Comments on "Flaw in the Security Analysis of Leakage-resilient Authenticated Key Exchange Protocol from CT-RSA 2016 and Restoring the Security Proof"

Rongmao Chen[1,2], Yi Mu[1], Guomin Yang[1], Willy Susilo[1], and Fuchun Guo[1]

[1] Centre for Computer and Information Security Research
School of Computing and Information Technology
University of Wollongong, Australia
[2] College of Computer
National University of Defense Technology, China
{rc517, ymu, gyang, wsusilo, fuchun}@uow.edu.au

Abstract. In CT-RSA 2016, Chen, Mu, Yang, Susilo and Guo proposed a strongly leakage-resilient authenticated key exchange (AKE) protocol [2]. In a rencent work [1], Chakraborty et al. claimed that they identified a flaw in the security analysis of Chen et al.'s protocol. In the letter, we point out that the flaw identified by Chakraborty et al. is invalid and does not exist in the original proof presented in Chen et al.'s paper.

1 The Strongly Leakage-Resilient AKE Protocol in CT-RSA 2016

Fig. 1 describes Chen et al.'s strongly leakage-resilient AKE protocol. Suppose that k is the system security parameter. Let $\mathbb{G}$ be a group with prime order p and g is a random generator of $\mathbb{G}$. Let $\mathcal{SPHF} = (\mathsf{SPHFSetup}, \mathsf{HashKG}, \mathsf{ProjKG}, \mathsf{WordG}, \mathsf{Hash}, \mathsf{ProjHash})$ denote a 2-smooth Smooth Projective Hash Function Family over $\mathcal{L} \subset \mathcal{X}$ and onto the set $\mathcal{Y}$ such that the subset membership problem between $\mathcal{L}$ and $\mathcal{X}$ is hard. Denote the hashing key space by $\mathcal{HK}$, the projection key space by $\mathcal{HP}$, the auxiliary input space by $\mathcal{AUX}$ and the witness space by $\mathcal{W}$. Let $H_1 : \{0,1\}^* \to \mathcal{AUX}, H_2 : \mathbb{G} \to \mathcal{Y}$ denote two collision-resistant hash functions.

Let $\lambda_1 = \lambda_1(k)$ be the bound on the amount of long-term secret key leakage, $\lambda_2 = \lambda_2(k)$ be that of the ephemeral secret key leakage and $\mathsf{Ext}_1, \mathsf{Ext}_2, \mathsf{Ext}_3$ be strong extractors defined as follows: $\mathsf{Ext}_1 : \mathcal{HK} \times \{0,1\}^{t_1(k)} \to \{0,1\}^{l_1(k)}$ is an average-case $(|\mathcal{HK}| - \lambda_1, \epsilon_1)$-strong extractor; $\mathsf{Ext}_2 : \{0,1\}^{u(k)} \times \{0,1\}^{t_2(k)} \to \{0,1\}^{l_2(k)}$ is an average-case $(k - \lambda_2, \epsilon_2)$-strong extractor; and $\mathsf{Ext}_3 : \mathcal{Y} \times \{0,1\}^{t_3(k)} \to \{0,1\}^{l_3(k)}$ is an average-case $(|\mathcal{Y}| - \lambda_1, \epsilon_3)$-strong extractor.

Let $\widehat{\mathsf{F}}$ and $\overline{\mathsf{F}}$ be PRF families and $\widetilde{\mathsf{F}}$ be a πPRF family defined as follows where the space of a long-term public key is denoted by Λ_k.

$\widehat{\mathsf{F}}^{k,\sum_{\widehat{\mathsf{F}}},\mathcal{D}_{\widehat{\mathsf{F}}},\mathcal{R}_{\widehat{\mathsf{F}}}} : \sum_{\widehat{\mathsf{F}}} = \{0,1\}^{l_1(k)}, \mathcal{D}_{\widehat{\mathsf{F}}} = \{0,1\}^{u(k)}, \mathcal{R}_{\widehat{\mathsf{F}}} = \mathcal{W} \times \mathbb{Z}_p,$

$\overline{\mathsf{F}}^{k,\sum_{\overline{\mathsf{F}}},\mathcal{D}_{\overline{\mathsf{F}}},\mathcal{R}_{\overline{\mathsf{F}}}} : \sum_{\overline{\mathsf{F}}} = \{0,1\}^{l_2(k)}, \mathcal{D}_{\overline{\mathsf{F}}} = \{0,1\}^{t_1(k)}, \mathcal{R}_{\overline{\mathsf{F}}} = \mathcal{W} \times \mathbb{Z}_p,$

