INTRODUCTION TO SECURITY REDUCTION I: BASICS

SHUANGJUN ZHANG

Content

1. *Encryption: A Formal Definition*
2. *What Does Security Mean in Cryptography*
3. *Proofs by Reduction*
4. *Stronger Security Notions*
5. *Structuring security proofs as sequences games*

Content

1. *Encryption: A Formal Definition*
2. *What Does Security Mean in Cryptography*
3. *Proofs by Reduction*
4. *Stronger Security Notions*
5. *Structuring security proofs as sequences games*

1. Encryption: A Formal Definition

- *1.1 Private-Key Encryption Scheme*

A private-key encryption scheme is a tuple of probabilistic polynomial-time algorithms (Gen, Enc, Dec) such that:

- 1. The key-generation algorithm Gen takes as input 1^n and outputs a key k; we write $k \leftarrow Gen(1^n)$. We assume without loss of generality that any key k output by $Gen(1^n)$ satisfies $|k| \geq n$.
- 2. The encryption algorithm Enc takes as input a key k and a plaintext message $m \in \{0,1\}^*$, and outputs a ciphertext c. Since Enc may be randomized, we write this as $c \leftarrow Enc_k(m)$.
- 3. The decryption algorithm Dec takes as input a key k and a ciphertext c, and outputs a message m, We assume that Dec is deterministic, and so write $m := Dec_k(c)$

It is required that for every n, every key k output by $Gen(1^n)$, and every $m \in \{0,1\}^*$, it holds that $Dec_k(Enc_k(m)) = m$.

1. Encryption: A Formal Definition

- *1.2 Public-Key Encryption Scheme*

A public-key encryption scheme is a tuple of probabilistic polynomial-time algorithms (Gen, Enc, Dec) such that:

- 1. The key-generation algorithm Gen takes as input 1^n and outputs a pair of keys (pk, sk). we write $(pk, sk) \leftarrow Gen(1^n)$. We refer to the first of these as the public key and the second as the private key. We assume for convenience that pk and sk each has length at least n.
- 2. The encryption algorithm Enc takes as input a public key pk and a plaintext message $m \in \{0,1\}^*$, and outputs a ciphertext c. Since Enc may be randomized, we write this as $c \leftarrow Enc_{pk}(m)$.
- 3. The decryption algorithm Dec takes as input a private key sk and a ciphertext c, and outputs a message m, We assume that Dec is deterministic, and so write $m := Dec_{sk}(c)$

It is required that for every n, every key (pk, sk) output by $Gen(1^n)$, and every $m \in \{0,1\}^*$, it holds that $Dec_{sk}(Enc_{pk}(m)) = m$.

Content

1. *Encryption: A Formal Definition*
2. *What Does Security Mean in Cryptography?*
3. *Proofs by Reduction*
4. *Stronger Security Notions*
5. *Structuring security proofs as sequences games*

2. What Does Security Mean in Cryptography

- The goal of encryption is to keep the plaintext hidden from an eavesdropper who can monitor the communication channel and observe the ciphertext.
- How should we define security?
- After getting the ciphertext:
  - *The adversary can't get plaintext.*
  - *The efficient adversary can't get plaintext.*
  - *The efficient adversary can't get any bit of plaintext.*
  - *The efficient adversary can't get any bit of plaintext except with a very small probability.*
- How should we define efficient adversary and very small probability?
  - *Efficient adversary: probabilistic polynomial-time adversary (polynomial-size circuit family)*
  - *Small probability: negligible probability*
- The probabilistic polynomial-time adversary can't get any bit of plaintext except with a negligible probability.

2. What Does Security Mean in Cryptography

- Formal Definition of Negligible Function:
  - *A function f from the natural numbers to the non-negative real numbers is negligible if for every positive polynomial p there is an N such that for all integers $n > N$ it holds that $f(n) < \frac{1}{p(n)}$.*
- Formal Definition of Security:
  - *An encryption scheme (Gen, Enc, Dec) is security(in private-key model) for messages of length l, if for every probabilistic polynomial-time adversary A and all $i \in \{1,2, \dots, l\}$, there is a negligible function $negl$ such that*

$$\Pr\left[A\left(1^n, Enc_k(m)\right) = m^i\right] \leq \Pr\left[A(1^n, \epsilon) = m^i\right] + negl(n) = \frac{1}{2} + negl(n)$$

