A Suite of Enhanced Security Models for Key Compromise Impersonation Resilience and ID-based Key Exchange

Robert W. Zhu, Xiaojian Tian, and Duncan S. Wong

Department of Computer Science
City University of Hong Kong
Hong Kong, China
{zhuwei,xjtian,duncan}@cs.cityu.edu.hk

Abstract. Canetti and Krawczyk proposed a security model (CK-model) for authentication and key exchange protocols in 2001 based on a modeling approach proposed by Bellare et al. in 1998. The model not only reasonably captures the power of practical attackers but also provides a modular approach to the design of secure key exchange protocols. However, the model does not capture the property of Key Compromise Impersonation (KCI) Resilience, which has been studied elaborately with respect to key exchange protocols. Until now, analysis concerning this property has mostly been performed heuristically and it has been difficult to apply existing security models and formal analysis methods to the study of KCI attacks. In this paper, we solve this problem by proposing an enhancement of the CK-model for capturing KCI attacks.

With the revival of interest in identity-based (ID-based) cryptography, there have been many new ID-based key exchange protocols proposed. Despite the fact that some of them have been proven in some restricted versions of a model proposed by Bellare and Rogaway in 1993 and some others have been proven in the CK-model, there is no security model specifically formalized for ID-based key exchange protocols. In particular, Forward Secrecy against compromised Key Generation Server (KGS-FS) has never been captured even though this notion is more important and also stronger than the perfect forward secrecy in ID-based cryptography. For this, we further extend our model to the ID-based cryptographic setting and capture the KGS-FS.

Finally, we provide some formal security analyses for several identity-based key exchange protocols under our models.

1 Introduction

In 1993, the first security model using a computational approach to proofs of key exchange (KE) protocols was proposed by Bellare and Rogaway [4]. It defines a hostile environment which has an active adversary with various attacking capabilities such as party corruption and session key reveal, and the environment also supports simultaneous executions of several copies (or sessions) of a protocol. This model has later been adapted or extended to other settings [5,6,7]. Even today,

it is still one of the most commonly used models, for example, for analyzing the security of identity-based KE protocols [14,19,26,15]

In 1998, Bellare, Canetti and Krawczyk [1] proposed a different approach for formalizing the security of a KE protocol. Later in 2001, Canetti and Krawczyk [12] extended this model and developed a more general model (here, we call it the CK-model) with the definition of secure key exchange changed from simulation-based to indistinguishability-based. The key element in the CK-model is that a *modular approach* is used to design and analyze KE protocols. Their approach is to treat authentication and key exchange separately. It provides a two-step methodology by which a KE protocol may first be proven secure in an ideally authenticated network, then the protocol is transformed into a variant which will be secure even in a more realistic unauthenticated network. This transformation is done by encapsulating the message flows of the original KE protocol for the ideally authenticated network using some authentication protocols called authenticators. The major advantage of this approach is that the proved building blocks can be reused to construct new provably secure protocols, and these protocols are also not limited to key exchange.

By applying the CK-model, we allow the proved KE protocols to capture many desirable security properties that include known key security, perfect forward secrecy (PFS), unknown key-share resilience, etc. The CK-model is also flexible enough as it can make some security properties optional. For example, a little adjustment on a condition in the CK-model can allow the model to remove the requirement of PFS for the proved KE protocols. In this paper, we will extend the CK-model so that it can capture another desirable security property called Key Compromise Impersonation (KCI) Resilience [11,6,18]. We will also make it optional by allowing it to be *turned off* if necessary.

KCI Resilience is a security property that compromising the long-term secret of one party should not allow an adversary to masquerade *to* the party as a different party. Let us consider a two-party protocol: suppose an adversary has compromised a party A's long-term secret, but not B's. Obviously, this allows the adversary to impersonate A to B. If, in addition, the adversary is able to impersonate *to* A as B, then such a protocol is said to be vulnerable to KCI attacks. We consider KCI attacks for public-key cryptographic protocols only, rather than symmetric key cryptographic protocols. This is because in the symmetric key model, once the adversary has compromised the long-term secret between A and B, the adversary can obviously impersonate A to B or B to A. Although KCI attacks have been studied elaborately with respect to key exchange protocols, analysis concerning this property has mostly been performed heuristically and it has been difficult to apply existing security models and formal analysis methods to the study of KCI attacks. Currently, the CK-model does not capture KCI resilience. In this paper, we solve the problem for CK-model by enhancing it with a new query called key compromise. The query is different from the existing party corruption query. Despite that the KCI resilience is traditionally considered as a desirable security property for KE protocols only, we notice that it actually has a much broader scope. Instead of limited to KE protocols, we can consider it as a desirable property for any authentication protocols in which the KE protocols are corresponding to just one type of authentication protocols. We will show that our enhanced CK-model,

called the KCIR-enhanced CK-model, not only captures all the security properties as captured in the original CK-model, but also captures the property of KCI resilience.

In recent years, with the revival of interest in identity-based (ID-based) cryptography [22], there have been many new ID-based KE protocols proposed (to name a few, they include [24,14,10,16,19,26,15,27]). However, due to the lack of a formally defined security model specifically for constructing and analyzing ID-based KE protocols, many of these protocols do not have any proof provided for supporting their security claims, and some others (such as [23,16]) have been shown to be insecure with respect to their original security claims. In [9] a summary on many recently proposed ID-based KE protocols on whether they have attempted to give security proofs or not has been given. As noted, many of these protocols that have security proofs actually have proofs only in a restricted version of the original model (exclusively using the Bellare-Rogaway model [4] mentioned above). In the restricted version, the adversary is prevented from asking any Reveal query. On the other side, for some other protocols [10,15,26], even proofs are given in the original Bellare-Rogaway model or CK-model, some properties that are only specific for ID-based KE protocols cannot be captured, for example, providing forward secrecy against compromised Key Generation Server (KGS) [17,14,19]. This property, referred as the KGS forward secrecy (KGS-FS), requires forward secrecy to be maintained even after the *master secret* of the KGS is compromised. KGS-FS is a stronger notion than that of PFS as compromising the master secret of the KGS implies compromising the secret keys of all parties in the context of ID-based cryptography. In this paper, we further extend our KCIR-enhanced CK-model for ID-based cryptographic protocols and capturing the property of KGS-FS for ID-based KE protocols.

Contributions. In this paper, we propose a KCIR-enhanced CK-model to capture the property of KCI Resilience in the CK-model. In particular, we extend the notion of KCI resilience, which is normally considered with respect to secure KE protocols only, to a more general one so that it becomes a desirable (and of course is still an optional) property for all authentication protocols which also include the KE protocols as defined in [12].

We further propose an ID-based extension onto our KCIR-enhanced CK-model. The new extension captures the general setting of ID-based authentication protocols (hence including ID-based KE protocols). The model, when used for analyzing ID-based KE protocols, also captures the property of KGS-FS. To exemplify the use of this new extension, we provide a proof of security for a recently proposed non-pairing based ID-based KE protocol [27] and show that it supports KCI resilience as well as KGS-FS. We also show that Smart's protocol [24], after some slight modification, is secure with KCI resilience but not PFS (perfect forward secrecy) in our model.

