Efficient eCK-Secure Authenticated Key Exchange Protocols in the Standard Model

Zheng Yang

Horst Grtz Institute for IT Security
Ruhr-University Bochum, Germany
zheng.yang@rub.de

Abstract. The extended CanettiKrawczyk (eCK) security model, is widely used to provide security arguments for authenticated key exchange protocols that capture leakage of various kinds of secret information like the long-term private key and session-specific secret state. In this paper, we study the open problem on constructing eCK secure AKE protocol without random oracles and NAXOS like trick. A generic construction GC-KKN satisfying those requirements is first given relying on standard cryptographic primitives. On the second a concrete protocol is proposed which is the first eCK secure protocol in the standard model under both standard assumptions and post-specified peer setting. Both proposed schemes can be more efficiently implemented with secure device than previous eCK secure protocols in the standard model, where the secure device might be normally used to store the long-term private key and implement algorithms of protocol which require to be resilience of state leakage.

Keywords: eCK model, authenticated key exchange, standard model, key encapsulation mechanism, non-interactive key exchange.

1 Introduction

Authenticated Key Exchange (AKE) is a fundamental cryptographic primitive which forms a crucial component in many network protocols. The security model for two party AKE and associated definitions have been evolved over years subjecting to increasing security requirements. Recently the extended Canetti-Krawczyk (eCK) [9] model is widely used to provide security arguments for AKE protocols. The eCK model is known to be one of the strongest AKE models that covers the most desirable security attributes for AKE including resistance to key compromise impersonation (KCI) attacks, leakage of secret states and chosen identity and public key (CIDPK) attacks and provision of weak perfect forward secrecy (wPFS). Nevertheless the eCK model leaves out the definition of session state or ephemeral key to specific protocols. Since it is hard to define session state in a general approach, which is independent of any protocols and corresponding implementation scenarios. However the ambiguities on session state may yield a lot of potential problems in either the protocol construction or its security

S. Qing et al. (Eds.): ICICS 2013, LNCS 8233, pp. 185–193, 2013.

analysis. If any implementer realizes a specific AKE protocol in a *careless* way allowing it to leak non-trivial session state to attackers, then it would trivially invalidate the security proof in such strong model. On the other hand, to our best of knowledge, no AKE protocol is secure in the eCK model if *all* session states can be revealed. Namely some session states of AKE protocols should be leakage resilience.

Implementation Model vs. Session States. In order to fulfil the gap that often exists between formal models and practical security, Sarr et al. [12] introduced two implementation scenarios for the situation that at each party an untrusted host machine is used together with a secure device such as tamper-proof smart card. A secure device may usually be used to store long-term cryptographic authentication keys and at least be able to fulfil a library of mathematical functions which are necessary to implement cryptographic operations or primitives. Hence based on secure device we are able to adopt a 'All-and-Nothing' strategy to define the states that can be revealed without leaving any ambiguity. General speaking we could assume that all intermediate states and ephemeral keys generated on host machine are susceptible to the maximum state leakage (MSL) attacks, but we treat the secure device as a black-box which is immune to leakage of internal states. On the other side, our goal is to define the maximum states that can be leaked. As those secure devices might be short in both storage capacity and computational resource, the algorithm on secure device is often causing performance bottleneck of systems. In addition, the communication round between host machine and secure device (which is called *HS-round* for short) might cause another efficiency problem, since the serial I/O bus of most secure devices is too slow. Due to those facts, it is necessary to optimize AKE protocols when they are realized involving secure device.

Motivating Problem. So far there are only few AKE protocols which are provably secure without random oracles in the eCK model. Although the protocols [11,10] have been proven to be eCK secure in the standard model, they require a rather strong class of pseudo-random function family with pairwise independent random sources (which is referred to as $\pi\mathsf{PRF}$) as key derivation function (KDF). Most recently, Fujioka et al. [7] introduced a generic construction for two-message AKE from key encapsulation mechanism (KEM) which is generalized from BCNP [3]. Although the FSXY scheme [7] has been shown to be CK+ (eCK) secure in the standard model, it is built relying on a special twisted-PRF trick (which is a variant of NAXOS trick and is first used in [11]).[1] However to securely implement the FSXY protocol, one might need to distribute all computations related to NAXOS trick on secure device in order to resist with the leakage of corresponding states (see detail discussion in [13]). This would lead to inefficiencies in the implementation of the FSXY protocol with secure device. Another drawback of FSXY protocol is not a one-round AKE protocol. Since two session participants cannot execute the FSXY protocol instances

[1] The CK+ model can be seen as a variant of eCK model in which the StateReveal query is used instead of EphemeralKeyReveal query to model MSL attacks.

simultaneously. To our best of knowledge it is still an open question to construct eCK secure one-round AKE protocols without random oracles and NAXOS trick, and under standard assumptions (e.g. without πPRF).

