MIT/LCS/TR-379

THE NOTION OF SECURITY FOR PROBABILISTIC PUBLIC-KEY CRYPTOSYSTEMS

ROBERT HAL SLOAN

OCTOBER 1986

The Notion of Security for Probabilistic Public-key Cryptosystems

by

Robert Hal Sloan

B.S., Yale University

(1983)

Submitted to the Department of
Electrical Engineering and Computer Science
in Partial Fulfillment of the
Requirements of the Degree of

Master of Science

in

Electrical Engineering and Computer Science

at the

Massachusetts Institute of Technology

September 1986

Signature of Author Robert Hal Sloan
Department of Electrical Engineering and Computer Science
June 13, 1986

Certified by
Silvio Micali
Associate Professor of Computer Science

Accepted by
Arthur C. Smith
Chairman, Departmental Committee on Graduate Students

The Notion of Security for Probabilistic Public Key Cryptosystems

by

Robert Hal Sloan

Submitted to the Department of Electrical Engineering and Computer Science on June 13, 1986 in partial fulfillment of the requirements for the Degree of Master of Science in Electrical Engineering and Computer Science

Abstract

The purpose of a cryptosystem is to allow people to communicate securely over an open channel. Before one can discuss whether a cryptosystem meets this goal, however, one must first rigorously define what is meant by security.

Three very different formal definitions of security for public-key cryptosystems have been proposed—two by Goldwasser and Micali and one by Yao. In this thesis, it is shown that the three definitions are essentially equivalent.

As originally proposed, the three definitions are not equivalent. The inequivalence, however, is caused only by some minor technical choices. After rectifying those choices, we prove all three definitions to be equivalent. This equivalence provides evidence that the right formalization of the notion of security has been reached.

Portions of this thesis represent joint work with Silvio Micali.

Thesis Supervisor: Prof. Silvio Micali

Title: Associate Professor of Computer Science

This research was supported by a General Electric Foundation Fellowship, NSF grant DCR-8509905, and by a National Science Foundation Fellowship.

Acknowledgments

I have been extremely fortunate to have Silvio Micali as my research advisor. He has been an outstanding source of ideas and guidance. The technical content of this thesis has benefitted greatly from his insights; the prose in this thesis has benefitted greatly from his abundant help in editing.

I have also benefitted from numerous technical discussions with Shafi Goldwasser. I want to thank her for her insight and helpfulness.

I'd like to thank Ravi Boppana for his help with Chernoff bounds, and Tom Cormen for his help in navigating the typesetting program.

Contents

1 Introduction **7**

1.1 Public-Key Cryptography 7

1.2 The RSA Public-key system 8

1.3 Probabilistic Encryption 9

1.4 Security 10

2 Notation and Public-key Scenarios **11**

2.1 Notation and Conventions for Probabilistic Algorithms. . . 11

2.2 Cryptographic Scenarios 12

2.3 Passes 13

3 Definitions of Security **15**

3.1 Informal Discussion 15

3.2 GM-security (3-pass) 16

3.3 Semantic Security (3-pass) 17

3.4 Y-security (3-pass) 18

3.5 The original definitions vs. mine 20

3.6 Inequivalence of the original definitions 22

4 Main Results **24**

4.1 Semantic Security Implies GM-security 24

4.2 Y-security implies GM-security 25

4.3 GM-security implies Y-security 26

5 One-Pass Scenarios **30**

5.1 GM-security (1-pass) 30

5.2 Semantic Security (1-pass) 31

5.3 Y-security (1-pass) 32

5.4 Equivalence 32

A Proof of Lemma 1 **34**

Chapter 1

Introduction

1.1 Public-Key Cryptography

The era of modern cryptography began with Diffie and Hellman's famous 1976 paper [4] which presented the concept of public-key cryptography. Informally, there is a community of users, A, B, ... In the Diffie-Hellman paradigm, each user U in the system selects a pair of encryption/decryption algorithms (E_U, D_U) such that for all x, $D_U(E_U(x)) = x$. User U publishes E_U (in an ad hoc public file) but keeps D_U secret. Any other user, in order to send securely a binary string m to U, first looks up E_U, then computes $y = E_U(m)$, and finally sends y to U.

Diffie and Hellman insisted that such a system be secure against any adversary who wiretaps the communications channels, intercepts the cyphertext y and tries to compute $D_U(y)$. Note that the concern here is only with *passive* adversaries; our adversary is not, for example, allowed either to alter the messages sent or to inject his own messages into the system.

The security of cryptosystems based on the Diffie-Hellman model, and in fact the security of every cryptosystem I will discuss in this thesis, is based on complexity theory. That is to say, statements such as "No adversary can extract the plaintext" or "No adversary can compute any information about the plaintext" really mean that it is computationally infeasible for an adversary to do such a thing.

The first concrete implementation of a cryptosystem based on Diffie and Hellman's idea was the RSA scheme of Rivest, Shamir, and Adleman [8]. A brief description of their cryptosystem is given in below.

1.2 The RSA Public-key system

Alice's Preprocessing steps:

1. Find two random distinct primes p_1, p_2.
2. Compute $n = p_1 p_2$.
3. Compute $\varphi(n) = (p_1 - 1)(p_2 - 2)$.
4. Compute s, t such that $st \equiv 1 \bmod \varphi(n)$. Thus $(s, \varphi(n)) = 1$. (Given a modulus m, there is a fast algorithm for computing multiplicative inverses mod m. Hence all we need to do is pick s at random from $\mathbf{Z}^*_{\varphi(n)}$ and then compute its inverse.)
5. Publish s, n in some public file ("phone book"), but keep t secret.

Instructions for Bob:

1. Bob has a message $m =$ "Hi! ..." that he wishes to send to Alice. First he must of course represent m as a binary number in some agreed upon way.
2. Bob then computes $y = m^s \bmod n$ and sends y to Alice.

Instructions for Alice:

Alice can easily recover the plaintext by computing $m = y^t \bmod n$.

1.3 Probabilistic Encryption

The RSA scheme—and indeed, any cryptosystem following the Diffie-Hellman model—is *deterministic*. That is to say, any given plaintext message has a unique encryption. As Goldwasser and Micali pointed out [5], discussing the security of a deterministic public-key cryptosystem is a tricky business. For instance, a deterministic public-key cryptosystem cannot be used to send securely a given small set of messages, say $\{0, 1, \ldots, 10\}$. In fact, any "code breaking algorithm" may, on input E and a cyphertext y, check first whether $E(i) = y$ for $i = 0, \ldots, 10$.