2. What Does Security Mean in Cryptography

- Definition of Semantic Security:
  - *An encryption scheme (Gen, Enc, Dec) is semantic security (in private-key model) for messages of length l, if for every probabilistic polynomial-time adversary A there exist a ppt algorithm A' such that for any polynomial-time computable functions f and h, there is a negligible function $negl$ such that*

$$|\Pr[A(1^n, Enc_k(m), h(m)) = f(m)] - \Pr[A'(1^n, |m|, h(m)) = f(m)]| = negl(n)$$

- The adversary A is given the ciphertext $Enc_k(m)$ as well as the external information $h(m)$, and attempts to guess the value of $f(m)$. Algorithm A' also attempts to guess the value of $f(m)$, but is given only $h(m)$ and the length of m.
- The security requirement states that A's probability of correctly guessing $f(m)$ is about the same as that of A'. Intuitively, then, the ciphertext $Enc_k(m)$ does not reveal any additional information about the value of $f(m)$.

2. What Does Security Mean in Cryptography

- Indistinguishable Encryption
- Consider the following game played between an adversary and challenger:
  - *1. The challenger B runs $Gen(1^n)$ and gets a key k.*
  - *2. The adversary A is given input 1^n, and outputs a pair of messages m_0, m_1 with $|m_0| = |m_1|$.*
  - *3. The challenger B flips a random coin $b \in \{0,1\}^*$ and computes ciphertext $c \leftarrow Enc_k(m_b)$, gives the ciphertext to A. We refer to c as the challenge ciphertext.*
  - *4. The adversary A outputs a guess b' of b.*

The advantage of an adversary A in this game is defined as $|\Pr[b = b'] - \frac{1}{2}|$.

Theorem 2.1 An encryption scheme is semantic security if and only if all PPT adversaries have at most a negligible advantage in the above game.

We can therefore use indistinguishability as our working definition, while being assured that the guarantees achieved are those of semantic security.

2. What Does Security Mean in Cryptography

- Construction 2.1(One-Time Pad):
- Define a private-key encryption scheme for messages of length n as follows:
  - *1. Gen: on input 1^n, choose uniform $k \in \{1,0\}^n$ and output as the key.*
  - *2. Enc: on input a key $k \in \{0,1\}^n$ and a message $m \in \{0,1\}^n$, output the ciphertext*
    $$c = k \oplus m$$
  - *3. Dec: on input a key $k \in \{0,1\}^n$ and a ciphertext $c \in \{0,1\}^n$, output the message*
    $$m = k \oplus c$$
- Theorem 2.2 The construction 2.1 is a fixed-length private-key encryption scheme that has indistinguishable encryptions.
- If k is a true random string, then c is also a true random string. It means that c is independent of b, c reveals no information about b. The adversary A's advantage in this attack game is 0.

Content

1. *Encryption: A Formal Definition*
2. *What Does Security Mean in Cryptography*
3. *Proofs by Reduction*
4. *Stronger Security Notions*
5. *Structuring security proofs as sequences games*

3. Proofs by Reduction

- If we wish to prove that a given construction is computationally secure, then we must rely on unproven assumptions(i.e. Integer Factorization Problem, Discrete Logarithm Problem).
- Basic idea of proofs by reduction:
  - *If a ppt adversary A can break our scheme Π with a noticeable probability, then an ppt algorithm B can be constructed to solve a mathematical problem X by using A as a subroutine.*
  - *It means that if no ppt algorithm can solve problem X(X is a hard problem), then there is no ppt adversary A can break our scheme Π .*

3. Proofs by Reduction

- Construction 3.1:
- Let G be a pseudorandom generator with expansion factor l. Define a private-key encryption scheme for messages of length l as follows:
  - *1. Gen: on input 1^n, choose uniform $k \in \{1,0\}^n$ and output as the key.*
  - *2. Enc: on input a key $k \in \{0,1\}^n$ and a message $m \in \{0,1\}^{l(n)}$, output the ciphertext*

$$c = G(k) \oplus m$$

  - *3. Dec: on input a key $k \in \{0,1\}^n$ and a ciphertext $c \in \{0,1\}^{l(n)}$, output the message*

$$m = G(k) \oplus c$$

3. Proofs by Reduction

- Definition of a pseudorandom generator(PRG):
- Let l be a polynomial and let G be a deterministic polynomial-time algorithm such that for any n and any input $s \in \{0,1\}^n$, the result $G(s)$ is a string of length $l(n)$. We say that G is a pseudorandom generator if the following conditions hold:
  - *1. (Expansion:) For every n it holds that $l(n) > n$.*
  - *2. (Pseudorandomness:) For any ppt algorithm D, there is a negligible function $negl$ such that*

$$|\Pr[D(G(s)) = 1] - \Pr[D(r) = 1] \leq negl(n)$$

We call l the expansion factor of G.