$\widetilde{\mathsf{F}}^{k,\sum_{\widetilde{\mathsf{F}}},\mathcal{D}_{\widetilde{\mathsf{F}}},\mathcal{R}_{\widetilde{\mathsf{F}}}} : \sum_{\widetilde{\mathsf{F}}} = \{0,1\}^{l_3(k)}, \mathcal{D}_{\widetilde{\mathsf{F}}} = (\Lambda_k)^2 \times \mathcal{L}^2 \times \mathbb{G}^2 \times \{0,1\}^{2t_3(k)}, \mathcal{R}_{\widetilde{\mathsf{F}}} = \{0,1\}^{l_4(k)}.$

The system parameter is $(\mathsf{param}, \mathbb{G}, p, g, H_1, H_2, \mathsf{Ext}_1, \mathsf{Ext}_2, \mathsf{Ext}_3, \widehat{F}, \overline{F}, \widetilde{F})$ where $\mathsf{param} \leftarrow \mathsf{SPHFSetup}(1^k)$.

$\mathcal{A}$	$\mathcal{B}$
Long-Term Key Generation	
$\mathsf{hk} \xleftarrow{\$} \mathsf{HashKG}(\mathsf{param}, \mathcal{L})$,	$\mathsf{hk}' \xleftarrow{\$} \mathsf{HashKG}(\mathsf{param}, \mathcal{L})$,
$\mathsf{hp} \xleftarrow{\$} \mathsf{ProjKG}(\mathsf{param}, \mathcal{L}, \mathsf{hk})$,	$\mathsf{hp}' \xleftarrow{\$} \mathsf{ProjKG}(\mathsf{param}, \mathcal{L}, \mathsf{hk}')$,
$r_{\mathcal{A}_1} \xleftarrow{\$} \{0,1\}^{t_1(k)}, r_{\mathcal{A}_2} \xleftarrow{\$} \{0,1\}^{t_2(k)}$,	$r_{\mathcal{B}_1} \xleftarrow{\$} \{0,1\}^{t_1(k)}, r_{\mathcal{B}_2} \xleftarrow{\$} \{0,1\}^{t_2(k)}$,
$lsk_{\mathcal{A}} = \mathsf{hk}, lpk_{\mathcal{A}} = (\mathsf{hp}, r_{\mathcal{A}_1}, r_{\mathcal{A}_2})$.	$lsk_{\mathcal{B}} = \mathsf{hk}', lpk_{\mathcal{B}} = (\mathsf{hp}', r_{\mathcal{B}_1}, r_{\mathcal{B}_2})$.
Session Execution	
$esk_{\mathcal{A}} \xleftarrow{\$} \{0,1\}^{u(k)}, t_{\mathcal{A}} \xleftarrow{\$} \{0,1\}^{t_3(k)}$,	$esk_{\mathcal{B}} \xleftarrow{\$} \{0,1\}^{u(k)}, t_{\mathcal{B}} \xleftarrow{\$} \{0,1\}^{t_3(k)}$,
$\widehat{lsk}_{\mathcal{A}} = \mathsf{Ext}_1(lsk_{\mathcal{A}}, r_{\mathcal{A}_1})$,	$\widehat{lsk}_{\mathcal{B}} = \mathsf{Ext}_1(lsk_{\mathcal{B}}, r_{\mathcal{B}_1})$,
$\widehat{esk}_{\mathcal{A}} = \mathsf{Ext}_2(esk_{\mathcal{A}}, r_{\mathcal{A}_2})$,	$\widehat{esk}_{\mathcal{B}} = \mathsf{Ext}_2(esk_{\mathcal{B}}, r_{\mathcal{B}_2})$,
$(w_{\mathcal{A}}, x) = \widehat{F}_{\widehat{lsk}_{\mathcal{A}}}(esk_{\mathcal{A}}) + \overline{F}_{\widehat{esk}_{\mathcal{A}}}(r_{\mathcal{A}_1})$,	$(w_{\mathcal{B}}, y) = \widehat{F}_{\widehat{lsk}_{\mathcal{B}}}(esk_{\mathcal{B}}) + \overline{F}_{\widehat{esk}_{\mathcal{B}}}(r_{\mathcal{B}_1})$,
$W_{\mathcal{A}} = \mathsf{WordG}(\mathsf{param}, \mathcal{L}, w_{\mathcal{A}}), X = g^x$,	$W_{\mathcal{B}} = \mathsf{WordG}(\mathsf{param}, \mathcal{L}, w_{\mathcal{B}}), Y = g^y$,
Erase all state except $(esk_{\mathcal{A}}, W_{\mathcal{A}}, X, t_{\mathcal{A}})$.	Erase all state except $(esk_{\mathcal{B}}, W_{\mathcal{B}}, Y, t_{\mathcal{B}})$.

