klist}_i^{\mathcal{S}}$ (or klist_i) is their perceived set of ephemeral public keys.*

Definition 10. *To communicate in a multi-party session, each participant U_i needs to compute a symmetric **session key** $\mathrm{sk}_i^{\mathcal{S}}$ which should be computable by other parties participating in the chat or be transmitted confidentially to them. We say a participant enters the **accepted state** if they have computed $\mathrm{sk}_i^{\mathcal{S}}$ and have detected no error in the protocol.*

The essential defining factor is that part of sk_i^S should become common knowledge for the session participants at the end of AGKE execution, so they can communicate confidentially. Nevertheless, it is not necessary that all participants share the same secret sk_i^S among themselves and they can broadcast their messages encrypted using multiple keys. This decreases the efficiency as well as complicates the security analysis of the protocol. As such, we assume that at the end of running a correct AGKE, all participants possess a shared secret:

Definition 11. *Two accepted instances Π_i^S and Π_j^S are considered **partnered** if $\mathrm{sid}_i = \mathrm{sid}_j$ and $\mathrm{plist}_i = \mathrm{plist}_j$.*

Definition 12. *A **correct** AKGE algorithm is an AKGE where, when all Π_i^S instances of AKE algorithm are initiated with access to a network which correctly forwards all messages without modification, all participants ultimately are partnered and all compute equal sk_i^S's.*

After all instances of a session have partnered, they need to use the computed common symmetric key to communicate securely. Following the subprotocol can guarantee some of the security properties which $(n+1)$sec aims to promise.

Definition 13. ([BHMS] Definition 3.1) ***A* stateful authenticated encryption with associated data *(stateful AEAD) scheme*** *Π consists of:*

- *A probabilistic key generation algorithm (it is the AGKE in our case).*
- *A stateful probabilistic encryption $E(k, \mathrm{ad}, m, \mathrm{st}_E) \rightarrow (c, \mathrm{st}'_E)$.*
- *A deterministic decryption algorithm $D(k, \mathrm{ad}, c, \mathrm{st}_D) \rightarrow (\mathrm{ad}, m, \mathrm{st}'_D)$*

  which can output ad *or $\perp$ as error and message m or $\perp$ as error.*

Definition 14. *A **correct stateful AEAD scheme** is a stateful AEAD scheme Π, which can correctly decrypt a ciphertext c to the corresponding message m for any sequences of message or output error in case the ciphertext does not correspond to the output of $E(k)$.*

2 Adversarial Power

We will re-use these definitions to demonstrate similar routes for other adversaries considered by the threat models in later sections.

2.1 Adversarial power for AKE

The following set of functions models the AKE adversarial threats. The adversary for the authenticated key exchange can mount an attack through a sequence of calls to the functions, outlined below. The limitation on the order and condition of calling these functions is defined per adversary.

- Execute(plist): asks all parties in the plist to run (a new) AGKE protocol and $\mathcal{A}$ will receive the execution transcript, i.e. can eavesdrop.
- $\mathrm{Send}_{U_i}(\Pi_i^S)(m)$ sends a message m to the instance Π_i^S as user U_j. We assume that m contains information to identify the sender U_j. U_j will receive the reply transcript. Specifically, by sending plist messages it forces U_i to initiate Π_i^S.
- **SKE(ΠSi,spidi)**: asks ΠSi to compute the subgroup key for the spidi subsession. In response, ΠSi will either send a message or compute the subgroup key kspidi depending on the state of ΠSi. This can be invoked only once per input.

- **RevealGK()**: ΠSi gives ski to Aa if it has accepted (as described in Definition III.3).

- **RevealSK**: ΠSi gives the subkTi to Aa if it has been computed for subsession T.

- RevealPeer(Π_i^S,U_j): When the $\mathcal{A}$ calls this function, it will be provided with the $p2p$ key $k_{i,j}^S$, if it is already computed.

- **Corrupt(Ui)**: Ui gives its long-term secret key to Aa (but not the session key).

