DEPARTMENT OF COMPUTER SCIENCE

Separation between Games and UC for the case of Key Exchange Protocols

Shailesh Prakash

A dissertation submitted to the University of Bristol in accordance with the requirements of the degree of Master of Science in the Faculty of Engineering

September 2012 | CSMSC-12

ABSTRACT

In this paper we have investigated the relationship between two standard security models: Cryptographic games and Simulation based models. The obtained result is new and previously unexplored in the context of key exchange protocol. This is by pointing out inability to simulate specific corruption scheme in presence of adaptive adversary in the simulation based security model. The results we found here provide the basic step to form the relationship between these two models. Notion of both the security models are not new but unfortunately the relationship is not fully understood. In general, cryptographers believe that simulation-based model provides stronger security guarantee than cryptographic games. To understand this problem, we model security guarantee of key exchange protocol in both the models. As our aim is to compare these two security models instead of providing security guarantee of actual protocol, we proposed a new and simple to construct but sufficiently secure key exchange protocol, KE(ENC,SIG). The protocol is based on two fundamental public key cryptographic primitive, encryption and signature, to provide secrecy and authenticity respectively. We used classical Bellare-Rogaway model (BR model) for cryptographic games technique. For the simulation-based model, popular Canetti's Universal Compose-able framework has been used. As a main result, KE(ENC,SIG) is found secure for BR model in presence of the adaptive adversary. On the other hand UC model declare the same protocol unsecure under same adversarial entity. The security analysis focuses on key exchange and assumed long-term keys are pre-established and public keys are verifiable with protocol user's identities within the system. Our results are based on previous work done by Nielsen for the non-committing encryption to show separation between complexity theory and random oracle models. The reason for failure in UC framework security modeling is due to a fundamental issue of simulation in presence of adaptive adversary. The issue arises due to asynchronous communication channels and inability to simulate the real protocol corruption by just knowing the length of the transmitted message in a feasible computational system. The result draws a clear line of separation between two cryptographic security models and the wider formal conjunctures from earlier believe of a stronger modelling approach by simulations. Simulation based models provide nice modular security definitions for complex cryptosystems. Cryptographic games should not be ditched for the simulation based modeling technique just because of sophistication in the security definitions. One should first understand the relationship between the two.

CONTENTS

1 INTRODUCTION 1
1.1 Related work 4
2 BACKGROUND 5
2.1 Key exchange protocols 5
2.2 Security Models 8
2.2.1 Adversary in Security Model 8
2.2.2 Adaptive Corruption in a protocol 8
2.2.3 Cryptographic Games 8
2.2.4 Universal Composable Framework 9
3 SECURITY MODELS FOR KEY EXCHANGE 13
3.1 Cryptographic Games: Bellare-Rogaway Model 13
3.1.1 The Model 14
3.1.2 Mutual Authentication 14
3.2 Simulations: UC based Model 16
3.2.1 Joint State in UC 17
4 SECURITY ANALYSIS OF KE(ENC,SIG) 21
4.1 BR Secure Game 22
4.2 UC Security 26
4.2.1 $\widehat{\text{KE}}$: KE with JUC side/effect 26
5 EVALUATION 29
5.1 Main Results 29
5.2 Moving Relationship to Separation 31
6 CONCLUSION 35
6.1 Possible Future Work 35
A APPENDIX 37
A.1 Random Oracle Model 37
A.2 Probability of $\mathcal{B}$ guesses test session correctly 37

BIBLIOGRAPHY 39

LIST OF FIGURES

Figure 1 Communication between two parties in key exchange 5
Figure 2 Abstract model to run Games for security 9
Figure 3 Simulation based security model with real/random execution of π with $\mathcal{E}$ 9
Figure 4 Game execution state for protocol π 15
Figure 5 Adversarial query model for protocol π 16
Figure 6 Handling multiple session of π with/without joint state 18
Figure 7 Protocol KE(ENC,SIG) 21
Figure 8 Proof model for $\mathcal{B}$ breaks encryption if $\mathcal{A}$ breaks protocol 23
Figure 9 Execution model for UC Framework 26

LIST OF ALGORITHMS

Figure 1 $\mathcal{F}_{\mathsf{KE(ENC,SIG)}}$ 17
Figure 2 Corrupt($\mathtt{corrupted}_{\{0,1\}}$, message, adv, user, env) 19
Figure 3 Initialization(1^{λ}) 24
Figure 4 LeftoRightGame(m0, m1) 24
Figure 5 Dec(C^{*}) 24

LIST OF SYMBOLS

π real protocol . . . 10
P_i Party: Initiator . . . 13
P_j Party: Responder . . . 13
A Party: Initiator . . . 21
B Party: Responder . . . 21
C Party: Suspicious . . . 7
$\mathcal{A}$ Adversary . . . 22
$\mathcal{B}$ Adversary . . . 23
$\mathcal{C}$ Adversary . . . 23
$\mathcal{S}$ Simulator . . . 31
$\mathcal{E}$ Environment . . . 26
$\mathcal{F}$ Ideal Functionality . . . 26

ACRONYMS

ITM Interactive Turing Machine

PPT Probabilistic Polynomial Time

CDH Computational Diffie-Hellman

UC Universal Composability

JUC Universal Composability with Joint state

PKI Public key Infrastructure

KE Key Exchange

1

INTRODUCTION

Context. Cryptographic tasks protect valuable information and hence security guarantee of such task become really vital. In distributed environment like internet, an attacker can control the network to manipulate transmitted data and may also extract extra information which is not known by just reading the transmitting line. Intuitively, designers uses "design-then-break" paradigm to provide security to a system. In this approach, the system is considered to be secure till someone finds an error in the security. Then it is at risk till someone (else) can find a fix. The system enjoys it's security till the next cycle of finding loop holes in security. Alternatively, model based approach is a formal way to define security by using mathematical abstractions of the system. In this approach the system is considered as an algorithm, which is made of known security primitive. RSA problem is a good example for it, which is based on finding factorization of really large numbers rendering it hard. In a nutshell, the key exchange protocol is a cryptographic tool, which allows two or more entities to exchange common secret key over the public channel. The goal of the security is to keep this key secret from others. The security model should allow this task by considering real and practical intruders to show any weakness or variability in the protocol. Bellare and Rogaway[BR94]] showed how game based model can be used for symmetric, two party setup. In simulation-based model, security is defined by comparing protocols with adversary to similar *ideal process adversary* with a trusted third party, an ideal functionality.

Motivation. Modelling security of any cryptographic task with proveable way is not new. Still, two standard security models lack a formal relationship. Recently, Canetti proposed a simulation based model, Universal composeable framework[Can01]. This framework gaining popularity over standard way of modeling security, Cryptographic games. Basic advantage with UC is a modular approach to analyse security. Later, there are several changes and extensions were proposed to the original UC framework, as it was unable to model security guarantee for existing practical tasks. The framework is very promising while defining security to a new protocol. On the other hand cryptographic games providing good security guarantee for existing and new protocols. The dilemma of "which one is better" needs

to be investigated to get some conclusion.

Aims & Objectives. We are aiming to get implication and separation between two standard security models for key exchange pro-tocols. Forming relationship between security notions is not a trivial task. Symmetric cryptosystems like encryption, pseudorandom functions have a formal relationship for semantic security among each other but as system or corruption scheme become more complex these starts lacking formalism. To achieve the aim, tasks are as follows,

1. Model security for key exchange protocol, π in game base and simulation based framework considering an adaptive adversary and show corruption scheme for long term keys.
2. Construct ideal functionality $\mathcal{F}_{KE}$ for key exchange in Universal Composability (UC) framework.
3. Patch protocol π in UC for joint state to construct $\widetilde{\pi}$.[1]
4. Prove or disprove protocol π is game base secure but can not be shown UC secure.

We had decided to find or come up with a protocol that should be simple to analyse, allow practical corruptions and realistic conclusion about models can be obtained by modelling it. We finally proposed a new protocol KE(ENC,SIG) for exchanging keys. The new results from this work will allow research community to explore and establish relationship in other areas of secure communication and further to find a design pattern for such technique. We are focusing our work to a practical task, a protocol that is in daily practice over the internet. We deal with strong security threat, adaptive corruption for this analysis with key-exchange protocol. It is interesting to model security for such a widely used protocol under practical security threat.

The Process. We have presented a detail overview for game based model. First we took up standard definition of Bellare and Rogaway model but we needed to modify it a bit. According to original definition the attacker can test the real or random value at the end. We felt it restrictive for adaptive adversary. In our definition we allow attacker to make this query in between and continue playing the game till he reveal all but two sessions, session he wants to test and it's partner session 3.3 or

1 protocol needs to be modified for the case of long-term key as joint state in universal composition theorem.

the corruption of parties which runs these two sessions. Then we used reduction to get the final conclusion of protocols security in Games. Universal composable framework is too vast to discuss in detail in this report. We have enlighten the main attacked and key difference which are worth to explore without loosing any pace with the analysis. We provided a conjecture regarding the impossibility of proving the security of the protocol with informal explain why thus is the case, before taking final results.

Our Results.

1. Proposed key exchange protocol KE(ENC,SIG) 4 is BR Secure[BR94] but not UC Secure[Can01] in presence of adaptive adversary.

2. To improve efficiency and modularity Universal Composability with Joint state (JUC) theorem[CR03] patch needed to be added in universally composable theorem proposed by Canetti[Can01]. This modification provides security guarantee to a bit changed protocol than we analyse in first place.

3. Proving security by nesting more game is harder to formalise than proving a single game by reduction.

4. we got stronger result in games than we were hoping for. The results shows the adaptive adversary cannot guess key for any fresh session with matching conversation (we required it for only one test session, S_t).