Also, even when the tapper is not capable of retrieving m from E and $E(m)$, he may be able to compute some partial information about m, say the value of $P(m)$ for some predicate P. From this point of view, if messages are deterministically encrypted, then an adversary can always extract some information about the plaintext from the cyphertext. At the very least, an adversary can easily compute, given only E and the cyphertext, y, the following predicate P_E :

$$P_E(m) = \begin{cases} 0 & \text{if the last bit of the encryption } E(m) \text{ is } 0 \\ 1 & \text{if the last bit of the encryption } E(m) \text{ is } 1. \end{cases} \tag{1.1}$$

We are not claiming that P_E is an interesting predicate. We are simply pointing out the difficulties of discussing the security of deterministic cryptosystems.

To prevent an adversary from computing even such partial information about the plaintext from the cyphertext, Goldwasser and Micali suggested using probabilistic encryption algorithms. In other words, one may think of the encryption algorithm as an algorithm with two inputs, $E = E(\cdot, \cdot)$, the message to be transmitted and a random string (selected by the sender). If one chooses a probabilistic encryption algorithm properly, then every plaintext message will have many different encryptions, but a given cyphertext will still be the encryption of only one plaintext message.

This choice also allowed Goldwasser and Micali to introduce rigorous and convincing notions of security. Changing the scenario from deterministic to probabilistic becomes necessary as their security conditions cannot be met by any deterministic cryptosystem.

1.4 Security

The key desideratum for any cryptosystem is that encrypted messages must be secure. Before one can discuss whether a cryptosystem has this property, however, one must first rigorously define what is meant by security. Three different rigorous notions of security have been proposed. Goldwasser and Micali [5] suggested two different definitions, polynomial security and semantic security, and proved that the first notion implies the second. Yao [11] proposed a third definition, one inspired by information theory, and suggested that it implies semantic security.

Not completely knowing the relative strength of these definitions is rather unpleasant. For instance, several protocols have been proved correct adopting the notion of polynomial security. Are these protocols that are secure with respect to that particular definition, or are they secure in a more general sense? In other words, a natural question arises: Which of the definitions is the "correct" one? Even better: How should we decide the "correctness" of a definition?

The best possible answer to these questions would be to find that the proposed definitions—each attempting to be as general as possible—are all equivalent. In this case, one obviously no longer has to decide which one definition is best. Moreover, the equivalence suggests that one has indeed found a strong, natural definition.

In this thesis, I will show that the three notions are *essentially* equivalent. The three originally proposed definitions were not equivalent. However, as I will point out, this inequivalence was caused only by some minor technical choices. We can prove, after rectifying these marginal choices, the desired equivalences and keep the spirit of the definitions intact.

Chapter 2

Notation and Public-key Scenarios

2.1 Notation and Conventions for Probabilistic Algorithms.

The notation I present here is almost identical to that introduced by Goldwasser, Micali, Rivest [6].

I emphasize the number of inputs received by an algorithm as follows. If algorithm A receives only one input I write "$A(\cdot)$", if it receives two inputs "$A(\cdot,\cdot)$" and so on.

"PS" will stand for "probability space"; in this paper we only consider countable probability spaces. In fact, we deal almost exclusively with probability spaces arising from probabilistic algorithms.

If $A(\cdot)$ is a probabilistic algorithm, then for any input i, the notation $A(i)$ refers to the PS which assigns to the string σ the probability that A, on input i, outputs σ. Notice the special case where A takes no inputs; in this case the notation A refers to the algorithm itself, whereas the notation $A()$ refers to the PS defined by running A with no input. If S is a PS, denote by $\Pr_S(e)$ the probability that S associates with element e. Also, we denote by $[S]$ the set of elements which S gives positive probability. In the case that $[S]$ is a singleton set $\{e\}$ we will use S to denote the value e;

this is in agreement with traditional notation. (For instance, if $A(\cdot)$ is an algorithm that, on input i, outputs i^3, then we may write $A(2) = 8$ instead of $[A(2)] = \{8\}$.)

If $f(\cdot)$ and $g(\cdot, \cdots)$ are probabilistic algorithms then $f(g(\cdot, \cdots))$ is the probabilistic algorithm obtained by composing f and g (i.e. running f on g's output). For any inputs $x, y, \ldots$ the associated probability space is denoted $f(g(x, y, \ldots))$.

If S is any PS, then $x \leftarrow S$ denotes the algorithm which assigns to x an element randomly selected according to S; that is, x is assigned the value e with probability $\Pr_S(e)$. If F is a finite set, then the notation $x \leftarrow F$ denotes the algorithm which assigns to x an element randomly selected from the PS which has sample space F and the uniform probability distribution on the sample points. Thus, in particular, $x \leftarrow \{0, 1\}$ means x is assigned the result of a coin toss.

The notation $\Pr(p(x, y, \ldots) \mid x \leftarrow S;\ y \leftarrow T;\ \ldots)$ denotes the probability that the predicate $p(x, y, \ldots)$ will be true, after the ordered execution of the algorithms $x \leftarrow S$, $y \leftarrow T$, etc. I use analogous notation for expected value—$\mathrm{Ex}(f(x, y, \ldots) \mid x \leftarrow S;\ y \leftarrow T;\ \ldots)$—where now f is a function which takes numerical values.

Let $\mathcal{RA}$ denote the set of probabilistic polynomial-time algorithms. I assume that a natural representation of these algorithms as binary strings is used.

By 1^n we denote the unary representation of integer n, i.e.

$$\underbrace{11\ldots1}_{n}$$

2.2 Cryptographic Scenarios

Here I specify those elements that are necessary for all public-key cryptography.

A *cryptographic scenario* consists of the following components:

- A *security parameter* n which is chosen by the user when he creates his encryption and decryption algorithms. The parameter n will determine a number of quantities (length of plaintext messages, overall security, etc.).

- A sequence of *message spaces*, $M = \{M_n\}$ from which all plaintext messages will be drawn. M_n consists of all messages allowed to be sent if the security parameter has been set equal to n. In order to make our notation simpler, (but without loss of generality), we'll assume that $M_n = \{0,1\}^n$. There is a probability distribution on each message space, $\Pr_n : M_n \to [0,1]$ such that $\sum_{m \in M_n} \Pr_n(m) = 1$.

- A *public-key cryptosystem* is an algorithm $\mathcal{C} \in \mathcal{RA}$ that on input 1^n outputs the description of two polynomial-size circuits E and D such that:

  1. E has n inputs and $l(n)$ outputs, and D has $l(n)$ inputs and n outputs. (l is some polynomial that gives the length of the cyphertext.)
  2. E is probabilistic; D is deterministic.
  3. For all $m \in \Sigma^n$, $\Pr(D(\alpha) = m \mid (E, D) \leftarrow \mathcal{C}(1^n);\ \alpha \leftarrow E(m)) = 1$.

  Notice that $[E(m)]$ is a set which is typically quite large. Our notation requires us to write $\alpha \in [E(m)]$ to refer to α, a particular encryption of m. Nevertheless, we will sometimes sloppily write $E(m)$ for a particular encryption of m when the meaning is clear.

