From semantic security to chosen ciphertext security

by

Sahnghyun Cha

A thesis submitted to the graduate faculty

in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

Major: Computer Science

Program of Study Committee:
Clifford Bergman, Major Professor
David Fernandez-Baca
Pavan Aduri

Iowa State University

Ames, Iowa

2010

TABLE OF CONTENTS

ABSTRACT

A chosen ciphertext attack against the RSA encryption standard PKCS#1 v1.5 was introduced by Daniel Bleichenbacher at Crypto '98. This attack was the first example where an adaptive chosen ciphertext attack is not just a theoretical concept but a practical method to crack a semantically secure encryption scheme.

This paper reviews the notion of the semantic security which was believed to be secure enough in reality and the reason for which this belief was denied. The paper also presents a demonstration of the Bleichenbacher's attack by using a simplified version of PKCS#1 v1.5 format.

1 Introduction

For many years, the chosen-ciphertext security was a theoretical concept in cryptography. Before 1998, when the Cramer-Shoup cryptosystem [6] was introduced, secure encryptions against a chosen-ciphertext attack were often impractical or not provably secure. Most of the attacks were based on unintended software bugs instead of cryptanalytic methods. Thus, using an encryption scheme which provides the highest level security was probably not a priority for the engineers.

In public key cryptosystem, semantically secure [9] encryption, such as PKCS#1 v1.5 + RSA, was normally believed to be a *secure envelope*. Since there is no way that we can gain any information about the letter from the securely sealed envelope, this notion of security seemed quite enough to use. However, Daniel Bleichenbacher has proved that one of the most popular encryptions, which is semantically secure, was vulnerable against the chosen-ciphertext attack. The attack was a proper cryptanalytic attack, and it triggered people to realize the importance of the chosen-ciphertext security.

In this paper, we briefly review the concept of cryptanalytic attacks and the notion of semantic security. Then, we explain Bleichenbacher's attack in detail along with the idea of the attack. A small example of Bleichenbacher's attack will also be demonstrated which does not require a computer to perform the attack.

2 Definition

2.1 Possible Cryptanalytic Attacks

While transferring the data through the network, one must always assume that there is always an anonymous person with malicious purposes who eavesdrops on the conversation. The goal of cryptography is to prevent attacks from this adversary. This task may not seem very hard to achieve since there are various problems for which there is no known effective solution, and we can convert a plaintext into an unreadable data by using them. However, we should not expect that the adversary is merely a passive eavesdropper. For example, maybe

he has an ability to obtain some plaintexts and corresponding ciphertexts, or perhaps he has access to an encrypting machine without knowing the encryption key. Therefore, when we evaluate any cryptographic scheme as a solid tool, they must be secure enough against every possible cryptanalytic attack. These attacks are normally classified as follows:

1. *Ciphertext-only attack* is an attack in which the adversary is only an eavesdropper. In a ciphertext-only attack, the adversary may observe one or more ciphertexts and attempts to determine corresponding plaintexts or partial information about the plaintexts. This is the most fundamental of all attacks.

2. *Known-plaintext attack* is an attack in which the adversary is skillful enough to collect one or more pairs of plaintexts and corresponding ciphertexts. However, the adversary cannot choose the plaintexts or ciphertexts to be encrypted or decrypted. The adversary may be able to understand some relationship between plaintexts and ciphertexts, so he can decrypt a challenge ciphertext c based on this information. In worst case, this information may be used to identify the key, either encryption or decryption, of the cryptosystem.

3. *Chosen-plaintext attack* is an attack in which the adversary has the ability to access an encryption oracle. In this model, the adversary can choose one or more plaintexts and can obtain corresponding ciphertexts through the encryption oracle. The adversary does not have to understand the exact mechanism by which the plaintext is being encrypted. The goal of the adversary is to determine the plaintext of a challenge ciphertext c by using the ability to access an encryption oracle.