Organisation. The paper provides initial sketch of the work in this chapter which is followed by detailed discussion about key exchange & universal composable framework in the next chapter. In Section 2.1 key exchange protocol and it's security requirements are analyzed. We provide enough examples and informally discuss security loopholes before beginning with actual security modeling. Section 2.2 tells more about security models and adversaries. We outline UC framework by [Can01] and an example of idealised functionality to provide a clear idea about this sophisticated security model for subsequent Chapter 3 where we construct the model for proposed key exchange protocol along with Games. In Section 3.2 we introduce *Joint State* and it's treatment to multiple sessions share long-term key to actively use this advance topic in key exchange. In Chapter 4 we get all our results by providing formal proof of KE(ENC,SIG) in Games and discussed insecurity in UC model due to long-term key corruption. The Chapter 5 states direct results and then critically analyses and evaluates the work we

have carried so far. This includes the KE(ENC,SIG) and both the security models. We sum up all our work in Chapter 6 and discuss few open problems on the same path. We put some interesting and optional work in Appendix A. This includes details of Random oracle model which explains basis of idealised cryptographic functions. Later, we put a weak independent result with adaptive and non-adaptive adversary found in game based model.

1.1 RELATED WORK

The closest to our work is done by Nielsen[Nie02], showing separation results for non committing encryption between complexity theory and random oracle model. The results show possibility of adaptive adversary secure protocol for a non committing encryption is either by generating arbitrarily large keys (w.r.t. size of the message) or programmable random oracle. Our work uses this result but key exchange is way different form public key encryption scheme. Work done by Canetti & Krawczyk[CK02] shows a relation between cryptographic games (SK security) section **??** and simulation based model (UC framework) for key exchange protocol composed with secure channel but the work is done for non-adaptive adversary. They have shown a strong relation between SK secure game and relaxed UC secure model. Which is opposite to our result of separation with adaptive adversary. The negative results do not contradict their work but it stops to generalise the relationship further.

2

BACKGROUND

2.1 KEY EXCHANGE PROTOCOLS

Key exchange is the process by which parties, called as entities, distribute keys to use it later for other cryptographic tasks like building secure channel for message transmission. One of the security challenges is to partner with correct entity as they can be remotely distributed over network like internet where the communication channel is not secure. The second problem is to maintain the secrecy of the key to be exchanged and it should be transmitted from an initiator to a receiver correctly & untempered over the distributed network. In this protocol, parties generate a short-term key, termed as Session Key, derived from a secret key for the exchange. This secret key is called as long-term key.

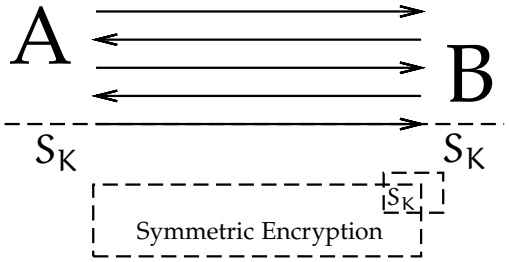

Figure 1: Communication between two parties in key exchange

The above approach improves security of the protocol, as only session keys are transmitted over the network and on compromise it can be freshly regenerated. But this improvement adds complexity in the protocol and hence scope of security problems also increases. By analyzing session keys one should not guess the long-term key and also if the long-term keys of small set of parties were exposed, it should not effect the overall security of the protocol. Moreover, the security definition should capture real world characteristics of current problems and efficiently usable for practice. There are different ways to exchange keys, two or more parties may distribute keys in a protocol. Even the distributed keys may be asymmetric or parties may agree on a shared symmetric key. In this paper, we consider two parties that want to exchange a symmetric secret key and the protocol is in a Public key Infrastructure (PKI) settings i.e.; if any cryptographic algorithm uses asymmetric keys

in the protocol, public key of entities are verified and bonded correctly and we do not need to provide security guarantee for it along with the protocol.

Definition 2.1 *Key Exchange protocol is a key transport mechanism to distribute keys among parties. This distribution may or may not include pre-established long live key. All parties may or may not contribute in key agreement.*

Communication in KE. A protocol run consists of communication flows between parties before agreed on a final key. Each flow is discrete in time and upon finish of one flow successfully, next stage starts except the final stage where protocol terminates. A complete one round of communication for a party is called as **Round** and maximum rounds for any of the party (A or B) is defines the rounds of given protocol. For any stage, party that send last message is called as **Owner** of the protocol till protocol moves to the next stage or it ends. Protocol is said to be finished successfully when the last message reaches to the other party. Parties which initiate the communication are termed as an **Initiator** and parties which receive first message sent by initiator are termed as **responder**. Any party can be an initiator, a responder or both. One set of communication is called as a **session**, which party runs locally as their part of the protocol . Parties can run multiple sessions to same or different party. Even a party can run the protocol to itself being an initiator and responder of that session.

KE in action. We present examples based on symmetric key and Diffie-Hellman key exchanged protocol. We also enquire security concerns related to these protocols. Protocols rely on some mathematical assumption made by the primitive cryptographic algorithm. Hardness of Computational Diffie-Hellman (CDH) or cipher text looks like random output (for encryption) is the basis of following examples,

1. One pass key transport[MvOV96]:A point-to-point key update process, depends on symmetric encryption for secure key transport.
$$A \rightarrow B : E_{key}(R_A) \qquad (1)$$
The protocol derives session key, S based on a priori shared long-term symmetric key K between parties A and B. To maintain freshness optional time-stamp/sequence number can be sent to B. A redundant data, B will detect any modification in data if sent along with R_A. Session key will remain same, $S = R_A$
$$A \rightarrow B : E_{key}(R_A, t_i, B) \qquad (1')$$

2. challenge-response[MvOV96]:
$$A \leftarrow B : N_B \quad (1)$$
$$A \rightarrow B : E_{key}(R_A, N_B, B^*) \quad (2)$$
Dependency on time-stamp is not reliable for freshness but one can use a monotonic random number, *nonce* sequence to provide freshness. R_A is the derived session key, $S = R_A$.
If both the parties want to add a share in derived key, a random key R_B can be used in the same setup.
$$A \leftarrow B : N_B \quad (1)$$
$$A \rightarrow B : E_{key}(R_A, N_A, N_B, B) \quad (2)$$
$$A \leftarrow B : E_{key}(R_B, N_B, N_A, A) \quad (3)$$
This 3-step protocol provides authenticated key exchanged based on symmetric encryption scheme E. session key S $= (R_A, R_B)$

3. Diffie-Hellman key exchange[MvOV96]:
one time distribution of an appropriate prime q & a generator g of $\mathbb{Z}(2 \leqslant g \leqslant q-2)$
$$A \rightarrow B : g^a \bmod q \quad (\ldots cert_A) \quad (1)$$
$$A \leftarrow B : g^b \bmod q \quad (\ldots cert_B) \quad (2)$$
$$1 \leqslant a, b \leqslant q-2$$
Both A & B computes session key $S = g^{a.b} \bmod q$
Protocol's security relies on CDH assumption and exponentiation provides a trapdore[1]. It removes the dependency to pre-establish long term keys. The problem is with authentication in this key exchange protocol.
TEST. Let's assume C controls the network and able to modify the message. Consider C as C_B when it impersonate B to A and the same for B with C_B and C chooses c as $1 \leqslant c \leqslant q-2$
$$A \rightarrow C_B : g^a \bmod q \quad (1)$$
$$C_A \rightarrow B : g^c \bmod q \quad (1')$$
$$C_A \leftarrow B : g^b \bmod q \quad (2)$$
$$A \leftarrow C_B : g^c \bmod q \quad (2')$$
After protocol execution, both parties A and B will have key exchanged with C, $S_{AC} = g^{a.c} \bmod q$ & $S_{BC} = g^{b.c} \bmod q$ respectively. C can maintain this setup to get all transmitted data between A & B if S_{AC} & S_{BC} been used for later cryptographic tasks.

These security tests can be written as mathematical algorithms. The test relies on previously defined secure primitive and if the protocol does not leak any extra information during it's execu-

1 The CDH assumption is based on discreet logarithmic problem, it's hard to mathematically calculate logarithm of value base g (generator).

tion then it should remain secure as well. Last example vaguely explains power of external entity C, which impersonates the actual user and breaks the security of protocol. C does not brake any rules of the protocol hence we can say that the protocol is not secure if someone other than A and B can alter the channel of communication. Now the protocol must be updated to allow such kind of corruption if communication channel remains public & controllable by C and parties A & B still wants to do secure key exchange.

2.2 SECURITY MODELS

2.2.1 *Adversary in Security Model*

Now we define an adversary and classify it based on power given to it in a security model. *Adversary* is a polynomial time running machine with a goal to break the security of given cryptographic task. The system defines its power. When an eavesdropper can listen to the communication between parties and try to gets information based on it, we call it *passive* adversary. A more realistic adversary get control of the network and decides whether to read, alter, block or inject the on-going communication is *active* adversary. Another type of classification is depending on when to corrupt the parties to get the secret information. A *static* adversary can corrupt parties in the beginning of the protocol run while *adaptive* adversary can corrupt parties on the fly.

2.2.2 *Adaptive Corruption in a protocol*

An ***Adaptive Adversary*** is a type of adversary who can corrupt parties on the fly. This gives the adversary power to analyse previous communication if he can eavesdrops the communication for some extra information which is not known in general. By using this s̈ome extraïnformation it can choose the next party to corrupt for the final task breaking the protocol itself.