Contributions. We first present a one-round authenticated key exchange protocol (named GC-KKN) to solve the above open problem. As opposed to FSXY scheme, GC-KKN does not rely on any NAXOS like trick that yields a more efficient solution when it is implemented with secure device. We give compact game-based proofs reducing eCK security of GC-KKN to break the used cryptographic primitives without random oracles.

On the second we present a concrete and practical AKE protocol (P1) that is eCK secure under standard assumptions (e.g. without πPRF). The proposed protocol is based on bilinear pairings, target collision resistant hash function family, and pseudo-random function family. To be of independent interesting, P1 is able to run under post-specified peer setting [5] (i.e. without knowing any information of communication peer at session activation), unlike FSXY scheme and our GC-KKN scheme which might be executed under only pre-specified peer setting. Our construction idea of P1 is inspired by the GC-KKN. In order to securely implement P1, only one exponentiation is required on secure device that is the more efficient than any previous eCK secure protocols without random oracles.

2 Preliminaries

Notations. We let $\kappa \in \mathbb{N}$ denote the security parameter and 1^κ the string that consists of κ ones. Let a 'hat' on top of a capital letter denote an identity; without the hat the letter denotes the public key of that party. Let $[n] = \{1, \ldots, n\} \subset \mathbb{N}$ be the set of integers between 1 and n. If S is a set, then $a \xleftarrow{\$} S$ denotes the action of sampling a uniformly random element from S. Let $\mathcal{IDS}$ be an identity space. Let $\mathcal{K}_{\mathsf{AKE}}$ be the key space of session key, and $\{\mathcal{PK}, \mathcal{SK}\}$ be key spaces for long-term public/private key respectively. Those spaces are associated with security parameter κ.

In our constructions, we will make use of one-round passively secure key exchange protocols KE, IND-CCA secure key encapsulation mechanism schemes KEM, CKS-light secure [6] non-interactive key exchange protocols NIKE, strong randomness extractor SEXT, pseudo-random function family PRF, symmetric bilinear groups and Bilinear Decisional Diffie-Hellman (BDDH) assumption.

Meanwhile, a one-round $\mathsf{KE} = (\mathsf{KE.Setup}, \mathsf{KE.EGen}, \mathsf{KE.SKGen})$ protocol consists of three algorithms, where KE.EGen is the ephemeral key generator and KE.SKGen is the session key generator. In our construction should satisfy the following two conditions: (i) the protocol is executed without any long-term keys; (ii) all ephemeral public/secret key are chosen freshly and randomly from corresponding key spaces for each protocol instance. A $\mathsf{KEM} = (\mathsf{KEM.Setup}, \mathsf{KEM.Gen}, \mathsf{KEM.EnCap}, \mathsf{KEM.DeCap})$ scheme consists of four polynomial time algorithms, where KEM.Setup is the initiation

algorithm used to generate the system parameters, KEM.Gen is the key generation which outputs a pair of long-term keys, KEM.EnCap is the encryption algorithm and KEM.DeCap is the decryption algorithm. Furthermore, a NIKE = (NIKE.Setup, NIKE.KGen, NIKE.ShKey) scheme consists of three algorithms , where NIKE.KGen is the long-term key generation algorithm and NIKE.ShKey is the algorithm that is used to compute the long-term shared secret key between two parties. The corresponding security definitions are detailed in the full version of this paper [13].

3 Security Model

In this section we present the eCK security model for two party PKI-based authenticated key-exchange (AKE) protocol. We provide an 'execution environment' for active adversaries following an important research line research [2,4,9,8] which is initiated by Bellare and Rogaway [1].