Definition 15. ***AKE-Security of P2P Keys****: Let $\mathcal{P}$ be a* GKE + P *protocol and b a uniformly chosen bit. Adversary $\mathcal{A}_{p2p}$ is allowed to invoke all adversarial queries. At some point the Adversary runs* TestPeer(Π_i^S, U_j) *for some fresh instance User pair* (Π_i^S, U_j) *which remains fresh. $\mathcal{A}_{p2p}$ is allowed to continue the adversarial queries provided the test pair remains fresh. Finally $\mathcal{A}_{p2p}$ outputs a bit b'. The adversarial advantage is defined as*

$$\mathrm{Adv}_{\mathcal{A}_{p2p}}(\mathcal{P}) := |2\mathrm{Pr}(b' = b) - 1|$$

We say the $\mathcal{P}$ is secure if the advantage is negligible.

Definition 16. ([ACMP] Definition 5 AKE-Security of group Keys) *: Let $\mathcal{P}$ be a correct GKE+P protocol and b a uniformly chosen bit. By* $\mathrm{Game}_{\mathcal{A}_{\mathrm{GKE}}}(\mathcal{P},\kappa)$*,we define the following adversarial game, which involves a PPT adversary $\mathcal{A}_{\mathrm{GKE}}$ that is given access to all queries:*

– $\mathcal{A}_{\mathrm{GKE}}$ interacts via queries;

– at some point $\mathcal{A}_{\mathrm{GKE}}$ asks a TestGK(Π_i^S) *query for some instance Π_i^S which is (and remains) fresh;*

– $\mathcal{A}_{\mathrm{GKE}}$ continues interacting via queries;

– when $\mathcal{A}_{\mathrm{GKE}}$ terminates, it outputs a bit, which is set as the output of the game. We define:

$$\mathrm{Adv}_{\mathcal{A}_{\mathrm{GKE}}}(\mathcal{P},\kappa) := |2\mathrm{Pr}[\mathrm{Game}_{\mathcal{A}_{\mathrm{GKE}}}(\mathcal{P},\kappa) = b] - 1|$$

We say that $\mathcal{P}$ provides GKE-security if the maximum of this advantage over all possible PPT adversaries $\mathcal{A}_{\mathrm{GKE}}$ is negligible.

Definition 17. ([ACMP] Definition 6 AKE Security of subgroup keys) *: Let P be a correct GKE+S protocol and b a uniformly chosen bit. By Game ake-s,b*

A,P (κ) we define the following adversarial game, which involves a PPT adversary A that is given access to all queries:

– A interacts via queries;

– at some point A asks a TestSK(Π i s , spid si) query for some instance-subgroup pair

(Π i s , spid si) which is (and remains) fresh;

– A continues interacting via queries;

– when A terminates, it outputs a bit, which is set as the output of the game.

We define

$$\mathrm{Adv}_{\mathcal{A}_{S-\mathrm{GKE}}}(\mathcal{P},\kappa) := |2\mathrm{Pr}[\mathrm{Game}(\kappa) = b] - 1|$$

and denote with Adv ake-s(κ) the maximum advantage over all PPT adversaries A. We say that P provides AKE-security of subgroup keys if this advantage is negligible.

2.2 Secure Multi-party Channel Adversary

The desirable way to define an adversary for a multi-party chat session is a secure channel model similar to the two-party secure channels described in [CK], [JKSS] and [KPW]. As such, we set the *authenticated and confidential channel establishment* (ACCE) protocol as our starting point. In this regard, we would like to prove that $(n+1)$sec is an ACCE protocol. It is argued in [JKSS] that if a scheme provides a secure AKE and the symmetric encryption of the session communication satisfies the suitable confidentiality and integrity criteria, then one can conlude that the scheme is an ACCE protocol (although the inverse statement is not true). Following this path, we define the adversary for the communication phase of a secure multi-party chat session. We use the Definition 3.2 from [BHMS] instead of Definition 6 in [JKSS], because hiding the length of the conversation is not considered as a security property of $(n+1)$sec.

2.2.1 Definition of Adversaries and their advantages

Based on Definition 13, the adversary against an AEAD is defined as follows:

Definition 18. ([BHMS] Definition 3.2) *: Let* Π *be a stateful AEAD scheme and let A be a PPT adversarial algorithm. Let* $i \in \{1, \ldots, 4\}$ *and let* $b \in \{0, 1\}$*. The stateful AEAD experiment for* Π *with condition* cond_i *and bit b is given by* $\mathrm{Exp}^{\mathrm{aead}_i - b}(\Pi, \mathcal{A})$ *as defined in [BHMS] Figure 4. The adversaries' advantage is defined as*

$$\mathrm{Adv}^{\mathrm{aead}_i}()_{\Pi, \mathcal{A}_{\mathrm{aead}_i}} := \Pr\left[\mathrm{Exp}^{\mathrm{Exp}^{\mathrm{aead}_i-1}}(\Pi, \mathcal{A}) = 1\right] - \Pr\left[\mathrm{Exp}^{\mathrm{Exp}^{\mathrm{aead}_i-0}}(\Pi, \mathcal{A}) = 1\right]$$