2.2.3 *Cryptographic Games*

Definition 2.2 *Cryptographic Games is a PPM ITM [2] which accepts query q from a finite set of predefined queries Q and return responses to the output tape. In between the game progress, a special query Test*

2 PPM ITM is deterministic turing machine with an added random tape which interacts with I/O tapes on-line.

can be fired to result the game play. The game overs when player returns back with the guess query. The event of occurrence of guess query strictly later than the test query.

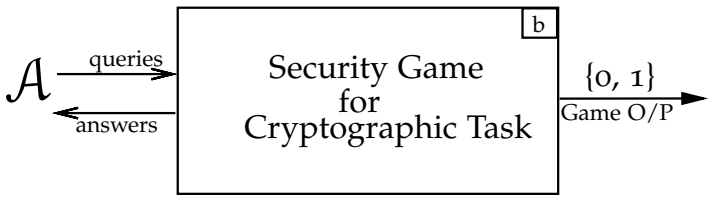

Figure 2: Abstract model to run Games for security

Henceforth we call cryptographic games as *Game* in general. It interacts with the adversary, $\mathcal{A}$ as a player to proceed with the game, eventually it allows $\mathcal{A}$ to guess the internal *game bit* b and $\mathcal{A}$ win/lose the game on his guess.

2.2.4 *Universal Composable Framework*

Simulations. Simulation based security model is based on real/ideal model and any external adversarial entity, environment try to distinguish between these two execution. The system is considered to be secure if environment cannot tell the difference by looking at global output of these two models.

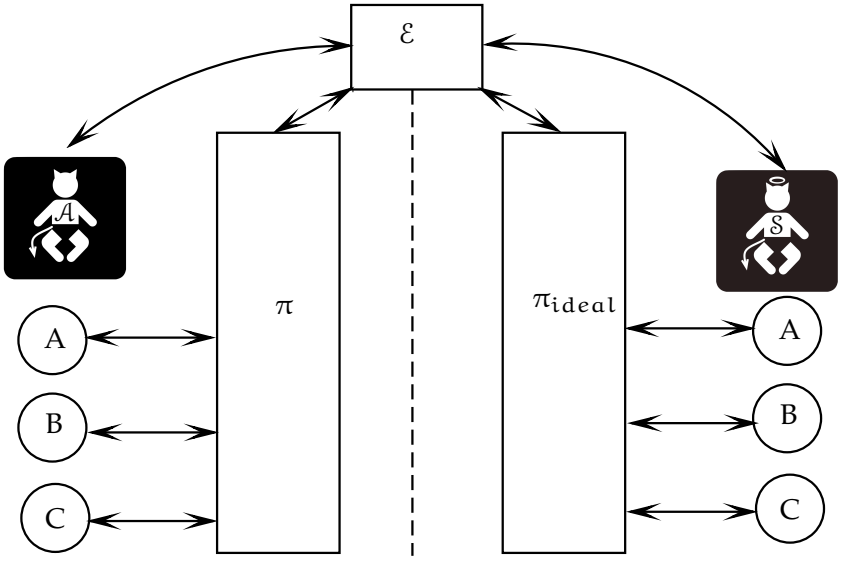

Figure 3: Simulation based security model with real/random execution of π with $\mathcal{E}$

Universal Composition framework is a simulation based security framework. It addresses security issues of a cryptographic protocol by making no assumption of other protocols in the

system. Those can be independent or concurrent executions of same or different protocol instances in that system. Even it allows protocol to compose with other unknown/dynamic components in the system. These components can be unknown at the time of analysis. Informally, in UC framework[Can01], real adversary $\mathcal{A}$ is emulated in ideal world by a simulator $\mathcal{S}$ and parties $\mathcal{P}_\jmath$ interact according to the protocol by giving their input and taking output without any external interaction with the trusted third party, idealised functionality. This is mimic by an algorithm in ideal world, called as *ideal functionality* $\mathcal{F}$ in context of UC. Here i is identifier of each party for particular execution of the protocol instance. All operations are held in the presence of an entity termed as *environment*, $\mathcal{E}$ which acts as communicator. $\mathcal{A}$ controls the communication channel and corrupt real $\mathcal{P}_\jmath$. $\mathcal{E}$ gives inputs to all parties, reads back result and communicates with $\mathcal{S}$. The protocol is considered to be secure if $\mathcal{A}$ interacts with real parties and $\mathcal{S}$ interacts with $\mathcal{F}$ in this above setup such that the $\mathcal{E}$ can not distinguish between the two executions interactively.

Definition 2.3 *[Can01]* Universal Composable Framework *consists of* $(\{P_i\}_{\{1,..,n\}}, \mathcal{A}, \{\mathcal{F}_{id}\}_{\{1,..,m\}}, \mathcal{S}, \mathcal{E})$ *over three main components real-model, ideal process and any outside world adversarial entity, to define security guarantee of any protocol* π *in any arbitrary system.*

The tuples of framework are defined as follows,

- $\{P_i\}_{\{1,..,n\}}$, where $n \in N$ of ITMs may run as parties of a protocol in real-model. n is predefined and N is a finite number in polynomial size of security parameter.
- $\mathcal{A}$ is an adversarial ITM who tries to break the protocol in real-model.
- $\{\mathcal{F}_{id}\}_{\{1,..,m\}}$ is instances of ideal functionality of the given protocol π which is conceptually incorruptible & $m \in M$. M is a finite number in polynomial size of security parameter. Informally, ideal functionality is a trusted third party who runs the security primitive and interacts with parties via some secure channel.
- $\mathcal{S}$ is an ITM, which tries to mimic the $\mathcal{A}$ by interacting with $\mathcal{F}$ to get information as per security definition.
- $\mathcal{E}$ is an adversarial decision making ITM which tries to "distinguish interactively" the real execution from ideal process. $\mathcal{E}$ interacts with all party and adversaries. In ideal world, due to secure channel between parties and trusted

third party, its acts as a dummy party which bypass messages between $\mathcal{E}$ and $\mathcal{F}$. The framework also imposes online interaction only for adversaries to $\mathcal{E}$ to stop rewinding of I/O tapes of TM.

Theorem 2.4 *[Can01] A protocol π realizes a functionality f securely if for any real-life adversary $\mathcal{A}$ there exist an ideal process simulator $\mathcal{S}$ such that the execution of real-life π with $\mathcal{A}$ is indistinguishable with running of the idealised model with simulator $\mathcal{S}$ by seeing the global output of these two models.*

The definition here is informal for clarity and brevity, mostly referred from the Overview section of Canetti paper[c01]. See there for full details.
Intuitively, this looks a strong security definition as it does not care about the system and other concurrent task running with the protocol . The notion of the security somewhat resembles with the definition of Session-Key(SK) security **??**. In SK-security two games UM (unauthenticated model) and AM (authenticated model) used to provide security by indistinguishability. Original protocol runs in AM and a protocol emulates the original in UM. To test "difference by a outside observer an authenticator provides description of emulated protocol. But definition provided by UC is strictly stronger than SK security as it does not put any condition on the distinguisher. The formal definition of SK-security and relationship with UC is defined in [CK02].

3

SECURITY MODELS FOR KEY EXCHANGE

Considering a two party protocol, we can fix an integer n of size polynomial in the security parameter, k. The parties are identified by some integer *i*, with $0 \leqslant i \leqslant n$. Let parties run an algorithm, ξ locally to compute session key as output There is no guaranty of identifying sessions uniquely. Parties list down the local session identifiers *lsid* $\in \mathbb{Z} \times \mathbb{Z} \times \mathbb{Z} \times \mathbb{B}$ is defined. *lsid* = (i,j,k,b) refers k^{th} session between parties i and j and if binary value b is set j is the session initiator else i initiates the session. In this two party protocol what we want is for the involved parties exclusively compute a session key and it should be independent of other similar communication.

Definition 3.1 *[BFWW11] Key exchange protocol is a pair of cryptographic algorithms ($\mathcal{K}g$, ξ) where $\mathcal{K}g$ is a randomization algorithm which generates keys in some key space based on input security parameter k. ξ is the algorithm which parties run locally as their part in the protocol execution.*

3.1 CRYPTOGRAPHIC GAMES: BELLARE-ROGAWAY MODEL

Definition 3.2 *[BR94] Two parties P_i & P_j are said to be in **Matching Conversation** if,*

- *Initiate one session each locally.*
- *Asynchronously send /receive message in order to agreed protocol in that session.*
- *Last message reaches successfully to the other party.*

Where $i, j \in \{1, n\}$.

Definition 3.3 *Two instance A & B of the protocol π is **Partner Sessions** if,*

- *A & B are in* Matching Conversation *during session instances $\pi(A,B,b)$ & $\pi(C,D,b')$ respectively.*
- *Instances $\pi(A,B,b)$ & $\pi(C,D,b')$ computes same session key, K as output.*
- *K is independently generated with respect to similar communication with other instances. Where A = D, B = C & $b' = \overline{b} = 1 - b$.*

3.1.1 *The Model*

Now we model the protocol based on the technique proposed by Bellare and Rogaway (BR Model).[BR94] Our model fulfils the demands of their notion as session can be created on the fly like real scenario and two local parties exchange session key using matching conversation in the protocol .

To *SETUP* the game for adversary, first party states are allocated and initialized. All parities p_i to p_n are initialized as honest parties by unset δ^* bit of their state. Asymmetric keys for encryption and signing and verification key for signature are generated by Kg(η) for each party. An empty set to contain lsid is also associated with all the parties . The predicate bit b is set to 0. To validate queries before execute, an algorithm *Valid* checks validity of arguments. In this setup we take query request from $\mathcal{A}$ in serial manner. The *Game* will accept queries from input tape and to maintain atomic progress, a tri-state variable, e $\in$ {accept, running, reject} tracks game running state.