- The *number of "allowed passes."* This number specifies how A and B agree upon an encryption algorithm E output by the public-key cryptosystem. To this crucial notion (surprisingly neglected so far), we devote the next section.

2.3 Passes

Within the public-key model, A and B can alternate communicating back and forth as many times as they feel are necessary to achieve security. Call each alternation a *pass.*

Any number of passes are, of course, permissible. I concentrate on what I believe are the two most interesting and important cases, one and three passes. I do not consider more than three passes, because, if trapdoor permutations exist, a well designed probabilistic encryption scheme can achieve as much security as is possible using only three passes.

Three-pass systems

The three-pass case is, perhaps, the most natural to think about. It corresponds to a telephone conversation. A has a message m that she wants to securely communicate to B. A calls up B and says, "I have a message I'd like to send to you." B, so alerted, proceeds to generate an encryption/decryption algorithm pair, (E, D), and tells A, "Please use E to encrypt your message." A then uses E to encrypt her message and tells B "$E(m)$."

Notice the key property of a three-pass system: The message and the encryption algorithm are selected independently of one another. We are nevertheless in a public-key model, since anyone tapping the phone line gets to hear B tell E to A.

One-pass systems

A one-pass system corresponds to what is commonly called a public file system. In the one-pass model, A simply looks up B's public encryption algorithm, E, in a "phone book" and uses it to encrypt her message. (One pass is a slight misnomer. At some point, in what we may view as a preprocessing stage, B must have communicated his encryption algorithm, presumably by telling it to whomever publishes the phone book of encryption algorithms, and thus indirectly to A. "One and a half passes" might be more accurate. "Half" refers to the preprocessing stage that needs to be performed only once.) In this case, the choice of message *can* depend on E.

Chapter 3

Definitions of Security

3.1 Informal Discussion

The main result of this thesis is

> GM-security, semantic security, and Y-security (all formally defined later in this chapter) are equivalent for both three-pass and one-pass cryptosystems.

Interestingly, the equivalence still holds in the one-pass scenario, but the notions of security vary between the one-pass and three-pass scenarios. This point has not been given the proper attention, because people frequently confuse the notion of one-pass public-key cryptography with public key cryptography in general.

The distinction, however, is crucial for avoiding errors, particularly in cryptographic protocols. Let us informally state the two definitions of security that are achievable in the two scenarios if trapdoor permutations exist.

3-pass A cryptosystem is secure if, for every message m in the message space, it is impossible to efficiently distinguish an encryption of m from random noise.

1-pass A cryptosystem is secure if, for every message m that is efficiently computable on input the encryption algorithm alone, it is impossible to efficiently distinguish an encryption of m from random noise.

In other words, in the one-pass scenario one cannot just blithely write, "For all messages m." For instance, if one closely analyzes all known public-key cryptosystems, it is *conceivable* that if (E, D) is an encryption/decryption pair, then D can be easily computed from $E(D)$. For instance, the constructive reduction of security to quadratic residuosity given by Goldwasser and Micali [5] for their cryptosystem would *vanish* if the encrypted message is allowed to be D itself.[1]

Such problems cannot arise in the three-pass scenario because the encryption algorithm E is selected after and independently of the message m.

In this thesis, we will only prove the desired equivalences in detail for the three-pass scenario. The proof for the one-pass scenario is sketched in the final chapter. The reason for this choice is that the definitions of security are much more easily stated for three-pass systems. It is much more convenient to say, "For all messages m," than "For all messages m that are efficiently computable given the encryption algorithm as an input."

3.2 GM-security (3-pass)

This definition is essentially what Goldwasser and Micali [5] called polynomial security.

A *line tapper* is a family of polynomial-size probabilistic circuits $T = \{T_n\}$. Each T_n takes four strings as input and outputs either 0 or 1. However, to make our next equation more readable, we will treat T_n's output as being either its second or third input (0 or 1 respectively).

[1] Notice that if Bob publishes an encryption algorithm E in the public file while keeping its associated decryption algorithm D secret, then any other user, being limited to efficient computation and ignorant of D, necessarily selects her message m efficiently from the input E—maybe without even looking at E—and perhaps other inputs altogether independent of (E, D). However, in designing cryptographic protocols, one would often like to be able to transmit things like $E(D)$. For instance, if that type of message were allowed, one would have a trivial solution to the problem of verifiable secret sharing [3].

Definition Let $\mathcal{C}$ be a public-key cryptosystem. $\mathcal{C}$ is *GM-secure* if for all line tappers T and $c > 0$, for all sufficiently large n, for every $m_0, m_1 \in \{0,1\}^n$

$$\Pr(\mathrm{T_n}(\mathrm{E}, \mathrm{m_0}, \mathrm{m_1}, \alpha) = \mathrm{m} \mid \mathrm{m} \leftarrow \{m_0, m_1\}\,; \mathrm{E} \leftarrow \mathcal{C}(1^n);\ \alpha \leftarrow \mathrm{E(m)}) < \frac{1}{2} + \mathrm{n}^{-c}. \tag{3.1}$$

Remark: In reading the above definition, one should pay close attention to our notation. Upon casual consideration of 3.1, one might conclude that there aren't any GM-secure cryptosystems! After all, the definition says that the encryption E must be secure against *any* m_0 and m_1, both of which are given as inputs to the line tapper. What happens if we put $m_0 = D$, a description of the decryption algorithm? The answer to this question is that our notation specifies that *first* we choose m from $\{m_0, m_1\}$ (and thus m_0 and m_1 already had been set), and *then* we choose our encryption algorithm. If $\mathcal{C}$ is GM-secure, then the probability that $\mathcal{C}(1^n)$ assigns to any given output is quite small, say $O(2^{-n})$. Thus there's little worry that $\mathcal{C}$ will just happen to output a decryption algorithm $D = m_0$. Notice how the above definition (via our notation) models the three-pass scenario.

3.3 Semantic Security (3-pass)

Again, this definition is essentially the same as in [5]. It can be viewed as a polynomial time bounded version of Shannon's "perfect secrecy" [10]. This definition makes use of the probability distributions $\Pr_n$ over the sets of messages M_n. Informally, let f be *any* function, $f : M \to V = \{$any values the adversary likes$\}$. Intuitively, f should be thought of as some information about the plaintext that the adversary would like to be able to compute from the cyphertext—say the first seventeen bits of the plaintext. A cryptosystem is semantically secure if no adversary, on input $E(m)$ can compute $f(m)$ more accurately than by random guessing.

