Security Enhancement and Modular Treatment towards Authenticated Key Exchange*

Jiaxin Pan[1,2], Libin Wang[1,2,**], and Changshe Ma[1,2]

[1] School of Computer, South China Normal University, Guangzhou 510631, China
[2] Shanghai Key Laboratory of Integrate Administration Technologies for Information Security, Shanghai 200240, China
csplator@gmail.com, lbwang@scnu.edu.cn, changshema@gmail.com

Abstract. We present an enhanced security model for the authenticated key exchange (AKE) protocols to capture the pre-master secret replication attack and to avoid the controversial random oracle assumption in the security proof. Our model treats the AKE protocol as two relatively independent modules, the secret exchange module and the key derivation module, and formalizes the adversarial capabilities and security properties for each of these modules. We prove that the proposed security model is stronger than the extended Canetti-Krawczyk model. Moreover, we introduce NACS, a two-pass AKE protocol which is secure in the enhanced model. NACS is practical and efficient, since it reqires less exponentiations, and, more important, admits a tight security reduction with weaker standard cryptographic assumptions. Finally, the compact and elegant security proof of NACS shows that our method is reasonable and effective.

Keywords: Provable security; authenticated key exchange protocol; extended Canetti-Krawczyk model.

1 Introduction

Authenticated Key Exchange (AKE) protocol is the cryptographic primitive that enables two parties to establish a secure common session key via a public network. To date, a great number of provably secure AKE protocols have been proposed in the literature, and many of them were subsequently broken. Hence, we need to study the limitations of formal security models and to propose an appropriate model in order to capture some possible fatal attacks.

Recently, Cremers [4] proposed an important kind of attacks that the extended Canetti-Krawczyk (eCK) model [5] fails to capture. We name this kind of attacks as *the pre-master secret replication (PSR) attacks*. The goal of the adversary is to

* This work was supported by the National Natural Science Foundation of China under Grant #60703094, and the Opening Project of Shanghai Key Laboratory of Integrate Administration Technologies for Information Security.
** Corresponding author.

M. Soriano, S. Qing, and J. López (Eds.): ICICS 2010, LNCS 6476, pp. 203–217, 2010.

force two distinct non-matching sessions with the same communication parties to agree on the similar pre-master secrets which will be input into the key derivation function to derive the session keys. After that the adversary chooses one session as the test session, and learns the pre-master secret of the other. By doing this, the adversary can break the security of some eCK-secure protocol (e.g. NAXOS [4]). The eCK model can not describe such attack, since the adversarial capability definition and the eCK-freshness permit the adversary to compute the pre-master secret which belongs to the peer of the test session.

We believe that resistance to the PSR attack is a desirable security property of AKE protocols. For an AKE protocol, the static private keys of the honest parties reside in e.g. a tamper-proof module (TPM) or cryptographic coprocessor, while the remainder of the protocol computations are done in regular (unprotected) memory. During the protocol execution, the adversary can read the regular memory to learn the pre-master secret and present PSR attack.

According to this problem, we introduce a new oracle query named as PremasterSecretReveal to capture the leakage of pre-master secret and to enhance the adversarial capabilities. Moreover, an appropriate freshness definition is given to allow the adversary to learn the pre-master secret of a session which does not have a matching test session. Our security notion is strong, since the PSR attack can be described in the proposed enhancement.

In addition, the formal model should provide facilitation for the security proof to avoid the controversial random oracle assumption [1], since the security proof is tightly related to the formal model definition. Actually, an AKE protocol has two sequential phases, the secret exchange phase and the key derivation phase: the secret exchange phase enables two honest parties to exchange a pre-master secret through the public communication channel; and the key derivation phase enables the honest parties to derive a session key from the pre-master secret separately. The traditional models treat these two phases as a whole model. To give a security reduction, we always either search the adversary's guess for the pre-master secret in the random oracle list, or propose some rather strong standard cryptographic assumptions (e.g. πPRF [9]).

To avoid controversial random oracle assumption, a modular treatment towards AKE is presented. We divide an AKE protocol into the secret exchange module and the key derivation module, and define adversarial capabilities and security for each module. Concluding the security of these two modules, a modular security model is defined. In the secret exchange security model, TestS query is defined to capture the adversary capability to compute pre-master secret, and also allows the simulator to learn the adversary guess for the pre-master secret in the security reduction without searching the random oracle list. We view this as the major contribution of the modular treatment.

Finally, we propose a two-pass AKE protocol, called NACS, which provably meets our enhanced and modular AKE security under the Gap Diffie-Hellman assumption [10]. The design motivation is to prevent PSR attacks, to avoid the random oracle assumption, and to admit tight security reduction under weaker standard assumptions. We name the proposed AKE protocol as NACS, since

the computation of the ephemeral public key use the NAXOS trick [5], and the pre-master secret is computed similarly to CMQV [13] and SMQV [12]. Compared with the related work, NACS combines the higher security level with comparable efficiency.

Summary of Our Contributions

1. We provide an enhanced AKE security model which is stronger and captures more attacks than the extended Canetti-Krawczyk model.
2. We give a modular treatment towards the AKE protocol such that the security proof can get rid of the random oracles with the help of this treatment.
3. We propose a two-pass AKE protocol, called NACS, which is provably secure in the enhanced model under weaker standard cryptographic assumptions. Compared with the related work in standard model, NACS has higher efficiency due to less exponentiations.

Organization. Section 2 establishes the enhanced security model and proves our model is stronger than the eCK model. The NACS protocol is described and the security proof ideas are outlined in Section 3. Finally, we draw a conclusion in Section 4.

2 Enhanced Security Model

In this section, the enhanced eCK model, which allows the adversary to learn the shared pre-master secret, is proposed. The enhanced model divides the AKE protocol into two modules, the secret exchange module and the key derivation module. For each module, a particular security model is specified by formalizing four aspects of the model: 1) the session identifier and matching sessions; 2) the adversarial capability; 3) the fresh session; 4) the adversarial goal. Note that the definitions of session identifiers and matching sessions for these two modules are the same, thus we only describe them in the model of secret exchange. Combining the security models of these two modules, an enhanced eCK-security is presented. We prove our security model is stronger than the original eCK model.