Execution Environment. In the execution environment, we fix a set of honest parties $\{\mathsf{ID}_1, \ldots, \mathsf{ID}_\ell\}$ for $\ell \in \mathbb{N}$, where ID_i ($i \in [\ell]$) is the identity of a party which is chosen uniquely from space $\mathcal{IDS}$. Each identity is associated with a long-term key pair $(sk_{\mathsf{ID}_i}, pk_{\mathsf{ID}_i}) \in (\mathcal{SK}, \mathcal{PK})$. Each honest party ID_i can sequentially and concurrently execute the protocol multiple times with different intended partners, this is characterized by a collection of oracles $\{\pi_i^s : i \in [\ell], s \in [d]\}$ for $d \in \mathbb{N}$. Moreover, we assume each oracle π_i^s maintains a list of independent internal state variables with following semantics: (i) Ψ_i^s – storing the identity of its communication partner; (ii) Φ_i^s – denoting the decision $\Phi_i^s \in \{\texttt{accept}, \texttt{reject}\}$; (iii) K_i^s – recording the session key $K_i^s \in \mathcal{K}_{\mathsf{AKE}}$; (iv) st_i^s – storing the maximum secret states that allow to be revealed; (v) sT_i^s – recording the transcript of messages sent by oracle π_i^s; (vi) rT_j^t – recording the transcript of messages received by oracle π_i^s. All those variables of each oracle are initialized with empty string which is denoted by the symbol $\emptyset$. At some point, each oracle π_i^s may complete the execution always with a decision state $\Phi_i^s \in \{\texttt{accept}, \texttt{reject}\}$. Furthermore, we assume that the session key is assigned to the variable K_i^s (such that $K_i^s \neq \emptyset$) iff oracle π_i^s has reached an internal state $\Phi_i^s = \texttt{accept}$.

Adversarial Model. An adversary $\mathcal{A}$ in our model is a PPT Turing Machine taking as input the security parameter 1^κ and the public information (e.g. generic description of above environment), which may interact with these oracles by issuing the following queries.

- $\mathsf{Send}(\pi_i^s, m)$: The adversary can use this query to send any message m of his own choice to oracle π_i^s. The oracle will respond the next message m^* (if any) to be sent according to the protocol specification. Oracle π_i^s would be initiated as *initiator* via sending the oracle the first message $m = (\top, \widetilde{\mathsf{ID}_j})$ consisting of a special initialization symbol $\top$ and a value $\widetilde{\mathsf{ID}_j}$, where the $\widetilde{\mathsf{ID}_j}$ is either the identity ID_j of intended partner or empty string $\emptyset$. After answering a Send query, the internal state variables of π_i^s will be updated depending on the specific protocol.

- $\mathsf{RevealKey}(\pi_i^s)$: Oracle π_i^s responds with the contents of variable K_i^s.
- $\mathsf{StateReveal}(\pi_i^s)$: Oracle π_i^s responds with the contents of variable st_i^s.[2]
- $\mathsf{Corrupt}(\mathsf{ID}_i)$: Oracle π_i^1 responds with the long-term secret key sk_{ID_i} of party ID_i if $i \in [\ell]$; otherwise a failure symbol $\perp$ is returned.
- $\mathsf{EstablishParty}(\mathsf{ID}_\tau, pk_{\mathsf{ID}_\tau})$: This query allows the adversary to register an identity ID_τ ($\ell < \tau < \mathbb{N}$) and a static public key pk_{ID_τ} on behalf of a party ID_τ, if ID_τ is unique.
- $\mathsf{Test}(\pi_i^s)$: If the oracle has state $\Phi_i^s = \texttt{reject}$ or $K_i^s = \emptyset$, then the oracle π_i^s returns some failure symbol $\perp$. Otherwise it flips a fair coin b, samples a random element K_0 from key space $\mathcal{K}_{\mathsf{AKE}}$, and sets $K_1 = K_i^s$. Finally the key K_b is returned.

Secure AKE Protocols. To formalize the notion that two oracles are engaged in an on-line communication, we define the partnership via *matching sessions*.

Definition 1. *We say that an oracle π_i^s has a* matching session *to oracle π_j^t, if π_i^s has sent all protocol messages and all the following conditions hold: (i) $\Psi_i^s = \mathsf{ID}_j$ and $\Psi_j^t = \mathsf{ID}_i$, (ii) $sT_i^s = rT_j^t$ and $rT_i^s = sT_j^t$.*

Correctness. We say an AKE protocol Σ is correct, if two oracles π_i^s and π_j^t accept with matching sessions, then both oracles hold the same session key, i.e. $K_i^s = K_j^t$.