Definition Let $\mathcal{C}$ be a public-key cryptosystem, $M = \{M_n\}$ a sequence of message spaces, and V be any set. Let $\mathcal{F} = \left\{f_n^E : M_n \to V \mid E \in [\mathcal{C}(1^n)]\right\}$ be any set of functions. For $v \in V$, we denote by $f_n^{E^{-1}}(v)$ the inverse image of v; that is, the set $\{m \in M_n \mid f_n^E(m) = v\}$. Then the probability of the most probable value for $f(m)$ is $p_n^E = \max\left\{\sum_{m \in f_n^{E^{-1}}(v)} \Pr_n(m) \mid v \in V\right\}$. p_n^E is the maximum probability with which one could guess $f_n^E(m)$ without having any idea whatsoever what m is.

C is *semantically secure* if for every family of polynomial-size probabilistic circuits $A = \{A_n(\cdot,\cdot)\}$, for all $c > 0$, and for all sufficiently large n

$$\Pr(\, A_n(E,\alpha) = f_n^E(m) \mid m \leftarrow M_n;\ E \leftarrow C(1^n);\ \alpha \leftarrow E(m)\,) < p_n^E + n^{-c}. \tag{3.2}$$

3.4 Y-security (3-pass)

Yao's definition [11] is inspired by information theory, but its context differs from classical information theory in that the communicating agents, A(lice) and B(ob), are limited to probabilistic polynomial-time computations.

An intuitive explanation of Yao's definition is the following: A has a series of n^k messages, selected from a probability space known to both A and B, and an encryption of each message. She wishes to transmit enough bits to B so that he can (in polynomial time with very high probability) compute all the plaintexts. A cryptosystem is Y-secure if the average number of bits A must send B is the same regardless of whether B possesses a copy of the cyphertext.

I now make this notion precise, first by defining "Alice and Bob," and then eventually defining Y-security itself.

Let $M = \{M_n\}$ be a sequence of message spaces. Each M_n is $\{0,1\}^n$ with a fixed probability distribution. (Note that an information theorist would consider M to be a sequence of *sources*.)

Let $\varepsilon(n)$ be any function that vanishes faster than n^{-c} for all positive c.

For the sake of compactness of notation, the expression $\vec{m}$ will denote a particular series of n^k messages. That is, $\vec{m}$ stands for $m_1, m_2, \ldots, m_{n^k}$.

Let f be any positive function such that $f(n) \leq n$. Intuitively, $f(n)$ is the number of bits per message that A must transmit to B in order for B to recover the plaintexts. Recall that all the messages in M_n have length n.

Definition An $f(n)$ *c/d pair* (c/d for compressor/decompressor) for M is a pair of families of probabilistic polynomial-size circuits, $(\{A_n\}, \{B_n\})$, satisfying the following three properties for some constant k and all sufficiently large n:

1. "B_n understands A_n."

$$\begin{aligned} \Pr(\vec{m} = y \mid m_1 \leftarrow M_n; \ldots; m_{n^k} \leftarrow M_n; \beta \leftarrow A_n(\vec{m}); & \\ y \leftarrow B_n(\beta)) &= 1 - O(\varepsilon(n)). \end{aligned} \tag{3.3}$$

2. "A_n transmits only $f(n)$ bits per message."

$$\mathrm{Ex}\left[\frac{|\beta|}{n^k} \mid m_1 \leftarrow M_n; \ldots; m_{n^k} \leftarrow M_n; \beta \leftarrow A_n(\vec{m})\right] \leq f(n). \tag{3.4}$$

3. "The output of A_n can be parsed."
For all polynomials Q there exists a probabilistic polynomial-time Turing machine S^Q such that S^Q takes as input n and a concatenated string of $Q(n)$ βs, each of which is a good output from A_n, and separates them. That is, its input is $\beta_1\beta_2\ldots\beta_{Q(n)}$ and its output is $\beta_1\#\beta_2\#\ldots\#\beta_{Q(n)}$. We require that

$$\Pr(S^Q \text{ correctly splits } \beta_1\beta_2\ldots\beta_{Q(n)}) = 1 - O(\varepsilon(n)). \tag{3.5}$$

Remark: The requirement that S^Q exist is a technical requirement. It creates a finite analogue of classical information theory's requirement that messages be transmitted one bit at a time, in an infinite sequence of bits.

We say that the *cost of communicating M is less than or equal to $f(n)$*, in symbols $C(M) \leq f(n)$, if there exists an $f(n)$ compressor/decompressor pair for M.

We define $C(M) > f(n)$ to be the negation of $C(M) \leq f(n)$—that is, *any* circuits "communicating M" must use at least $f(n)$ bits. The definition of $C(M) = f(n)$ is analogous.

Let $\mathcal{C}$ be a cryptosystem. We define $C(M \mid E_{\mathcal{C}}(M))$, *the cost of communicating M given encryptions from $\mathcal{C}$* in a manner analogous to $C(M)$. The only difference is that now *both A_n and B_n* also get E and the n^k values of some encryption function $E \in [\mathcal{C}(1^n)]$ as inputs. (We now call $(\{A_n\}, \{B_n\})$ a *shared cyphertext c/d pair.*) That is, for this definition we must rewrite Equation 3.3 above to read:

$$\begin{aligned}\Pr(\vec{m} = y \mid m_1 \leftarrow M_n; \ldots; m_{n^k} \leftarrow M_n; E \leftarrow \mathcal{C}(1^n);& \\ \alpha_1 \leftarrow E(m_1); \ldots; \alpha_{n^k} \leftarrow E(m_{n^k});& \\ \beta \leftarrow A_n(E, \vec{m}, \vec{\alpha}); y \leftarrow B_n(E, \beta, \vec{\alpha})) &= 1 - O(\varepsilon(n)).\end{aligned} \tag{3.6}$$

An analogous change must also be made to Equation 3.4.

Notice that for this definition, the probabilities involved must be taken over the different choices of E from $\mathcal{C}$ as well as everything else.

Definition Let $\mathcal{C}$ be a public-key cryptosystem. Fix a sequence of message spaces $M = \{M_n\}$ (and thus the probability distribution on each M_n). We say that $\mathcal{C}$ is *Y-secure with respect to M* if

$$C(M) = C(M \mid E_{\mathcal{C}}(M)) + O(\varepsilon(n)). \tag{3.7}$$

We say that $\mathcal{C}$ is *Y-secure* if for all M, $\mathcal{C}$ is Y-secure with respect to M.

3.5 The original definitions vs. mine

As I discussed in Chapter 1, I made minor changes in the cryptographic scenario from [5] and [11]. Here I will spell out those changes are and why they were made.

Changes to Goldwasser and Micali's Definition

There are two ways a cryptosystem (the server that generates encryption/decryption algorithm pairs) can achieve security:

1. The cryptosystem gets a description of a message space M (and thus its probability distribution) as one of its inputs and will output an encryption/decryption algorithm pair to securely encrypt M.

2. The cryptosystem is told nothing about the message space. The encryption algorithms it outputs are supposed to be secure for every possible message space.