Theorem 3.2 If G is a pseudorandom generator, then the construction 3.1 is a fixed-length private-key encryption scheme that has indistinguishable encryptions.

3. Proofs by Reduction

- Proof of Theorem 3.2:
  - *Suppose the adversary A's advantage in the attack game is $\epsilon(n)$, we construct a PPT distinguisher D for guessing that w is a pseudorandom string or a true random string.*
  - *D is given as input a string $w \in \{0,1\}^{l(n)}$*
    - Run $A(1^n)$ to obtain a pair of massage $m_0, m_1 \in \{0,1\}^{l(n)}$
    - Choose a random bit $b \in \{0,1\}$, set $c = w \oplus m_b$
    - Give c to A and obtain output b', output 1 if $b' = b$, and output 0 otherwise.
  - *Analyzing the behavior of D*
  - *Case 1: if w is a true random string, then c is also a true random string. It means that c is independent of b, c reveals no information about b. The adversary A's advantage in this attack game is 0. We can get*

$$\Pr[D(w) = 1] = \Pr[b' = b] = \frac{1}{2}$$

  - *Case 2: if w is a pseudorandom string , that is, $w = G(s)$ for a true random string $s \in \{0,1\}^n$. Then, c is a valid ciphertext from construction 3.1. The adversary A's advantage in this attack game is $\epsilon(n)$. We can get*

$$\Pr[D(G(s)) = 1] = \Pr[b' = b] = \frac{1}{2} + \epsilon(n)$$

3. Proofs by Reduction

- *Put them all, we get*

$$|\Pr[D(w) = 1] - \Pr[D(G(s)) = 1]| = \left|\frac{1}{2} - \left(\frac{1}{2} + \epsilon(n)\right)\right| = \epsilon(n)$$

- *Therefore, if G is a pseudorandom generator, then $\epsilon(n)$ is negligible, and the advantage of A is negligible.*

Content

1. *Encryption: A Formal Definition*
2. *What Does Security Mean in Cryptography*
3. *Proofs by Reduction*
4. *Stronger Security Notions*
5. *Structuring security proofs as sequences games*

4. Stronger Security Notions

- *In our discussion above, the adversary is passive and have access only to a ciphertext.*
- *Consider an active adversary, he can choose random plaintexts to be encrypted and obtain the corresponding ciphertexts. The goal of the attack is to gain some further information which reduces the security of the encryption scheme. This attack model is called chosen plaintext attack(CPA) model.*
- *In the formal definition we model chosen-plaintext attacks by giving the adversary A access to an encryption oracle, viewed as a "black-box" that encrypts messages of A's choice using a key k that is unknown to A.*

4. Stronger Security Notions

- *Indistinguishable Encryption in CPA model*
- *Consider the following game played between an adversary and challenger:*
  - 1. The challenger B runs $Gen(1^n)$ and gets a key k.
  - 2. The adversary A is given input 1^n and oracle access to $Enc_k(\cdot)$, and outputs a pair of messages m_0, m_1 with $|m_0| = |m_1|$.
  - 3. The challenger B flips a random coin $b \in \{0,1\}^*$ and computes ciphertext $c \leftarrow Enc_k(m_b)$, gives the ciphertext to A. We refer to c as the challenge ciphertext.
  - 4. The adversary A continues to have oracle access to $Enc_k(\cdot)$, and outputs a guess b' of b.
- *The advantage of an adversary A in this game is defined as* $\left|\Pr[b = b'] - \frac{1}{2}\right|$*. An encryption scheme is CPA-Secure if all PPT adversaries have at most a negligible advantage in the above game.*