Definition 2. *Let π_i^s be an accepted oracle with $\Psi_i^s = \mathsf{ID}_j$. Let π_j^t be an oracle (if it exists), such that π_i^s has a matching session to π_j^t. Then the oracle π_i^s is said to be* fresh *if none of the following conditions holds: (i) $\mathcal{A}$ queried $\mathsf{EstablishParty}(\mathsf{ID}_j, pk_{\mathsf{ID}_j})$; (ii) $\mathcal{A}$ queried either $\mathsf{RevealKey}(\pi_i^s)$ or $\mathsf{RevealKey}(\pi_j^t)$ (if π_j^t exists); (iii) $\mathcal{A}$ queried both $\mathsf{Corrupt}(\mathsf{ID}_i)$ and $\mathsf{StateReveal}(\pi_i^s)$; (iv) If π_j^t exists, $\mathcal{A}$ queried both $\mathsf{Corrupt}(\mathsf{ID}_j)$ and $\mathsf{StateReveal}(\pi_j^t)$; (v) If π_j^t does not exist, $\mathcal{A}$ queried $\mathsf{Corrupt}(\mathsf{ID}_j)$.*

Security Experiment $\mathsf{EXP}_{\Sigma,\mathcal{A}}^{\mathsf{AKE}}(\kappa)$: On input security parameter 1^κ, the security experiment is proceeded as a game between a challenger $\mathcal{C}$ and an adversary $\mathcal{A}$ based on an AKE protocol Σ, where the following steps are performed:

1. At the beginning of the game, the challenger $\mathcal{C}$ implements the collection of oracles $\{\pi_i^s : i \in [\ell], s \in [d]\}$, and generates ℓ long-term key pairs $(pk_{\mathsf{ID}_i}, sk_{\mathsf{ID}_i})$ for all honest parties ID_i for $i \in [\ell]$ where the identity $\mathsf{ID}_i \in \mathcal{IDS}$ of each party is chosen uniquely. $\mathcal{C}$ gives adversary $\mathcal{A}$ all identities, public keys $\{(\mathsf{ID}_1, pk_{\mathsf{ID}_1}), \ldots, (\mathsf{ID}_\ell, pk_\ell)\}$ as input.
2. $\mathcal{A}$ may issue polynomial number of Send, $\mathsf{StateReveal}$, $\mathsf{Corrupt}$, $\mathsf{EstablishParty}$ and $\mathsf{RevealKey}$ queries. At some point, $\mathcal{A}$ may issue a $\mathsf{Test}(\pi_i^s)$ query to an oracle π_i^s during the game with only once.

[2] We stress that the exact meaning of the $\mathsf{StateReveal}$ must be defined by each protocol separately, and each protocol should be proven secure to resist with such kind of state leakage as its claimed, i.e., the content stored in the variable st during protocol execution. In other word, each protocol should define the protocol steps processed on secure device. Our goal is to define the maximum states that can be leaked.

3. At the end of the game, the $\mathcal{A}$ may terminate with returning a bit b' as its guess for b of Test query. Finally, 1 is returned if all following conditions hold: (i) $\mathcal{A}$ has issued a Test query to a fresh oracle π_i^s without failure, and (ii) $\mathcal{A}$ returned a bit b' which equals to b of Test-query; Otherwise 0 is returned.

Definition 3. *A correct AKE protocol Σ is called (t, ϵ)-secure if probability bound $|\Pr[\mathsf{EXP}_{\Sigma,\mathcal{A}}^{\mathsf{AKE}}(\kappa) = 1] - 1/2| \leq \epsilon$ holds for all adversaries $\mathcal{A}$ running within time t in the above security experiment and for some negligible probability $\epsilon = \epsilon(\kappa)$ in the security parameter κ.*

4 A Generic One-Round AKE Construction from KE, KEM and NIKE

In this section, we present a generic one-round authenticated key exchange protocol (denoted by GC-KKN), that is more suitable to be implemented for providing eCK security than previous works.

Protocol Description. In our generic construction, the following building blocks are required: (i) One-round key exchange scheme KE; (ii) Key encapsulation mechanism scheme KEM; (iii) Non-interactive key exchange scheme NIKE; and (iv) Strong randomness extractor SEXT; and (v) Pseudo-random function family PRF.