We will call the former cryptosystems *aware* and the latter *oblivious*.

Goldwasser and Micali consider aware cryptosystems for both of their definitions of security [5]; Yao doesn't make it clear which type of cryptosystem he is assuming for his definition of security [11]. I believe it makes more sense to consider oblivious cryptosystems, for both theoretical and applied reasons.

The theoretical reason for preferring oblivious cryptosystems is that all three definitions of security are equivalent. (See Chapter 4.) This is a desirable property that fails to hold for aware cryptosystems, as we will show in the next section.

The practical reason for preferring oblivious cryptosystems is that, although it is certainly conceivable that having knowledge of the message space would allow one to design a better encryption algorithm, cryptographers have in fact normally tried to design cryptosystems that are secure for all message spaces. For example, consider the cryptosystem based on arbitrary trapdoor predicates proposed by Goldwasser and Micali [5]. Although they only considered security in the aware cryptosystem sense, their cryptosystem is in fact secure in the stronger, oblivious sense.

Changes to Yao's Definition

In [11], Yao assumes deterministic private key cryptography, but the definition is immediately extended to probabilistic public-key cryptography.

Yao defines the compressor A and decompressor B to be Turing machines, not circuits. I have switched to circuits because it is not clear that there are *any* secure cryptosystems with respect to probabilistic Turing machines. It might be that one can always achieve greater polynomial-time compression given the cyphertext simply because having a shared random (enough) string (in this case the cyphertext!) helps. If it does help, however, having made the compressor and decompressor nonuniform circuits, we can always hardwire in a shared random string of bits.

3.6 Inequivalence of the original definitions

In this section, we point out that, for aware cryptosystems, GM-security is a notion stronger than either semantic security or Y-security. We do this in the following two claims, each supported by an informal argument. These claims can be easily transformed to theorems after formalizing the discussed security notions in terms of aware cryptosystems, a tedious effort once we have realized that the aware setting is not the "right" one.

Claim 1 *If any GM-secure aware public-key cryptosystem exists, then there exist aware public-key cryptosystems that are semantically secure but not GM-secure.*

Let $C(\cdot,\cdot)$ be any GM-secure (and thus semantically secure) aware cryptosystem. We'll construct a $C'(\cdot,\cdot)$ that is still semantically secure, but is not GM-secure.

C' behaves identically to C for all message spaces, except for the message space $\{0,1\}^n$ with uniform probability distribution. In this case, C' runs C to compute an encryption algorithm E, and then outputs the algorithm E' defined by:

$$E'(x) = \begin{cases} 0^n & \text{if } x = 0^n \\ 1^n & \text{if } x = 1^n \\ E(x) & \text{otherwise} \end{cases} \tag{3.8}$$

C' is clearly not GM-secure, because, for the special message space described above, there are two messages, 0^n and 1^n, that are easily distinguished by their encryptions. However, C' is still semantically secure. Those two messages have such a low probability weight that they won't give an adversary any significant advantage—on average—in computing a function of the plaintext on input the cyphertext. □

Claim 2 *If any GM-secure aware public-key cryptosystem exists, then there exist aware public-key cryptosystems that are Y-secure but not GM-secure.*

We construct exactly the same C' as we did for the previous claim. C' is of course not GM-secure. However, the two "weak messages" have such low probability that they basically don't affect the average number of bits necessary to communicate messages from the message space. Thus C' is Y-secure.

$\square$

Chapter 4

Main Results

In this chapter we provide the proof of the equivalence of GM-security, semantic security, and Y-security. We choose to do these proofs by showing that GM-security is equivalent to Y-security and that GM-security is equivalent to semantic-security. We present here only three of the four necessary implications. The proof that GM-security implies semantic security may be found in [5]. We'll present the three proofs in order of increasing difficulty and technical complexity.

4.1 Semantic Security Implies GM-security

This proof is quite simple. If a cryptosystem is not GM-secure, then there exist two messages, m_1 and m_2, which we can easily distinguish. If we make a new message space in which these are the only messages, then given only cyphertext, one has a better than random chance of figuring out which of the two plaintext messages the cyphertext represents.

Theorem 1 *Let C be a public-key cryptosystem. If C is semantically secure, then C is GM-secure.*

Proof Again we prove the contrapositive. Let C be a public-key cryptosystem that is not GM-secure. We will prove that C is not semantically secure.

Formally, $\mathcal{C}$ is not GM-secure means that there exist a line tapper T and a $c > 0$ such that for infinitely many n there are $m_1^n, m_2^n \in M_n$ for which

$$\Pr(\mathrm{T_n}(\mathrm{E}, m_1^n, m_2^n, \alpha) = \mathrm{m} \mid \mathrm{m} \leftarrow \{m_1^n, m_2^n\}\,;\ \mathrm{E} \leftarrow \mathcal{C}(1^{\mathrm{n}});\ \alpha \leftarrow \mathrm{E}(\mathrm{m})) \geq \frac{1}{2} + \frac{1}{\mathrm{n^c}}. \tag{4.1}$$

We construct a new message space M_n as follows: For those n for which equation 4.1 holds, $\Pr_n(m_1^n) = 1/2$ and $\Pr_n(m_2^n) = 1/2$.

We've set up the message space so one can simply guess the plaintext by seeing the cyphertext. More precisely, a circuit guessing the plaintext from the cyphertext can use T as a subroutine and thus obtain a polynomial advantage. On the other hand, without seeing the cyphertext, circuits with no input can only randomly guess the plaintext. Q.E.D.

4.2 Y-security implies GM-security

In the proof of the next theorem, we use a technical lemma that is a variation of Chernoff's bound [2]. The derivation of the lemma from Chernoff's bound is in the appendix.

Lemma 1 *Let X be a random variable having binomial distribution, with n trials and probability of success p. For $0 \leq \alpha \leq 1/2 \leq p \leq 1$, we have $\Pr(X \leq \alpha n) \leq e^{-2(p-\alpha)^2 n}$.*

Theorem 2 (Rackoff [7]) *Let $\mathcal{C}$ be a cryptosystem. If $\mathcal{C}$ is Y-secure, then $\mathcal{C}$ is GM-secure.*

Proof Again we will prove the contrapositive. Let $\mathcal{C}$ be a cryptosystem that is not GM-secure for some message space M. There exists a family of line tappers $T = \{T_n\}$ such that for infinitely many n, there are $m_0^n, m_1^n \in M_n$ such that T_n can distinguish between them. Consider now a new message space M' that, for those n, has $\Pr_n(m_0^n) = 1/2$, $\Pr_n(m_1^n) = 1/2$, and $\Pr_n(m) = 0$ for all other $m \in \{0,1\}^n$.

Clearly $C(M') = 1$: any circuits not sharing cyphertext will need one bit per message to communicate outputs from M'. This fact follows from classical information theory considerations.