3.1.2 *Mutual Authentication*

Definition 3.4 *π is a secure key exchange protocol with any polynomial time adversary $\mathcal{A}$,*

1. *if oracle $\pi^s_{A,B}$ & $\pi^t_{B,A}$ and having matching conversation and only both of these accept same session key.*[1]
2. *There is negligible probability of acceptance of a non-matching conversation by both the parties .*
3. *There is negligible probability gaining extra information about session key. e.i; there exists a negligible function negl(k) in the security parameter k such that*

$$Adv^{\pi}_{E} \leqslant negl(k)$$

To model the above protocol in the *Game*, we need to define extra state information which may not be a part of the protocol but required to run the game deterministically. Game state consists,

1. A predicate bit, b to store response to Test query.
2. A tri-state variable, e to tell game running state.
3. A set of identifier for parties which can be initiator or responder.

1 $\pi^s_{A,B,b}$: party A runs a session s with party B.

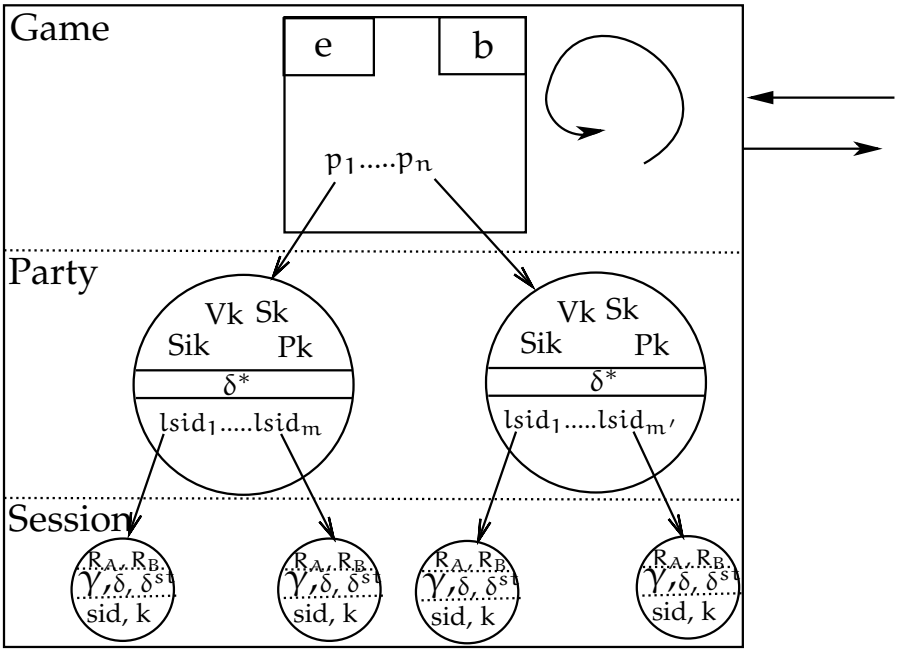

Figure 4: Game execution state for protocol π

4. Collections of Party's state mapped with each of the above identifier.

5. Party state consists long term keys for encryption and signature, a bit δ^* to identify if these keys are corrupted or not and a dynamic set of lsid.

6. Collection of session's state mapped with each of above lsid. Each element contains session id, sid and two bit, δ & δ^* to keep track of "Reveal query" for this session & partner session respectively. While $\gamma \in$ {uninitialized, negotiation, derived} is to check if session is not begun yet or still under negotiation or completed. Any random nonce generated for the session or partner's nonce to be use for generating signature is also stored in this state.

7. An additional collection to map sid to lsid for fast lookup of local session state.

The arrow in above figure denotes input/ output tapes with arrow direction to the game-box. The internal arc arrow denotes random tape.

To identify party, instead of using P_i, we will use alphabets like A,B,C for better understanding. The state associated with a party is it's long-term keys which is set at the beginning of the game run. Any K^{th} session run by a party is identified by lsid of that session.
Here is the full list of queries, Q that $\mathcal{A}$ can fire to the game.

1. New(A, B, i): Ask to start a new session for party A and B where A is initiator if i=0 else A is responder.

2. Send(sid, M, U): Sends a message M to party U in session sid.

3. Reveal(sid): Reveals the session key if it finishes the derivation for both parties for any session with identifier sid.

4. Corrupt(U): Corrupt a part U by giving secret key pair of ENC and SIG.

5. Test(S_T): Test for winning of the adversary with two uncorrupted parties involved in the derived session key of session S_T.

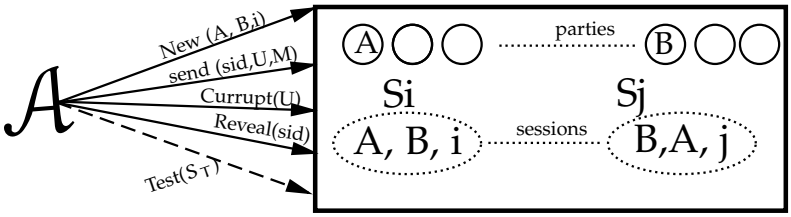

Figure 5: Adversarial query model for protocol π

Winning of adversary is decided by key indistinguishability. The game allows it via *Test* query and result is compared by *Guess* query, as follows

1. $\mathcal{A}$ issues a *Test* query for a session, S_T.

2. The S_T or its partner session, defined through matching conversation is not corrupted by the adversary.

3. The parties running these sessions are also not corrupted ie; long-term keys are not revealed for both the parties.

4. A fair coin is flipped and it is assigned to b. depending on b, it sends response as session key or random key.

5. The adversary fires *Guess* query with bit b'. If $b' = b$, $\mathcal{A}$ win else $\mathcal{A}$ loss.

3.2 SIMULATIONS: UC BASED MODEL

Real protocol execution progress same as per the protocol rules. The real-world adversary $\mathcal{A}$ interacts with the execution looking for any security problems. To model **Ideal world execution** we need to define *Ideal Functionality* of KE(ENC,SIG) first. Let parties $P_1, ..., P_n$ and idealised process adversary $\mathcal{S}$ running on security parameter k.

Algorithm 1 $\mathcal{F}_{\text{KE(ENC,SIG)}}$

[CK02]

Wait to receive a value (exchange,sid,P_i, P_j, b) from some party P_i
Wait to receive a value (exchange,sid,P_j, P_i, b') from party P_j.
if $b = b' = \perp$ **then**
  choose $K \xleftarrow{R} \{0,1\}'$
else
  if $b \neq \perp$ **then**
    set k = b;
  elsek = b'
    Send (key,id,k) to P_i
    Send (key,id,k) to P_j
    Notify $\mathcal{S}$ **return**

3.2.1 *Joint State in UC*

Canetti & Rabin [CR03], extended the scope of [Can01], as the proposed framework is not suitable for existing protocols. Different Protocols may share common security primitive such as common secret key within a system and can not be analyzed efficiently in isolation. The extension allows composition of protocols if they share some amount of common secret or in generality *some state* called as JUC(*composition operation with joint state*). The new proposed composition operation, JUC claims to preserve security irrespective of whether different components have some amount of joint state and joint randomness. It could be useful in practice as for example, some protocols use Public Key Infrastructure. JUC can provide security guarantee in such scenarios. Informally, system consists of π (high level protocol) and ρ (sub-protocol) as multiple instances with some joint state. JUC theorem states that, we can construct a protocol $\hat{\rho}$, which functionally behaves same as multiple copies of ρ. To provide security framework for such protocols, π and ρ runs same as in UC then replaces all instances of ρ with single instance of $\hat{\rho}$. Now we can execute the protocol π with $\hat{\rho}$ in environment $\mathcal{E}$ as of multiple ρ for JUC operation. The *ideal functionality* is modified to do joint state operation and stated as $\mathcal{F}$*-hybrid*. This is by adding an additional sub-session identifier, ssid to the $\mathcal{F}_{\text{sid}}$ to make it $\mathcal{F}_{\{\text{sid,ssid}\}}$. While the newly added ssid is locally unique within a party. This construction improves efficiency significantly as only single copy of $\mathcal{F}$ with joint state will replace multiple instances of previous $\mathcal{F}$. For example multi-session protocol no longer needs to get exclusive copy of ideal functionality for each session of a party, these can share a single $\mathcal{F}$. JUC

gives security definition in $\mathcal{F}$-hybrid maodel which is nothing but real model and parties has access to idealised functionality, $\mathcal{F}$ same as dummy parties. *Long-term key corruption*. Corruption

