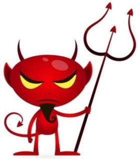
Introduction to Security Reduction

Lecture 1: Definitions

(Algorithm and Security Model)



Adversary

My IQ is up to 186.

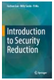
My interest is breaking schemes.

You want me to help you solve problem?

Fool me first!

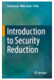
Lecture 12: Flaws in Papers
Lecture 11: Revision of Security Reduction
Lecture 10: Security Proofs for Encryption (Computational)
Lecture 9: Security Proofs for Encryption (Decisional)
Lecture 8: Security Proofs for Digital Signatures
Lecture 7: Analysis (Towards A Correct Reduction)
Lecture 6: Simulation and Solution
Lecture 5: Difficulties in Security Reduction
Lecture 4: Entry to Security Reduction
Lecture 3: Preliminaries (Hard Problem and Secure Scheme)
Lecture 2: Preliminaries (Field, Group, Pairing, and Hash Function)
Lecture 1: Definitions (Algorithm and Security Model)

Computational Complexity Theory

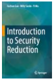


Outline

1. Classical Cryptography vs Modern Cryptography
2. How to Define Algorithms
3. How to Define Security Models
4. Examples for Practice

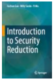


Outline



Classical Cryptography vs Modern Cryptography

- Classical cryptography was referred to as art.
- After the late 20th century (digital computers), cryptography was studied as science and called Modern Cryptography.
- Modern cryptography is distinguished from classical cryptography by its emphasis on formal definitions, models and proofs.
  - Definitions of algorithms, hard assumptions, advantage and so on.
  - Computational model, security model and so on.
  - Security reduction, game-hopping proof and so on.



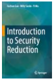
Clarified Concepts

Cryptology= Cryptography + Cryptanalysis

- Cryptography, such as public-key cryptography, group-based cryptography, and elliptic-curve cryptography, is a security mechanism to provide security services for authentication, confidentiality, integrity, etc.
- Cryptanalysis is to analyze existing cryptography.
- Cryptosystem, such as digital signatures, public-key encryption, and identity-based encryption, is a suite of algorithms that provides ONE security service or more.
- Scheme, such as the BLS scheme[1], is a specific construction of the corresponding algorithms for ONE cryptosystem.

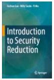
---

[1] Dan Boneh, Ben Lynn, Hovav Shacham (2004). "Short Signatures from the Weil Pairing". Journal of Cryptology. 17: 297-319. doi:10.1007/s00145-004-0314-9.



Clarified Concepts

- Cryptology
  - Cryptanalysis
  - Cryptography
    - Cryptosystem-a
      - Scheme-1
      - Scheme-2
      - ⋮
      - Scheme-k
    - Cryptosystem-b
      - Scheme-1
      - Scheme-2
      - ⋮
      - Scheme-k
    - ⋮
    - Cryptosystem-z
      - Scheme-1
      - Scheme-2



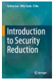
Outline

1 Classical Cryptography vs Modern Cryptography

2 How to Define Algorithms

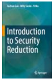
3 How to Define Security Models

4 Examples for Practice



How to Define Algorithms for a Cryptosystem?

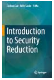
- Fully understand the security service(s) (motivation).
- Which entities (e.g., signer) are involved.
- How many algorithms are involved.
- What is the name of each algorithm.
- Who will runs each algorithm.
- What are inputs and outputs of each algorithm (objects).
- What correctness must the algorithms satisfy?



How to Define Algorithms for Digital Signatures?

Motivation

- A party, say Alice, wants to convince all other parties that a message m is published by her.
- To do so, Alice generates a public/secret key pair (pk, sk) and publishes the public key pk to all verifiers.
- To generate a signature σ_m on m, she digitally signs m with her secret key sk.
- Upon receiving (m, σ_m), any receiver who already knows pk can verify the signature σ_m and confirm the origin of the message m.