Paper Organization. In Sec. 2, we review the CK-model. In Sec. 3, we describe the enhancement of the CK-model for supporting KCI resilience. In Sec. 4, we further extend the model to support ID-based settings and KGS-FS. In Sec. 5, we review several ID-based KE protocols and analyze their security under our model.

2 The Canetti-Krawczyk Model (CK-Model)

We review the model proposed by Canetti and Krawczyk [12] to some extent so that it is sufficient for understanding the rest of the paper. For full details, readers can refer to the full papers of [1,12].

2.1 Message Driven and Key Exchange (KE) Protocols

The CK-model is defined over message driven protocols. An n-party message driven protocol is a collection of n programs, each of them is run by a different party. A program is first invoked with some initial input which includes a security parameter and some random input. Once invoked, it can be activated by two types of events:

1. The arrival of an incoming message.
2. An action request which models information coming from other programs run by the party. Typical examples of action requests include requesting for sending a message or triggering a key exchange request with some specified party. For each action request, the response of the program should be defined explicitly in the protocol specification.

At the end of each activation, a local output value is generated.

A Key Exchange (KE) protocol is a message driven protocol. Each execution of the KE protocol is modeled as a series of activations on two parties, P_i and P_j. For example, after all the programs of parties have been invoked, the program running in P_i will take an action request of the form establish session$(P_i, P_j, s, role)$ where P_j is the party with whom the key is to be exchanged, s is the session ID, and $role$ is either initiator or responder. The program of P_i will then fork a sub-process and ask it to handle the subsequent execution procedure of the protocol. The sub-process is called a KE-session with input $(P_i, P_j, s, role)$. After the sub-process is forked, if P_i is activated by an action request to send a message m to P_j in session s (i.e. there is a send-message action request received with inputs m, P_j and s), the corresponding KE-session will handle it. The KE-session is *completed* when the corresponding sub-process returns with output (P_i, P_j, s, κ) where κ is a non-null *session key*. The KE-session can label κ as 'secret' which prevents an adversary from getting the value of it. When a session *completes*, the internal state of the corresponding sub-process is assumed to be erased.

In this model, each program running in a party can fork multiple sub-processes to handle multiple KE-sessions. If in one KE protocol execution, party P_i has a KE-session with input $(P_i, P_j, s, role)$ and party P_j has a KE-session with input $(P_j, P_i, s', role')$, and $s = s'$, then we say that the two KE-sessions are *matching*.

The Initialization Function I. This function in a message driven protocol models a perfectly secure bootstrapping of cryptographic functions. The function takes a random input r and a security parameter k, and outputs a vector $I(r,k) = I(r,k)_0 \cdots I(r,k)_n$, where $I(r,k)_0$ will become known to all parties and the adversary; $I(r,k)_i$, for $i > 0$, will become known only to party P_i.

2.2 Two Adversarial Models: UM and AM

For capturing various capabilities and activities of adversaries, in the CK-model, there are two adversarial models defined. They are the *unauthenticated-links model (UM)* and the *authenticated-links model (AM)*.

The Unauthenticated-links Adversarial Model (UM). Consider a message driven protocol π with n parties $P_1, \cdots, P_n$. A UM-adversary $\mathcal{U}$ is a PPT (probabilistic polynomial-time) Turing machine which controls all the communication links of the system and schedules almost all the protocol activities, including party activation and message delivery (except the protocol initialization which is controlled by the initialization function I). In particular, $\mathcal{U}$ can activate π within some party P_i by either incoming messages or action requests. $\mathcal{U}$ can also arbitrarily generate, inject, modify, and deliver any messages of his own will. In addition, $\mathcal{U}$ can access all the local output of a party except a secret output, e.g. the session key of a key exchange session. The adversary $\mathcal{U}$ can also do the following activities by making some appropriate queries.

- party corruption on P_i: $\mathcal{U}$ learns all the internal state information of P_i. It includes the long-term secret of P_i and all the state information of its sessions. P_i will no longer be activated and does not generate further local output. P_i is said to be *corrupted*.
- session-state reveal: $\mathcal{U}$ learns all the current state of a specified session.
- session-output reveal: $\mathcal{U}$ learns all outputs that are labeled 'secret', from a specified session.

The global output denoted by $UNAUTH_{\pi,\mathcal{U}}$ of a protocol is the cumulative output of $\mathcal{U}$ and $P_1, \cdots, P_n$. The output of the adversary is specified in the protocol specification. For example, in the definition of session-key security in Sec. 2.4, the output of $\mathcal{U}$ is its guess for the coin value tossed by the game simulator.

Remark: For KE protocols, the session-key reveal query is used instead of the session-output reveal query and an additional adversarial activity called session expiration query is added. A session-key reveal query can be scheduled by $\mathcal{U}$ for a *completed* KE-session. In this case, $\mathcal{U}$ learns the session key of the specific KE-session. $\mathcal{U}$ can also issue a session expiration query for any *completed* KE-session. In this case, the session key is erased from the party's memory and $\mathcal{U}$ is no longer allowed to issue a session-key reveal query on an *expired* KE-session.

Session Exposure. A KE-session with input $(P_i, P_j, s, role)$ is called *locally exposed* if $\mathcal{U}$ performs any of the following activities: (1) a session-state reveal (2) a session-key reveal (3) a party corruption on P_i before the session expired. However, a party corruption on P_i *after* session $(P_i, P_j, s, role)$ expired is not taken into account. A KE-session is called *exposed* if either the session or its matching session is *locally exposed*.

The Authenticated-links Adversarial Model (AM). An AM-adversary $\mathcal{A}$ is similar to the UM-adversary $\mathcal{U}$ while $\mathcal{A}$ is not allowed to inject or modify messages unless the message sender is corrupted or the message belongs to an exposed session. $\mathcal{A}$ is restricted to deliver messages faithfully, and each message

can only be delivered *at most* once. Also note that the sender of a message can never be changed by $\mathcal{A}$ even if the origin session is exposed (which includes the case that the sender is corrupted). The global output denoted by $AUTH_{\pi,\mathcal{A}}$ is the cumulative output of $\mathcal{A}$ and $P_1, \cdots, P_n$.

Definition 1 ([12]). *Let π and π' be two n-party message driven protocols. We say that π' emulates π in UM if given any adversary $\mathcal{U}$ in UM against protocol π', there exists an adversary $\mathcal{A}$ in AM such that $AUTH_{\pi,\mathcal{A}}$ and $UNAUTH_{\pi',\mathcal{U}}$ are computationally indistinguishable.*

Since the authentication in AM is explicitly ensured, if π' emulates π in UM, the authentication in UM is also ensured.

2.3 Authenticators

An authenticator $\mathcal{C}$ is an algorithm that for any protocol π in AM, the protocol $\mathcal{C}(\pi)$ emulates π in UM. One way of constructing an authenticator is given in [1], where a layered approach is used. According to [1, Theorem 3], an authenticator $\mathcal{C}_\lambda$, which is called a **layered authenticator** in [1], can be constructed from an MT-authenticator λ which emulates the basic message transmission (MT) protocol. The basic idea is that whenever a party P_i wants to send or receive a message (using an MT protocol) in AM, the MT-authenticator emulates it in UM using λ. Formally, an MT protocol in AM is carried out as follows. Upon activation within a party P_i on an action request of the form send(P_i, P_j, s, m), P_i sends the message (P_i, P_j, s, m) to party P_j, and outputs "P_i sent m to P_j in session s". Upon receipt of a message (P_i, P_j, s, m), P_j outputs "P_j received m from P_i in session s". Suppose we have an MT-authenticator λ which emulates this MT protocol. For a protocol π, a layered authenticator $\pi' = \mathcal{C}_\lambda(\pi)$ can be constructed as follows. For each message that π sends, λ is invoked and activated with an action request for sending that message to the corresponding recipient in UM. When π' is activated with some incoming message, $\mathcal{C}_\lambda$ also activates λ with the corresponding incoming message. When λ outputs, for example, "P_j received m from P_i in session s", π is activated with incoming message m from P_i.