$$\xrightarrow{(\widehat{B}, \widehat{A}, W_{\mathcal{A}}, X, t_{\mathcal{A}})}$$

$$\xleftarrow{(\widehat{A}, \widehat{B}, W_{\mathcal{B}}, Y, t_{\mathcal{B}})}$$

Session Key Ouput	
Set $\mathsf{sid} = (\widehat{A}, \widehat{B}, W_{\mathcal{A}}, X, t_{\mathcal{A}}, W_{\mathcal{B}}, Y, t_{\mathcal{B}})$	Set $\mathsf{sid} = (\widehat{A}, \widehat{B}, W_{\mathcal{A}}, X, t_{\mathcal{A}}, W_{\mathcal{B}}, Y, t_{\mathcal{B}})$
$aux = H_1(\mathsf{sid}), K_{\mathcal{A}_1} = Y^x$,	$aux = H_1(\mathsf{sid}), K_{\mathcal{A}_1} = X^y$,
$K_{\mathcal{A}_2} = \mathsf{ProjHash}(\mathsf{param}, \mathcal{L}, lpk_{\mathcal{B}}, W_{\mathcal{A}}, w_{\mathcal{A}}, aux)$,	$K_{\mathcal{B}_2} = \mathsf{Hash}(\mathsf{param}, \mathcal{L}, lsk_{\mathcal{B}}, W_{\mathcal{A}}, aux)$,
$K_{\mathcal{A}_3} = \mathsf{Hash}(\mathsf{param}, \mathcal{L}, lsk_{\mathcal{A}}, W_{\mathcal{B}}, aux)$,	$K_{\mathcal{B}_3} = \mathsf{ProjHash}(\mathsf{param}, \mathcal{L}, lpk_{\mathcal{A}}, W_{\mathcal{B}}, w_{\mathcal{B}}, aux)$,
$s_{\mathcal{A}} = Ext_3(H_2(K_{\mathcal{A}_1}) \oplus K_{\mathcal{A}_2} \oplus K_{\mathcal{A}_3}, t_{\mathcal{A}} \oplus t_{\mathcal{B}})$,	$s_{\mathcal{B}} = Ext_3(H_2(K_{\mathcal{B}_1}) \oplus K_{\mathcal{B}_2} \oplus K_{\mathcal{B}_3}, t_{\mathcal{A}} \oplus t_{\mathcal{B}})$,
$SK_{\mathcal{A}} = \widetilde{F}_{s_{\mathcal{A}}}(\mathsf{sid})$.	$SK_{\mathcal{B}} = \widetilde{F}_{s_{\mathcal{B}}}(\mathsf{sid})$.

Fig. 1. Framework for CLR-eCK secure AKE

1.1 Security Proof in [2]

Theorem 1. *The AKE protocol following the general framework is* (λ_1, λ_2)-CLR-eCK-*secure if the underlying smooth projective hash function is* 2-smooth, *the DDH assumption holds in* $\mathbb{G}$, H_1, H_2 *are collision-resistant hash functions,* $\widehat{\mathsf{F}}$ *and* $\overline{\mathsf{F}}$ *are PRF families and* $\widetilde{\mathsf{F}}$ *is a* π*PRF family. Here* $\lambda_1 \leq \min\{|\mathcal{HK}| - 2\log(1/\epsilon_1) - l_1(k), |\mathcal{Y}| - 2\log(1/\epsilon_3) - l_3(k)\}, \lambda_2 \leq u(k) - 2\log(1/\epsilon_2) - l_2(k)$.