On the other hand, we will now show that $C(M' \mid E_C(M')) \leq 1 - 1/n^k$. (The value of the constant k will be specified below.) This value is achieved by a shared cyphertext c/d pair that transmits n^k messages at a time.

A_n gets n^k messages in both plain and cleartext as its input. Since there are only two messages in M'_n, each message can be considered to be a bit b and each cyphertext the encryption of a bit. That is to say, A_n's input is $b_1, b_2, \ldots, b_{n^k}, , \alpha_1, \alpha_2, \ldots, \alpha_{n^k}$ where $\alpha_i \in [E(b_i)]$. A_n now XORs each adjacent pair of messages (bits). That is, put $c_i = b_i \oplus b_{i+1}$ for $i = 1, 2, \ldots, n^k - 1$. Put $\beta = c_1 c_2 \cdots c_{n^k-1}$. This β is the "hint" that A_n sends to B_n. Obviously, $|\beta|/n^k = 1 - 1/n^k$.

Now, can B_n, given β and the α_is as its input, determine the plaintext with probability $1 - O(\varepsilon(n))$? Yes. The "hint", β, constrains B_n to only two possible choices of values for the b_i. That is, if B_n decides that $b_1 = 0$, then it knows the value of all the bits—say $v_1 v_2 \ldots v_{n^k}$. On the other hand, if B_n decides that $b_1 = 1$, then the whole series of messages must of have been $\bar{v}_1 \bar{v}_2 \ldots \bar{v}_{n^k}$ (where $\bar{v}$ is the compliment of v).

B_n also has a line tapper, T_n, that it can use to test the α_i. B_n runs T_n on each α_i and obtains T_n's opinion as to what each bit was. Call this sequence $t_1 t_2 \ldots t_{n^k}$. Since $\mathcal{C}$ is not GM-secure, each t_i is correct with probability $p = \frac{1}{2} + 1/n^j$, for some fixed j. By Lemma 1 (with $\alpha = 1/2$ and "$n = n^k$"), if we make $k \geq 2j + 1$, then the majority of t_is will be correct with probability $1 - O(e^{-n})$. B_n compares the t_i to both the v_i and the $\bar{v}_i$, and decides either $b_1 = 0$ if the majority of t_i coincide with the v_i, or $b_1 = 1$ if the majority of the t_i coincide with the $\bar{v}_i$. Q.E.D.

4.3 GM-security implies Y-security

Theorem 3 *Let $\mathcal{C}$ be a public-key cryptosystem. If $\mathcal{C}$ is GM-secure, then $\mathcal{C}$ is Y-secure.*

Proof We'll prove the contrapositive. A bird's eye view of our proof is as follows. Assuming that $\mathcal{C}$ is not Y-secure, there exists a good shared cyphertext c/d pair that manages to communicate using "few" bits. This pair will allow us to test (for some special pair of messages m_1 and m_2) whether a particular α is the encryption of either m_1 or m_2 thus violating the GM-security condition. Namely, if the pair works successfully on inputs α and m_1, we declare α to be an encryption of m_1; otherwise we declare α to be an encryption of m_2.

Let us proceed formally. Since C is not Y-secure, there is a particular message space sequence, $M = \{M_n\}$, such that C is not Y-secure for M. That is to say, there exist a shared cyphertext c/d pair $AB = (\{A_n\}, \{B_n\})$, a positive integer k, and a polynomial P such that

($\star$) A_n communicates n^k messages from M_n to B_n using "few" bits per message—on top of the cyphertext which they get to share for free.

($\star\star$) Furthermore, for every c/d pair AB', there exists an infinite subset $N' \subseteq \mathbf{N}$, such that for all $n \in N'$, on average AB' uses at least $1/P(n)$ more bits per message than AB does.

We're now going to run a series of experiments to see how AB behaves on inputs that it doesn't "expect." We begin, however, by running a control experiment:

In experiment n-EXP_0, we pick n^k messages m_i at random from M_n and an E at random from $[C(1^n)]$, and run A_n on input

$$\begin{array}{ccccc} m_1 & m_2 & m_3 & \dots & m_{n^k} \\ E(m_1) & E(m_2) & E(m_3) & \dots & E(m_{n^k}) \end{array}$$

(The output will be a string β such that B_n, on inputs β and $E(m_1), \dots, E(m_{n^k})$ will output $m_1, \dots, m_{n^k}$ with overwhelming probability.)

Now consider the following experiment, n-EXP_i: This time we again pick n^k messages and an E at random, but we also pick one more message, r, at random from M_n, and set $\rho = E(r)$. Now we run A_n with i copies of ρ replacing the first i cyphertexts in its input, and then run B_n on A_n's output. A "picture" of A_n's input is

$$\begin{array}{ccccc} m_1 & \dots & m_i & m_{i+1} & \dots & m_{n^k} \\ \rho & \dots & \rho & E(m_{i+1}) & \dots & E(m_{n^k}) \end{array}$$

Definition We define the *difference between n-EXP_i and n-EXP_j*, $d_n(i,j)$ to be the maximum of the average difference between

1. the length of the βs output by A_n in the two experiments, and
2. the frequency with which B_n recovers the correct plaintexts in the two experiments.

Claim 1: There exists a polynomial Q such that for an infinite subset $N'' \subseteq \mathbf{N}$ and for all $n \in N''$ $d_n(0, n^k) > 1/Q(n)$.

Proof of Claim 1: By contradiction. Assume that for all sufficiently large n, n-EXP_0 is indistinguishable from n-EXP_{n^k}. Then A_n and B_n still function successfully on input

$$\begin{array}{ccccc} m_1 & m_2 & m_3 & \ldots & m_{n^k} \\ \rho & \rho & \rho & \ldots & \rho \end{array}$$

where m_i, r, ρ, and E are as above. We now construct A'_n, B'_n to violate $(\star\star)$. We simply hardwire the encryption of some random string into a pair of circuits identical to A_n and B_n but not sharing cyphertext. By assumption, these circuits are a a c/d pair violating $(\star\star)$. $\square$

Claim 2: For all $n \in N''$, there is a polynomial Q' and an i, $0 \le i \le n^k - 1$, such that $d_n(i, i+1) > 1/Q'(n)$.

Proof of Claim 2: Fix $n \in N''$. $d_n(0,0) = 0$ and $d_n(0, n^k) \ge 1/Q(n)$. Therefore, there must be an i such that $d_n(i, i+1) \ge \frac{1}{n^k Q(n)}$. $\square$

Let $n \in N''$. For simplicity, but WLOG, consider the case where $i = 0$ in Claim 2, and $d_n(0,1)$ is due to a difference in the length of A_n's output (rather than B_n's success rate).