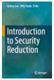
How to Define Algorithms for Digital Signatures?

Motivation

- A party, say Alice, wants to convince all other parties that a message m is published by her.
- To do so, Alice generates a public/secret key pair (pk, sk) and publishes the public key pk to all verifiers.
- To generate a signature σ_m on m, she digitally signs m with her secret key sk.
- Upon receiving (m, σ_m), any receiver who already knows pk can verify the signature σ_m and confirm the origin of the message m.

Which entities (not names) are involved.

Signer, Verifier



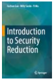
How to Define Algorithms for Digital Signatures?

Motivation

- A party, say Alice, wants to convince all other parties that a message m is published by her.
- To do so, Alice generates a public/secret key pair (pk, sk) and publishes the public key pk to all verifiers.
- To generate a signature σ_m on m, she digitally signs m with her secret key sk.
- Upon receiving (m, σ_m), any receiver who already knows pk can verify the signature σ_m and confirm the origin of the message m.

How many algorithms are involved.

1 (system parameter generation) + 3



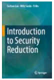
How to Define Algorithms for Digital Signatures?

Motivation

- A party, say Alice, wants to convince all other parties that a message m is published by her.
- To do so, Alice generates a public/secret key pair (pk, sk) and publishes the public key pk to all verifiers.
- To generate a signature σ_m on m, she digitally signs m with her secret key sk.
- Upon receiving (m, σ_m), any receiver who already knows pk can verify the signature σ_m and confirm the origin of the message m.

What is the name of each algorithm.

SysGen, KeyGen, Sign, Verify



How to Define Algorithms for Digital Signatures?

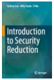
Motivation

- A party, say Alice, wants to convince all other parties that a message m is published by her.
- To do so, Alice generates a public/secret key pair (pk, sk) and publishes the public key pk to all verifiers.
- To generate a signature σ_m on m, she digitally signs m with her secret key sk.
- Upon receiving (m, σ_m), any receiver who already knows pk can verify the signature σ_m and confirm the origin of the message m.

Who runs each algorithm.

SysGen (Authority), KeyGen (Signer), Sign (Signer), Verify (Verifier)

A signer can generate a system parameter for him/her to use alone so that SysGen is run by Signer.



How to Define Algorithms for Digital Signatures?

SysGen$(\lambda) \rightarrow SP$.

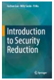
KeyGen$(SP) \rightarrow (pk, sk)$.

Sign$(SP, sk, m) \rightarrow \sigma_m$.

Verify$(m, \sigma_m, SP, pk) \rightarrow \{0, 1\}$.

input and output OBJECTS only for algorithm definition.

Note: Which objects must be defined in each algorithm should be well considered. For example, can the verification algorithm include sk as the input?



How to Define Algorithms for Digital Signatures?

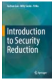
SysGen: The system parameter generation algorithm takes as input a security parameter λ. It returns the system parameters SP.

KeyGen: The key generation algorithm takes as input the system parameters SP. It returns a public/secret key pair (pk, sk).

Sign: The signing algorithm takes as input a message m from its message space, the secret key sk, and the system parameters SP. It returns a signature of m denoted by σ_m.

Verify: The verification algorithm takes as input a pair (m, σ_m), the public key pk, and the system parameters SP. It returns "accept" if σ_m is a valid signature of m signed with sk; otherwise, "reject."

Note: All algorithms are probabilistic polynomial time algorithms.



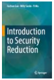
Remarks on Algorithm Definition

The algorithm definition must satisfy some Correctness.

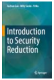
- In digital signatures, all generated signatures can be verified.
- In encryption, all generated ciphertexts can be decrypted.

The correctness should hold for all outputs such as all key pairs.

Note: The correctness is related to services of cryptosystems.

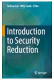


Outline



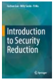
Questions Before Security Model

- How to analyze the security of a scheme?
- Can a scheme (secure in a security model) resist any attack?
- Is a security model defined for a scheme?