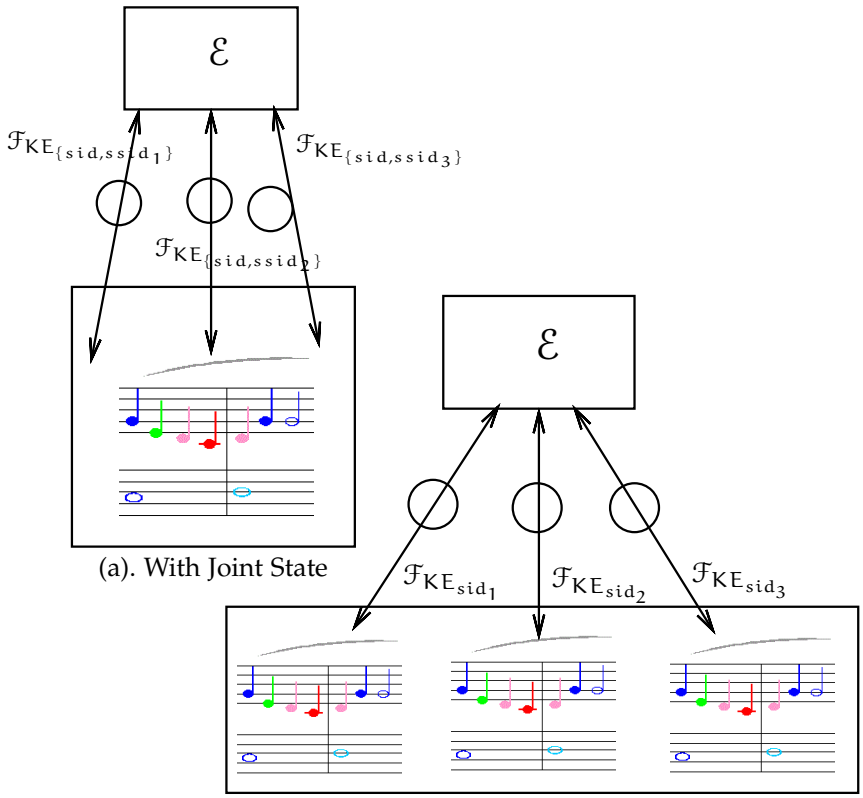

Figure 6: Handling multiple session of π with/without joint state

of idealised functionality is not very precisely specified in the UC model. We are using a macro which can plug to the ideal functionality to provide long-term key corruption for adaptive adversary. This approach is similar to the work of [KTo8]. In their work a generalised macro is written for non-adaptive and adaptive adversary. While we have specifically done it for adaptive adversary.

long-term corruption scheme for KE(ENC,SIG)

- $\mathcal{E}$ can check for the corruption state.
- $\mathcal{A}$ can corrupt an uncorrupted party. On corruption, all further communication with other parties will be send to $\mathcal{A}$
- Other parties can communicate to the corrupted party in the protocol.

Algorithm 2 Corrupt($corrupted_{\{0,1\}}$, message, adv, user, env)

Require: initialised
Ensure: len = 0
Ensure: $List_{user} \neq \phi$
  if recv(IsCorrupted) from env **then**
    send(corrupted) to env
  if recv(corrupt) from adv and ¬corrupted **then**
    corrupted ← true
    send(corrupted, message) to adv
  if corrupted and recv(send, msg, user) from adv **then**
    assert user ∈ $List_{user}$
    assert $0 < |msg| \leqslant len$
    result ← 0
    send msg to user
  if corrupted and recv(msg) from user **then**
    assert user ∈ $List_{user}$
    result ← 0
    send(Recv, msg, user)
  if corrupted and recv(Recv, r) from env **then**
    send(Recv, r) to adv
    len ← |r|

4

SECURITY ANALYSIS OF KE(ENC,SIG)

KE(ENC,SIG) is a secure key exchange protocol with parties p_i where $0 < i \leqslant n$. To provide secure communication it uses security primitive such as Encryption & Signature. Parties locally run ξ to evaluate session key. It uses public-key signature, SIG(1^*,Sik,Vk) and asymmetric encryption, ENC(1^*,Sk,Pk) to provide authentication and secrecy respectively but we need to show the overall protocol is also secure if and only if the security primitive is secure enough[1]. Key pairs (Sk_i, Pk_i), (Sik_i, Vk_i) are used as encryption keys and signature keys respectively for any party p_i and generated by the key generation algorithm. *Certificate Authority* is a trusted third party which

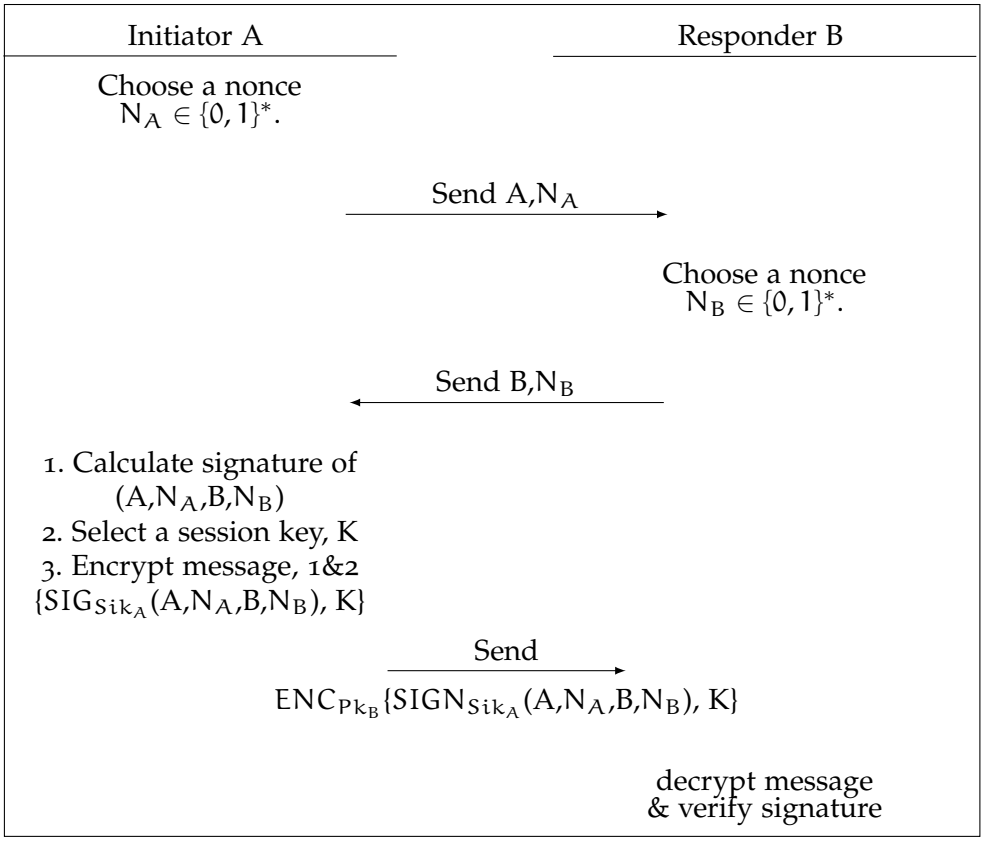

Figure 7: Protocol KE(ENC,SIG)

generates $cert_i$ for each party, P_i. This provides a mechanism of binding long-term state information to any party. Basically, it prevents two honest user to acquire same P_i and prevent adversary to fake any party's identification. For example, internet

1 Pr[breaking of algorithm] $\leqslant$ f(n) + negl. Where f(n) is fair chance of predicate to be unsecured, possibly zero in most cases

uses email id and other physical details of registered domain to provide SSL certificate. The enforcement and binding is not part of our proof so we do not consider these for our security model.

Definition 4.1 *Protocol KE(ENC,SIG) is a key exchange protocol if it satisfies following conditions:*

- *Kg output quadruple of keys in pair (Sk, Pk) & (Sik, Vk) for all parties* p_i *where* $o < i \leqslant n$.
- *Any instance of* ξ *outputs n-bit string, session key on successfully finishing the protocol steps. Otherwise a special fail symbol,* $\perp$ *is output by the instance.*
- *Two instance of protocol* π_1 *&* π_2 *are* Partner Session *and produces keys S1 & S2 locally such as;*

$$S1 = \xi(1^n, P_1, Sik_1, P_2, Pk_2, k, b)$$
$$S2 = \xi(1^n, P_2, Sk_2, P_1, Vk_1, k', \overline{b})$$
$$S1 = S2$$

Lets assume parties run algorithm π in protocol KE(ENC,SIG) and two parties A and B have keys (Sk_A, Pk_A), (Sik_A, Vk_A) & (Sk_B, Pk_B), (Sik_B, Vk_B) respectively. As a shorthand an instance can be represented as $\pi(A,B,k_i,b)$. The last instance between parties A & B is represented by $\pi(A,B,b)$.[2] At the end of execution both of them will have session key, K_i as an output.

4.1 BR SECURE GAME

Theorem 4.2 *If ENC is IND-CCA secure and SIG is UF-CMA secure then KE(ENC, SIG) is a secure key exchange protocol in the presence of an adaptive adversary,* $\mathcal{A}$.

Lemma 4.3 *If SIG is UF-CMA secure then at most one session accepts signature of message,* $\{A, N_A, B, N_B\}$ *in the protocol in presence of an adaptive adversary* $\mathcal{A}$.