Let us restate Claim 2 in a more convenient form. Consider the following *joint* experiment, n-EXP_{01}. Randomly draw $r, m_1, \ldots, m_{n^k}$ from M_n and set $E \leftarrow \mathcal{C}(1^n)$. Run both n-EXP_0 and n-EXP_1 on the *same* inputs. That is, run n-EXP_0 on input

$$\begin{array}{ccccc} m_1 & m_2 & m_3 & \ldots & m_{n^k} \\ E(m_1) & E(m_2) & E(m_3) & \ldots & E(m_{n^k}) \end{array}$$

to compute A_n's output β_0 and run n-EXP_1 on input

$$\begin{array}{ccccc} m_1 & m_2 & m_3 & \ldots & m_{n^k} \\ E(r) & E(m_2) & E(m_3) & \ldots & E(m_{n^k}) \end{array}$$

to compute A_n's output on this input, β_1. The output of n-EXP_{01} is $|\beta_1| - |\beta_2|$. Then, by the linearity of the average we get that the expected value of the output of n-EXP_{01} is at least $1/Q'(n)$.

From this it immediately follows that

$(\star\star\star)$ there exist $\bar{r}, \bar{m}_1, \bar{m}_2, \ldots, \bar{m}_{n^k}$ in M_n such that the expected value of the output of n-EXP_{01} is still greater than $1/Q'(n)$ when the average of the length of β is computed only over the choice of $E \leftarrow C(1^n)$ and of encryptions of messages.

Now for all $n \in N''$ we can build a tapper T_n that will succeed in distinguishing two messages m_1^n and m_2^n, described below.

Fix $\bar{r}$ and $\bar{m}_i$ to be messages that fit the requirements of $(\star\star\star)$. We set T_n's inputs: $m_1^n = \bar{m}_1$ and $m_2^n = \bar{r}$. T_n gets as inputs $E \in [C(1^n)]$, m_1^n, m_2^n, and α, where either $\alpha \in [E(m_1^n)]$ or $\alpha \in [E(m_2^n)]$

T_n picks $m \in \{m_1^n, m_2^n\}$ at random and runs A_n on input

$$\begin{array}{ccccc} m & \bar{m}_2 & \bar{m}_3 & \ldots & \bar{m}_{n^k} \\ \alpha & E(\bar{m}_2) & E(\bar{m}_3) & \ldots & E(\bar{m}_{n^k}) \end{array}$$

to compute a β. There is some threshold length value v for the experiment described at $(\star\star\star)$ such that if $|\beta| < v$ it is more likely that $\alpha \in [E(\bar{m}_1)]$ and if $|\beta| > v$ it is more likely that $\alpha \in [E(\bar{r})]$. Thus T_n compares $|\beta|$ to v and outputs its verdict accordingly. Q.E.D.

Notice that at several points in the proof we took advantage of the fact that T_n is nonuniform. v is hardwired into T_n, as are $\bar{r}, \bar{m}_1, \ldots, \bar{m}_{n^k}$. In fact, most of these uses of nonuniformity could be replaced by polynomial size Monte Carlo experiments. However, T_n must be nonuniform since A_n and B_n are nonuniform.

Chapter 5

One-Pass Scenarios

In this chapter we present the proper definitions for one-pass cryptography, and then go on to show that these definitions are all equivalent to one another. (They are *not* equivalent to the three-pass definitions.) These definitions are all considerably more complicated than the analogous definitions for the three-pass scenario.

5.1 GM-security (1-pass)

As discussed at the beginning of Chapter 3, for a one-pass cryptosystem, we must change from requiring security "for all messages m," to requiring security for every message m that is efficiently computable on input the encryption algorithm alone. In order to do this, we introduce an adversary called a message finder.

A *message finder* is a family of polynomial-size probabilistic circuits $F = \{F_n(\cdot)\}$ each of which takes the description of an encryption algorithm as its input and has two messages of length n as its output. Intuitively, on input E, F_n tries to find m_0 and m_1 such that it's easy for a fellow adversary (a line tapper) to distinguish encryptions of m_0 from encryptions of m_1.

Definition Let C be a public-key cryptosystem. C is *GM-secure (one-pass)* if for all message finders F, line tappers T, and $c > 0$, for all sufficiently large n,

$$\begin{aligned}\Pr(T_n(E, m_0, m_1, \alpha) = m \mid E \leftarrow \mathcal{C}(1^n); m_0, m_1 \leftarrow F_n(E);& \\ m \leftarrow \{m_0, m_1\}; \alpha \leftarrow E(m)) \;\leq\;& \frac{1}{2} + n^{-c}.\end{aligned} \tag{5.1}$$

5.2 Semantic Security (1-pass)

To change the definition of semantic security to fit the one-pass scenario, we need to introduce something like the message finders of the previous section. For semantic security, however, we're concerned not with finding two "weak" messages, but rather with the probability distribution of the entire message space. Thus our second adversary will not pick out particular messages, but instead set the probability distribution of the message space. Furthermore, we now explicitly give the other adversary a description of that probability distribution.

A *message space enemy* is a family of polynomial-size probabilistic circuits $B = \{B_n(\cdot)\}$. Each B_n takes the description of a encryption algorithm as its input, and outputs the description of a probabilistic Turing machine $N()$. N outputs elements of $\{0,1\}^n$ with some probability distribution.

As in the three-pass definition, we let V be any set and let $\mathcal{F} = \{f_n^E : M_n \rightarrow V \mid E \in [\mathcal{C}(n)]\}$ be any set of functions. Again set p_N^E to be the probability of the most probable value for $f(m)$; set $p_n^E = \max\{\sum_{m \in f_n^{E^{-1}}(v)} \Pr_n(m) \mid v \in V\}$.

Definition Let $\mathcal{C}$ be a public-key cryptosystem. $\mathcal{C}$ is *semantically secure* if for every message space enemy B, family of polynomial-size probabilistic circuits $A = \{A_n(\cdot,\cdot,\cdot)\}$, and $c > 0$, for all sufficiently large n

$$\begin{aligned}\Pr(A_n(E, N, \alpha) = f_n^E(m) \mid E \leftarrow \mathcal{C}(1^n);\ N \leftarrow B_n(E);& \\ m \leftarrow N();\ \alpha \leftarrow E(m)) \;<\;& p_n^E + \frac{1}{n^c}.\end{aligned} \tag{5.2}$$

5.3 Y-security (1-pass)

The changes that must be made to the definition of Y-security are completely analogous to the changes we made to the definition of semantic security.

5.4 Equivalence

The proofs that the three definitions of security are all equivalent are quite similar to the proofs for the three-pass case. Here we will redo only the proof of the easiest of the four implications, semantic security implies GM-security. This proof shows the additional details that must be taken into consideration when working with the one-pass scenario.