It is assumed that each message transmitted in the network contains the identities of the sender and the receiver, as well as the session IDs of the sender's and the receiver's sessions. Throughout this paper, we also assume that the sender and the receiver share the same session ID and each message contains only one copy of it. When the identities of the sender and the receiver are implicitly specified in the context, we omit them in our description. In the original papers of [1,12], the authors use m to denote a message which comprises both the actual message content and a session ID. This causes certain inconvenience in presentation, as an attacker may choose to modify only the session ID or only the actual message content. In this paper, we adopt a more explicit approach. We use m to denote the actual content of a message only. It does not include sender or receiver identity, or the session ID.

2.4 Session-Key (SK) Security

For defining the session key security of a KE protocol, the capability of the adversary ($\mathcal{U}$ in UM or $\mathcal{A}$ in AM) is extended by allowing it to make a test-session query on a KE-session that is *completed*, *unexpired* and *unexposed*. Let κ be the session key (labeled as 'secret' in the session output) of the queried KE-session. A coin $b \in_R \{0,1\}$ is tossed by the game simulator. If $b = 0$, κ is returned to the adversary; otherwise, a random value chosen according to the probability distribution of session keys of the protocol is returned. The adversary can still carry out regular actions but is not allowed to expose the test-session (namely, neither of the test-session and its matching session can be locally exposed). Note that the adversary will be allowed to corrupt a partner to the test-session as soon as the test-session (or its matching session) expires at that party. This helps capture the forward secrecy property of a KE protocol. At the end of its run, the adversary outputs a bit b' (as its guess for b).

Definition 2 ([12]). *A KE protocol π is SK-secure if the properties below hold.*

1. *If two uncorrupted parties complete matching sessions, then they both output the same session key;*
2. *The probability that the adversary guesses correctly the bit b' (i.e., $b' = b$) is no more that $1/2$ plus a negligible fraction in the security parameter.*

According to [12, Theorem 6], it states that if π is a SK-secure KE protocol in AM and λ is an MT-authenticator, $\pi' = \mathcal{C}_\lambda(\pi)$ is a SK-secure KE protocol in UM. Therefore, a modular approach of using SK-secure KE protocol in AM and MT-authenticators to the design of SK-secure KE protocols in UM is obtained. Also, the MT-authenticators can be reused to construct new KE protocols.

3 The KCIR-enhanced CK-model

For a KE protocol, the property of Key Compromise Impersonation (KCI) Resilience [11,6,18] requires that compromising the long-term secret of a party A should not allow an adversary to masquerade *to* A as another party B, under the assumption that the adversary does not have the long-term secret of B. KCI resilience is an important property for KE protocols. In the CK-model, KCI resilience is not captured. As an example of a proven SK-secure KE protocol in the CK-model but being vulnerable to KCI attacks, we refer to [10, Protocol 1]. The protocol does not support KCI resilience as knowing the long-term secret of either one of the two communicating parties will allow the adversary to impersonate any of these two parties.

To see why a proven SK-secure KE protocol in the CK-model does not necessarily imply resistance to KCI attacks, we notice that compromising the long-term secret of a party in the CK-model corresponds to corrupting the party and this party can no longer participate in any further protocol interactions. As a result, the CK-model cannot capture KCI attacks in which the adversary is trying to impersonate *to* a party whose long-term secret is compromised.

A New Query. In order to incorporate the notion of KCI resilience into the CK-model, we introduce a new query called key compromise query into the model. When an adversary issues a key compromise query for a specified party, say P_i, the adversary will only learn the long-term secret of P_i but not any other internal information of P_i. A special note will also be written in P_i's local output which indicates that P_i is *compromised*. Sometimes, we may say that a session is compromised (or uncompromised). In that case, it is referring to the corresponding party of the session of being compromised (or not being compromised yet).

The difference between a party corruption query and a key compromise query is that, if a party is corrupted, then the adversary gets the long-term secret and all the internal states of the party, and the party stops being activated. While if a party is compromised, the attacker gets only the long-term secret of the party, and the party can still be activated and queried. Note a party can be uncorrupted but compromised.

Below is the definition for a secure protocol which supports KCI Resilience.

Definition 3. *In the enhanced CK-model described above, let π and π' be two n-party message driven protocols. We say that π' emulates π in UM if given any adversary $\mathcal{U}$ in UM against protocol π', there exists an adversary $\mathcal{A}$ in AM against π such that $AUTH_{\pi,\mathcal{A}}$ and $UNAUTH_{\pi',\mathcal{U}}$ are computationally indistinguishable.*

In other words, Def. 1 applies directly to the KCIR-enhanced CK-model. In the CK-model (original or enhanced), we define a protocol π' in UM to be "authenticated" if there exists protocol π in AM such that π' emulates π in UM. If a protocol π is authenticated in the enhanced CK-model above, then the protocol π also supports KCI resilience. To understand this statement, we should notice that KCI attacks can never happen in AM even when the key compromise query is allowed. However, if a protocol π is only authenticated in the original CK-model, but not authenticated in the enhanced CK-model, then π does not support KCI resilience.

We believe that KCI Resilience is a property related to "authentication". It applies to any authentication protocols, not necessarily KE protocols. From now on, we call our enhanced CK-model as "KCIR-enhanced CK-model".

Layered Authenticators. The definition of an authenticator is the same as before except that it is now under the context of the KCIR-enhanced CK-model. The way of constructing an authenticator is also the same as that of a layered authenticator given in [1] which is reviewed in Sec. 2.3.

Theorem 1. *If λ is an MT-authenticator in the KCIR-enhanced CK-model then $\mathcal{C}_\lambda$ constructed based on λ using the layered approach as described in [1] and reviewed in Sec. 2.3 is an authenticator.*

The proof is similar to that of [1, Theorem 3]. As an example of an MT-authenticator in the KCIR-enhanced CK-model, we can see that the signature-based MT-authenticator described in [1, Sec. 3.1] can be shown to be an authenticator of the MT protocol in the KCIR-enhanced CK-model. A proof can be obtained by following the approach given in [1, Proposition 4].

SK Security. With the additional key compromise query, the adversary now has one more method (besides party corruption) to obtain the long-term secret of a

party. Therefore, we should also consider the effect of key compromise when defining the SK security of a KE protocol. First, we require that the adversary can only make a test-session query on a KE-session that is completed, unexpired, unexposed and *uncompromised*. After completing the query, the adversary can continually carry out regular actions but is not allowed to *expose* the test-session. However, the adversary can now compromise the party. Since compromising a party only lets the adversary get the long-term secret of the party, hence once the test-session is completed, the adversary is allowed to issue key compromise to the party, that will have the same effect as corrupting the party after the test-session expires.