The protocol π based on asymmetric encryption and public-key signature scheme. To show π is secure, we need to show that it maintains secrecy if ENC is secure by giving proof for *Theorem* 4.2. Before showing the proof, first we need to argue about *Lemma* 4.3. Even if $\mathcal{A}$ breaks the encryption, still parties A & B can do conversation authentically and $\mathcal{A}$ can only infringement the conversation during protocol steps. The protocol maintains authentication if public-key signature is UF-CMA

2 If we consider parties in two isolated set of initiator and responder, we can omit requirement of bit b, the protocol still be valid for key exchange protocol .

secure. To show proof for this, we can construct an adversary $\mathcal{C}$ which runs in time t'' using blackbox access of $\mathcal{A}$ and will able to break the public-key signature scheme, SIG of protocol π. The correctness of protocol run by $\mathcal{A}$ relies on simulation of $\mathcal{B}$ or $\mathcal{C}$. If $\mathcal{A}$ evaluates that the simulated game environment has cheated it, $\mathcal{A}$ can abort the execution. Let $\mathcal{A}$ is playing the game while encryption scheme used in protocol π is IND-CCA secure. Suppose $\mathcal{A}$ runs the game and wins in time t with an advantage more than negl.

$$\text{Advantage}[\mathcal{A}] = |\Pr[\mathcal{A} = 1|b = 0] - \Pr[\mathcal{A} = 1|b = 1]| > \mathit{negl}$$

Using $\mathcal{A}$, we can construct another adversary $\mathcal{B}$ which runs in time t' using black-box access of $\mathcal{A}$ and will able to break the encryption scheme, ENC of protocol π.

$$\begin{aligned}\Pr[\text{Break the ENC}] &= |\Pr[\mathcal{B}^{\mathcal{O}^{\pi}_{k,b=1}} = 1] - \Pr[\mathcal{B}^{\mathcal{O}^{\pi}_{k,b=0}} = 1]| \\ &+ |\Pr[\mathcal{B}^{\mathcal{O}^{\pi}_{k,b=0}} = 0] - \Pr[\mathcal{B}^{\mathcal{O}^{\pi}_{k,b=1}} = 0]| < \text{negl(k)} \ldots\text{eq(1)}\end{aligned}$$

This is a contradiction of our initial assumption that π uses IND-CCA secure encryption scheme hence $\mathcal{A}$ is not able to break the system. Informally, If $\mathcal{A}$ is able to distinguish between an encrypted message and a random message then $\mathcal{B}$ will be able to distinguish between $m_1 \& m_2$ to break encryption scheme of IND-CCA game. And if $\mathcal{A}$ can change session key in last message to a responder then $\mathcal{C}$ can break the signature scheme. To show such contradiction we will prove it by reduction as follows, to proceed to game, $\mathcal{B}$ needs to provide simulated envi-

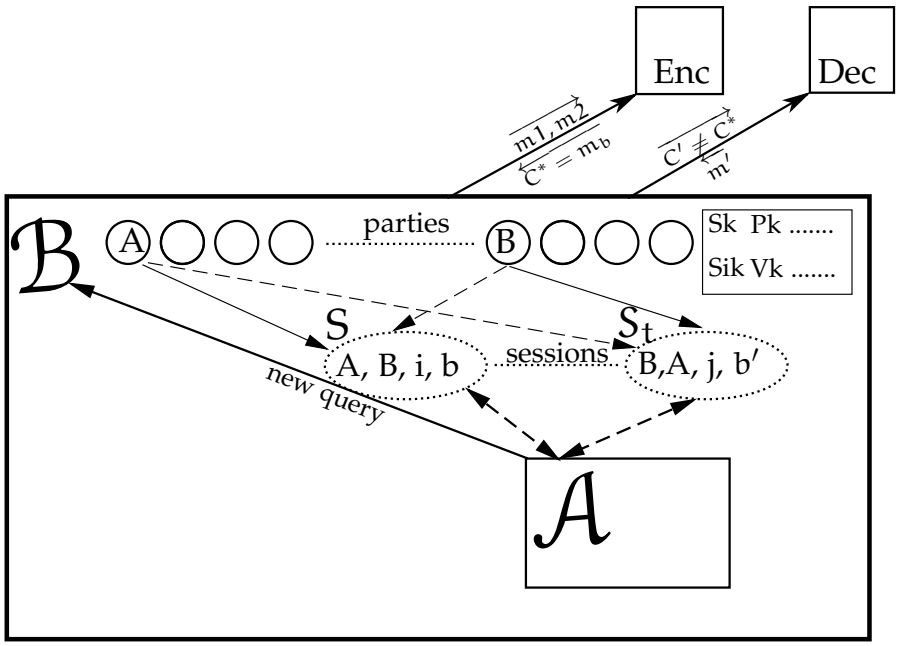

Figure 8: Proof model for $\mathcal{B}$ breaks encryption if $\mathcal{A}$ breaks protocol

ronment to $\mathcal{A}$. As a setup $\mathcal{A}$ needs public keys of all parties . $\mathcal{B}$

can guess on which session $\mathcal{A}$ can run *Test* query and involve parties A(initiator), B(responder) in polynomial time. If $\mathcal{B}$ can't guess the test session correctly it will simply output a random choice by flipping a coin. For every party $\mathcal{B}$ chooses random encryption keys and signing key except responder B's public key, which it will provide from the experiment of IND-CCA game. To run this game, $\mathcal{B}$ also need to track protocol progress (keep transcript). To do so it maintains several maps to store nonce and session key it generated using keyed with lsid and a map of lsid keyed with sid for fast lookup. We are assuming that query input is not needed to validate here and at any point if $\mathcal{B}$ receives an invalid sid it aborts with a random guess for IND-CCA game. GAME G1 for $\mathcal{B}$ to simulate for $\mathcal{A}$'s game. **Oracle**

Algorithm 3 Initialization(1^{λ})

Require: vector $\leftarrow$ [ ]
Require: map $\leftarrow$ [ ]
for i $\leftarrow$ 0 to n **do**
$((pk_i, sk_i), (sik_i, vk_i)) \leftarrow Kg$
add (pk_i, sk_i) to map as key-value pair
add (vk_i, sik_i) to map as key-value pair **return** vector$((pk_1, vk_1), ..., (pk_n, vk_n))$result()

if $\mathcal{A}$ finishes gracefully **then return** bit b, result of game played by $\mathcal{A}$
else##[$\mathcal{A}$ aborted in between] **return** $b' \leftarrow_{\$} \{0, 1\}$

procedure for IND-CCA Game to $\mathcal{B}$

Algorithm 4 LeftoRightGame(m0, m1)

C $\leftarrow$ ENC(m_b,pk)
output C

Algorithm 5 Dec(C^*)

if then$C^* \neq C$
$m' \leftarrow$ Dec(C^*,sk) **return** m'
elsereturn $\perp$

Queries response of $\mathcal{B}$ **to** $\mathcal{A}$

- *New*(A, B): $\mathcal{B}$ notes down by adding an entry in directory that it started a session sid, lsid(A,B,k^{th},b) and responds with sid.
- *Send*(sid, message): $\mathcal{B}$ will lookup in the map for it's own or partner's nonce for validity and respond with nonce

and identity for first two stages. It can also produce encryption and signature as it has all the keys to generate message for final stage except the session it guesses as *Test* session. For final stage of initiator, $\mathcal{B}$ generates a random session key, K_{lsid}. Compute the signature and encrypt these two with public key of responder and send it to $\mathcal{A}$.

- *Reveal*(sid): $\mathcal{B}$ will give away the session key, K_{lsid} to $\mathcal{A}$.
- *Corrupt*(sid): Except for party other than B (essentially involve in *Test* query), $\mathcal{B}$ will give away the encryption & signature keys.

Let $\mathcal{B}$ is a t-time adversary who plays the **IND-CCA** game.

$$Adv^{IND-CCA}(\mathcal{B}) = Pr[\mathcal{G}^{\mathcal{B}}_{IND-CCA} = 1] \leqslant \frac{1}{2} + \varepsilon$$

$Pr[\mathcal{G}^{\mathcal{B}}_{IND-CCA} = 1]$ depends on following conditions.

- $\mathcal{A}$ distinguishes between real and random keys.
- $\mathcal{A}$ generates signature by itself.

Condition 1: $\mathcal{A}$ distinguishes between real and random keys
Condition 2: $\mathcal{A}$ generates signature by itself

Proof of Theorem 4.2. We have equation(1) as,

$$\begin{aligned} Pr[\text{Break the ENC}] = {} & |Pr[\mathcal{B}^{\mathcal{O}^{\pi}_{k,b=1}} = 1] - Pr[\mathcal{B}^{\mathcal{O}^{\pi}_{k,b=0}} = 1]| \\ & + |Pr[\mathcal{B}^{\mathcal{O}^{\pi}_{k,b=0}} = 0] - Pr[\mathcal{B}^{\mathcal{O}^{\pi}_{k,b=1}} = 0]| \text{ ...from eq(1)} \end{aligned}$$

The first term & third of RHS is same as *condition 1* stated above. If $\mathcal{A}$ can distinguish real/random then $\mathcal{B}$ will output correct result.
The second term and fourth term of RHS (absolute values) is the probability that third message is send and received not by parties (by the adversary), as it breaks the protocol rule. Let total C_S session are used by $\mathcal{A}$ to win the game and out of these session $\mathcal{A}$ uses a session S_T as test session, which is *condition 2*. From *Lemma 4.3*, we can substitute these two condition in equation (1) as follows,

$$\begin{aligned} Pr[\text{Break the ENC}] = {} & |Pr[\mathcal{B}^{\mathcal{O}^{\pi}_{k,b=1}} = 1] + Pr[\mathcal{B}^{\mathcal{O}^{\pi}_{k,b=0}} = 0]| \\ & \text{- } |Pr[\mathcal{B}^{\mathcal{O}^{\pi}_{k,b=0}} = 1] + Pr[\mathcal{B}^{\mathcal{O}^{\pi}_{k,b=1}} = 0]| \\ Pr[\text{Break the ENC}] = {} & Pr[\mathcal{A} \text{ Win}] \text{ - } \frac{1}{C_N} Pr[\text{Forge the SIG}] \end{aligned}$$

If winning of $\mathcal{A}$ is non-negligible then breaking of ENC is also non-negligible.[3].
contradicting(1). Hence **Pr[$\mathcal{A}$ Win] < negl(k)** □
Also, by rearranging the terms,

3 negligible functions are associative

$$\Pr[\mathcal{A}\ \text{Win}] = \Pr[\text{Break the ENC}] + \frac{1}{C_N}\Pr[\text{Forge the SIG}]$$

Where C_N is no of session initiate by the adversary.