2.3 Message Origin Authentication Adversary

Any manipulation of data by an outsider is modeled in the AEAD adversary as described in Definition 18. $(n+1)$sec, however, also needs to protect insiders from forging messages on behalf of each other. That is why each participant executes a sign and encrypts a function before sending their authenticated ephemeral signing key. The message origin adversary model is based on a typical adversary for a signature scheme such as the one presented in [BPVY].

2.3.1 Adversarial power

In addition to adversarial functions defined in Section 2, we must define the following function to allow for the adversary using the chosen-message attack.

- **MakeSend($\Pi_i^{\mathcal{S}}$, $\Pi_j^{\mathcal{S}}$, m)** causes the $\Pi_i^{\mathcal{S}}$ to sign and send a valid message m to instance $\Pi_j^{\mathcal{S}}$. $\mathcal{A}_{\mathrm{orig}}$ will receive the transcript including the signature.

2.3.2 Definition of the Adversary

Definition 19. ***Message Origin Authentication Adversary:*** $\mathcal{A}_{\mathrm{orig}}$ *is a polynomial time algorithm which has access to the* ***Corrupt***, ***Send***, ***Reveal*** *and* ***MakeSend*** *functions. The output of the algorithm should be a message m sent to instance* $\Pi_j^{\mathcal{S}}$*. The scheme is secure against the message origin adversary if the probability in which* $\Pi_j^{\mathcal{S}}$ *believes that m has originated from an uncorrupted participant* U_i *is negligible under assumption of the hardness of the Discrete Logarithm Problem.*

3 Security of Triple Diffie-Hellman Authentication

3.1 The Triple Diffie-Hellman Protocol

Assuming that A and B are represented by long-term public keys g^A and g^B respectively:

3.2 The deniablity of TDH

We will prove a parallel to Theorem 4 [RGK] which proves the deniability of SKEME. We use the notation introduced in [RGK]. Following the same notation:

Definition 20. *By* $\mathrm{Adv}^*_{\mathrm{deny}}$ *we refer to the party which represents the interaction of the Simulator* Sim *with the adversary. In other words,* $\mathrm{Adv}^*_{\mathrm{deny}}$ *has access to all information which* $\mathrm{Adv}_{\mathrm{deny}}$ *possesses.*

Theorem 21. *If Computational Diffie-Hellman (CDH) is intractable, then Triple DH Algorithm is deniable.*

Proof. We build a Sim which interacts with $\mathrm{Adv}_{\mathrm{deny}}$. We show that if $\mathcal{J}$ is able to distinguish $\mathrm{Trans}_{\mathrm{Sim}}$ from $\mathrm{Trans}_{\mathrm{Real}}$, they should be able to solve CDH as well.

Intuitively, when $\mathcal{A}_{\mathrm{deny}}$ sends g^a to $\mathcal{S}_{\mathrm{deny}}$, $\mathcal{S}_{\mathrm{deny}}$ inquires $\mathcal{A}_{\mathrm{deny}}$ for a, in this way $\mathcal{S}_{\mathrm{deny}}$ also can compute the same key k by asking $\mathcal{A}^*_{\mathrm{deny}}$. If $\mathcal{A}_{\mathrm{deny}}$ has chosen $g^a \in \mathrm{Tr}(B)$ or has just chosen a random element of the group without knowing its DLP, then $\mathcal{S}_{\mathrm{deny}}$ will choose a random exponent a' and compute the key k based on that and the confirmation value using k. Due to the difficulty of CDH, this value is indistinguishable from a k generated by B.