4. *Chosen-ciphertext attack* is an attack in which the adversary has the ability to access a decryption oracle. In this case, the adversary can obtain the decryption of one or more ciphertexts which he has chosen. Again, the decryption is computed through the decryption oracle, so we assume that the adversary does not have any understanding of the decryption mechanism. We also assume that the adversary cannot use this decryption oracle to directly decrypt the challenge ciphertext c.

2.2 Security Notions For Public Key Cryptosystem

Before we start considering the security notions, we first define the syntax of a public key encryption scheme, which we will use for the rest of this section.

Definition 1 *A public key encryption scheme is given by a triple of algorithms,* $\Pi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$*, such that*

- $\mathcal{K}$*, the key generation algorithm, is a probabilistic algorithm that takes a security parameter* $k \in \mathbb{N}$ *as an input and outputs a pair of keys* (k_1, k_2) *which are a public and secret key respectively,*

- $\mathcal{E}$*, the encryption algorithm, is a probabilistic algorithm that takes* k_1 *and a plaintext* m *as an input and outputs a ciphertext* c*,*

- $\mathcal{D}$*, the decryption algorithm, is a deterministic algorithm that takes* k_2 *and a ciphertext* c *as an input and outputs a plaintext* m*,*

where $\mathcal{D}_{k_2}(\mathcal{E}_{k_1}(m)) = m$ *for all* m *and* (k_1, k_2)*.*

2.2.1 Semantic Security

To determine the security level of a public key cryptosystem, one would set a standard notion as a potential goal of this system. Goldwassar and Micali [9] first introduced a notion of security for a public key cryptosystem, named semantic security. To achieve this level of security, a system should not leak any partial information about the plaintext of a given corresponding ciphertext. To be more specific, let f be a polynomial time computable function and $m \in \{0,1\}^n$ be a plaintext. Then, the public key cryptosystem is semantically secure if the probability that an adversary can guess $f(m)$ in polynomial time given the ciphertext is *almost* the same as the probability of guessing $f(m)$ without the ciphertext. We say that these two probabilities are *almost* the same if the difference is less than $1/p(n)$ for every polynomial p on input n. Thus we can conclude that in a semantically secure public key cryptosystem,

whatever information a polynomial time bounded adversary can compute about the plaintext given the ciphertext is also computable even without the ciphertext.

This idea can be formally verified through the following experiment called polynomial-time Indistinguishability (a.k.a. polynomial security [9]) experiment against an eavesdropper. Let $\Pi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ be the public key encryption scheme and let $\mathcal{A}$ be a polynomial-time adversary. For any $n \in \mathbb{N}$ as a security parameter:

Experiment $\mathrm{Exp}^{\mathrm{eav}}_{\Pi,\mathcal{A}}(n)$ **:**

1. The adversary $\mathcal{A}$ outputs a pair of plaintexts (m_0, m_1). Both m_0 and m_1 are n-bit string.
2. Choose a key $k \leftarrow \mathcal{K}(1^n)$ and a random $b \in \{0, 1\}$. Information of k and b cannot be leaked to $\mathcal{A}$.
3. Compute an encryption $c \leftarrow \mathcal{E}_k(m_b)$.
4. The adversary $\mathcal{A}$ receives c and guesses $b' \in \{0, 1\}$.
5. Return 1 if $b' = b$. Otherwise, return 0.

If the adversary outputs b' at random, the probability that $\mathrm{Exp}^{\mathrm{eav}}_{\Pi,\mathcal{A}}(n) = 1$ should be $1/2$. This cryptosystem has indistinguishable encryptions against an eavesdropper if any adversary whose computation power is no better than a probabilistic polynomial-time Turing machine cannot guesses $b' = b$ with significantly higher probability than $1/2$; that is, the probability that an adversary guesses $b' = b$ is, again, *almost* $1/2$.