The definition of SK-secure for KE protocols should also be modified.

Definition 4. *A KE protocol π is called SK-secure in KCIR-enhanced CK-model if the following properties hold.*

1. *If two uncorrupted and* ***uncompromised*** *parties complete matching sessions, then they both output the same session key;*
2. *The probability that the adversary guesses correctly the bit b' (i.e., $b' = b$) is no more that $1/2$ plus a negligible fraction in the security parameter.*

By following the proof of [12, Theorem 6], we can also show that if there exists a SK-secure KE protocol in AM, by using an MT-authenticator for the KCIR-enhanced CK-model (such as the signature-based MT-authenticator mentioned above), a SK-secure KE protocol in UM can be derived and the protocol will then support KCI resilience. For example, Protocol SIG-DH [12] is SK-secure and also resistant to KCI attacks. This is because Protocol 2DH [12] can still be shown to be SK-secure in AM under the KCIR-enhanced CK-model; then by applying the signature-based MT-authenticator mentioned above, which is the same as the original authenticator of [1], to Protocol 2DH, we obtain Protocol SIG-DH.

For the scenario where perfect forward secrecy (PFS) is not needed, we need to prevent the adversary from getting the long-term secrets of involving parties of the test-session. Besides requiring that the test-session never expire (for preventing the adversary from issuing party corruption query onto the parties of the test-session or its matching session), we also need to prohibit the adversary issue any key compromise query on these parties.

There are also some scenarios where KCI resilience is not necessary or may not be possible. For example, in some applications an adversary would not gain any advantage when masquerading to the compromised party as a different party, or a conventional symmetric-key based KE protocol simply does not support KCI resilience. In these cases, we should remove key compromise from the adversary's list of allowable queries and the model will fall back to the original CK-model.

In the next section, we further extend the KCIR-enhanced CK-model for supporting the general ID-based setting and KGS-FS for ID-based KE protocols.

4 The ID-based Enhanced CK-model

In an ID-based setting, there is a key generation server (KGS) which has a *master key*. The KGS uses the master key to generate the long-term secret key for each

of the parties in the system so that the secret key of any party can be derived directly from the master key or the KGS and the unique identity of the party. In the context of ID-based KE protocols (or authentication protocols in general), additional security concerns are raised by considering this extra party, KGS, in the system and the property above. In particular, for ID-based KE protocols, the notion of KGS forward secrecy (KGS-FS) is to require a session key should remain secure even after the master key of the KGS is compromised. This is at least as strong as the perfect forward secrecy (PFS) of a non-ID-based KE protocol as knowing the KGS' master key implies knowing the secret keys of all parties in the system.

In the following, we describe the changes that need to be made on the KCIR-enhanced CK-model for extending it to an ID-based enhanced model.

4.1 ID-based Message Driven and KE Protocols

An ID-based message driven protocol $\tilde{\pi}$ is a collection of programs, each program is to be run by a different party which is *created* by the KGS. We emphasize that it is the KGS who creates parties during the protocol execution. Each party can perform the same set of actions as described in Sec. 2.1. An ID-based KE protocol is an ID-based message driven protocol with similar specification to that of a KE protocol described in Sec. 2.1.

4.2 The Adversarial Models

Similar to the CK-model (original or KCIR-enhanced), there are two models, UM and AM, with adversary $\tilde{\mathcal{U}}$ and $\tilde{\mathcal{A}}$, respectively. The sets of actions that can be carried out by these two adversaries are similar to their counterparts in the KCIR-enhanced CK-model. In addition to these, we introduce two additional adversarial activities to our models for capturing the existence of the KGS. They are create party query and corrupt KGS query.

The UM. Let k be a security parameter. The system has the UM-adversary $\tilde{\mathcal{U}}$, a PPT machine called KGS and a number of parties denoted by $P_1, \cdots, P_N$ (also modeled as PPT machines) for the ID-based message driven protocol $\tilde{\pi}$. The number of parties in the system is N where N is a polynomial in k. We assume that $\tilde{\mathcal{U}}$ does not *create* (which will be defined later) more than N parties. The initialization function I is changed to give out a pair of outputs only: $I(r,k) = (I(r,k)_0,\ I(r,k)_{KGS})$, where $I(r,k)_0$ is the public information which becomes known to all parties, the KGS and the adversary while $I(r,k)_{KGS}$ becomes known only to the KGS and it is called the *master key* of the KGS. In the system, we also assume that there is a secure channel between the KGS and each of the N parties. Hence the KGS can make use of this secure channel to send information of a particular party so that the information will only be known to the party and the KGS.

When $\tilde{\mathcal{U}}$ makes a create party query with a specified party, say P_i with identity ID_i, the KGS is activated to assign the identity ID_i to P_i, which is not yet created. Then, KGS computes a long-term secret sk_i from ID_i and the master key of the KGS using an algorithm called user key generation algorithm which should be

defined explicitly in the protocol specification. sk_i is then sent to P_i through the secure channel. A special note is also appended to the KGS' local output which specifies that P_i has been created with identity ID_i. Party P_i is said to be *created*. The create party query can only be made once for each of the parties, and the number of created parties is not fixed, instead it grows with the number of create party queries made by the adversary.

Only after a party is created, $\tilde{\mathcal{U}}$ can start activating the party either by incoming message queries or action request queries, and perform all the adversarial actions described in Sec. 2.2 and key compromise queries described in Sec. 3 on the party. Note that all queries are only allowed to be made on created parties.

There is an additional adversarial action for capturing the property of KGS-FS. By issuing a corrupt KGS query, $\tilde{\mathcal{U}}$ can learn the master key of the KGS. This event is recorded through a special note in the KGS' local output. From this point on, the KGS cannot be activated anymore and there is no further local output generated. The KGS is said to be *corrupted*.

The global output $UNAUTH_{\tilde{\pi},\tilde{\mathcal{U}}}$ now consists of the cumulative outputs of all *created* parties as well as the outputs of $\tilde{\mathcal{U}}$ and KGS.

The AM. An AM-adversary $\tilde{\mathcal{A}}$ has all the capabilities of $\mathcal{A}$ described in Sec. 2.2 and Sec. 3 with the additional adversarial actions, create party and corrupt KGS. The global output $AUTH_{\tilde{\pi},\tilde{\mathcal{A}}}$ is analogous to $UNAUTH_{\tilde{\pi},\tilde{\mathcal{U}}}$, where the computation is carried out in the AM.

4.3 Authenticators

The definition of an authenticator remains unchanged except that it should now be defined under the context of the ID-based enhanced CK-model. To construct an authenticator, we can also use the layered approach to construct a **layered authenticator** from an MT-authenticator as described in Sec. 2.3.

Theorem 2. *If λ is an MT-authenticator in the ID-based enhanced CK-model, then $\mathcal{C}_\lambda$ constructed based on λ using the layered approach as described in [1] and reviewed in Sec. 2.3 is an authenticator.*

The proof is similar to that of [1, Theorem 3]. As an example of an MT-authenticator in the ID-based enhanced CK-model, we can see that the signature-based MT-authenticator described in [1, Sec. 3.1] can be shown to be an authenticator of the MT protocol in the ID-based enhanced CK-model after changing the signature scheme of the initiator to an ID-based signature scheme [22,20,13,2]. The security requirement of the ID-based signature scheme is the ID-based extension of the conventional existential unforgeability against adaptive chosen message (euf-cma). It concerns about existential unforgeability against adaptive chosen message attack as well as chosen identity attack [13,2] (euf-cma-ida). A proof for this MT-authenticator can be obtained by following the approach given in [1, Proposition 4].