2.1 On the Limitations of eCK Model

The extended Canetti-Krawczyk (eCK) model proposed by LaMacchia *et al.* [5] is an extension of Canetti-Krawczyk (CK) model [2]. The eCK model replaces the SessionStateReveal query in CK model with the EphemeralKeyReveal query, which is used to capture the leakage of the ephemeral private keys, to define a stronger AKE security. However, recent papers [4,3,14,12] show many limitations on the eCK model, especially for the EphemeralKeyReveal query.

More recently, Cremers [4] argues that the EphemeralKeyReveal query is not stronger than the SessionStateReveal query, and presents a kind of attacks against some eCK-secure protocols using SessionStateReveal query, which is equivalent to say some eCK-secure protocols are not CK-secure. During the attack, the

adversary forces two distinct non-matching sessions to have the similar pre-master secrets. In that case, the adversary can select one of the sessions as the test session and use SessionStateReveal to learn the pre-master secret of the other. At this time the test session is fresh, since these two sessions are non-matching and no SessionStateReveal query is asked towards the test session. We call such kind of attacks as the pre-master secret replication (PSR) attacks.

The original eCK model fails to capture the PSR attacks. The reason is that, although the adversary can compute the pre-master secret after revealing both static private key and ephemeral private key for the same party, due to the adversarial capability and the eCK-freshness definition, the adversary is permitted to reveal the static private key of the peer for the test session. More precisely, for two non-matching sessions with the same communication participants, *sid* and sid^*, if the adversary chooses *sid* as the test session without loss of generality, then the eCK model permits the adversary to learn the shared pre-master secret of sid^*.

Thus, in the design of security model, we need to capture the PSR attacks, and to study the security properties of the pre-master secret independently.

2.2 Security of Secret Exchange Module

The secret exchange module is a communication protocol that enables two honest parties to agree on a common pre-master secret which might be a group element or a triple of group elements, and so on.

Parties. In this module, there are $n(\lambda)$ parties each modeled by a probabilistic Turing machine where $\lambda \in \mathbb{N}$ is a security parameter and $n(\cdot)$ is polynomial. Each party has a static public-private key pair together with a certificate that binds the public key to that party. $\hat{A}$ ($\hat{B}$) denotes the static public key A (B) of party $\mathcal{A}$ ($\mathcal{B}$) together with a certificate. We do not assume that the certifying authority (CA) requires parties to prove possession of their static private keys, but we require that the CA verifies that the static public key of a party belongs to the domain of public keys.

Session. In this module, two parties exchange certified static public keys $\hat{A}$, $\hat{B}$ and ephemeral public keys X, Y. A party $\mathcal{A}$ can be activated to execute an instance of the protocol called a *session*. $\mathcal{A}$ is activated by an incoming message with one of the following forms: (i) $(\hat{A}, \hat{B})$ or (ii) $(\hat{A}, \hat{B}, Y)$ where Y is $\mathcal{B}$'s ephemeral public key. If $\mathcal{A}$ is activated by $(\hat{A}, \hat{B})$ then $\mathcal{A}$ is the session *initiator*, otherwise the session *responder*.

Definition 1 (Session identifier). *The session is identified via a session identifier* $sid = (role, CPK, CPK', EPK, EPK')$*, where role is the role performed by the session (here initiator or responder),* CPK *is the certified static public key of the participant executing sid,* CPK' *is the certified static public key of the intended communication partner, and* EPK *and* EPK' *are the ephemeral public keys transferred in the protocol.*

Definition 2 (Matching sessions for two-party protocols). *For a two-party protocol, sessions sid and* sid^* *are said to be match if and only if there*

exists roles role and role (role ≠ role*), certified static public keys CPK and CPK*, and ephemeral public keys EPK and EPK*, such that the session identifier of sid is* $(role, CPK, CPK^*, EPK, EPK^*)$ *and the session identifier of sid* is* $(role^*, CPK^*, CPK, EPK, EPK^*)$.

For example, sessions $(\mathcal{I}, \hat{A}, \hat{B}, X, Y)$ and $(\mathcal{R}, \hat{B}, \hat{A}, X, Y)$ are matching, where $\mathcal{I}$ denotes initiator and $\mathcal{R}$ denotes responder.

Adversarial capability. The adversary $\mathcal{M}$ is modeled as a probabilistic Turing machine and makes oracle queries to honest parties. It means that the adversary can control all communications between the protocol participants. The adversary presents parties with incoming messages via $\mathsf{Send}(message)$, thereby controlling the activation of sessions. Here we give the adversary additional capability to reveal the pre-master secret. In order to formalize possible leakage of private information, we define the following oracle queries:

- $\mathsf{EphemeralKeyReveal}(sid)$: The adversary obtains the ephemeral private key held by the session *sid*.
- $\mathsf{Pre\text{-}masterSecretReveal}(sid)$: The adversary obtains the pre-master secret for the session *sid*. If the corresponding secret does not exist, $\perp$ is returned.
- $\mathsf{StaticKeyReveal}(party)$: The adversary learns the static private key of the party.
- $\mathsf{Establish}(party)$: This query allows the adversary to register a static public key on behalf of a party. In this way the adversary totally controls that party. Parties against whom the adversary does not issue this query are called *honest*.

Adversarial goal. The aim of the adversary $\mathcal{M}$ against the secret exchange module is to compute the pre-master secret of a *fresh session* (which is defined in Definition 3). We capture this by defining a special query $\mathsf{TestS}(sid, \sigma')$:

- $\mathsf{TestS}(sid, \sigma')$: This query can be asked by the adversary only once during its entire execution. σ' is the adversary's guess for the pre-master secret of session *sid*. If no pre-master secret σ of session *sid* is defined, or the session *sid* is not fresh, then an error symbol $\perp$ is returned. Otherwise, if $\sigma' = \sigma$ then return 1; if $\sigma' \neq \sigma$ then return 0.