Set-up: To initiate the system, the public system parameters $pms := (pms^{ke}, pms^{kem}, pms^{nike}, k_{\mathsf{SEXT}})$ are firstly generated via performing $pms^{ke} \leftarrow \mathsf{KE.Setup}(1^\kappa)$, $pms^{kem} \leftarrow \mathsf{KEM.Setup}(1^\kappa)$, $pms^{nike} \leftarrow \mathsf{NIKE.Setup}(1^\kappa)$ and $k_{\mathsf{SEXT}} \xleftarrow{\$} \mathcal{S}_{\mathsf{SEXT}}$ where k_{SEXT} is the secret key of SEXT and $\mathcal{S}_{\mathsf{SEXT}}$ is a key space.

Long-term Key Generation: A party $\hat{A}$ may run algorithms $(pk_{\hat{A}}^{kem}, sk_{\hat{A}}^{kem}) \xleftarrow{\$} \mathsf{KEM.Gen}(pms^{kem})$ and $(pk_{\hat{A}}^{nike}, sk_{\hat{A}}^{nike}) \xleftarrow{\$} \mathsf{NIKE.KGen}(pms^{nike}, \hat{A})$ to generate the long-term key pair.

Protocol Execution: On input pms, the protocol between party $\hat{A}$ and party $\hat{B}$ is depicted in Fig. 1.

Session States and Implementaton Senario. We now define the session states in terms of implementation model with secure device. Basically, all states of $\mathsf{KE.EGen}$, $\mathsf{KE.SKGen}$ and $\mathsf{KEM.EnCap}$ algorithms would be stored in the state variable st. However, we assume no secret states related to $\mathsf{KEM.DeCap}$ and $\mathsf{NIKE.ShKey}$ algorithms can be revealed. This can be realized by doing all computations involving long-term private key of $\mathsf{KEM.DeCap}$ and $\mathsf{NIKE.ShKey}$ algorithms, and final session key generation on secure device.

Security Analysis. We assume without loss of generality that the maximum probability for the event that two oracles output the same ciphertext C or ephemeral public key epk, is a negligible fraction $1/2^\lambda$ where $\lambda \in N$ is a large

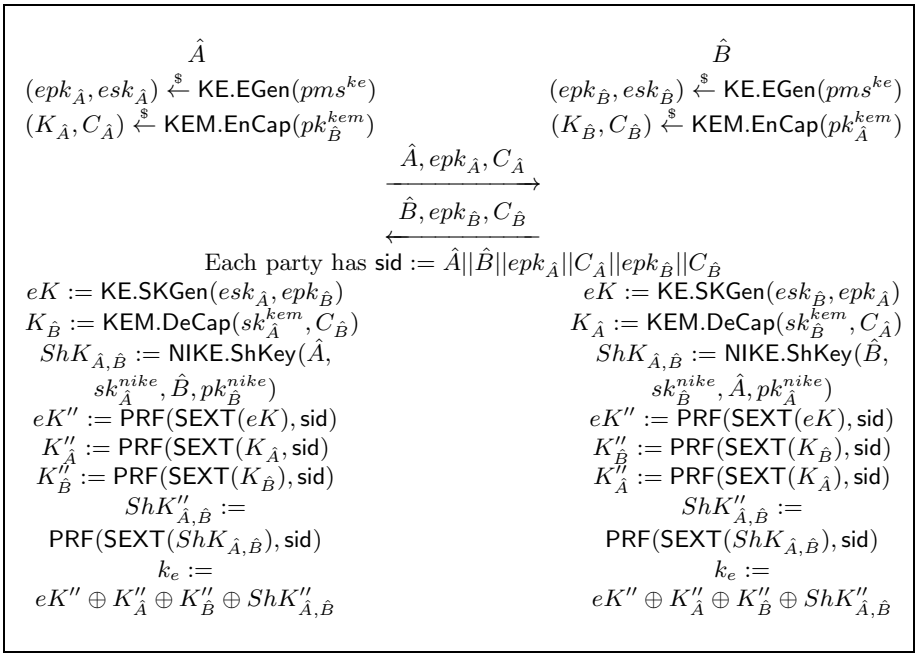

Fig. 1. Generic One-round AKE Protocol from KE, KEM and NIKE

enough integer in terms of the security parameter κ. Let $\mathsf{MAX}(X_1, X_2, X_3)$ denote the function to obtain the maximum values from variables X_1, X_2 and X_3.