Let session $\mathsf{sid}^* = (\widehat{A}, \widehat{B}, W^*_{\mathcal{A}}, X^*, t^*_{\mathcal{A}}, W^*_{\mathcal{B}}, Y^*, t^*_{\mathcal{B}})$ be the target session chosen by adversary $\mathcal{M}$. $\mathcal{A}$ is the owner of the session sid^* and $\mathcal{B}$ is the peer. We then analyze the security of the AKE protocol in the following two disjoint cases.

Case I. *There exists a matching session,* $\overline{\mathsf{sid}^*}$, *of the target session* sid^*. Based on the definition, we can see that for each party, either long-term or ephemeral secret key remains unknown to the adversary. Without loss of generality, suppose that the adversary obtains at most λ_2-bits of the ephemeral secret key of target session sid^*, we have that $\widehat{esk}^*_{\mathcal{A}} = \mathsf{Ext}_2(esk^*_{\mathcal{A}}, r_{\mathcal{A}_2}) \overset{s}{\equiv}_{\epsilon_2} \widehat{esk}'_{\mathcal{A}} \xleftarrow{\$} \{0,1\}^{l_2(k)}$. Therefore, $(w^*_{\mathcal{A}}, x^*) = \widehat{F}_{\widehat{lsk}_{\mathcal{A}}}(esk^*_{\mathcal{A}}) + \overline{F}_{\widehat{esk}^*_{\mathcal{A}}}(r_{\mathcal{A}_1}) \overset{c}{\equiv} (w'_{\mathcal{A}}, x') \xleftarrow{\$} \mathcal{W} \times \mathbb{Z}_p$. Similarly, suppose that the adversary obtains at most λ_2-bits of the ephemeral secret key of matching session $\overline{\mathsf{sid}^*}$, we have that $\widehat{esk}^*_{\mathcal{B}} = \mathsf{Ext}_2(esk^*_{\mathcal{B}}, r_{\mathcal{B}_2}) \overset{s}{\equiv}_{\epsilon_2} \widehat{esk}'_{\mathcal{B}} \xleftarrow{\$} \{0,1\}^{l_2(k)}$, and thus $(w^*_{\mathcal{B}}, y^*) =$

$\widehat{F}_{\widehat{lsk}_{\mathcal{B}}}(esk^*_{\mathcal{B}}) + \overline{F}_{\widehat{esk}^*_{\mathcal{B}}}(r_{\mathcal{B}_1}) \overset{c}{\equiv} (w'_{\mathcal{B}}, y') \overset{\$}{\leftarrow} \mathcal{W} \times \mathbb{Z}_p$. Therefore, regardless of the type of the reveal query and leakage query, (x^*, y^*) are uniformly random elements in $\mathbb{Z}_p^2$ from the view of adversary $\mathcal{M}$. Therefore, $K^*_{\mathcal{A}_1} = K^*_{\mathcal{B}_1} = g^{x^* y^*}$ is computationally indistinguishable from a random element in $\mathbb{G}$ according to the DDH assumption and hence $H_2(K^*_{\mathcal{A}_1})$ is a uniform random string from the view of $\mathcal{M}$ who is given $X^* = g^{x^*}, Y^* = g^{y^*}$. We then have that the seed $s^*_{\mathcal{A}}$ for the πPRF function is uniformly distributed and unknown to the adversary and thus the derived session key $SK^*_{\mathcal{A}}$ is computationally indistinguishable from a random string. It is worth noting that in this case we only require $\tilde{F}$ to be a normal PRF.

Case II. *There exists no matching session of the test session* sid^*.