What is Security Model?

- When we propose a scheme for a cryptosystem, we do not analyze its security against a list of attacks, such as replay attack and collusion attack. Instead, we analyze that the proposed scheme is secure in a security model.
- A security model can be seen as an abstract of multiple attacks for a cryptosystem. If a proposed scheme is secure in a security model, it is secure against any attack that can be described and captured in this security model.
- Abstracted attacks focus on what information the adversary learns instead of how the adversary learns these information.
- A security model is defined for a cryptosystem not for a scheme.



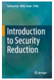
How to Define a Security Model?

We can use a game played between adversary and challenger to describe a security model.

- The challenger is the secret key owner of a cryptosystem.
- The adversary is trying to break the cryptosystem.

A security model defines:

- The adversary's capabilities:
  - What information the adversary can query
  - When the adversary can query information
- The adversary's security goal:
  - How the adversary wins the game (breaks the scheme).

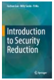


How to Define a Security Model?

The definition of a security model is composed of four parts:

- Setup: The initialization between the adversary and the challenger.
- Capabilities: Describe what and when the adversary queries.
- Security Goal: The condition of wining the game for the adversary.
- Advantage: Define a parameter satisfying.
  - If the parameter is negligible, the cryptosystem is secure.
  - Otherwise, the parameter is non-negligible and it is insecure.

Note: Advantage and (non)negligible will be introduced in later lectures.



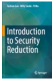
When Defining a Security Model

When defining a security model, we consider input and output only the same as algorithm definition. The definition must be applicable to all potential schemes (proposed for a cryptosystem).

For example, when the adversary queries a signature on a message m, the challenger must respond to this query by running the signing algorithm with (pk, sk, m) as input to output signature σ_m.

- We don't care how to generate the signature σ_m.
- We don't care what the signature looks like [2].
- We only care what the adversary queried (i.e. the message m)

[2] The signature is just a symbol denoted by σ_m.



When Defining a Security Model

When defining a security model, we don't consider

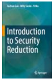
- The adversary's trivial attack that can help the adversary break the cryptosystem easily. For example, the adversary asks the challenger to share all secrets (e.g. all secret keys) with it.

- The adversary's strategy about how it obtains a piece of information. Therefore, when the adversary makes a query, the challenger must respond to this query correctly and honestly.

When defining a security model, make sure that

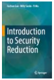
the advantage is negligible.

(Therefore, the adversary is restricted in some queries.)



Security Model for Digital Signatures

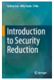
- Setup: A key pair is generated and the adversary knows pk.
- Capabilities: The adversary obtains signatures of some messages.
- Goal: The adversary cannot forge signature of a new message.
- Advantage: The probability of successful forgery.



Security Model for Digital Signatures

- Setup: A key pair is generated and the adversary knows pk.
- Capabilities: The adversary obtains signatures of some messages.
- Goal: The adversary cannot forge signature of a new message.
- Advantage: The probability of successful forgery.

Trivial Attack: The adversary queries the signing/secret key. It can easily win the game. Any attack that will make advantage=1 must be forbidden.



Security Model of Digital Signatures

The security model of existential unforgeability against chosen-message attacks (EU-CMA) can be described as follows.

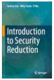
Setup. Let SP be the system parameters. The challenger runs the key generation algorithm to generate a key pair (pk, sk) and sends pk to the adversary. The challenger keeps sk to respond to signature queries.

Query. The adversary makes signature queries on messages that are adaptively chosen by the adversary itself. For a signature query on the message m_i, the challenger runs the signing algorithm to compute σ_{m_i} and then sends it to the adversary.

Forgery. The adversary returns a forged signature σ_{m^*} on some m^* and wins the game if

- σ_{m^*} is a valid signature of the message m^*.
- A signature of m^* has not been queried in the query phase.

The advantage ϵ of winning the game is the probability of returning a valid forged signature.