Theorem 1. *Suppose that the* SEXT *is* $(\kappa, \epsilon_{\mathsf{SEXT}})$*-strong randomness extractor, the* KEM *is* $(q_{kem}, t, \epsilon_{\mathsf{KEM}})$*-secure (key indistinguishable) against adaptive chosen message attacks and* KE *is* $(t, \epsilon_{\mathsf{KE}})$*-passively secure, and the* PRF *is* $(q_{prf}, t, \epsilon_{\mathsf{PRF}})$*-secure, and the* NIKE *is* $(t, \epsilon_{\mathsf{NIKE}})$*-CKS-light-secure. And we assume that either* KE *key or* KEM *key or* NIKE *key has* κ*-min-entropy. Then the proposed protocol is* (t', ϵ)*-secure in the sense of Definition 3 with* $t' \leq t$*,* $q_{kem} \geq d$ *and* $q_{prf} \geq d+1$*, and* $\epsilon \leq \frac{(d\ell)^2}{2^\lambda} + 3(d\ell)^2 \cdot (\mathsf{MAX}(\epsilon_{\mathsf{KE}}, \epsilon_{\mathsf{KEM}}, \epsilon_{\mathsf{NIKE}}) + \epsilon_{\mathsf{SEXT}} + \epsilon_{\mathsf{PRF}})$. [3]

The proof of this theorem can be found in [13].

5 An Efficient One-Round AKE Protocol under Standard Assumptions

In this section we present a concrete eCK secure AKE protocol P1 in the standard model.

[3] The integer q_{kem} and q_{prf} are the numbers of oracle queries that can be issued in corresponding security experiment.

Protocol Description. The proposed protocol relies on standard bilinear pairings $\mathcal{PG} = (\mathbb{G}, g, \mathbb{G}_T, p, e)$ [13] along with random values $(u_1, u_2, u_3, u_4) \xleftarrow{\$} \mathbb{G}$, target collision resistant hash function family TCR and pseudo-random function family PRF. The variable pms stores the public system parameters $pms := (\mathcal{PG}, \{u_i\}_{1 \le i \le 4}, hk_{\mathsf{TCR}})$ where hk_{TCR} is the hash key of TCR and is chosen uniformly at random.

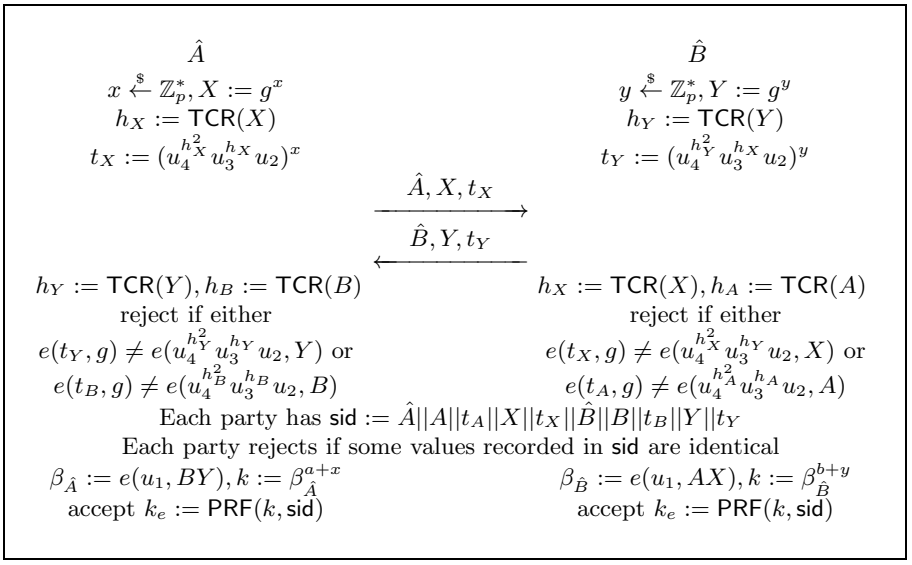

Fig. 2. Pairing-based AKE Protocol under Standard Assumptions

Long-term Key Generation: A party $\hat{A}$ may run an efficient key generation algorithm to generate the long-term key pair as: $sk_{\hat{A}} = a \xleftarrow{\$} \mathbb{Z}_p^*, pk_{\hat{A}} = (A, t_A)$ where $A = g^a$, $t_A := (u_4^{h_A^2} u_3^{h_A} u_2)^a$ and $h_A = \mathsf{TCR}(A)$.[4]

Protocol Execution: On input pms, the protocol between parties $\hat{A}$ and $\hat{B}$ is depicted in the Fig. 2.