Now we suppose that the TDH is not deniable and we build a solver for CDH. First we note that if $\mathcal{A}_{\mathrm{deny}}$ engages in an honest interaction with B, there is no way that $\mathcal{J}$ can distinguish between the $T(\mathcal{A}_{\mathrm{deny}}(\mathrm{Aux}))$ and $T(\mathcal{S}_{\mathrm{deny}}(\mathrm{Aux}))$. This is because $\mathcal{A}_{\mathrm{deny}}$ is able to generate the very exact transcript without the help of B. Therefore, logically, the only possibility for $\mathcal{J}$ to distinguish $T(\mathcal{A}_{\mathrm{deny}}(\mathrm{Aux}))$ and $T(\mathcal{S}_{\mathrm{deny}}(\mathrm{Aux}))$ is when $\mathcal{A}_{\mathrm{deny}}$ presents $\mathcal{J}$ with a transcript that $\mathcal{A}_{\mathrm{deny}}$ is not able to generate itself. The only variable that $\mathcal{A}_{\mathrm{deny}}$ has control over in the course of the exchange is g^a and therefore the only way $\mathcal{A}_{\mathrm{deny}}$ is able to claim that it was unable to generate the genuine $T(\mathcal{A}_{\mathrm{deny}}(\mathrm{Aux}))$ is by sending g^a in which $\mathcal{A}_{\mathrm{deny}}$ itself is not aware of the exponent a.

In such case, assuming the undeniability of TDH, we have an ε such that

$$\max_{\text{all } \mathcal{J}} |2\Pr(\mathrm{Output}(\mathcal{J}, \mathrm{Aux}) = b) - 1| > \varepsilon$$

The solver $\mathcal{A}_{\mathrm{CDH}}$ receives a triple (g, g^a, g^b) and should compute g^{ab}. To that end, assuming long-term identity g^A for some $\mathcal{A}_{\mathrm{deny}}$, it engages in a TDH key exchange with a hypothetical automated party $\mathcal{A}^*$ with long-term private key B who generates g^b as the ephemeral key as well. $\mathcal{A}_{\mathrm{CDH}}$ then tosses a coin and, based on the result, it either chooses a random a' and computes $g' = g^{a'}$ or sets $g' = g^a$, then it submits $h_0 = H(g^{b\,A}, g'^B, g^{ba'})$ alongside with (g^B, g^b) to the $\mathcal{J}$ as a proof of engagement with $\mathcal{A}^*$. Due to the undeniability assumption,

$$\mathrm{Output}(\mathcal{J}, \mathrm{Aux})(h_0, (A, g^a, B, g^b)) = b$$

with significant probability, as it means $\mathcal{J}$ is able to distinguish $T(\mathcal{A}_{\mathrm{deny}}(\mathrm{Aux}))$ and $T(\mathcal{S}_{\mathrm{deny}}(\mathrm{Aux}))$ with high probability. Therefore $\mathcal{J}$ is able to decide if:

$$h_0 \overset{?}{=} H(g^{bA}, (g^a)^B, (g^a)^b)$$

Round 1	$A \rightarrow B$: $''A''$, g^a	$B \rightarrow A$:$''B''$, g^b
Key Computation	$k \leftarrow H((g^b)^A \vert (g^B)^a \vert (g^b)^a)$	$k \leftarrow H((g^A)^b \vert (g^a)^B \vert (g^a)^b)$
Round 2	$\mathrm{Enc}_k(H(k, A))$	$\mathrm{Enc}_k(H(k, B))$

Table 1. Triple Diffie-Hellman protocol

Because H is a random oracle, the only way the judge is able to distinguish the second value from the real value is to have knowledge about the exact pre-image: g^{bA}, $(g^a)^B$, $(g^a)^b$. Using the information in the transcript, $\mathcal{J}$ can compute $g^{bA}, (g^a)^B$, but still has to compute g^{ab} using g^a and g^b with high probability without knowing a or b. At this point, $\mathcal{A}_{\mathrm{CDH}}$ is publishing the value of $g^{a\,b}$.

□

3.3 Security of TDH as a two-party Authenticated Key Exchange

In this section we prove that TDH is a secure two-party authenticated key exchange. we do so in the AKE security model proposed in [Man]. This is because (n+1)sec's key exchange protocol is a variant of the protocol proposed in [ACMP], which is designed to satisfy all three AKE models proposed in [Man] and [ACMP]. Furthermore, based on the security properties required by (n+1)sec as a secure multi-party chat protocol, we believe these models provide adequate security for real-world threat scenarios.

Theorem 22. *If the GDH problem is hard in $\mathbb{G}$, then TDH protocol explained in Table 1, is secure in AKE model, with the advantage of the adversary bounded by:*

$$\mathrm{Adv}_{\mathcal{A}_{p2p}(k) \leqslant \mathcal{O}(q^2)} / Q$$

Where q is the maximum number of queries by the adversary.