Formally, the goal of $\mathcal{M}$ is to compute a guess σ' for the fresh session *sid* and make $\mathsf{TestS}(sid, \sigma')$ output 1, which means $\mathcal{M}$ wins the SE-game. For an AKE protocol $\mathcal{P}$, we denote by $\mathbf{Succ}^{\mathsf{se}}_{\mathcal{P}}(\mathcal{M})$ the probability that $\mathcal{M}$ correctly compute the secret of a fresh session; more precisely $\mathbf{Succ}^{\mathsf{se}}_{\mathcal{P}}(\mathcal{M}) = \Pr[\mathsf{TestS}(sid, \sigma') = 1]$

Before we define the security of secret exchange module, the freshness notion captures the intuitive fact that a pre-master secret can not be trivially known to the adversary.

Definition 3 (Fresh session). *Let sid be the session identifier of a completed session, owned by an honest party $\mathcal{A}$ with peer $\mathcal{B}$, who is also honest. Let sid* be the session identifier of the matching session of sid, if it exists. Define sid to be fresh if none of the following conditions holds:*

1. $\mathcal{M}$ *issues a* Pre-masterSecretReveal(sid) *or a* Pre-masterSecretReveal(sid^*) *query (if* sid^* *exists).*
2. sid^* *exists and* $\mathcal{M}$ *makes either of the following queries:*
   - *both* StaticKeyReveal$(\mathcal{A})$ *and* EphemeralKeyReveal(sid)*, or*
   - *both* StaticKeyReveal$(\mathcal{B})$ *and* EphemeralKeyReveal(sid^*)*;*
3. sid^* *does not exist and* $\mathcal{M}$ *makes either of the following queries:*
   - *both* StaticKeyReveal$(\mathcal{A})$ *and* EphemeralKeyReveal(sid)*, or*
   - StaticKeyReveal$(\mathcal{B})$*.*

Definition 4 (SE security). *The secret exchange module* $\mathsf{SE}_{\mathcal{P}}$ *of an AKE protocol* $\mathcal{P}$ *is secure if both the following conditions hold:*

1. *If two honest parties complete matching sessions then, except with negligible probability, they both compute the same pre-master secret (or both output indication of protocol failure).*
2. *The polynomially bounded adversary* $\mathcal{M}$ *can compute the pre-master secret with a negligible probability, namely,* $\mathbf{Succ}^{\mathsf{se}}_{\mathcal{P}}(\mathcal{M})$ *is negligible in* λ*.*

2.3 Security of Key Derivation Module

In the key derivation module, the users put the pre-master secret and some auxiliary information into a key derivation function to derive a session key. This module amplifies the security of the secret exchange module. We present the adversarial capability of the key derivation module, and define the security that this module should satisfy.

Adversarial capability. The key derivation module $\mathsf{KD}_{\mathcal{P}}$ of the protocol $\mathcal{P}$ can be described as a function $\mathsf{KDF}_{\mathcal{P}}$ with the inputs σ produced by the secret exchange module and the optional string O (may be the user's id, and the protocol transcripts). More precisely,

$$\mathsf{KDF}_{\mathcal{P}} : \Delta \times \{0,1\}^{l_o} \rightarrow \{0,1\}^{l_{sk}}$$

where Δ is the given domain, and l_o is the length of the optional string, and l_{sk} is the length of the session key. The adversary against this module is given several queries:

- SessionKeyReveal(sid): The adversary obtains the session key for a session sid.
- TestK(sid): This query tries to capture the adversary's ability to tell apart a real session key from a random one. This query can be asked by an adversary only once during its entire execution. If no secret σ for the session sid is defined, or the session sid is not fresh, then an error symbol $\perp$ is returned. Otherwise, a private coin bk is flipped, and $\mathsf{KDF}_{\mathcal{P}}(\sigma, O)$ is returned if $bk = 1$ or a random l_{sk}-bit string is returned if $bk = 0$.

Since the SessionKeyReveal query is introduced, the definition of freshness described in Section 2.2 should be changed.

Definition 5 (Fresh session). *We say a session sid is fresh if it satisfies the Definition 3 and no* SessionKeyReveal(*sid*) *or* SessionKeyReveal(sid^*) *(if sid's matching session sid^* exists) is asked by the adversary $\mathcal{M}$.*

Adversarial goal. The goal of the adversary $\mathcal{M}$ is to distinguish the session key for the fresh session from a random key, which means $\mathcal{M}$ wins the KD-game. More precisely, the adversary $\mathcal{M}$ wants to guess a bit bk' such that $bk' = bk$ where bk is the private bit involved in TestK query. We define the advantage of the adversary $\mathcal{M}$ by $\mathbf{Adv}^{\mathsf{kd}}_{\mathcal{P}}(\mathcal{M}) = |2\Pr[bk' = bk] - 1|$.

Definition 6 (KD security). *The key derivation module $KD_{\mathcal{P}}$ of an AKE protocol $\mathcal{P}$ is secure if $\mathbf{Adv}^{kd}_{\mathcal{P}}(\mathcal{M})$ is negligible for any polynomially bounded adversary $\mathcal{M}$.*

2.4 Enhanced eCK-Security

After defining the security of secret exchange module and key derivation module, a modular security definition is proposed by treating an AKE protocol as two relative but independent modules. We will show that the enhanced eCK-security is stronger than the original eCK-security [5].

Definition 7 (Enhanced eCK-security). *We say an AKE protocol $\mathcal{P}$ is secure if its secret exchange module and key derivation module are both secure.*

One can see that the enhanced security is stronger than the eCK-security, since more adversarial capabilities are defined. The following theorem shows the fact.

Theorem 1. *The enhanced eCK-security is stronger than eCK security. More precisely, both the following cases hold:*

1. *If any AKE protocol $\mathcal{P}$ satisfies the enhanced eCK-security, then $\mathcal{P}$ also satisfies the original eCK-security.*
2. *There exists an AKE protocol $\mathcal{P}$ satisfies the original eCK-security, but does not satisfy the enhanced eCK-security.*

Proof. We prove the two cases in Theorem 1 both hold.