4.4 SK Security for ID-based KE Protocols

The definition of a secure ID-based KE protocol can be formalized by following Def. 4 and including the impacts brought in by the two additional adversarial actions, i.e. create party and corrupt KGS. First, we require that the adversary can only make a test-session query on a KE-session that is completed, unexpired, unexposed, uncompromised and also with the KGS being *uncorrupted*. After completing the query, the adversary can continually carry out regular actions but is not allowed to expose the test-session. However, the adversary can now compromise the party or even issue a corrupt KGS query. The effect would be similar to corrupting the party corresponding to the test-session after it expires.

The definition of SK-secure should also be modified.

Definition 5. *An ID-based KE protocol $\tilde{\pi}$ is called SK-secure in ID-based enhanced CK-model if the following properties hold.*

1. *With KGS uncorrupted, if two uncorrupted and uncompromised but* ***created*** *parties complete matching sessions, then both output the same session key;*
2. *The probability that the adversary guesses correctly the bit b' (i.e., $b' = b$) is no more than $1/2$ plus a negligible fraction in the security parameter.*

By following the proof of [12, Theorem 6], we can also show that if there exists a SK-secure KE protocol in AM of the ID-based enhanced CK-model, by using an MT-authenticator for the ID-based enhanced CK-model, a SK-secure KE protocol in UM can be derived.

KGS Forward Secrecy (KGS-FS). The original CK-model captures the notion of perfect forward secrecy (PFS), that is, a previously established session key should remain secure even after the long-term secrets of both involving parties are compromised. In the context of ID-based KE protocols, KGS forward secrecy (KGS-FS) [17,14,19] provides an even stronger protection to session keys than that of PFS. By KGS-FS, previously established session keys will still remain secure even after the master key of the KGS is compromised. The ID-based enhanced CK-model above captures the notion of KGS-FS by allowing the adversary to issue corrupt KGS once after the test-session query returns.

For scenarios where KGS-FS is not needed and only PFS is required, we need to remove corrupt KGS from the adversary's list of allowable queries. Note that although the adversary is not allowed to corrupt KGS, he can still issue a party corruption query on a party after the test-session is expired. Hence PFS is captured.

5 SK-Secure ID-Based KE Protocols

In this section, we review two ID-based KE protocols and show how to show their security under the ID-based enhanced CK-model. The first one is a protocol which supports KCI Resilience and KGS-FS while the second one supports KCI Resilience, but not KGS-FS nor PFS.

5.1 An ID-based KE Protocol supporting KCI Resilience and KGS-FS

In [27], Zhu et al. proposed an ID-based KE protocol which does not use bilinear pairings. They also showed the security of their protocol in the original CK-model. This implies that it supports PFS. However, it does not show that the protocol satisfies KGS-FS. In the following, we use the ID-based enhanced CK-model to show that the protocol supports KGS-FS and also has the property of KCI Resilience.

Let $k \in \mathbb{N}$ be a security parameter. The initial information $I(r,k)_0$ consists of a finite field $\mathbf{F}$, an elliptic curve $\mathscr{C}$ defined over $\mathbf{F}$, an element P of large prime order q in $\mathscr{C}$, and also the public key of the KGS denoted by mpk. This public key is generated as $mpk = mskP$ where $msk \in_R \mathbb{Z}_q$. The secret information $I(r,k)_{KGS}$ for the KGS is therefore the value of msk which is the master key of KGS. There is also an ID-based signature scheme associated with the protocol. For our discussion in this paper, we do not go into the details of the scheme and simply denote a signature generated by the scheme as $IDSign_A(m)$ where m is the message and the subscript A indicates that the signature is generated by party A using its long-term secret. The signature can be verified using A's "public key", that is, A's identity denoted by ID_A. Note that the scheme is shown to be euf-cma-ida.

Let A and B be the initiator and responder identified by $ID_A, ID_B \in \{0,1\}^*$. Let sk_A and sk_B be the long-term secrets of A and B, respectively. They are generated by the KGS when A and B are created. A generation is done by using the Schnorr signature scheme [21] on the party's identity as the message under the master key of the KGS. Suppose A and B already have a unique session-id s shared. The protocol proceeds as follows.

1. The initiator, A, on input (A,B,s), chooses $a \in_R \mathbb{Z}_q$ and sends $(A,s,T_A = aP)$ to B.
2. Upon receipt of (A,s,T_A), the responder B chooses $b \in_R \mathbb{Z}_q$ and sends $(B,s,T_B = bP)$ together with its signature $IDSign_B(s,T_B,T_A,A)$; it also computes the session key $K = bT_A$ and erases b.
3. Upon receipt of (B,s,T_B) and B's signature, party A checks the correctness of each component in the incoming message and checks whether the signature is valid with respect to the "public key" ID_B. If the verification succeeds, A sends $(A,s,IDSign_A(s,T_A,T_B,B))$ to B. A then computes $K' = aT_B$, erases a, and outputs the session key K' under session-id s.
4. Upon receipt of (A,s,sig), B checks the correctness of each component in the incoming message and determines if the signature sig is valid with respect to the "public key" ID_A. If yes, B outputs the session key K under session-id s.

Security Analysis. The protocol can be shown to be SK-secure (with KCI Resilience and KGS-FS) in the ID-based enhanced CK-model in two steps. First, an ID-based KE protocol is proposed and proven SK-secure in AM. Second, some appropriate MT-authenticators are applied to the SK-secure KE protocol in AM using the layered approach for obtaining a SK-secure ID-based KE protocol in UM.

For the first step, we begin with Protocol 2DH [12, Sec. 5.1] and describe it in the context of elliptic curve cryptography. It is the classical two-move Diffie-Hellman key exchange protocol. When formalized in AM, we describe it as having

the initiator A send $(A, s, T_A = aP)$ to B and then having the responder B send $(B, s, T_B = bP)$ back to A. The session key is abP. According to [12, Theorem 8], Protocol 2DH is SK-Secure in AM under the Decisional Diffie-Hellman assumption. In our ID-based enhanced CK-model, we can also construct a Distinguisher $\tilde{\mathcal{D}}$ which proceeds in the same way as the Distinguisher $\mathcal{D}$ in the proof of [12, Theorem 8] except with the following additional actions: whenever the adversary $\tilde{\mathcal{A}}$ creates a new party, $\tilde{\mathcal{D}}$ creates a party accordingly; whenever a *created* party is compromised or the KGS is corrupted, $\tilde{\mathcal{D}}$ hands the related information of that party or KGS to $\tilde{\mathcal{A}}$, respectively. The proof will then follow and we will get the contradiction we want for showing the SK security of Protocol 2DH in AM of the ID-based enhanced CK-model.