Note that in a public key cryptosystem, an eavesdropper can attempt a chosen plaintext attack(CPA) by simply encrypting any plaintext with the given public key. Therefore, if a public key cryptosystem is indistinguishable against an eavesdropper, then this system is also indistinguishable under CPA (i.e. $\Pr[\mathrm{Exp}^{\mathrm{CPA}}_{\Pi,\mathcal{A}}(n) = 1] = \Pr[\mathrm{Exp}^{\mathrm{eav}}_{\Pi,\mathcal{A}}(n) = 1]$.) Moreover, this probability never reaches exactly to $1/2$ no matter how large the message space is because every public key cryptosystem is vulnerable to an adversary performing chosen plaintext attack by computing every possible probabilistic encryption of each plaintext in the message space.

In [9], the definition of semantic security was only given against CPA. In fact, the equivalence between semantic security(SS) and indistinguishability under CPA(IND-CPA) was proven (IND-CPA $\rightarrow$ SS by [9], SS $\rightarrow$ IND-CPA by [11].)

In order to make a formal definition of Indistinguishability against CPA attack model, we consider the following two definitions:

Definition 2 *A function $f : \mathbb{N} \rightarrow \mathbb{R}, f(n) \geq 0$ for $n \in \mathbb{N}$ is* negligible *if for every polynomial p on input n, $f(n) < 1/p(n)$ holds.*

Definition 3 (Indistinguishability-CPA) *A public key encryption scheme $\Pi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ has an indistinguishable encryption under CPA if for every polynomial-time adversary $\mathcal{A}$, there is a negligible function f such that*

$$\Pr[\mathrm{Exp}^{\mathrm{CPA}}_{\Pi,\mathcal{A}}(n) = 1] - \frac{1}{2} \leq f(n).$$

Later on, Watanabe et al [17] have shown that semantic security and indistinguishability against other attack models are also equivalent by using extended definition of semantic security based on the security framework given by Bellare et al [1].

2.2.2 Chosen Ciphertext Attack

In a public key cryptosystem, every eavesdropper can perform a chosen plaintext attack as well as known plaintext attack by using the public key as a key for the encryption oracle. Thus, now we consider a final attack model: Chosen ciphertext attack(CCA). There are two different types of CCA model. In chronological order, these are named non-adaptive chosen ciphertext attack(CCA1) and adaptive chosen ciphertext attack(CCA2). Recall that an adversary under chosen-cipher attack has an ability to access a decryption oracle so that he can decrypt the ciphertext that he has chosen. The difference between CCA1 and CCA2 lies in whether or not this ability is restricted by the challenge ciphertext c. Under CCA1, due to Naor and Yung [12], the adversary can access the decryption oracle only before he receives the challenge ciphertext c. That is, the adversary's queries to a decryption oracle cannot be adapted to the challenge c. This is the reason why the term 'non-adaptive' is appended in the name of CCA1. Under

CCA2, due to Rackoff and Simon [13], the adversary has an ability to access a decryption oracle, and now he can use this decrypting function even after he obtains the challenge ciphertext c. The only restriction to this model is that an adversary cannot send c to the decryption oracle to verify the real message. In this model, an adversary can decide queries for the decryption oracle based on c, which is an extremely strong notion of security.

3 Semantically Secure Encryption

Semantically secure encryption is often described as a secure envelope. Once you seal the envelope, no attacker can gain any information on the content of the letter unless an authorized receiver breaks the seal. Assume that you wrote an important message for your parents and asked your friend to seal it with a secure envelope and to place it on the desk. Since all secure envelopes appear identical to each other, it is impossible to distinguish which one contains the letter you wrote to your parents if there was another sealed envelope on the desk. Even though you knew how to seal the envelope for yourself, this ability would not be of any help in finding your letter. The only way to distinguish the message is to unseal the envelopes. This illustrates the indistinguishability under chosen plaintext attack (IND-CPA), which is identical to the semantic security.

Since the semantic security looks to ba a fairly strong notion of security in public key encryption, IND-CPA might seem too difficult to achieve when constructing a cryptosystem. Then, why do we have to endeavor to accomplish semantic security? Throughout this section, I provide an answer to this question by showing vulnerabilities of a particular cryptosystem which has a certain level of security but is not semantically secure.