Case 1: We prove this by contradiction. Assume there exist a probabilistic polynomial-time (PPT) adversary $\mathcal{M}$ can break the eCK-security of $\mathcal{P}$ with a non-negligible probability. Then we can use $\mathcal{M}$ as a subroutine to construct another PPT adversary $\mathcal{T} = (\mathcal{T}_1, \mathcal{T}_2)$ that attacks the enhanced eCK-security of $\mathcal{P}$ with a non-negligible, where $\mathcal{T}_1$ is the adversary playing the SE-game and $\mathcal{T}_2$ is the adversary playing the KD-game. The construction of $\mathcal{T}$ is shown as follows:

(1) $\mathcal{T}$ simulates all the oracle queries asked by $\mathcal{M}$. Since the Send, EphemeralKeyReveal, StaticKeyReveal and Establish queries defined in the SE-game are equivalent to those in the eCK model, $\mathcal{T}_1$ can ask the SE-game challenger to answer the corresponding query asked by $\mathcal{M}$. Similarly, SessionKeyReveal

and TestK queries defined in the KD-game are equivalent to SessionKeyReveal and Test in eCK model, hence $\mathcal{T}_2$ can ask the KD-game challenger to answer the corresponding query asked by $\mathcal{M}$

(2) $\mathcal{T}$ runs $\mathcal{M}$ as a subroutine and answers $\mathcal{M}$'s oracle queries.

(3) After $\mathcal{M}$ queries Test(sid) and answers with its guess b' for the private coin involved in Test query, $\mathcal{T}_2$ set $bk' = b'$ and takes bk' as $\mathcal{T}_2$'s guess for the private coin involved in TestK.

From the construction, if $\mathcal{M}$ correctly guesses the private coin involved in Test query with a non-negligible probability, then $\mathcal{T}_2$ can guess the private coin involved in TestK query with a non-negligible probability. By Definition 7, that is equivalent to say if $\mathcal{M}$ breaks eCK-security with a non-negligible probability, then $\mathcal{T}$ breaks the enhanced eCK-security with a non-negligible probability. Thus, case 1 holds.

Case 2: We prove this by showing an eCK-secure protocol named as NAXOS [5] is insecure in the enhanced eCK-model. The attacking idea is similar to [4], but here we use the Pre-masterSecretReveal query to describe the leakage of the shared pre-master secret. Due to space limitation, we recommend Cremers' work [4] to the readers.

3 The Proposed AKE Protocol: NACS

In this section, we obtain a new AKE protocol named as NACS from NAXOS [5], CMQV [13] and SMQV [12]. Compared with the related work in the standard model [9,7], NACS has the following advantages: (i) it is provably secure with a tight reduction under weaker cryptographic assumptions; (ii) it requires less exponentiations.

NACS design principles are briefly explained before we describe the protocol.

1. The most important design goal is to prevent the pre-master secret replication attacks. Our solution is hashing the session identifiers into the exponent of the pre-master secret such that the pre-master secrets for two non-matching sessions are distinct. The pre-master secret computation method is derived from CMQV and SMQV.
2. We introduce $\mathsf{CDH}(X, Y)$ in the pre-master secret, in order to get rid of the Forking Lemma [11] and provide a tight security reduction. Due to space limitation, we recommend the readers compare Appendix A with the security proof in [13,12] to get more details.
3. We use the NAXOS trick [5] to compute the ephemeral public keys, which can simplify the security proof. Recently, it is argued that the NAXOS trick is insecure [7], since the adversary can still learn the exponent of X through the side channel attacks. We assume such side channel attack is impossible. We also emphasize this assumption is reasonable, since if such attack is possible then the exponent of pre-master secret can be also leaked out in the same way and the security of the AKE protocol are totally broken.

3.1 Description

Initialization. Let $\lambda \in \mathbb{N}$ be a security parameter and $\mathbb{G} = \langle g \rangle$ be a cyclic group of prime order p with $|p| = \lambda$, which makes the Gap Diffie-Hellman (GDH) assumption [10] hold. Let H be a target collision resistant (TCR) hash function family (due to space limitation, see the definition in [9] for more details), and $h_1 \xleftarrow{\$} \mathsf{KH}_\lambda$ be an index of a TCR hash function $H_1 := \mathsf{H}_{h_1}^{\lambda,\mathcal{D}_H,\mathcal{R}_H}$, where $\mathcal{D}_H := \{0,1\}^\lambda \times \mathbb{Z}_p$ and $\mathcal{R}_H := \mathbb{Z}_p$, and $h_2 \xleftarrow{\$} \mathsf{KH}_\lambda$ be an index of a TCR hash function $H_2 := \mathsf{H}_{h_2}^{\lambda,\mathcal{D}'_H,\mathcal{R}_H}$, where $\mathcal{D}'_H := \{\mathcal{I},\mathcal{R}\} \times (\Pi_\lambda)^2 \times \mathbb{G}^2$ and Π_λ denotes the space of certified static public keys. Let F be a pseudo-random function (PRF) family (see the definition in [9] for more details) and $F := \mathsf{F}^{\lambda,\Sigma_F,\mathcal{D}_F,\mathcal{R}_F}$ where $\Sigma_F := \mathbb{G}^2$, $\mathcal{D}_F := \mathbb{G}^2 \times (\Pi_\lambda)^2$ and $\mathcal{R}_F := \{0,1\}^\lambda$. The system parameter of the proposed AKE protocol is $(\mathbb{G}, H_1, H_2, F)$.

The static private key of party $\mathcal{A}$ is $a \xleftarrow{\$} \mathbb{Z}_p$ and the static public key of $\mathcal{A}$ is $A := g^a$. Similarly, the static key pair of $\mathcal{B}$ is $(b, B := g^b)$.

NACS is treated as two modules: the secret exchange module and the key derivation module.