4.2 UC SECURITY

4.2.1 $\widehat{\text{KE}}$: *KE with JUC side/effect*

$\widehat{\text{KE}}$ is the joint state modification of KE(ENC,SIG) To overcome the side-effect of joint state, KE(ENC,SIG) needs to be changed to $\text{KE}(\widehat{\text{ENC}}, \text{SIG})$, we can do it by including process id of ITM which runs the current session of protocol. We can do this for parties A and B by

$$\begin{aligned}&\text{ENC}_{\text{Pk}_i}\{\text{SIGN}_{\text{Sik}_A}(\text{A},\text{N}_\text{A},\text{B},\text{N}_\text{B}),\ \text{K}\} \Rightarrow \\ &\text{ENC}_{\text{Pk}_i}\{\text{SIGN}_{\text{Sik}_A}(\text{A},\text{N}_\text{A},\text{B},\text{N}_\text{B},\text{pid}),\ \text{K},\text{pid}\}\end{aligned}$$

The first one uses process id, pid in encryption only which is quite sufficient to change the overall output of KE. Other wise we can use the pid in both signature and encryption.

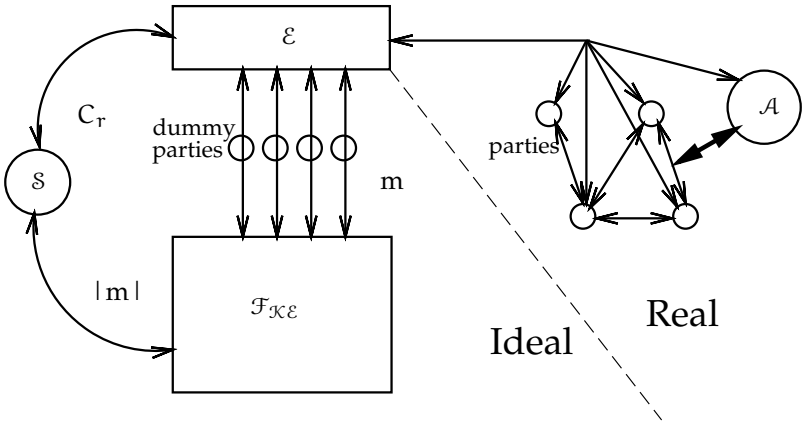

Figure 9: Execution model for UC Framework

Theorem 4.4 *If KE(ENC,SIG) is secure in Games it is also UC secure.*

Disproof by long term corruption.
Sketch[Nieo2]. Consider two parties P_i and P_j acting as receiver and initiator respectively. Consider the environment $\mathcal{E}$ which activates P_j with an arbitrary session key sk, assume total length of encrypted message is $l_m(k)$, where $l_m(k)$ is some polynomial. Consider the adversary $\mathcal{A}$ does not corrupts any party but just waits for P_i and P_j to finish the first two stage and for P_j to send a message c at final stage of the protocol to P_i. The adversary outputs c to the environment and then corrupts P_i before c arrives to party. The adversary outputs to the

environment the value of private keys (Sik_i, Pk_i) of P_i by this information gain. As message with sk is committed to the private key of P_i (ENC is semantically secure). If the environment runs the code of P_i from private keys and with input c to verify, then c will decrypt to earlier message created with sk with overwhelming probability. We assumed KE(ENC,SIG) is UC Secure so by the definition of UC secure protocol, there should exist a simulator $\mathcal{S}$ such that $\mathcal{S}$ simulates $\mathcal{A}$ in idealised world with trusted third party $\mathcal{F}$ in presence of the same environment $\mathcal{E}$ which interact with $\mathcal{A}$ earlier for the long-term key corruption. If $\mathcal{E}$ detects an output distinguishable from these two adversary. But in the ideal-world abstraction of secure communication given by $F_{KE(ENC,SIG)}$, the simulator, $\mathcal{S}$ does only know length of message with sk during the execution as long as both parties are uncorrupted. The simulator must therefore generate c on its own by just knowing the length. On the corruption of P_i, the simulator knows the message and calculates Pk_i to give to the environment. $\mathcal{E}$ running P_i with key Pik_i will open earlier message given c with non-negligible probability. Nielson shows that it is more than $frac12$.
Since ENC is a trapdoor permutation, c will decrypt to only one value except than negligible probability. Hence we can construct an injective map such that,

$$\begin{aligned} m_1, Pk_{i_1} &= c \\ m_2, Pk_{i_2} &= c \\ m_3, Pk_{i_3} &= c \\ &\ldots\ldots\ldots\ldots \\ m_n, Pk_{i_n} &= c \end{aligned}$$

Intuitively this means that the length of Pk_i must be at least l(m), length of the message itself. P_i is a Probabilistic Polynomial Time (PPT) Interactive Turing Machine (ITM) and it can send message in polynomial length of security parameter k during the entire execution. For a polynomial size l(m), length of Pk_i must be super-polynomial. Which contradicts earlier assumption of P_i to be a PPT ITM. Hence $\mathcal{E}$ can distinguish the random output for the final step produced by $\mathcal{S}$ from actual one. That proves KE(ENC,SIG) is not UC secure.

5 EVALUATION

5.1 MAIN RESULTS

1. UC framework fails to provide security guaranty of the protocol Key Exchange (KE) due to it's requirement of freshly generated long term key (shared state) for different session in presence of adaptive adversary. Even we assumed a global session identifier for UC which is not a practical case for real protocol[1] like SSL/TLS.

2. We needed to add JUC theorem patch to universally composable theorem proposed by Canetti[Cano1] to address long term key sharing with multiple session of the party. This modification changes initial protocol a bit, which is used as a single copy.

3. We found that cryptographic games is simple to construct but if reduction needs nested games, it is hard to formalize. To overcome this issue we calculated conditional probability than separate reduction of Signature scheme and mix two games later.

4. The above simplification does not affect security guaranty though and KE is proven secure in cryptographic games. We got even a stronger result which we were hoping for, the results shows the adaptive adversary cannot guess key for any fresh session with matching conversation (we required it for only one test session, S_t).

Theorem 5.1 *[$Game^{\pi}_{secure} \not\Rightarrow UC^{\pi'}_{secure}$]: If a protocol π is secure in game based model than there exist a protocol π' which is secure in UC model for non-adaptive adversary but which is not secure in presence of adaptive adversary.*

Remarks on Protocol KE(ENC,SIG).

- *KE(ENC,SIG) is not Efficient!*: Our protocol uses ENC and SIG for maintaining secrecy and authenticity. Overall KE is not very efficient as it is using both public key encryption and public key signature. What we want for our analysis is a simple and easy to analyse protocol which falls

1 UC demands session identifier to be globally i,e; say n parties running m session locally needs to maintaion nxm collision free session identifiers

just in the boundary between cryptographic games and simulation based models so that we can see how these models bereaves.

- *Only responder is secure!*: The security definition allows all queries from adversary to initiator. We only provide security analysis for receiver and never discuss for initiator. Key design of the KE(ENC,SIG) is to use any security primitive in final stage only. Random nonce is exchanged in earlier stages along with party's identifier but it flows in clear. Due to this fact no extra information can be obtained from initiator before starting final communication flow. Intuitively, if adversary tries to mangle with either of the values, it cannot lead him to finish the protocol as responder will detect it on arrival of the final message. Hence, we did not analyse it in detail and left it as an exercise.

- *Analysed $\widehat{\pi}$ is not π given!*: Applying JUC theorem in hybrid model[2] changes the actual protocol and security guarantee came for $\widehat{\pi}$ instead of π. This is a side effect of UC with joint state and [CR03] argued that π is also secure. If one is still suspicious (which is a good thing), either needs to wait for some straight implementation of joint state in UC theorem or use some alternative like GNUC.[HS11] Authors use hierarchical implementation for the common resource in GNUC. A quick analogy will be running several virtual machines inside a single operating system.

- *Anonymous user can join!*: It is good to make model extendable. Anonymous user can join the system after initialization(after distributing public keys). This can be modelled via considering special party process with id P_0. This party can be an honest/dishonest or an adversary itself. Other parties at least need to know the public keys. In practice, certificates expire and redistributed so this is a possible case. P_0 is just for giving idea as to how to accommodate more parties and it can be extended to include more. However, the security guarantee is only for the parties who join at the beginning, we are arguing that even an anonymous user can join and the system will remain secure.

- *Corrupted party's partner or partner session security is not guaranteed!*: We have shown here two party key exchange protocol . If one of the parties is under adversarial control (Adaptive adversary), there is no longer honest

2 parties in real model has secure access to ideal functionality

majority in the protocol . There is no known general protocol which is adaptively secure for honest minority in public channel in both the models. In case of reveled partner session or corrupted party, the adversary knows the session key[3], which is the prime goal of security. Hence there is no point of maintaining security if session key itself is exposed to adversary.

5.2 MOVING RELATIONSHIP TO SEPARATION

Real world problem. Real-world execution in simulation based model uses asynchronous communication between parties, adversary and the environment. $\mathcal{A}$ enjoys this non-interactive communication property to send new message before delivering the earlier ones. And this is not limited to parties but $\mathcal{E}$ as well. Due to this $\mathcal{A}$ convinces $\mathcal{E}$(decisional entity) that he knows the secret for any party P_i by adaptively corrupting and providing transcript of (cipher text, secret key) list. On the other hand cryptographic games corruption operation is atomic i.e; the *Game*(decisional entity) knows and update its internal state once adversary asks to corrupt a party.

This is a generic property of simulation based model and provides very strong security. But due to this **"too strong"** security, no semantically secure generic encryption scheme can be proven secure in presence of adaptive adversary.