In this case, the adversary cannot issue $\mathsf{LongTermKeyReveal}$ query to reveal the long-term secret key of $\mathcal{B}$ but may issue the leakage query $\mathsf{LongTermKeyLeakage}$ to learn some bit-information of $lsk_{\mathcal{B}}$. We prove the security of the AKE protocol as follows. In the simulation, we modify the security game via the following steps to obtain a new game. We first replace $K^*_{\mathcal{A}_2} = \mathsf{ProjHash}(\mathsf{param}, \mathcal{L}, lpk_{\mathcal{B}}, W^*_{\mathcal{A}}, w^*_{\mathcal{A}}, aux^*)$ by $K^*_{\mathcal{A}_2} = \mathsf{Hash}(\mathsf{param}, \mathcal{L}, lsk_{\mathcal{B}}, W^*_{\mathcal{A}}, aux^*)$, and then choose $W^*_{\mathcal{A}} \in \mathcal{X} \setminus \mathcal{L}$ instead of deriving it from $\mathcal{L}$ through the algorithm WordG. One can see that the new game is identical to the original game from the view of adversary $\mathcal{M}$ due to the fact that $\mathsf{ProjHash}(\mathsf{param}, \mathcal{L}, lpk_{\mathcal{B}}, W^*_{\mathcal{A}}, w^*_{\mathcal{A}}) = \mathsf{Hash}(\mathsf{param}, \mathcal{L}, lsk_{\mathcal{B}}, W^*_{\mathcal{A}})$, and due to the difficulty of the subset membership problem which ensures that the distribution of $\mathcal{X} \setminus \mathcal{L}$ is indistinguishable from $\mathcal{L}$.

Note that adversary $\mathcal{M}$ may activate a session sid, which is not matching to session sid^*, with $\mathcal{B}$. Precisely, $\mathcal{M}$ can choose $W \in \mathcal{X} \setminus \mathcal{L}$ (e.g., by replaying $W^*_{\mathcal{A}}$), send W to $\mathcal{B}$ and issue $\mathsf{SessionKeyReveal}(\mathsf{sid})$ query to learn the shared key. According to the property of 2-smooth of the underlying smooth projective hash function, we have that $K^*_{\mathcal{A}_2}$ is pairwisely independent from any other such key (denoted by $\widetilde{K}$) and all public information (i.e., $\mathsf{param}, \mathcal{L}, lpk_{\mathcal{B}}, W^*_{\mathcal{A}}, aux^*$) and hence $\widetilde{\mathrm{H}}_\infty(K^*_{\mathcal{A}_2} | \widetilde{K}, \mathsf{param}, \mathcal{L}, lpk_{\mathcal{B}}, W^*_{\mathcal{A}}, aux^*) = |\mathcal{Y}|$. Suppose that the leakage of $lsk_{\mathcal{B}}$ is at most λ_1-bits (denoted by $\widetilde{lsk_{\mathcal{B}}}$), and therefore (see *Lemma 1*), $\widetilde{\mathrm{H}}_\infty(K^*_{\mathcal{A}_2} | \widetilde{K}, \mathsf{param}, \mathcal{L}, lpk_{\mathcal{B}}, W^*_{\mathcal{A}}, aux^*, \widetilde{lsk_{\mathcal{B}}}) \geq \widetilde{\mathrm{H}}_\infty(K^*_{\mathcal{A}_2} | \widetilde{K}, \mathsf{param}, \mathcal{L}, lpk_{\mathcal{B}}, W^*_{\mathcal{A}}, aux^*) - \lambda_1 = |\mathcal{Y}| - \lambda_1$. Therefore, by using the strong extractor Ext_3, it holds that $s^*_{\mathcal{A}} = \mathsf{Ext}_3(H_2(K_{\mathcal{A}_1})^* \oplus K^*_{\mathcal{A}_2} \oplus K^*_{\mathcal{A}_3}, t^*_{\mathcal{A}} \oplus t^*_{\mathcal{B}}) \overset{s}{\equiv}_{\epsilon_3} s'_{\mathcal{A}} \overset{\$}{\leftarrow} \{0,1\}^{l_3(k)}$. One can see that $\mathcal{A}$ obtains a variable $s^*_{\mathcal{A}}$ which is pairwisely independent from any other such variables and thus the derived session key $SK^*_{\mathcal{A}}$ is computationally indistinguishable from a truly random element from $\mathcal{M}$'s view due to the application of πPRF, which completes the proof.

Simulation for Non-test Session. Note that for the two cases above, we have to simulate the non-test session correctly with the adversary. Specifically, when adversary $\mathcal{M}$ activates a non-test session with $\mathcal{A}$ or $\mathcal{B}$, the session execution simulated should be identical to the session run by $\mathcal{A}$ or $\mathcal{B}$ from the view of $\mathcal{M}$. One can note that this can be easily guaranteed when the query $\mathsf{LongTermKeyReveal}(\mathcal{A})$ or $\mathsf{LongTermKeyReveal}(\mathcal{B})$ is issued in the game. Since we know the long-term secret key of $\mathcal{A}$ or $\mathcal{B}$, we can just select an ephemeral secret key and compute the ephemeral public key correctly by using the long-term secret key and long-term public key. Nevertheless, if the query