Theorem 4 *GM-security (1-pass), semantic security (1-pass), and Y-security (1-pass) are all equivalent.*

Proof that semantic security (1-pass) implies GM-security (1-pass): We will, as usual, prove the contrapositive. Let $\mathcal{C}$ be a public-key cryptosystem that is not GM-secure. We know that there exist a message finder family of circuits $F = \{F_n\}$ and a line tapper family of circuits $T = \{T_n\}$. We will use the F_n as subroutines (circuit components to be precise) for building our message space enemy circuits and then use the T_n to do the distinguishing.

Our message space enemy, B_n, on input an encryption algorithm $E \in [\mathcal{C}(1^n)]$, runs F_n with input E. F_n outputs two messages, $m_0, m_1 \in \{0,1\}^n$. B_n outputs the design of a Turing machine $N()$ such that

$$N \text{ outputs } \begin{cases} m_0 & \text{with probability } 1/2 \\ m_1 & \text{with probability } 1/2 \end{cases}$$

An adversary A who uses T_n as a subroutine can distinguish encryptions of m_0 from encryptions of m_1. In other words, on the message space defined by the output of $N()$, A can compute the function $f(m) = m$ (with probability greater than at random) given only an encryption of m. A gets

E and α where either $\alpha \in [E(m_0)]$ or $\alpha \in [E(m_1)]$. However, T_n also requires m_0 and m_1 as inputs. A can obtain that information for T_n simply by running N a few times.

Formally, since $\mathcal{C}$ is not GM-secure we know that there exists a $c > 0$ such that for infinitely many n

$$\begin{aligned} \Pr(T_n(E, m_0, m_1, \alpha) = m \mid E \leftarrow \mathcal{C}(1^n); \\ m_0, m_1 \leftarrow F_n(E);\ m \leftarrow \{m_0, m_1\}\,; \\ \alpha \leftarrow E(m)) \ > \ \frac{1}{2} + n^{-c}. \end{aligned} \tag{5.3}$$

For those n, Equation 5.3 in fact says that Pr(A computes $f(m) = m$ correctly) $> \frac{1}{2} + n^{-c}$. Q.E.D.

Appendix A

Proof of Lemma 1

In this appendix we provide a proof of Lemma 1.

Let X be a random variable having a binomial distribution with n trials and probability of success p. For $0 \leq p, \alpha \leq 1$, define

$$f(p, \alpha) = \alpha \log \frac{\alpha}{p} + (1 - \alpha) \log \left(\frac{1 - \alpha}{1 - p} \right). \tag{A.1}$$

Chernoff [2] gave the following upper bound for estimating $\Pr(X \leq \alpha n)$.

Theorem 5 (Chernoff) *For* $0 \leq \alpha \leq p$, *we have* $\Pr(X \leq \alpha n) \leq e^{-f(p,\alpha)}$.

The following useful fact was pointed out to me by Ravi Boppana [1].

Theorem 6 *For* $0 \leq \alpha \leq \frac{1}{2} \leq p \leq 1$, *we have* $f(p, \alpha) \geq 2(p - \alpha)^2$.

Proof We first compute

$$\frac{\partial f}{\partial \alpha} = \log \frac{\alpha}{p} - \log \left(\frac{1 - \alpha}{1 - p} \right). \tag{A.2}$$

Notice that for all $p \in [0, 1]$, $f(p, p)$ and $\frac{\partial f}{\partial \alpha}(p, p)$ are equal to zero. Taylor's theorem (see, for instance, [9]) states that for any "nice" real function g defined on $[\alpha, \beta]$,

$$\exists x \in [\alpha, \beta] : \; g(\beta) = \sum_{k=0}^{n-1} \frac{g^{(k)}(\beta)}{k!} (\alpha - \beta)^k + \frac{g^{(n)}(x)}{n!} (\alpha - \beta)^n. \tag{A.3}$$

Thus, taking Equation A.3 with $n = 2$ we see that there is a $x \in [\alpha, p]$ such that

$$f(p,\alpha) = \frac{\partial^2 f}{\partial \alpha^2}(p,x)\frac{(\alpha - p)^2}{2}. \tag{A.4}$$

Differentiating f again, we see that $\frac{\partial^2 f}{\partial \alpha^2}(p,x) = \frac{1}{x(1-x)}$, so

$$\begin{aligned} f(p,\alpha) &= \frac{1}{2x(1-x)}(p-\alpha)^2 && \text{(A.5)} \\ &\geq \left[\min_{x\in[\alpha,p]}\left(\frac{1}{2x(1-x)}\right)\right](p-\alpha)^2. && \text{(A.6)} \end{aligned}$$

The function $x \mapsto \frac{1}{2x(1-x)}$ is increasing on $[1/2, 1]$. Thus inequality A.6 still holds if we take $x = 1/2$ and rewrite the right hand side as $2(p-\alpha)^2$, which completes our proof. Q.E.D.

Recall that Lemma 1 states that for $0 \leq \alpha \leq 1/2 \leq p \leq 1$,

$$\Pr(X \leq \alpha n) \leq e^{-2(p-\alpha)^2 n}. \tag{A.7}$$

Inequality A.7 follows as an immediate corollary of Theorems 5 and 6.

Bibliography

[1] R. Boppana, private communication.

[2] H. Chernoff, "A measure of asymptotic efficiency for tests of a hypothesis based on the sum of observations," *Annals of Mathematical Statistics* 23 (1952).

[3] B. Chor, S. Goldwasser, S. Micali, and B. Awerbuch, "Verifiable Secret Sharing and Achieving Simultaneity in the Presence of Faults," *Proceedings of the 23rd Annual Symposium on Foundations of Computer Science*, 1985, pp. 383–95.

[4] W. Diffie and M. E. Hellman, "New direction in cryptography," *IEEE Trans. Inform. Theory,"* Vol. IT-22, No. 6, 1976, pp. 644–54.

[5] S. Goldwasser and S. Micali, "Probabilistic Encryption," *JCSS*, vol. 28, No. 2, April 1984, pp. 270–99.

[6] S. Goldwasser, S. Micali, and R. Rivest, "A Digital Signature Scheme Secure Against Adaptive Chosen Message Attack," to appear.

[7] C. Rackoff, private communication.

[8] R. Rivest, A. Shamir, and L. Adleman, "A Method for Obtaining Digital Signatures and Public-Key Cryptosystems," *Communications of the ACM*, Vol. 21, 1978, pp. 120–26.

[9] W. Rudin, *Principles of Mathematical Analysis*, 3rd ed., McGraw-Hill, New York, 1976.

[10] C. E. Shannon, "Communication theory of secrecy systems," *Bell System Tech. J.*, 28 (1949), pp. 656–749.

[11] A. C. Yao, "Theory and Applications of Trapdoor Functions (extended abstract)," *Proceedings of the 23rd Annual Symposium on Foundations of Computer Science*, 1982, pp. 80–91.