Secrete Exchange Module

1. Upon activation $(\hat{A}, \hat{B})$, party $\mathcal{A}$ (the initiator) performs the steps: (a) Select an ephemeral private key $\tilde{x} \xleftarrow{\$} \{0,1\}^\lambda$; (b) Compute $x = H_1(\tilde{x}, a)$ and the ephemeral public key $X = g^x$; (c) Initiate session $sid = (\mathcal{I}, \hat{A}, \hat{B}, X, -)$ and send $(\hat{B}, \hat{A}, X)$ to $\mathcal{B}$.
2. Upon receiving $(\hat{B}, \hat{A}, X)$, party $\mathcal{B}$ (the responder) verifies that $X \in \mathbb{G}$. If so, $\mathcal{B}$ performs the following steps: (a) Select an ephemeral private key $\tilde{y} \xleftarrow{\$} \{0,1\}^\lambda$; (b) Compute $y = H_1(\tilde{y}, b)$ and the ephemeral public key $Y = g^y$. (c) Compute $d = H_2(\mathcal{I}, \hat{A}, \hat{B}, X, Y)$ and $e = H_2(\mathcal{R}, \hat{B}, \hat{A}, X, Y)$; (d) Compute the secret $\sigma_B = ((X \cdot A^d)^{y+eb}, X^y) = (\mathsf{CDH}(X \cdot A^d, Y \cdot B^e), \mathsf{CDH}(X, Y))$; (e) Destroy y; (f) Complete session $sid^* = (\mathcal{R}, \hat{B}, \hat{A}, X, Y)$ with secret σ_B and send $(\hat{A}, \hat{B}, X, Y)$ to $\mathcal{A}$.
3. Upon receiving $(\hat{A}, \hat{B}, X, Y)$, party $\mathcal{A}$ checks if he owns a session with identifier $(\mathcal{I}, \hat{A}, \hat{B}, X, -)$. If so, $\mathcal{A}$ verifies that $Y \in \mathbb{G}$ and performs the following steps: (a) Compute $d = H_2(\mathcal{I}, \hat{A}, \hat{B}, X, Y)$ and $e = H_2(\mathcal{R}, \hat{B}, \hat{A}, X, Y)$; (b) Compute $\sigma_A = ((Y \cdot B^e)^{x+d \cdot a}, Y^x) = (\mathsf{CDH}(X \cdot A^d, Y \cdot B^e), \mathsf{CDH}(X, Y))$; (c) Destroy x; (d) Complete session $sid = (\mathcal{I}, \hat{A}, \hat{B}, X, Y)$ with secret σ_A.

Key Derivation Module

4. Party $\mathcal{A}$ computes the session key $K_A = F_{\sigma_A}(X, Y, \hat{A}, \hat{B})$. Similarly, $\mathcal{B}$ computes the session key $K_B = F_{\sigma_B}(X, Y, \hat{A}, \hat{B})$

If any verification fails the party erases all session specific information, which includes the ephemeral private key, from its memory and aborts the session. A graphical description of NACS is shown in Figure 1.

It is easy to see two honest users who execute NACS directly can agree on the same session key.

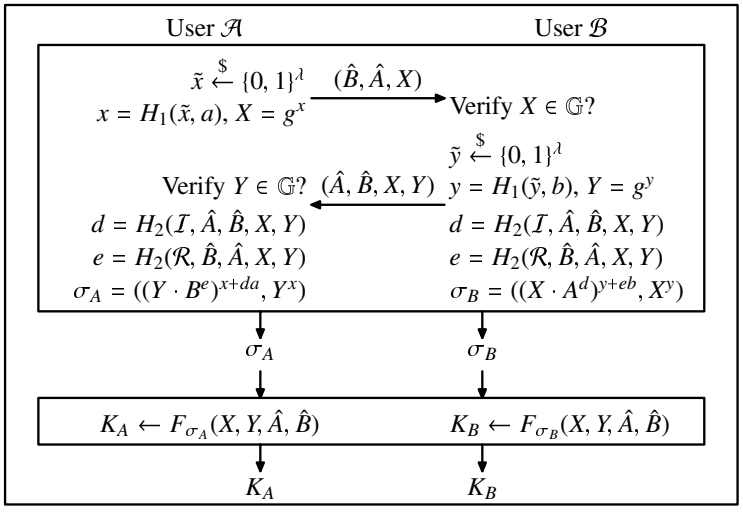

Fig. 1. An honest execution of NACS

3.2 Security

We use the enhanced security model defined in Section 2 to analyze the security of NACS. The security of NACS is presented according to the following lemmas.

Lemma 1. *The secret exchange module of NACS named as SE_{NACS} is secure (in the sense of Definition 4). More precisely, for any probabilistic polynomial-time (PPT) adversary $\mathcal{M}$ against the security of SE_{NACS} that runs in time at most t_{se}, involves at most $n(\lambda)$ honest parties and activates at most $s(\lambda)$ sessions, the success probability of $\mathcal{M}$ is*

$$\mathbf{Succ}^{\mathsf{se}}_{NACS}(\mathcal{M}) \leq \frac{s(\lambda)^2 n(\lambda)^2}{2n(\lambda)^2 + s(\lambda)} \cdot \mathbf{Succ}^{\mathsf{gdh}}_{\lambda,\mathbb{G}}(\mathcal{S}) + \frac{s(\lambda)}{2n(\lambda)^2 + s(\lambda)} \cdot \mathbf{Succ}^{\mathsf{tcr}}_{\lambda,H}(\mathcal{S}')$$

where $\mathcal{S}$ is a GDH solver and $\mathcal{S}'$ is a TCR hash function family breaker.

Proof idea: We prove this lemma by reduction. We assume an adversary $\mathcal{M}$ who can break the SE-security of SE_{NACS} with non-negligible probability. Then, given a CDH challenge (U, V) and DDH oracle, we run $\mathcal{M}$ as a subroutine to construct another adversaries to attack the GDH problem and the TCR hash function family respectively with non-negligible probability. The challenge is embedded in the simulation of test session. In the SE-game, $\mathcal{M}$ may choose a test session with or without a matching session. For the matching session case, (U, V) is embedded in the ephemeral public keys; for the non-matching session case, U is embedded in the ephemeral public key of the test session and V is embedded in the static public key of $\mathcal{B}$ who is the communication peer of the test session. With the help of TestS query, if $\mathcal{M}$ can break the SE-security then we can solve the GDH problem successfully. A detailed proof is given in the Appendix A.

Lemma 2. *The key derivation module of NACS named as KD_{NACS} is secure (in the sense of Definition 6). More precisely, for any PPT adversary $\mathcal{M}$ against the security of KD_{NACS} that runs in time at most t_{kd}, the advantage of $\mathcal{M}$ is*

$$\mathbf{Adv}^{kd}_{NACS}(\mathcal{M}) \leq 2\mathbf{Adv}^{prf}_{\lambda,\mathsf{F}}(\mathcal{S})$$

where $\mathcal{S}$ is a PRF family breaker.