For the second step, in Sec. 4.3, we mention that the signature-based MT-authenticator described in [1, Sec. 3.1] can be converted into an MT-authenticator for the ID-based enhanced CK-model after changing the signature scheme to an euf-cma-ida secure ID-based signature scheme. In the following, we illustrate this MT-authenticator where $IDSign_A(\cdot)$ denotes the ID-based signature generated under A's long-term secret.

$$
\begin{aligned}
&A \rightarrow B : s, m \\
&A \leftarrow B : s, m,\ N_B \\
&A \rightarrow B : s, m,\ IDSign_A(s, m, N_B, B)
\end{aligned}
$$

Therefore in step two, we can transform Protocol 2DH in AM to a protocol in UM by applying the MT-authenticator above to each of the two messages of Protocol 2DH using the layered approach. By using the optimization technique [1,12] and eliminating some redundant components, we can obtain the resultant protocol which is identical to Zhu et al.'s protocol reviewed above.

5.2 An ID-based KE Protocol supporting KCI Resilience but not PFS

In 2001, Boneh and Franklin proposed the first ID-based encryption scheme [8] from the Weil pairing, which opens a new era in ID-based cryptography. Since then a number of pairing-based ID-based KE protocols have been proposed [24,14,10,19,15]). In [24], Smart proposed one of the first pairing-based ID-based KE protocols. As noted by Smart in [24], the protocol does not support KGS-FS nor PFS. It only supports *partial* forward secrecy, namely the session key will remain secure if only one of the two involving parties is compromised. There is no security proof given in the paper. In the following, we briefly review Smart's protocol with the key confirmation steps included, and show that with a slight modification, we can show that the *modified* protocol of Smart's is SK-secure in our ID-based enhanced CK-model if the test-session never expires and corrupt KGS as well as key compromise on parties of the test-session and its matching session are not allowed.

For simplicity, instead of using the notations above for denoting the communicating parties, we follow Smart's paper to use A to represent the initiator and B to represent the responder. Let $\mathbb{G}_1$ and $\mathbb{G}_2$ be two cyclic groups of prime order q. Let P be the generator of $\mathbb{G}_1$. Let $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_2$ be a computable and non-degenerate bilinear map, and $H_1 : \{0,1\}^* \rightarrow \mathbb{G}_1$ be a map-to-point hash function. The KGS

public key P_{KGS} is computed as $s_{KGS}P$ where $s_{KGS} \in_R \mathbb{Z}_q$. When creating a party A with identity ID_A, the user's long-term secret is set to $S_A = s_{KGS}Q_A$ where $Q_A = H_1(ID_A)$. Let MAC be a secure message authentication code. The Smart's ID-based KE protocol [24] for a session s is shown as follows.

$$\begin{aligned} &A \rightarrow B : s,\ T_A = aP \\ &A \leftarrow B : s,\ T_B = bP,\ MAC_\sigma(2, B, A, R) \\ &A \rightarrow B : s,\ MAC_\sigma(3, A, B, R) \end{aligned}$$

Here $a, b \in_R \mathbb{Z}_q$ are randomly chosen, and R is computed as $\hat{e}(aT_B, P_{KGS})$ and $\hat{e}(bT_A, P_{KGS})$ by A and B, respectively. At the end of the protocol, A and B compute $\kappa_A = \hat{e}(aQ_B, P_{KGS}) \cdot \hat{e}(S_A, T_B) = \hat{e}(bQ_A, P_{KGS}) \cdot \hat{e}(S_B, T_A) = \kappa_B$. Then a pair of keys is computed as $(K, \sigma) = H(\kappa_A) = H(\kappa_B)$, where $H : \{0,1\}^* \rightarrow \{0,1\}^{2k}$ is a key derivation function for security parameter k. K is the session key and σ is the MAC key. In the original protocol description [24], the range of the key derivation function is defined over $\{0,1\}^*$. For quantifying the security analysis of the session key and the MAC key, we herewith explicitly specify the length of the keys generated by the key derivation function in terms of the security parameter k.

In the above, as $\kappa_A = \kappa_B = \hat{e}(S_A, T_B) \cdot \hat{e}(S_B, T_A)$, we can see that if both A and B are corrupted via party corruption after the corresponding sessions at A and B are expired, the adversary can compute the session key. Hence it does not support PFS.

Also notice that Smart's protocol is vulnerable to session corruption attack [25]. That is, by making use of session-state reveal queries, the UM-adversary will have non-negligible advantage in guessing the bit b' under the definition of SK security (Def. 5). In [14], Chen and Kudla also remarked that Smart's protocol can be shown secure in a *restricted* Bellare-Rogaway model [4]. The restricted model does not allow the adversary to make any Reveal query. This is related to the session corruption attack.

In the following, we slightly modify Smart's protocol so that it can be proven SK-secure in our ID-based enhanced CK-model provided that a test-session would never expire and corrupt KGS as well as key compromise on parties of test-session (and its matching session) are not allowed. In other words, the modified protocol will be secure against session corruption attack although PFS is still not supported. To maintain the original structure of Smart's protocol, we tend not to remove anything from the protocol but only insert some additional components for strengthening its security.

In the modification, we put the session ID, T_A and T_B into the MACs of the second and third messages. The key derivation function $H : \{0,1\}^* \rightarrow \{0,1\}^k$ now only generates the session key K, namely $K \leftarrow H(\kappa)$ where $\kappa = \hat{e}(S_A, T_B) \cdot \hat{e}(S_B, T_A)$. Let $H' : \{0,1\}^* \rightarrow \{0,1\}^k$ be a different key derivation function from H. The MAC key for the message from B to A now becomes $\sigma_1 \leftarrow H'(B, A, s, T_B, T_A, \kappa)$, and the MAC key for the message from A to B is generated as $\sigma_2 \leftarrow H'(A, B, s, T_A, T_B, \kappa)$. The resulting protocol is illustrated as follows.

$$A \rightarrow B : s,\ T_A = aP$$
$$A \leftarrow B : s,\ T_B = bP,\ MAC_{\sigma_1}(s, 2, B, A, R, T_B, T_A)$$
$$A \rightarrow B : s,\ MAC_{\sigma_2}(s, 3, A, B, R, T_A, T_B)$$

Security Analysis. For security analysis, we consider H_1, H and H' to be random oracles [3]. To show that the modified protocol of Smart's supports KCI Resilience and is also SK-secure (but without KGS-FS nor PFS), we follow the following two steps. First, an ID-based KE protocol is proposed and proven SK-secure (without KGS-FS nor PFS) in AM provided that test-session never expires and corrupt KGS as well as key compromise on parties of test-session (and its matching session) are not allowed. Second, some MT-authenticator, which is shown in the ID-based enhanced CK-model with KCI Resilience (i.e. allowing key compromise), is applied to the protocol in AM using the layered approach for constructing a protocol in UM.

For the first step, we begin with the two-move Smart's protocol [24] which is simply an exchange of T_A and T_B between A and B (with their identities and session ID attached in each of the message flow). We call this protocol the **2-Move Smart Protocol**.

Bilinear Diffie-Hellman (BDH) Assumption: Given $(P, xP, yP, zP) \in \mathbb{G}_1$ for some x, y, z randomly chosen from $\mathbb{Z}_q$, it is difficult to find $\hat{e}(P, P)^{xyz} \in \mathbb{G}_2$.