The dichotomy problem of Simulator. Ideal world process execution is magically secure. This magical security comes from information theory and to make this model useful, adversarial powers are worth to know. Simulations define power of $\mathcal{S}$ the same as $\mathcal{E}$ which restricts $\mathcal{S}$ to gain extra advantage.

The *impossibility task* of simulator is to guess the message by just knowing the length of a message. Which arguably provides strong construction to ideal functionality.[4] The adaptiveness of $\mathcal{A}$ creates this dichotomy and to overcome this $\mathcal{S}$ either needs more power than $\mathcal{E}$ or an assistance setup. The first open path of more powerful simulator than environment hurts the security definition. But there is significant work done with the additional setup.

The demanding change. The quest of stronger security guarantee than standard *Games* comes at price so far. The simulation based models demand to pre-distribute global session identifier

3 KE(ENC,SIG) does not provide perfect forward security

4 argued secrecy in ideal world is same as perfect encryption scheme e.g; One-Time Pad.

for modelling. If the protocol has joint state (which is practically true) the security guarantee is provided to a slightly modified protocol.[5]

FINAL NOTE. Cryptographic games evolve over time since Goldwasser and Micali used provable security for modelling security guarantee of encryption scheme.[GM84] It is popularly used by the cryptographer due to its simplicity and adaptiveness to new tool. But due to this adaptiveness, there is no strong abstract framework known for *Games*. Cryptographic tools are getting much more complex day by day and it demands comprehensive security modeling technique like the current simulation models are offering. To move forward in the discussion needs an abstract modeling technique in *Games*, which also provides practical requirement like composition or modularity.

5 joint state implementation is to provide modularity otherwise classic UC can also model security without joint state but inefficiently.

ï»¿

6

CONCLUSION

Notion of security models as Cryptographic games and Simulation based model, especially UC framework is not new but we present a new and interesting concept from previous work. It is discussed earlier that UC security model is stronger than the cryptographic games but we found it is not true at least for the case of Key exchange protocol . We have presented a simple two party key exchange protocol and shown it to be secure in classic cryptographic game based model presented by Bellare Rogaway (BR Model). Then we have shown the same protocol is not secure in Universal Composable framework, a standard simulation based framework. Then we commented out the problem with generic simulation and we support the argument that it is the case for all common simulation based framework which uses ideal and real world dichotomy. This draws a clear line of separation between these two security models and to conclude which one of these two models is stronger will require further investigation.

6.1 POSSIBLE FUTURE WORK

There is good research motive to find the "Holy Grail" of security modelling, a formal relationship to demonstrate which one of these two is Stronger in the same provable sense. We put the stepping stone for key exchange protocols and one could find easier to start from key exchange then hoping for a generalisation of the relationship to make final comment on two models discussed. We also find it interesting not having any generic standard abstract model for games. This is not related to the work we carried so far but one could further investigate this open problem.

A

APPENDIX

A.1 RANDOM ORACLE MODEL

Random oracle model remains one of the most controversial security modelling technique. In brief history, Bellare and Rogaway uses random oracles for practical security modelling. The security proof in this model is simpler because of the construction of a fully random oracle. Authors suggests to show secure a cryptographic tool in random oracle model and later replace it with ongoing secure enough actual hash function (like MD5 or SHA1). Bellare, Boldyreva and Palacio presented an asymmetric encryption scheme, which is secure in ROM but unsecured in standard games. Later, the results is analysed by Koblits and Menezes and shown that previous analysis break a fundamental rule in construction of uninstantiable encryption scheme. They further provides evidence to support the model but these discussion is out of the scope of current work.

Random oracle $\mathcal{O} : \{0,1\}^* \rightarrow \{0,1\}^k$, where $k \in M$ chosen independently and uniformly by selecting $\mathcal{O}(m) \in \{0,1\}^k$ for $m \in \{0,1\}^*$. The random oracle model uses this oracle to provide any randomness. This a very strong assumption about randomness as the only way to get a collision is to send same message twice to random oracle. Further the construction of the model uses a map of input and out put valuse follows,

- if input value is queried for the first time, select $r \in \{0, 1\}$ randomly and add into the map for later compararision of input quries.
- if input value is quried before, return the value from map which is preveously choosen randomly.

The oracle is accessed by all the parties including adversary and environment. In this model, protocol π uses random oracles to implement some functionality $\mathcal{F}$. If π is proven secure in the model then in real world all random oracle is replaced by practical function who mimic the random oracle, mainly a hash function.

A.2 PROBABILITY OF $\mathcal{B}$ GUESSES TEST SESSION CORRECTLY

Here we present a independent result to the main proof. It gives a weak relation between static and adaptive adversary

for KE(ENC,SIG).
$\mathcal{A}$ chooses Test session not statically so $\mathcal{B}$ is not able to know it in advance. It could be any random session for $\mathcal{B}$ hence probability of guessing is,

$$Pr[\mathcal{B}^{ST}_{Guess}] = \frac{1}{\text{total no of session}}$$

But the IND-CCA game played by $\mathcal{B}$ needs to guess party only to match the long term key use for the Test session ST. The real catch is if $\mathcal{A}$ ask to corrupt the guessed party then it will look like a fake experiment to $\mathcal{A}$. In presence of adaptive adversary we can at most say that two parties will not be corrupted (Parties involve in session ST) and either of them will be owner of last step of the protocol , hence

$$Pr[\mathcal{B}^{ST}_{Guess}] = \frac{2}{\text{total no of parties}} \cdot \frac{1}{2} = \frac{1}{n}$$

Proposition A.1 *Let there is an adaptive adversary $\mathcal{A}$ and Protocol KE(ENC,SIG) is at least as secure as adversary $\mathcal{A}$ is* ***non-adaptive****.*

$$Adv^{IND-CCA}_{Adaptive}(\mathcal{B}) \leqslant \frac{k}{n} Adv^{IND-CCA}_{Static}(\mathcal{B}) \qquad 2 < k \leqslant n$$

Current proof can be easily leads to the statement. We argue that if $\mathcal{A}$ is not adaptive than session which are corrupted, known in the beginning of the game. In that case,

$$Pr[\mathcal{B}^{ST}_{Guess}] = \frac{\text{total no of non-corrupt parties}}{\text{total no of parties}} = \frac{1}{n}$$

which is greater than current case if static adversary does not corrupt all parties except parties involve in ST.

BIBLIOGRAPHY

[BFWW11] Christina Brzuska, Marc Fischlin, Bogdan Warinschi, and Stephen C. Williams. Composability of bellare-rogaway key exchange protocols. In Yan Chen, George Danezis, and Vitaly Shmatikov, editors, *ACM Conference on Computer and Communications Security*, pages 51–62. ACM, 2011.

[BR94] Mihir Bellare and Phillip Rogaway. Entity authentication and key distribution. In Douglas Stinson, editor, *Advances in Cryptology CRYPTO 93*, volume 773 of *Lecture Notes in Computer Science*, pages 232–249. Springer Berlin / Heidelberg, 1994.

[Can01] Ran Canetti. Universally composable security: A new paradigm for cryptographic protocols. In *FOCS*, pages 136–145. IEEE Computer Society, 2001.

[CK02] Ran Canetti and Hugo Krawczyk. Universally composable notions of key exchange and secure channels. In Lars R. Knudsen, editor, *EUROCRYPT*, volume 2332 of *Lecture Notes in Computer Science*, pages 337–351. Springer, 2002.

[CR03] Ran Canetti and Tal Rabin. Universal composition with joint state. In Dan Boneh, editor, *CRYPTO*, volume 2729 of *Lecture Notes in Computer Science*, pages 265–281. Springer, 2003.

[GM84] Shafi Goldwasser and Silvio Micali. Probabilistic encryption. *Journal of Computer and System Sciences*, 28(2):270 – 299, 1984.

[HS11] Dennis Hofheinz and Victor Shoup. Gnuc: A new universal composability framework. *IACR Cryptology ePrint Archive*, 2011:303, 2011.

[Knu74] Donald E. Knuth. Computer Programming as an Art. *Communications of the ACM*, 17(12):667–673, December 1974.

[KT08] Ralf Kusters and Max Tuengerthal. Joint state theorems for public-key encryption and digital signature functionalities with local computation. *2012*

IEEE 25th Computer Security Foundations Symposium, 0:270–284, 2008.

[MvOV96] Alfred Menezes, Paul C. van Oorschot, and Scott A. Vanstone. Key establishment protocols. In *Handbook of Applied Cryptography*, pages 489–541. CRC Press, 1996.

[Nie02] JesperBuus Nielsen. Separating random oracle proofs from complexity theoretic proofs: The non-committing encryption case. In Moti Yung, editor, *Advances in Cryptology, CRYPTO 2002*, volume 2442 of *Lecture Notes in Computer Science*, pages 111–126. Springer Berlin Heidelberg, 2002.

GLOSSARY

adversary	a attacker who wants to break the security. 4
flow	a peice of information transmitted over communication channel between parties. 6
IND-CCA	Indistinguishability under Chosen Ciphertext Attack. 22–24, 38
MD5	Message Digest Five. 37
protocol	a multi-party sequential algorithm which runs in asynchronous steps to commonly achieve a goal. 1, 29
SHA1	secure hash algorithm 1. 37
SK	Session Key. 4
SSL	Secure Socket Layer. 29
ST	Test Session. 38
TLS	Transport Layer Security. 29
TM	Turing Machine: a finite state machine with infinite long working tape to read or write one at a time. 11
UF-CMA	Unforgeability under Chosen Message Attack. 22