4. Stronger Security Notions

- *It is easy to see that the construction 3.1 is not CPA-Secure. Since A can use the encryption oracle to get a ciphertext of m_0 and m_1(c_0 and c_1). If the challenge ciphertext $c = c_0$, then output 0, else output 1.*
- *We can construct a CPA-Secure encryption scheme by using Pseudorandom Functions(PRF)*
- *First, We introduce the important notion of pseudorandom functions(PRF). A pseudorandom function (PRF) F is a deterministic algorithm that has two inputs: a key k and an string x, it outputs $y = F(k, x) = F_k(x)$.*
- *Now, instead of considering "random-looking" strings we consider "random-looking" functions. Thus, we must instead refer to the pseudorandomness of a distribution on functions.*
- *Consider a set $Func[X, Y]$ denotes the set of all functions $f: X \to Y$, then $|Func[X, Y]| = |Y|^{|X|}$. Choose a uniform function $f \in Func[X, Y]$, f is a true random function.*
- *We call F pseudorandom if the function F_k (for a uniform key k) is indistinguishable from a function chosen uniformly at random from the set of all functions having the same domain and range.*

4. Stronger Security Notions

- Definition of PRF:
  - *Let $F : \{0,1\}^* \times \{0,1\}^* \to \{0,1\}^*$ be an efficient, length-preserving, keyed function. F is a pseudorandom function if for all PPT distinguishers D, there is a negligible function $negl$ such that:*

$$\left|\Pr\left[D^{F_k(\cdot)}(1^n) = 1\right] - \Pr\left[D^{f(\cdot)}(1^n) = 1\right]\right| \leq negl(n)$$

- Construction 4.1
- Let F be a PRF. Define a private-key encryption scheme for messages of length n as follows:
  - *1. Gen: on input 1^n, choose uniform $k \in \{1,0\}^n$ and output as the key.*
  - *2. Enc: on input a key $k \in \{0,1\}^n$ and a message $m \in \{0,1\}^n$, choose uniform $r \in \{0,1\}^n$ and output the ciphertext*

$$c =< r, F_k(r) \oplus m >$$

  - *3. Dec: on input a key $k \in \{0,1\}^n$ and a ciphertext $c =< r, s >$, output the message*

$$m = F_k(r) \oplus s$$

4. Stronger Security Notions

- Theorem 4.2 If F is a PRF, then Construction 4.1 is a CPA-secure private-key encryption scheme for messages of length n.
- Proof of Theorem 4.2
  - *Suppose the adversary A's advantage in the attack game is $\epsilon(n)$, we construct a PPT distinguisher D for the pseudorandom function F. The distinguisher D is given oracle access to some function O, and its goal is to determine whether this function is "pseudorandom".*
  - *Distinguisher D(D is given input 1^n and access to an oracle $O : \{0,1\}^n \to \{0,1\}^n$.)*

1. Run $A(1^n)$, Whenever A queries its encryption oracle on a message $m \in \{0,1\}^n$, answer this query in the following way:
   - a) Choose a uniform $r \in \{0,1\}^n$
   - b) Query $O(r)$ and obtain response y
   - c) Return the ciphertext $\langle r, y \oplus m \rangle$ to A.

2. When A outputs messages $m_0, m_1 \in \{0,1\}^n$, choose a uniform bit b, and then:
   - a) Choose a uniform $r \in \{0,1\}^n$
   - b) Query $O(r)$ and obtain response y
   - c) Return the challenge ciphertext $\langle r, y \oplus m_b \rangle$ to A.

3. Continue answering encryption-oracle queries of A as before until A outputs a bit b'. Output 1 if $b' = b$, and 0 otherwise.

4. Stronger Security Notions

- *let $q(n)$ be an upper bound on the number of queries that $A(1^n)$ makes to its encryption oracle. (Note that q must be upper-bounded by some polynomial.) D runs in polynomial time since A does.*
- *Analyzing the behavior of D*
- *Case 1: if the D's oracle is a PRF, the challenge ciphertext c is a valid ciphertext from the construction 4.1. The adversary A's advantage in this attack game is $\epsilon(n)$. That is*

$$\left|\Pr[b = b'] - \frac{1}{2}\right| = \epsilon(n)$$

  - D output 1 if and only if $b = b$', so

$$\left|\Pr\left[D^{F_k(\cdot)}(1^n) = 1\right] - \frac{1}{2}\right| = \left|\Pr[b = b'] - \frac{1}{2}\right| = \epsilon(n)$$

4. Stronger Security Notions

- *Case 2: if the D's oracle is a true random function, we claim in this case A's advantage is at most* $\frac{q(n)}{2^n}$.