$\mathsf{LongTermKeyReveal}(\mathcal{A})$ or $\mathsf{LongTermKeyReveal}(\mathcal{B})$ is not issued, that is, without the long-term secret key of $\mathcal{A}$ or $\mathcal{B}$, the simulation of the non-test session owned by $\mathcal{A}$ or $\mathcal{B}$ can no longer be simulated as shown above. In this case, we simulate the session as follows. Suppose that we are to simulate the session owned by $\mathcal{A}$ without knowing $lsk_{\mathcal{A}}$, we pick $(r_1, r_2) \overset{\$}{\leftarrow} \mathcal{W} \times \mathbb{Z}_p$ and then compute $W_{\mathcal{A}} = \mathsf{WordG}(\mathsf{param}, \mathcal{L}, r_1), X = g^{r_2}$. We say that the session simulated in this way can be identical to the real session from $\mathcal{M}$'s view due to the pseudo-randomness of the PRF. To be more precise, even when $\mathcal{M}$ obtains at most λ_1-bits of $lsk_{\mathcal{A}}$ through $\mathsf{LongTermKeyLeakage}(\mathcal{A})$, the variable $\widehat{lsk}_{\mathcal{A}}$, which comes from $\mathsf{Ext}_1(lsk_{\mathcal{A}}, r_{\mathcal{A}})$ and inputs to the pseudo-random function $\widehat{F}$, still remains unknown to adversary $\mathcal{M}$. Therefore, the value of $\widehat{F}_{\widehat{lsk}_{\mathcal{A}}}(esk_{\mathcal{A}})$ is computationally indistinguishable from a random element.

2 The Invalid "Flaw" Shown in [1]

Chakraborty et al. claimed that there is a "flaw" in the security proof of the above AKE protocol. Fig. 2 illustrates the scenario they used to describe the "problem".

The "flaw" illustated in Fig. 2 basically claims that Chen et al.'s protocol cannot be proven secure under the DDH assumption because when the simulator embeds the DDH problem instance (g, g^a, g^b, Z) into the simulation, there is a situation that the simulator cannot answer a session key reveal query made by the adversary. Let $\pi_B^{t^*}$ denote the test (i.e., target) session run by user B and π_A^s denote a session of user A. The scenario described in Fig. 2 can be summarized as follows:

1. The simulator embeds the DDH problem into the simulation by setting $X = g^a$ for π_A^s and $Y = g^b$ for $\pi_B^{t^*}$;
2. The attacker relays X to $\pi_B^{t^*}$ which obtains a transcript containing (X, Y);
3. The attacker modifies Y to $\tilde{Y} = g^c$ and sends $\tilde{Y}$ to π_A^s which obtains a transcript containing $(X, \tilde{Y})$;
4. The attacker issues a session key reveal query to π_A^s.
5. Since the simulator cannot compute g^{ac} without knowing the value of a or c, the simulator cannot answer the session key reveal query. But the adversary can compute g^{ac} (and hence the final session key of π_A^s) with the knowledge of c.

The "Flaw" Is Invalid. In the above scenario, the test session $\pi_B^{t^*}$ has no matching session. Below we cite the original definition of "matching session" in the eCK model [3] which is adopted by Chen et al. in [2]:

"As in the Canetti-Krawczyk model, we define the matching session to an AKE session to be the session executed by the other party with the same communications being transmitted, albeit in different order. For example, in a 2-round protocol, if A executes the session (O,A, B, commA, commB), then the matching session is executed by B and has session identifier (P, B,A, commA, commB)."

Although the message $\pi_B^{t^*}$ has received is generated by π_A^s, $\pi_B^{t^*}$ and π_A^s are not matching sessions and therefore the adversary is allowed to make a session key reveal query to π_A^s. Also, with overwhelming probability, there is no other session simulated by the simulator that has the communication transcript (X, Y) since one of X and Y will be freshly generated by the simulator in a different session although the adversary can replay the other one.