Proof idea: The proof idea is reduction, to use $\mathcal{M}$ to build the attacker against the PRF family in such a way that if $\mathcal{M}$ succeeds in breaking the security of KD_{NACS} then we can break the PRF family. Our proof consists of a sequence of games, starting at the original attack game and ending in the game where the advantage of the adversary is 0, we bound the difference in the adversary's advantage between any two consecutive games. The proof is similar to [15].

By Definition 7 and Lemma 1 and 2, we can conclude the following theorem.

Theorem 2. *NACS is secure in the enhanced eCK model (See Definition 7) if the GDH assumption holds for $\mathbb{G}$, H is a TCR hash function family, and F is a PRF family.*

Remark 1. Using a pseudo-random function as a key derivation function seems to be a natural way to avoid random oracles. However, without our modular treatment (especially for the TestS query in our model), it is not easy to construct a security proof under standard assumptions, since the security reduction heavily relies on the capability of revealing the adversary's guess for pre-master secret, which is modeling as a new query (named as TestS query) in our model.

3.3 Comparison

We compare CMQV [13], SMQV [12], Okamoto [9] and Moriyama-Okamoto [7] protocols in terms of efficiency (number of exponentiations per party), security and underlying cryptographic assumptions. The comparison is presented in Table 1. eCK, seCK and eeCK stand for the extended Canetti-Krawczyk (eCK) [5], strengthened eCK [12], and enhanced eCK defined in Section 2 respectively; RO is short for random oracle model. Here the number of group exponentiations is counted in the naive way without accounting for possible improvements in the computations (such as Shamir's trick, the Algorithm 14.88 in [6], and Exponent Combination Method [8]). Several advantages of NACS can be summarized:

1. Compared with CMQV and SMQV, NACS is provably secure without the controversial random oracle assumption and the security reduction of NACS is tight, although NACS requires one more exponentiation. The security reductions of CMQV and SMQV are loose, since they rely on the Forking Lemma [11] which may introduce a wider gap in the reduction.
2. Compared with Okamoto protocol and Moriyama-Okamoto protocol which are the related works in the standard model, NACS requires less exponentiations and weaker cryptographic assumptions. πPRF used in Okamoto protocol and Moriyama-Okamoto protocol is much stronger than PRF used in

Table 1. Protocols Comparison

Protocol	Efficiency	Security	Assumption
CMQV [13]	3	eCK	GDH, RO
SMQV [12]	3	seCK	GDH, RO
Okamoto [9]	6	eCK	DDH, TCR, πPRF
Moriyama-Okamoto [7]	12	eCK	DDH, CR, πPRF
NACS	4	eeCK	GDH, TCR, PRF

NACS and it seems impossible to instantiate πPRF in practice. TCR defined in this paper is weaker than CR used in Moriyama-Okamoto protocol.

4 Conclusion

The ambition of our work is to introduce security enhancement and modular treatment towards the AKE protocols, in order to capture more attacks and provide facilitation for security proof to avoid the controversial random oracle assumption. The proposed model treats an AKE protocol as a combination of the secret exchange module and the key derivation module. By utilizing the TestS query defined in the model, we can learn the pre-master secret guessed by the adversary and use it to do security reduction in the standard model without searching the random oracle list or proposing some rather strong cryptographic assumptions. To prevent the pre-master secret replication attacks, we also propose a two-pass AKE protocol, NACS, which is provably secure in the proposed model. Compared with the related work in the standard model, NACS is more efficient and admits tight reduction with weaker assumptions. In a forthcoming stage, we will be interested in analyzing the existing AKE protocols using the enhanced security model.

References

1. Canetti, R., Goldreich, O., Halevi, S.: The random oracle methodology, revisited. Journal of the ACM 51(4), 557–594 (2004)
2. Canetti, R., Krawczyk, H.: Analysis of key-exchange protocols and their use for building secure channels. In: Pfitzmann, B. (ed.) EUROCRYPT 2001. LNCS, vol. 2045, pp. 453–474. Springer, Heidelberg (2001)
3. Cremers, C.J.: Formally and practically relating the CK, CK-HMQV, and eCK security models for authenticated key exchange. Cryptology ePrint Archive, Report 2009/253 (2009), http://eprint.iacr.org/
4. Cremers, C.J.: Session-state Reveal is stronger than Ephemeral Key Reveal: Attacking the NAXOS authenticated key exchange protocol. In: Abdalla, M., Pointcheval, D., Fouque, P.-A., Vergnaud, D. (eds.) ACNS 2009. LNCS, vol. 5536, pp. 20–33. Springer, Heidelberg (2009)
5. LaMacchia, B., Lauter, K., Mityagin, A.: Stronger security of authenticated key exchange. In: Susilo, W., Liu, J.K., Mu, Y. (eds.) ProvSec 2007. LNCS, vol. 4784, pp. 1–16. Springer, Heidelberg (2007)