Theorem 3. *The 2-Move Smart Protocol is SK-secure in AM of the ID-based enhanced CK-model if the BDH assumption holds, provided that test-session never expires and corrupt KGS as well as key compromise on parties of test-session and its matching session are not allowed.*

Proof. When two uncorrupted and uncompromised parties A and B complete two matching sessions, they output the same session key $K \leftarrow H(\kappa)$ where $\kappa = \hat{e}(aQ_B, P_{KGS}) \cdot \hat{e}(S_A, T_B) = \hat{e}(aS_B + bS_A, P) = \hat{e}(bQ_A, P_{KGS}) \cdot \hat{e}(S_B, T_A)$. So Condition 1 in Def. 5 holds.

We prove that Condition 2 also holds by the way of contradiction. Suppose that an adversary $\tilde{\mathcal{A}}$ succeeds in guessing the bit b' with non-negligible advantage, then we can use $\tilde{\mathcal{A}}$ to construct an adversary $\mathcal{S}$ which solves the BDH problem with non-negligible probability, contradicting the BDH assumption. Given the tuple $(\mathbb{G}_1, \mathbb{G}_2, \hat{e})$ and (P, xP, yP, zP), $\mathcal{S}$ is to find the value of $\hat{e}(P, P)^{xyz}$.

$\mathcal{S}$ starts by running $\tilde{\mathcal{A}}$ on the following simulated interaction with KGS. $\mathcal{S}$ first sets P_{KGS} to xP. The public key of each party is computed on request as $Q_i = H_1(ID_i) = r_i P$ ($r_i \in_R \mathbb{Z}_q$) except for some randomly chosen parties P_f and P_g whose public keys are set to yP and zP respectively. $\mathcal{S}$ then distributes long-term secret $S_i = r_i \cdot xP$ to each party except for parties P_f and P_g. The public information are also given to each party and the adversary following the protocol specification except that $Q_f = yP$, $Q_g = zP$. Next, when $\tilde{\mathcal{A}}$ activates any party, $\mathcal{S}$ follows the protocol specification on behalf of that party with the exception that when $\tilde{\mathcal{A}}$ activates party P_f to communicate with P_g, $\mathcal{S}$ will let P_f send $(1/2)yP$ to P_g and let P_g reply with $(1/2)zP$. As a result the shared key between P_f and P_g will be

$$H(\hat{e}((1/2)y \cdot zP, xP) \cdot \hat{e}((1/2)z \cdot yP, xP)) = H(\hat{e}(P, P)^{xyz})$$

When dealing with oracle H, $\mathcal{S}$ maintains a list L, which stores all the inputs and the corresponding outputs to H, to consistently reply the oracle queries with a list of random values. If $\tilde{\mathcal{A}}$ issues a party corruption query or a key compromise query (at some allowable conditions) on party P_f or P_g, $\mathcal{S}$ fails and aborts. Note that $\tilde{\mathcal{A}}$ cannot corrupt KGS here.

Now, suppose that $\tilde{\mathcal{A}}$ succeeds with non-negligible advantage ϵ. Suppose there are at most ℓ sessions in the system, where ℓ is a polynomial in the security parameter k. Then the probability that $\tilde{\mathcal{A}}$ succeeds on the session above between P_f and P_g should be at least ϵ/ℓ. Since H is a random oracle, when $\tilde{\mathcal{A}}$ succeeds, $\tilde{\mathcal{A}}$ must have queried H on $\hat{e}(P,P)^{xyz}$. This implies that $\mathcal{S}$ is able to compute $\hat{e}(P,P)^{xyz}$ with probability at least ϵ/ℓ which contradicts the BDH assumption. □

For the second step, we propose the following MT-authenticator.

An MAC-based MT-authenticator. We use the same set of notations as the review of Smart's protocol. Suppose that there are two *created* parties A and B, with private keys $S_A = s_{KGS}Q_A$ and $S_B = s_{KGS}Q_B$ respectively. The MT-authenticator is illustrated as follows.

$$A \rightarrow B : s,\ m,\ T'_A = a'P$$
$$A \leftarrow B : s,\ m,\ T'_B = b'P$$
$$A \rightarrow B : s,\ m,\ MAC_\sigma(s, COUNT, A, B, R, m, T'_A, T'_B)$$

$a', b' \in_R \mathbb{Z}_q$ are randomly chosen. The MAC key σ is computed using a key derivation function $H' : \{0,1\}^* \rightarrow \{0,1\}^k$ as $\sigma \leftarrow H'(A, B, s, m, T'_A, T'_B, \kappa)$ where $\kappa = \hat{e}(a'Q_B, P_{KGS}) \cdot \hat{e}(b'Q_A, P_{KGS}))$. We can see that A can compute κ as $\hat{e}(a'Q_B, P_{KGS}) \cdot \hat{e}(S_A, T'_B)$ and B can compute it as $\hat{e}(b'Q_A, P_{KGS}) \cdot \hat{e}(S_B, T'_A)$. For security analysis, H' is considered to be a random oracle. $COUNT$ is a value which is distinct for each message sent. $R = \hat{e}(a'T'_B, P_{KGS}) = \hat{e}(b'T'_A, P_{KGS})$.

Theorem 4. *The MAC-based MT-authenticator described above emulates the MT protocol in UM of the ID-based enhanced CK-model if the BDH assumption holds.*

Proof (Sketch). We need to show that for any UM-adversary $\tilde{\mathcal{U}}$ against the MAC-based MT-authenticator above, there exists an AM-adversary $\tilde{\mathcal{A}}$ against the MT protocol so that $AUTH_{MT,\tilde{\mathcal{A}}}$ and $UNAUTH_{\lambda_{MAC},\tilde{\mathcal{U}}}$ are computationally indistinguishable.

$\tilde{\mathcal{A}}$ runs $\tilde{\mathcal{U}}$ on a simulated interaction with a set of imitated parties (*created*) running the MT-authenticator above and an imitated KGS'. Then, $\tilde{\mathcal{A}}$ emulates most of the activities of $\tilde{\mathcal{U}}$ according to the proof of [1, Proposition 4] until event $\mathcal{B}$ (defined in the proof of [1, Proposition 4]) occurs. Note that in the above emulation, when $\tilde{\mathcal{U}}$ issues a KGS corruption query in UM, $\tilde{\mathcal{A}}$ also corrupts KGS' in AM and then relays the *master secret* to $\tilde{\mathcal{U}}$. When $\tilde{\mathcal{U}}$ issues a create party query in UM, $\tilde{\mathcal{A}}$ issues the same query to KGS' in AM and passes the related information to $\tilde{\mathcal{U}}$. When $\tilde{\mathcal{U}}$ issues a key compromise query to an imitated party in UM, $\tilde{\mathcal{A}}$ issues the same query to the corresponding party in AM and passes the long-term secret to $\tilde{\mathcal{U}}$.