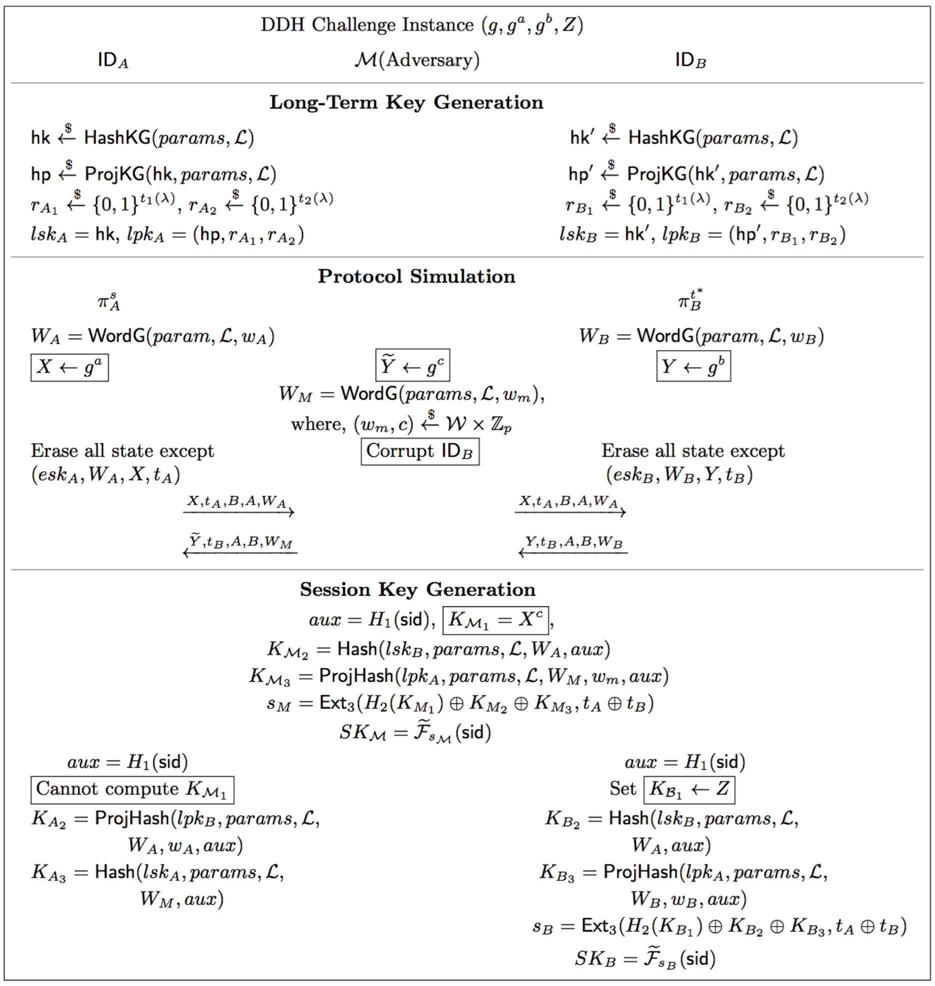
DDH Challenge Instance (g, g^a, g^b, Z)
ID_A
$\mathcal{M}$(Adversary)
ID_B
Long-Term Key Generation
$\mathsf{hk} \xleftarrow{\$} \mathsf{HashKG}(params, \mathcal{L})$
$\mathsf{hp} \xleftarrow{\$} \mathsf{ProjKG}(\mathsf{hk}, params, \mathcal{L})$
$r_{A_1} \xleftarrow{\$} \{0,1\}^{t_1(\lambda)}, r_{A_2} \xleftarrow{\$} \{0,1\}^{t_2(\lambda)}$
$lsk_A = \mathsf{hk}, lpk_A = (\mathsf{hp}, r_{A_1}, r_{A_2})$
$\mathsf{hk}' \xleftarrow{\$} \mathsf{HashKG}(params, \mathcal{L})$
$\mathsf{hp}' \xleftarrow{\$} \mathsf{ProjKG}(\mathsf{hk}', params, \mathcal{L})$
$r_{B_1} \xleftarrow{\$} \{0,1\}^{t_1(\lambda)}, r_{B_2} \xleftarrow{\$} \{0,1\}^{t_2(\lambda)}$
$lsk_B = \mathsf{hk}', lpk_B = (\mathsf{hp}', r_{B_1}, r_{B_2})$
Protocol Simulation
π_A^s
$\pi_B^{t^*}$
$W_A = \mathsf{WordG}(param, \mathcal{L}, w_A)$
$\boxed{X \leftarrow g^a}$
$\boxed{\widetilde{Y} \leftarrow g^c}$
$W_M = \mathsf{WordG}(params, \mathcal{L}, w_m),$
where, $(w_m, c) \xleftarrow{\$} \mathcal{W} \times \mathbb{Z}_p$
$\boxed{\text{Corrupt } \mathsf{ID}_B}$
$W_B = \mathsf{WordG}(param, \mathcal{L}, w_B)$
$\boxed{Y \leftarrow g^b}$
Erase all state except (esk_A, W_A, X, t_A)
Erase all state except (esk_B, W_B, Y, t_B)
$\xrightarrow{X, t_A, B, A, W_A}$
$\xleftarrow{\widetilde{Y}, t_B, A, B, W_M}$
$\xrightarrow{X, t_A, B, A, W_A}$
$\xleftarrow{Y, t_B, A, B, W_B}$
Session Key Generation
$aux = H_1(\mathsf{sid}), \boxed{K_{\mathcal{M}_1} = X^c},$
$K_{\mathcal{M}_2} = \mathsf{Hash}(lsk_B, params, \mathcal{L}, W_A, aux)$
$K_{\mathcal{M}_3} = \mathsf{ProjHash}(lpk_A, params, \mathcal{L}, W_M, w_m, aux)$
$s_M = \mathsf{Ext}_3(H_2(K_{M_1}) \oplus K_{M_2} \oplus K_{M_3}, t_A \oplus t_B)$
$SK_{\mathcal{M}} = \widetilde{\mathcal{F}}_{s_{\mathcal{M}}}(\mathsf{sid})$
$aux = H_1(\mathsf{sid})$
$\boxed{\text{Cannot compute } K_{\mathcal{M}_1}}$
$K_{A_2} = \mathsf{ProjHash}(lpk_B, params, \mathcal{L}, W_A, w_A, aux)$
$K_{A_3} = \mathsf{Hash}(lsk_A, params, \mathcal{L}, W_M, aux)$
$aux = H_1(\mathsf{sid})$
Set $\boxed{K_{B_1} \leftarrow Z}$
$K_{B_2} = \mathsf{Hash}(lsk_B, params, \mathcal{L}, W_A, aux)$
$K_{B_3} = \mathsf{ProjHash}(lpk_A, params, \mathcal{L}, W_B, w_B, aux)$
$s_B = \mathsf{Ext}_3(H_2(K_{B_1}) \oplus K_{B_2} \oplus K_{B_3}, t_A \oplus t_B)$
$SK_B = \widetilde{\mathcal{F}}_{s_B}(\mathsf{sid})$