6. Menezes, A., van Oorschot, P.C., Vanstone, S.A.: Handbook of Applied Cryptography. CRC Press, Boca Raton (1997)
7. Moriyama, D., Okamoto, T.: An eCK-secure authenticated key exchange protocol without random oracles. In: Pieprzyk, J., Zhang, F. (eds.) ProvSec 2009. LNCS, vol. 5848, pp. 154–167. Springer, Heidelberg (2009)
8. M'Raïhi, D., Naccache, D.: Batch exponentiation: a fast dlp-based signature generation strategy. In: Gong, L., Stern, J. (eds.) Proceedings of the 3rd ACM Conference on Computer and Communications Security, CCS 1996, pp. 58–61. ACM Press, New York (1996)
9. Okamoto, T.: Authenticated key exchange and key encapsulation in the standard model. In: Kurosawa, K. (ed.) ASIACRYPT 2007. LNCS, vol. 4833, pp. 474–484. Springer, Heidelberg (2007)
10. Okamoto, T., Pointcheval, D.: The gap-problems: a new class of problems for the security of cryptographic schemes. In: Kim, K.-c. (ed.) PKC 2001. LNCS, vol. 1992, pp. 104–118. Springer, Heidelberg (2001)
11. Pointcheval, D., Stern, J.: Security arguments for digital signatures and blind signatures. Journal of Cryptology 13(3), 361–396 (2000)
12. Sarr, A.P., Elbaz-Vincent, P., Bajard, J.-C.: A new security model for authenticated key agreement. In: Garay, J.A., De Prisco, R. (eds.) SCN 2010. LNCS, vol. 6280, pp. 219–234. Springer, Heidelberg (2010)
13. Ustaoglu, B.: Obtaining a secure and efficient key agreement protocol from (H)MQV and NAXOS. Designs, Codes and Cryptography 46(3), 329–342 (2008)
14. Ustaoglu, B.: Comparing SessionStateReveal and EphemeralKeyReveal for Diffe-Hellman protocol. In: Pieprzyk, J., Zhang, F. (eds.) ProvSec 2009. LNCS, vol. 5848, pp. 183–197. Springer, Heidelberg (2009)
15. Wang, L., Pan, J., Ma, C.: A modular proof technique for password-based authenticated key exchange protocols. In: INSCRYPT 2010 (2010)

A. Proof of Lemma 1

At the beginning of the proof, we exclude the case in which the adversary $\mathcal{M}$ guesses $str \neq (\tilde{x}, a)$ and $str' \neq (\tilde{y}, b)$ such that $H_1(\tilde{x}, a) = H_1(str)$ and $H_1(\tilde{y}, b) = H_1(str')$. After that $\mathcal{M}$ queries $\mathsf{StaticKeyReveal}(\mathcal{A})$ or $\mathsf{StaticKeyReveal}(\mathcal{B})$. In this way, $\mathcal{M}$ can compute σ_A (or σ_B) successfully, and we can derive another adversary $\mathcal{S}'$ to attack TCR hash function family with str or str'. However, by the definition of the TCR hash function family, the probability of this event is at most $\mathbf{Succ}^{\mathsf{tcr}}_{\lambda,\mathsf{H}}(\mathcal{S}')$.

We assume $\mathcal{M}$ can win SE-game with a non-negligible probability, then we can construct a Gap Diffie-Hellman (GDH) problem solver $\mathcal{S}$ which uses $\mathcal{M}$ as a subroutine. More precisely, $\mathcal{S}$ takes as input a GDH challenge (U, V), where we denote u as $\mathsf{DLOG}(U)$ and v as $\mathsf{DLOG}(V)$. Then $\mathcal{S}$ executes the SE-game with $\mathcal{M}$ against the secret exchange module $\mathsf{SE}_{\mathrm{NACS}}$ of NACS and modifies the data returned by the honest parties in such a way that if $\mathcal{M}$ breaks the SE-security of $\mathsf{SE}_{\mathrm{NACS}}$, then $\mathcal{S}$ can get the solution to the GDH problem from $\mathcal{M}$. In the following, two complementary cases are considered respectively: the test session has a matching session; and the test session do not have a matching session.

Matching Session Exists. Assume that $\mathcal{M}$ always selects a test session for which the matching session exists. The event SUCCS_1 denotes $\mathcal{M}$ wins SE-game in this case. Assume the success probability $\Pr[\text{SUCCS}_1]$ is non-negligible. In this case, $\mathcal{S}$ establishes $n(\lambda)$ honest parties and selects two sessions sid and sid^* at random. Suppose that $\mathcal{M}$ selects one of these sessions as the test session and the other as its matching session; if not then $\mathcal{S}$ aborts. The simulation follows the protocol description except for sessions sid and sid^*. For these sessions, $\mathcal{S}$ selects ephemeral private keys $\tilde{x}, \tilde{y} \xleftarrow{\$} \{0,1\}^\lambda$, and sets the ephemeral public keys X as U and Y as V. Both sessions are assigned the same pre-master secret $(T, T') \xleftarrow{\$} \mathbb{G}^2$. Moreover, the simulation of TestS query is changed as follows:

- $\mathsf{TestS}(sid, \sigma')$: Compile the σ' as the form (σ'_1, σ'_2). Query the DDH oracle with $(U \cdot A^d, V \cdot B^e, \sigma'_1)$ and (U, V, σ'_2). If $\mathsf{DDH}(U, V, \sigma'_2) = 1$ then we compute $\mathsf{CDH}(U, V) = \sigma'$ and answer the adversary with 1 if $\mathsf{DDH}(U \cdot A^d, V \cdot B^e, \sigma'_1) = 1$ or 0 if $\mathsf{DDH}(U \cdot A^d, V \cdot B^e, \sigma'_1) = 0$; otherwise, we answer the adversary with 0.

Analysis. The simulation is perfect except with negligible probability. Without loss of generality, assume sid is the test session with the matching session sid^*, and $\mathcal{A}$ is the initiator of session sid with peer $\mathcal{B}$. The simulation fails if and only if $\mathcal{M}$ reveals $(\tilde{x}, a)$ and finds out $g^{H_1(\tilde{x}, a)} \neq X$, or $\mathcal{M}$ reveals the pre-master secret of sid and learns that $T \neq \mathsf{CDH}(X \cdot A^d, Y \cdot B^e)$ and $T' \neq \mathsf{CDH}(X, Y)$. For the freshness of the test session, $\mathcal{M}$ can not query $\mathsf{Pre\text{-}masterSecretReveal}(sid)$, or query both $\mathsf{EphemeralKeyReveal}(sid)$ and $\mathsf{StaticKeyReveal}(\mathcal{A})$. Thus, the simulation will fail with a negligible probability.