Proof. Suppose that $k_{\mathrm{test}} = H((g^b)^A|(g^B)^a|(g^b)^a)$. Assuming that H is a PRF (SHA-256 in the case of (n+1)sec), the only way that adversary $\mathcal{A}_{p2p}$ can distinguish k_{test} from a random value k' is to compute all elements of the triplet $(g^b)^A, (g^B)^a, (g^b)^a$.

We show how to construct a GDH solver using an $\mathcal{A}_{p2p}$ that can compute all of the three above. Assuming that the test session is Fresh, then the adversary cannot corrupt either A or B and can request session reveal on the test session. Therefore it does not have either access to a or b.

Now suppose simulator $\mathcal{S}$ has access to the Adversary $\mathcal{A}$ oracle which is able to compute the Triple Diffie-Hellman inside the paranthesis. $\mathcal{S}$ needs to solve $g^{a\,b}$ for a given g^a and g^b. As such, it generates a transcript to set up a session between A and B while inserting g^a and g^b as exchanged keys.

Assuming the above, the adversary can compute the last token which is the solution to CDH.

□

Theorem 23. *If the GDH problem is hard in $\mathbb{G}$, then (n+1)sec protocol is secure against $\mathcal{A}_{p2p}$ adversary.*

Proof. We argue that the AKE security for the (n+1)Sec $p2p$ keys follows, similarly, from the proof of Theorem 8 [ACMP] which proves the security of BD+P protocol.

In fact, we follow the same sequence of games for games G_0 and G_1.

For game G_2, we note that contrary to mBP+P, which signs the second round message with LPK_i for authentication, the adversary has two ways to forge the authentication and force the other party to accept a wrong key. One is to forge the signature generated by the ephemeral key. This is basically covered by G_2. However, another way is to forge the authentication token we simulate in G_2'.

***G_2'*.** In this game, we abort the simulation if $\mathcal{A}_{p2p}$ queries $\mathrm{Send}(U_i, \mathrm{kc}_{i,j})$ with a valid confirmation where neither U_i or U_j is not corrupted. To do so, $\mathcal{A}_{p2p}$ needs to generate $H(k_{ij}|U_i)$. Assuming that H is PRF, this is only possible if $\mathcal{A}_{p2p}$ has successfully computed k_{ij}, which in part necessitates $\mathcal{A}_{p2p}$ computing $g^{b\,\mathrm{LPK}_i}$ to be able to impersonate A to B. Knowing neither secret b nor LPK_i, the advantage of $\mathcal{A}_{p2p}$ is bounded by its advantage in solving GDH. The adversary needs to solve all three GDH problems. Therefore we have:

$$|\Pr[\mathrm{Win}_2] - \Pr[\mathrm{Win}_{2'}]| < q\,(\mathrm{Succ}_{\mathbb{G}}^{\mathrm{GDH}}(\kappa))^3$$

In fact, the only difference in the proof is related to G_6. As k_{ij} is computed as $H(g^{Ab}|g^{Ba}|g^{ab})$. Therefore simulator delta will output $H'(g^A|g^B|g^a|g^b)$. However, because H is a perfect PRF, this remains indistinguishable unless the adversary has advantage on computing g^{Ab}, g^{Ba}, g^{ab}.

$$|\Pr[\mathrm{Win}_6] - \Pr[\mathrm{Win}_5]| < qH_p(\mathrm{Succ}_{\mathbb{G}}^{\mathrm{GDH}}(\kappa))^3$$

Consequently, the overall advantage of $\mathcal{A}_{p2p}$ bar its advantage in transition from G_2 to $G_{2'}$ is smaller than their advantage in the original mBD+P protocol:

$$\mathrm{Adv}_{(n+1)\mathrm{sec}}^{p2p}(\kappa) < \mathrm{Adv}_{\mathrm{mBD}+P}^{p2p}(\kappa) + q\,(\mathrm{Succ}_{\mathbb{G}}^{\mathrm{GDH}}(\kappa))^3$$

This proves that $\mathrm{Adv}_{(n+1)\mathrm{sec}}^{p2p}(\kappa)$ is asymptotically the same as $\mathrm{Adv}_{\mathrm{mBD}+P}^{p2p}(\kappa)$.