Let r^* denote the random string used when generating the challenge ciphertext $\langle r^*, f(r^*) \oplus m_b \rangle$.

There are two possibilities:

- The value r^* is never used when answering any of A's encryption-oracle queries: in this case $f(r^*)$ is a true random string in $\{0,1\}^*$ (since f is a true random function), and $f(r^*) \oplus m_b$ is also a true random string. This means that the challenge ciphertext reveals no information about b. So, in this case the advantage of A is 0. That is

$$\Pr[b = b'] = \frac{1}{2}$$

- The value r^* is used when answering at least one of A's encryption-oracle queries: In this case, A may easily determine whether m_0 or m_1 was encrypted. This is so because if the encryption oracle ever returns a ciphertext $< r^*, s >$ in response to a request to encrypt the message m, the adversary learns that $f(r^*) = s \oplus m$.
- Let S denote the event that the value r^* is used when answering at least one of A's encryption-oracle queries and let S_i denote the event that that the value r^* is used when answering A's ith encryption-oracle query. Clearly $\Pr[S_i] = \frac{1}{2^n}$. since A makes at most $q(n)$ queries to its encryption oracle, we have

$$\Pr[S] = \Pr[S_1 \cup S_2 \cup \cdots \cup S_{q(n)}] \leq q(n) \times \Pr[S_1] = \frac{q(n)}{2^n}$$

4. Stronger Security Notions

- So, In case 2, the advantage of A is
- $\left|\Pr[b' = b] - \frac{1}{2}\right|$

$$= \left|\Pr[b' = b \wedge S] + \Pr\left[b' = b \wedge \overline{S}\right] - \frac{1}{2}\right|$$

$$= \left|\Pr[b' = b|S] \cdot \Pr[S] + \Pr\left[b' = b\middle|\overline{S}\right] \cdot \Pr\left[\overline{S}\right] - \frac{1}{2}\right|$$

$$\leq \left|\Pr[S] + \Pr\left[b' = b\middle|\overline{S}\right] - \frac{1}{2}\right| = \left|\frac{q(n)}{2^n} + \frac{1}{2} - \frac{1}{2}\right| = \frac{q(n)}{2^n}$$

- D output 1 if and only if $b = b'$, so

$$\left|\Pr\left[D^{f(\cdot)}(1^n) = 1\right] - \frac{1}{2}\right| = \left|\Pr[b = b'] - \frac{1}{2}\right| \leq \frac{q(n)}{2^n}$$

4. Stronger Security Notions

- *Together Case 1 and Case 2 show that*

$$\left|\Pr\left[D^{F_k(\cdot)}(1^n)=1\right]-\Pr\left[D^{f(\cdot)}(1^n)=1\right]\right| \le \left|\left(\frac{1}{2}+\epsilon(n)\right)-\left(\frac{1}{2}-\frac{q(n)}{2^n}\right)\right| = \epsilon(n)+\frac{q(n)}{2^n}$$

- *Therefore, if F is a PRF, then there exists a negligible function $negl(n)$, such that*

$$\epsilon(n)+\frac{q(n)}{2^n}=negl(n)$$

- *Since $\frac{q(n)}{2^n}$ is negligible, $\epsilon(n)$ is also negligible. It means that the advantage of A is negligible and the construction 4.1 is CPA-Secure private-key encryption scheme.*

4. Stronger Security Notions

- *Consider a more powerful adversary, the adversary has the ability not only to obtain encryptions of messages of its choice (as in a chosen plaintext attack), but also to obtain the decryption of ciphertexts of its choice(with one exception discussed later). This attack model is called chosen ciphertext attack(CCA) model.*
- *Formally, we give the adversary access to a decryption oracle in addition to an encryption oracle.*

4. Stronger Security Notions

- *Indistinguishable Encryption in CCA model*
- *Consider the following game played between an adversary and challenger:*
  - 1. The challenger B runs $Gen(1^n)$ and gets a key k.
  - 2. The adversary A is given input 1^n and oracle access to $Enc_k(\cdot)$ and $Dec_k(\cdot)$, and outputs a pair of messages m_0, m_1 with $|m_0| = |m_1|$.
  - 3. The challenger B flips a random coin $b \in \{0,1\}^*$ and computes ciphertext $c \leftarrow Enc_k(m_b)$, gives the ciphertext to A. We refer to c as the challenge ciphertext.
  - 4. The adversary A continues to have oracle access to $Enc_k(\cdot)$ and $Dec_k(\cdot)$, but is not allowed to query the latter on the challenge ciphertext itself and outputs a guess b' of b.
- *The advantage of an adversary A in this game is defined as* $\left|\Pr[b = b'] - \frac{1}{2}\right|$*. An encryption scheme is CCA-Secure if all PPT adversaries have at most a negligible advantage in the above game.*
- *Question: Is Construction 4.1 a CCA-Secure private-key encryption scheme?*