Fig. 2. The proof reduction "problem" from [1]

According to the definition of a "fresh" session, when $\pi_B^{t^*}$ has no matching session, then the adversary cannot corrupt user A. As described in the original proof of the protocol given above, under this scenario which is referred to as Case 2 in the original proof, the security of the session key is based on the security of the SPHF rather than the difficulty of solving the DDH problem. **Therefore, in the above scenario, the**

simulator does not need to embed the DDH problem instance into the simulation at all. As described in the original proof, when there is no matching session, the security of the session key generated by the test session $\pi_B^{t^*}$ relies on K_{B_3} instead of the Diffie-Hellman key K_{B_1}. The simulator can simply generate $X = g^{r_1}$ and $Y = g^{r_2}$ with the knowledge of r_1 and r_2 and answer the session key reveal query to π_A^s without any issue.

In conclusion, the "flaw" described in [1] does not exist in the original proof of [2].

References

1. Chakraborty, S., Paul, G., Rangan, C.P.: Flaw in the security analysis of leakage-resilient authenticated key exchange protocol from ct-rsa 2016 and restoring the security proof. Cryptology ePrint Archive, Report 2016/862 (2016)
2. Chen, R., Mu, Y., Yang, G., Susilo, W., Guo, F.: Strongly leakage-resilient authenticated key exchange. In: Topics in Cryptology - CT-RSA 2016 - The Cryptographers' Track at the RSA Conference 2016, San Francisco, CA, USA, February 29 - March 4, 2016, Proceedings. pp. 19–36 (2016)
3. LaMacchia, B.A., Lauter, K.E., Mityagin, A.: Stronger security of authenticated key exchange. In: Provable Security. pp. 1–16 (2007)