□

4 Security of (n+1)sec authenticated group key exchange

In this section we prove the security of the (n+1)sec group key exchange in the proposed adversarial model. Because the key exchange is essentially FAGKE, with the only difference being that the traditional DH key exchange is replaced by TDH, we prove the security of the (n+1)sec GKE based on the security of FAKE.

4.1 Security of GKE

We recall that the GKE protocol in (n+1)sec is essentially the same as the FAGKE protocol, except that in (n+1)sec we have:

$$k_{i,i+1} = H(g^{\mathrm{LS}_i x_{i+1}}, g^{\mathrm{LS}_{i+1} x_i}, g^{x_i x_{i+1}})$$

Whereas in FAGKE we have:

$$k_{i,i+1} = g^{x_i x_{i+1}}$$

Therefore, to prove that $(n+1)$sec is secure, we need to prove Theorem 24:

Theorem 24. *If the GDH problem is hard, then the (n+1)sec key exchange provides AKE-security of group keys.*

Proof. We argue that the AKE security for the (n+1)sec group key follows, similarly, from the proof of Theorem 7 [ACMP], which proves the security of the BD+P protocol.

In fact, we follow the same sequence of games for games G_0 and G_1.

Similar to the case of $p2p$ argued in Theorem 23, we need to expand game G_2 into two games of G_2 and G_2' to account both for the forgery of the signature and the TDH token. With the transitional advantage of

$$|\Pr[\text{Win}_2] - \Pr[\text{Win}_{2'}]| < q\,(\text{Succ}_{\mathbb{G}}^{\text{GDH}}(\kappa))^3$$

We proceed similarly with game G_3. The difference in the proof is related to G_4. Δ responds with g^a and g^b from the values of the GDH challenge. In this game, instead of computing z_i' as $H(H(g^{Ab}|g^{Ba}|g^{ab}), \text{sid})$, simulator Δ will output $H'(g^A|g^B|g^a|g^b)$. However because H is a perfect PRF, this remains indistinguishable, unless the adversary has an advantage on computing g^{Ab}, g^{Ba}, g^{ab}. So we have

$$|\Pr[\text{Win}_6] - \Pr[\text{Win}_5]| < qH_p(\text{Succ}_{\mathbb{G}}^{\text{GDH}}(\kappa))^3$$

The remaining argument for game G_4 is the same as $\text{mBD} + P$ proof.

Consequently, the overall advantage of $\mathcal{A}_{(n+1)\text{sec}}^{\text{ake}-g}$ bar its advantage in transition from G_2 to $G_{2'}$, is smaller than their advantage in the original mBD+P protocol:

$$\text{Adv}_{(n+1)\text{sec}}^{\text{ake}-g}(\kappa) < \text{Adv}_{\text{mBD}+P}^{\text{ake}-g}(\kappa) + q\,(\text{Succ}_{\mathbb{G}}^{\text{GDH}}(\kappa))^3$$

This proves that $\text{Adv}_{(n+1)\text{sec}}^{\text{ake}-g}(\kappa)$ is asymptotically the same as $\text{Adv}_{\text{mBD}+P}^{\text{ake}-g}(\kappa)$.

□

5 Security of $(n+1)$sec as a secure channel

In this section we prove the following theorem.

Theorem 25. *(n+1)sec is an* authenticated and confidential channel establishment *(ACCE) protocol.*

Proof. Based on [JKSS], a protocol which establish the confidential authentication key using a secure AKE and provides security against a stateful AEAD adversary during the secure session using the established key provides a secure (confidential and authenticated) channel. We have already established the GKE security of $(n + 1)$sec. Accordingly, we only need to prove that $(n+1)$sec provides stateful AEAD security.

To do so, we use [BHMS] Theorem 3.1 to prove that $(n+1)$sec is a secure level-3 AEAD scheme.

First, we recall the format of (n+1)sec messages:

```
:o3np1sec:Base64EnocodedMessage
```

In which `Base64EnocodedMessage` is encoded as

```
sid (DTHash),  Signature (DTHashx2), Encrypted part of the message.
```

Where sid and Signature are associated data and the Encryption is provided by AES-GSM. Using the result of [MV] and [IOM], we know that AES-GSM is both IND-CCA and INT-CTXT. As such, $(n+1)$sec is a level -1 AEAD scheme.