Suppose that event $\mathcal{B}$ occurs with non-negligible probability ϵ, then we construct an adversary (a BDH solver) $\mathcal{S}$ to solve the BDH problem with non-negligible

probability, which contradicts the BDH assumption. The reduction is similar to that in the proof of Theorem 3 above. Below are the differences. For the BDH problem instance (P, xP, yP, zP), xP is assigned to P_{KGS}, yP is assigned to Q_f for some randomly chosen party P_f, and zP is assigned to T'_g in the fist outgoing message of some session s of some randomly chosen party P_g. If key compromise or party corruption is queried for P_f, the system aborts. By following the proof of Theorem 3, we will find that with non-negligible probability the adversary will query H with $\hat{e}(zQ_f, P_{KGS}) \cdot \hat{e}(a'Q_g, P_{KGS})$ where a' is the discrete logarithm of T'_f which is in the first outgoing message of P_f in the matching session s. Note that the BDH solver $\mathcal{S}$ knows the discrete logarithm of Q_g as it is generated by $\mathcal{S}$. Hence $\mathcal{S}$ can compute $a'Q_g$ from T'_f and Q_g even when T'_f could possibly be generated by $\tilde{\mathcal{U}}$. Finally $\mathcal{S}$ solves the BDH problem instance by computing $\hat{e}(zQ_f, P_{KGS}) = \hat{e}(P, P)^{xyz}$.

The reason why we use a different reduction approach from the one used in Theorem 3 is because we need to capture the key compromise impersonation (KCI) attack against P_g. In the simulation, P_f correspond to A and P_g correspond to B of the MAC-based MT-authenticator above. By using the reduction approach of Theorem 3, $\mathcal{S}$ cannot simulate the KCI attack against P_g (or B) because $\mathcal{S}$ does not know the value of xQ_g (which is xzP in the proof of Theorem 3) and therefore is unable to answer the key compromise query for P_g. Since event $\mathcal{B}$ also includes the KCI attack against P_g, we need to simulate this attack without aborting the simulation. □

Finally, we transform the 2-Move Smart Protocol in AM to a protocol in UM by applying the MAC-based MT-authenticator shown above using the layered approach. By using the optimization technique [1,12] and eliminating some redundant components such as T_A and T_B, which can be used both as the messages and the challenges, we can obtain the resultant protocol which is identical to the modified version of Smart's protocol described above.

References

1. M. Bellare, R. Canetti, and H. Krawczyk. A modular approach to the design and analysis of authentication and key exchange protocols. In *Proc. 30th ACM Symp. on Theory of Computing*, pages 419–428. ACM, May 1998. Full paper available at the first author's homepage.
2. M. Bellare, C. Namprempre, and G. Neven. Security proofs for identity-based identification and signature schemes. In *Proc. EUROCRYPT 2004*, pages 268–286. Springer-Verlag, 2004. LNCS 3027 (Full paper is available at Bellare's homepage URL: `http://www-cse.ucsd.edu/users/mihir`).
3. M. Bellare and P. Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In *First ACM Conference on Computer and Communications Security*, pages 62–73, Fairfax, 1993. ACM.
4. M. Bellare and P. Rogaway. Entity authentication and key distribution. In *Proc. CRYPTO 93*, pages 232–249. Springer-Verlag, 1994. LNCS 773.
5. M. Bellare and P. Rogaway. Provably secure session key distribution – the three party case. In *Proc. 27th ACM Symp. on Theory of Computing*, pages 57–66, Las Vegas, 1995. ACM.

6. S. Blake-Wilson, D. Johnson, and A. Menezes. Key agreement protocols and their security analysis. In *Sixth IMA International Conference on Cryptography and Coding*, pages 30–45. Springer-Verlag, 1997. LNCS 1355.
7. S. Blake-Wilson and A. Menezes. Entity authentication and authenticated key transport protocols employing asymmetric techniques. In *Security Protocols Workshop*, pages 137–158. Springer-Verlag, 1997. LNCS 1361.
8. D. Boneh and M. Franklin. Identity-based encryption from the Weil pairing. In *Proc. CRYPTO 2001*, pages 213–229. Springer-Verlag, 2001. LNCS 2139.
9. C. Boyd and K.-K. R. Choo. Security of two-party identity-based key agreement. In *Mycrypt 2005*, pages 229–243. Springer-Verlag, 2005. LNCS 3715.
10. C. Boyd, W. Mao, and K. G. Paterson. Key agreement using statically keyed authenticators. In *ACNS 2004*, pages 248–262. Springer-Verlag, 2004. LNCS 3089.
11. C. Boyd and A. Mathuria. *Protocols for Authentication and Key Establishment*. Springer-Verlag, 2003.
12. R. Canetti and H. Krawczyk. Analysis of key-exchange protocols and their use for building secure channels. In *Proc. EUROCRYPT 2001*, pages 453–474. Springer-Verlag, 2001. LNCS 2045. `http://eprint.iacr.org/2001/040`.
13. J. C. Cha and J. H. Cheon. An identity-based signature from gap Diffie-Hellman groups. In *Public Key Cryptography 2003*, pages 18–30. Springer-Verlag, 2002. LNCS 2567.
14. L. Chen and C. Kudla. Identity based authenticated key agreement protocols from pairings. Cryptology ePrint Archive, Report 2002/184 (Revised Date: 27 May 2004), 2002. `http://eprint.iacr.org/2002/184`.
15. K. Y. Choi, J. Y. Hwang, D. H. Lee, and I. S. Seo. ID-based authenticated key agreement for low-power mobile devices. In *Information Security and Privacy, 10th Australasian Conference (ACISP 2005)*, pages 494–505. Springer, 2005. LNCS 3574.
16. Y. J. Choie, E. Jeong, and E. Lee. Efficient identity-based authenticated key agreement protocol from pairings. *Applied Mathematics and Computation*, 162(1), 2005.
17. C. Günther. An identity-based key exchange protocol. In *Proc. EUROCRYPT 89*, pages 29–37. Springer-Verlag, 2000. LNCS 434.
18. M. Just and S. Vaudenay. Authenticated multi-party key agreement. In *Proc. ASIACRYPT 96*, pages 36–49. Springer-Verlag, 1996. LNCS 1163.
19. N. McCullagh and P. S. L. M. Barreto. A new two-party identity-based authenticated key agreement. In *CT-RSA 2005*, pages 262–274. Springer-Verlag, 2005. LNCS 3376.
20. T. Okamoto. Provably secure and practical identification schemes and corresponding signature schemes. In *Proc. CRYPTO 92*, pages 31–53. Springer-Verlag, 1993. LNCS 740.
21. C. P. Schnorr. Efficient identification and signatures for smart cards. In *Proc. CRYPTO 89*, pages 239–252. Springer, 1990. LNCS 435.
22. A. Shamir. Identity-based cryptosystems and signature schemes. In *Proc. CRYPTO 84*, pages 47–53. Springer, 1984. LNCS 196.
23. K. Shim. Efficient id-based authenticated key agreement protocol based on Weil pairing. *IEE Electronics Letters*, 39(8):653–654, April 2003.
24. N. Smart. Identity-based authenticated key agreement protocol based on Weil pairing. *IEE Electronics Letters*, 38(13):630–632, June 2002.
25. X. Tian and D. S. Wong. Session corruption attack and improvements on encryption based MT-authenticators. In *CT-RSA 2006*, pages 34–51. Springer-Verlag, 2006. LNCS 3860.
26. Y. Wang. Efficient identity-based and authenticated key agreement protocol. Cryptology ePrint Archive, Report 2005/108, 2005. `http://eprint.iacr.org/2005/108`.

27. R. W. Zhu, G. Yang, and D. S. Wong. An efficient identity-based key exchange protocol with KGS forward secrecy for low-power devices. In *The 1st Workshop on Internet and Network Economics (WINE 2005)*, pages 500–509. Springer-Verlag, 2005. LNCS 3828.