Content

1. *Encryption: A Formal Definition*
2. *What Does Security Mean in Cryptography*
3. *Proofs by Reduction*
4. *Stronger Security Notions*
5. *Structuring security proofs as sequences games*

5. Structuring security proofs as sequences games

- Security proofs in cryptography may sometimes be organized as sequences of games.
- To prove security using the sequence-of-games approach, one proceeds as follows.
  - *One constructs a sequence of games, $Game\ 0, Game\ 1, \ldots, Game\ n$, where $Game\ 0$ is the original attack game with respect to a given adversary.*
  - *We denote the event that the adversary A succeed in $Game\ i$ as S_i.*
  - *The proof shows that $\Pr[S_i]$ is negligibly close to $\Pr[S_{i+1}]$ for $i = 0, \ldots, n-1$, and that $\Pr[S_n]$ is equal (or negligibly close) to the "target probability"($\Pr[S_n] = 0$ or $\Pr[S_n] = \frac{1}{2}$).*
  - *From this, and the fact that n is a constant, it follows that $\Pr[S]$ is negligibly close to the "target probability," and security is proved.*

5. Structuring security proofs as sequences games

- Recall the Construction I.3.1, we will re-proof Theorem I.3.2 in a new way.
- Proof of Theorem 3.2:
  - *Game 0. This is the original attack game. We may describe the attack game algorithmically as follows:*
    - $k \leftarrow \{0,1\}^n$
    - $(m_0, m_1) \leftarrow A(1^n)$
    - $b \leftarrow \{0,1\}; c = G(k) \oplus m_b$
    - $b' \leftarrow A(c)$

  If we define S_0 to be the event that $b = b'$ in Game 0, then the adversary's advantage is $\left|\Pr[S_0] - \frac{1}{2}\right|$.

  - *Game 1. We now make one small change to the above game. Namely, instead of using PRG G, we use a true random string r.*
    - $r \leftarrow \{0,1\}^l$
    - $(m_0, m_1) \leftarrow A(1^n)$
    - $b \leftarrow \{0,1\}; c = r \oplus m_b$
    - $b' \leftarrow A(c)$

  If we define S_1 to be the event that $b = b'$ in Game 1.

5. Structuring security proofs as sequences games

- *Claim 1.* $\Pr[S_1] = \frac{1}{2}$
- *Claim 2.* $|Pr[S_0]\text{-}Pr[S_1]| = \epsilon_{prg}$ *where* ϵ_{prg} *is the PRG-advantage of some efficient algorithm.*
- *Proof of Claim 2.*
  - Consider a distinguishing algorithm D works as follows.
  - Algorithm D(w)
    - $(m_0, m_1) \leftarrow A(1^n)$
    - $b \leftarrow \{0,1\}; c = w \oplus m_b$
    - $b' \leftarrow A(c)$
    - *If b = b' output 1, else output 0*
  - Algorithm D effectively "interpolates" between Games 0 and 1.
  - If w(G(s)) is a pseudorandom string, then the adversary A is in Game 0, and therefore $\Pr[D(G(s)) = 1] = \Pr[S_0]$
  - If w(G(s)) is a true random string, then the adversary A is in Game 1, and therefore $\Pr[D(r) = 1] = \Pr[S_1]$
  - So, we get $|\Pr[S_0] - \Pr[S_1]| = |\Pr[D(G(s)) = 1] - \Pr[D(r) = 1]| = \epsilon_{prg}$

5. Structuring security proofs as sequences games

- *Combining Claim 1 and Claim 2, we see that*

$$\left|\Pr[S_0] - \frac{1}{2}\right| = \epsilon_{prg}$$

- *Since ϵ_{prg} is negligible, we get the advantage of A in game 0 is negligible.*

Exercise Organizing the proof of theorem 4.2 as sequences of games.