With probability at least $2/s(\lambda)^2$ the adversary selects one of the two sessions sid and sid^* as the test session and the other as its matching session. From the simulation of TestS query, if $\mathcal{M}$ wins the SE-game successfully, then $\mathcal{S}$ can correctly solve the GDH problem. The success probability of $\mathcal{S}$ is

$$\Pr[\mathcal{S} \text{ succeeds}] \geq \frac{2}{s(\lambda)^2} \Pr[\text{SUCCS}_1] \tag{1}$$

No Matching Session Exists. Assume now that $\mathcal{M}$ always selects a test session such that the matching session does not exist. The event SUCCS_2 denotes $\mathcal{M}$ wins SE-game in this case. Assume the success probability $\Pr[\text{SUCCS}_2]$ is non-negligible. In this case, $\mathcal{S}$ selects at random two distinct parties $\mathcal{A}$ and $\mathcal{B}$, and a session sid. Furthermore, for the party $\mathcal{B}$, $\mathcal{S}$ selects a random static private key b and assigns the static public key B equal to V. The remaining $n(\lambda) - 1$ parties are assigned random static key pairs. We suppose that $\mathcal{M}$ selects sid as the test session, and $\mathcal{A}$ is the owner of sid and $\mathcal{B}$ its peer; if not $\mathcal{S}$ aborts. The protocol simulation follows the definition. The exceptions are: (1) for the test session sid, $\mathcal{S}$ selects $\tilde{x} \xleftarrow{\$} \{0,1\}^\lambda$, sets the ephemeral public key X equal to U, and chooses a random pair (T, T') as the pre-master secret; (2) for the test session sid, if $\mathcal{M}$ establishes a session sid' ($sid' \neq sid$) which is not matching to sid but $H_2(sid') = H_2(sid)$, then $\mathcal{S}$ aborts and declares the adversary successful.

In this way, we can derive another adversary $\mathcal{S}'$ to attack TCR hash function family with sid'. By the definition of TCR hash function family, this event will happen with the probability at most $\mathbf{Succ}^{\mathsf{tcr}}_{\lambda,\mathsf{H}}(\mathcal{S}')$. Moreover, the simulation of TestS query is changed as follows:

– $\mathsf{TestS}(sid, \sigma')$: Compile σ' as the form (σ'_1, σ'_2). Query DDH oracle with $(U \cdot A^d, Y \cdot V^e, \sigma'_1)$ and (U, Y, σ'_2). If $\mathsf{DDH}(U \cdot A^d, Y \cdot V^e, \sigma') = 1$ and $\mathsf{DDH}(U, Y, \sigma'_2) = 1$ then we answer the adversary with 1 and compute $\mathsf{CDH}(U, V) = (\sigma'_1 \cdot Y^{-ad} \cdot V^{-ade}/\sigma'_2)^{e^{-1}}$; otherwise, we answer the adversary with 0.

Analysis. The simulation is perfect except with negligible probability. The simulation fails if and only if one of the following events has happened: (1) $\mathcal{M}$ learns that $B \neq g^b$; (2) $\mathcal{M}$ recognizes that $X \neq g^{H_1(\tilde{x},a)}$; (3) $\mathcal{M}$ finds out that $T \neq \mathsf{CDH}(X \cdot A^d, Y \cdot B^e)$ and $T' \neq \mathsf{CDH}(X, Y)$. By the freshness definition of the test session, for case (1), $\mathcal{M}$ can not query $\mathsf{StaticKeyReveal}(\mathcal{B})$ to get b; for case (2), $\mathcal{M}$ is permitted to query both $\mathsf{EphemeralKeyReveal}(sid)$ and $\mathsf{StaticKeyReveal}(\mathcal{A})$ to get $(\tilde{x}, a)$; for case (3), $\mathcal{M}$ can not query $\mathsf{Pre\text{-}masterSecret\text{-}Reveal}(sid)$. Thus, the simulation will fail with a negligible probability.

With probability at least $1/s(\lambda)$ $\mathcal{M}$ selects session sid as the test session. With probability at least $1/n(\lambda)^2$ $\mathcal{M}$ selects $\mathcal{A}$ as the owner and $\mathcal{B}$ as the peer of session sid. From the simulation of TestS query, one can see that if $\mathcal{M}$ correctly computes a σ' and makes the TestS query output 1, then $\mathcal{S}$ can solve the GDH problem. The reason is that if $\mathsf{DDH}(U \cdot A^d, Y \cdot V^e, \sigma'_1) = 1$ and $\mathsf{DDH}(U, Y, \sigma'_2) = 1$ then $\sigma'_1 = (g^{u+ad})^{y+ve} = g^{uy+uve+ady+adve}$ and $\sigma'_2 = g^{uy}$ (where $y = \mathsf{DLOG}(Y)$). Hence, $\Pi = \frac{\sigma'_1 \cdot Y^{-ad} \cdot V^{-ade}}{\sigma'_2} = g^{uve}$. Then, $\mathsf{CDH}(U, V) = \Pi^{e^{-1}}$. The success probability of $\mathcal{S}$ is

$$\Pr[\mathcal{S} \text{ succeeds}] \geq \frac{1}{s(\lambda)} \cdot \frac{1}{n(\lambda)^2} \cdot (\Pr[\textsc{Succ}S_2] - \mathbf{Succ}^{\mathsf{tcr}}_{\lambda,\mathsf{H}}(\mathcal{S}')) \tag{2}$$

Concluding Lemma 1. Due to these complementary cases, especially for formulas (1) and (2), we get

$$\Pr[\mathcal{S} \text{ succeeds}] \geq \frac{2}{s(\lambda)^2} \Pr[\textsc{Succ}S_1] + \frac{1}{s(\lambda)} \cdot \frac{1}{n(\lambda)^2} \cdot (\Pr[\textsc{Succ}S_2] - \mathbf{Succ}^{\mathsf{tcr}}_{\lambda,\mathsf{H}}(\mathcal{S}'))$$

By the definition of GDH and the SE-security, we rewrite the formula, and have

$$\mathbf{Succ}^{\mathsf{se}}_{NACS}(\mathcal{M}) \leq \frac{s(\lambda)^2 n(\lambda)^2}{2n(\lambda)^2 + s(\lambda)} \cdot \mathbf{Succ}^{\mathsf{gdh}}_{\lambda,\mathbb{G}}(\mathcal{S}) + \frac{s(\lambda)}{2n(\lambda)^2 + s(\lambda)} \cdot \mathbf{Succ}^{\mathsf{tcr}}_{\lambda,\mathsf{H}}(\mathcal{S}')$$