Implementation and Session States: We assume that only the ephemeral private key x (resp. y) would be stored in the state variable st. This can be guaranteed by performing the computations of key material k and session key k_e on secure device.

Security Analysis. We show the security result of proposed protocol in our strong security model via the following theorem.

Theorem 2. *Assume each ephemeral key chosen during key exchange has bit-size $\lambda \in \mathbb{N}$. Suppose that the* BDDH *problem is $(t, \epsilon_{\mathsf{BDDH}})$-hard in the symmetric*

[4] Please note that we allow arbitrary key registration.

bilinear groups $\mathcal{PG}$*, the* TCR *is* $(t, \epsilon_{\mathsf{TCR}})$*-secure, and the* PRF *is* $(q_{prf}, t, \epsilon_{\mathsf{PRF}})$*-secure. Then the proposed protocol is* (t', ϵ)*-secure in the sense of Definition 3 with* $t' \approx t$ *and* $q_{prf} \geq 2$ *and* $\epsilon \leq \frac{(d\ell)^2}{2^\lambda} + \epsilon_{\mathsf{TCR}} + 3(d\ell)^2 \cdot (\epsilon_{\mathsf{BDDH}} + \epsilon_{\mathsf{PRF}})$.

The proof of this theorem can be found in [13].

References

1. Bellare, M., Rogaway, P.: Entity authentication and key distribution. In: Stinson, D.R. (ed.) CRYPTO 1993. LNCS, vol. 773, pp. 232–249. Springer, Heidelberg (1994)
2. Blake-Wilson, S., Johnson, D., Menezes, A.: Key agreement protocols and their security analysis. In: Darnell, M. (ed.) Cryptography and Coding 1997. LNCS, vol. 1355, pp. 30–45. Springer, Heidelberg (1997)
3. Boyd, C., Cliff, Y., Gonzalez Nieto, J.M., Paterson, K.G.: Efficient one-round key exchange in the standard model. In: Mu, Y., Susilo, W., Seberry, J. (eds.) ACISP 2008. LNCS, vol. 5107, pp. 69–83. Springer, Heidelberg (2008)
4. 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)
5. Canetti, R., Krawczyk, H.: Security analysis of IKE's signature-based key-exchange protocol. In: Yung, M. (ed.) CRYPTO 2002. LNCS, vol. 2442, pp. 143–161. Springer, Heidelberg (2002), `http://eprint.iacr.org/2002/120/`
6. Freire, E.S.V., Hofheinz, D., Kiltz, E., Paterson, K.G.: Non-interactive key exchange. In: Kurosawa, K., Hanaoka, G. (eds.) PKC 2013. LNCS, vol. 7778, pp. 254–271. Springer, Heidelberg (2013)
7. Fujioka, A., Suzuki, K., Xagawa, K., Yoneyama, K.: Strongly secure authenticated key exchange from factoring, codes, and lattices. In: Fischlin, M., Buchmann, J., Manulis, M. (eds.) PKC 2012. LNCS, vol. 7293, pp. 467–484. Springer, Heidelberg (2012)
8. Jager, T., Kohlar, F., Schäge, S., Schwenk, J.: On the security of TLS-DHE in the standard model. In: Safavi-Naini, R., Canetti, R. (eds.) CRYPTO 2012. LNCS, vol. 7417, pp. 273–293. Springer, Heidelberg (2012)
9. LaMacchia, B.A., 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)
10. Moriyama, D., Okamoto, T.: An eck-secure authenticated key exchange protocol without random oracles. TIIS 5(3), 607–625 (2011)
11. Okamoto, T.: Authenticated key exchange and key encapsulation in the standard model (invited talk). In: Kurosawa, K. (ed.) ASIACRYPT 2007. LNCS, vol. 4833, pp. 474–484. Springer, Heidelberg (2007)
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. Yang, Z.: Efficient eck-secure authenticated key exchange protocols in the standard model (full version). Cryptology ePrint Archive, Report 2013/365 (2013), `http://eprint.iacr.org/`