Remarks on Security Model (1)

- A cryptosystem might have more than one security service.
- Each security service could need one security model.

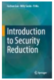
  For example:

  signcryption = signature + encryption

  Two security models:

  cannot break signature, cannot break encryption

- A security model could capture multiple security definitions (services). An example is anonymous receiver with message indistinguishability in IBE.

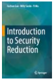


Remarks on Security Model (2)

Security models for a security service of a cryptosystem might have different definitions, depending on capabilities and security goal.

- Standard security model: the capability and security goal are acceptable by most applications (or by cryptography community).
- Strong security model: the adversary has a strong capability and/or a easier security goal than the standard one. (Resist more attacks)
- Weak security model: the adversary has a weak capability and/or a harder security goal than the standard one. (Resist less attacks)

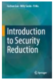
For example: in a weak security model, the adversary is only allowed to know signatures on some given messages, instead of querying signatures on any messages.



Remarks on Security Model (3)

- A scheme secure in a strong security model means that it can resist more powerful attacks that a scheme in a weak security model.
- Strong security model was proposed due to some applications that need strong security.
- Weak security model was proposed because
  - a weak security is enough and we can construct more efficient schemes in this model.
  - some very efficient schemes can only be proved secure in the weak security model.

Note: Breaking a scheme in a security model= solving a computing problem



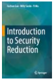
Outline

1 Classical Cryptography vs Modern Cryptography

2 How to Define Algorithms

3 How to Define Security Models

4 Examples for Practice

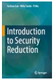


Online/Offline Signatures

Motivation

- To speed up the signature generation with pre-computations.
- In the offline phase, most of heavy computations can be pre-completed without knowing the messages to be signed.
- In the online phase, upon receiving the message to be signed, the signature can be quickly generated with a secret token computed in the offline phase.

Try to define algorithms and security model for this cryptosystem.

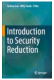


Identity-Based Signatures

Motivation

- Suppose we are in a system where there is only one key pair, called master public key and master secret key, generated by an authority.
- Each user in this system can apply for a key pair. The public key is the user's identity. For example, the public key of user Alice is ID="Alice". The private key of each user is computed with the master secret key and the user's public key.
- When signing a message, Alice can sign the message with her private key and the master public key.
- When verifying a signature from Alice, a verifier can use the identity Alice and the master public key to check the validity.

Try to define algorithms and security model for this cryptosystem.

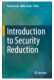


Public-Key Encryption

Motivation

- A party, say Bob, wants to send a sensitive message m to another party, say Alice, though they do not share any secret key.
- Alice first generates a public/secret key pair (pk, sk) and publishes her public key pk to all senders.
- With pk from Alice, Bob can then encrypt the sensitive message m and sends the resulting ciphertext to Alice.
- Alice can in turn decrypt the ciphertext with the secret key sk and obtain the message m.

Try to define algorithms and security model for this cryptosystem.

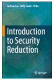


Identity-Based Encryption

Motivation

- Suppose we are in a system where there is only one key pair, called master public key and master secret key, generated by an authority.
- Each user in this system can apply for a key pair. The public key is the user's identity. For example, the public key of user Alice is ID="Alice". The private key of each user is computed with the master secret key and the user's public key.
- When encrypting a message for Alice, an encryptor can generate the ciphertext with the identity ID=Alice and the master public key.
- When Alice decrypts a ciphertext, she can use the private key and the master public key to extract the message.

Try to define algorithms and security model for this cryptosystem.

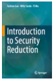


Identity-Based Broadcast Encryption

Motivation

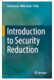
- An extension of identity-based encryption.
- We can encrypt a message for a set of users with their identities.
- Any user can decrypt the ciphertext with his/her private key and the set of identities, if his/her identity is within the identity set.
- The motivation of this notion is to generate a short ciphertext.

Try to define algorithms and security model for this cryptosystem.



Answers

The definitions of algorithms and security models for previous cryptosystems can be found from any related published papers.




END