By considering the fact that $(n+1)$sec messages have an `own_sender_id` which is strictly increasing for each sender one by one for each message, alongside with `session_id` and `nonce`, we prove that $(n+1)$sec encoding passes TEST4 described in [BHMS] Figure 3. Therefore, based on [BHMS] Theorem 3.1, (n+1)sec resists a level-4 stateful AEAD adversary.

Now using the result of Theorem 24, based on the conclusion of [JKSS], we conclude that (n+1)sec is an ACCE protocol.

□

6 Security of (n+1)sec against Message Origin Authentication Adversary

Using the result of Theorem 25, we know that the (n+1)sec session transcript is secure against outsiders' manipulation. Therefore, it only remains to study the ability of the insiders of the session in forging messages against each other. To prevent such scenario, $(n+1)$sec messages are signed by authenticated ephemeral keys. The authenticity of ephemeral keys is assured based on Theorem 23 and has been established before the session starts. Therefore we only need to prove that $(n+1)$sec provides security against signature forgery.

As each participant executes a sign and encrypt function before sending their authenticated ephemeral signing key, the message origin adversary model is based on a typical adversary for a signature scheme such as the one presented in [BPVY].

Theorem 26. $(n+1)$sec *is secure against* $\mathcal{A}_{\text{orig}}$.

Proof. $(n+1)$sec message is signed using the EdDSA signature scheme. According to [BDL+], the EdDSA system is a Schnorr based signature system and inherits the security properties of the Schnorr signature. According to [PS] Theorem 4, a chosen-message attack which can break the Schnorr scheme can solve the DLP of the underlying system in polynomial time. This will establish the security of $(n+1)$sec against the adversary defined in Definition 19.

□

7 Security of Transcript Consistency Assurance

Bibliography

[ACMP] Michel Abdalla, Céline Chevalier, Mark Manulis, and David Pointcheval. Flexible Group Key Exchange with On-Demand Computation of Subgroup Keys. Volume 6055 of *LNCS*, pages 351–368. Springer.

[BDL+] Daniel J. Bernstein, Niels Duif, Tanja Lange, Peter Schwabe, and Bo-Yin Yang. High-Speed High-Security Signatures. Volume 6917 of *Lecture Notes in Computer Science*, pages 124–142. Springer.

[BHMS] Colin Boyd, Britta Hale, Stig Frode Mjølsnes, and Douglas Stebila. From Stateless to Stateful: Generic Authentication and Authenticated Encryption Constructions with Application to TLS.

[BPVY] Ernest F. Brickell, David Pointcheval, Serge Vaudenay, and Moti Yung. Design Validations for Discrete Logarithm Based Signature Schemes. Pages 276–292.

[CK] Ran Canetti and Hugo Krawczyk. Analysis of Key-Exchange Protocols and Their Use for Building Secure Channels. Volume 2045 of *Lecture Notes in Computer Science*, pages 453–474. Springer.

[GBNM] M. Choudary Gorantla, Colin Boyd, Juan Manuel González Nieto, and Mark Manulis. Modeling key compromise impersonation attacks on group key exchange protocols. 14(4):28.

[IOM] Tetsu Iwata, Keisuke Ohashi, and Kazuhiko Minematsu. Breaking and repairing GCM security proofs. In *Advances in Cryptology–CRYPTO 2012*, pages 31–49. Springer.

[JKSS] Tibor Jager, Florian Kohlar, Sven Schäge, and Jörg Schwenk. On the security of TLS-DHE in the standard model. In *Advances in Cryptology–CRYPTO 2012*, pages 273–293. Springer.

[KPW] Hugo Krawczyk, Kenneth G. Paterson, and Hoeteck Wee. On the Security of the TLS Protocol: A Systematic Analysis. 2013:339.

[Man] Mark Manulis. Group Key Exchange Enabling On-Demand Derivation of Peer-to-Peer Keys. Pages 1–19.

[MV] David A McGrew and John Viega. The security and performance of the Galois/Counter Mode (GCM) of operation. In *Progress in Cryptology-INDOCRYPT 2004*, pages 343–355. Springer.

[PS] David Pointcheval and Jacques Stern. Security arguments for digital signatures and blind signatures. 13(3):361–396.

[RGK] Mario Di Raimondo, Rosario Gennaro, and Hugo Krawczyk. *Deniable Authentication and Key Exchange*. Published: Cryptology ePrint Archive, Report 2006/280 http://eprint.iacr.org/.