3.1 Plain RSA Is Not Semantically Secure

First, we focus on one of the most famous public key algorithms named RSA. The RSA scheme was originally introduced by Ron Rivest, Adi Shamir, and Leonard Adleman [14] in 1978. The basic concept of this scheme is the fact that we believe a factoring problem of a large integer n, where n is the product of large random primes p and q, is computationally hard.

The message space per single encryption is upper bounded in n, i.e., a block of the message should be an integer between 0 and $n-1$.

The RSA scheme involves a public key and a private key. These keys can be generated through the following procedure:

1. Pick two large primes p and q. Let $n = pq$.
2. Compute $\phi(n) = (p-1)(q-1)$, where ϕ is the Euler's totient function.
3. Choose $e, d \in \mathbb{Z}_{\phi(n)}$ such that $ed \equiv 1 \pmod{\phi(n)}$.
4. Public key: (n, e)
5. Private key: d

Let m be the plaintext, then the encryption of m is

$$c = m^e \bmod n.$$

After receiving the ciphertext c from the sender, the decryption of c can be computed as

$$m = c^d \bmod n.$$

It is believed that the private key d cannot be determined as long as $\phi(n)$ is kept secret, and it is computationally infeasible to compute $\phi(n)$ without factoring n. Thus, an attacker cannot illegally decrypt the ciphertext in a reasonable amount of time. However, even though the RSA cryptosystem is assumed to be secure, it is not secure enough to qualify as a secure envelope. There are some important weak points of the RSA encryption that we should focus on.

First of all, a ciphertext encrypted by the RSA leaks some of the information in the message. Clearly, any deterministic encryption would not be able to overcome this weakness because there is only a unique encoding for each message. This could be a fatal defect for the public key encryptions as they are always vulnerable to the dictionary attack – an attacker can construct a dictionary which contains encryptions of every possible plaintext in the message space. Assume

the case where an eavesdropper already knows that the message is the 3-digits of a credit card security code. Then, the attacker can undoubtedly create a dictionary filled with 1000 lines of encryptions and figure out the original security code.

Another known drawback of the plain RSA is its malleability. Any eavesdropper can construct the encryption of a transformed plaintext. Let m be the original plaintext and c be the encryption of the plaintext. Then, an adversary computes

$$c' = c \times t^e \equiv m^e t^e \equiv (mt)^e \bmod n$$

for any t. If an adversary happens to know that this modification has a proper meaning, then he can encrypt a distorted message even though he does not know the plaintext m. For instance, setting $t = \frac{1}{2}$ instantly reduces m into half, and this kind of attack could be very critical if the plaintext was a numerical value such as price, date, or time.

Intuitively, the secure envelope should not allow malleable encryption as a sealed envelope cannot be opened without breaking the seal. Moreover, the non-malleability is also an important concept of security since it is shown by [1] that non-malleability under CPA (NM-CPA) implies IND-CPA as well. In other words, a non-malleable public key cryptosystem is semantically secure. As we now know that this attack can be a practical threat, NM-CPA should be achieved along with IND-CPA when designing a secure encryption scheme.

3.2 Random Padding Scheme

We now describe a random padding scheme which will reinforce the security level of encryption schemes in order to approach the concept of secure envelope. As we already know from the last section, it is impossible to acheive semantic security by deterministic encryption such as plain RSA because of the limited message space which would cause the dictionary attack. To avoid the dictionary attack, we need to add randomness somewhere in the encryption process. One quick solution for this problem is to use a randomized padding scheme. As an easy example, we encrypt a message m along with a 10-bit random string r at the end of the message. With the help of the random string r, a ciphertext would become indistinguishable to the eavesdropper if the sender encrypts the same message twice and sends both ciphertexts to

the receiver. After the decryption of the ciphertext, the plaintext m can be restored by cutting off the last 10-bits string of the decrypted message.

In 1993, RSA laboratories introduced the RSA Encryption Standard (PKCS#1) version 1.5 which explains how to securely encrypt messages with RSA scheme. One of the key subjects of the standard is the message block formatting method as a part of the encryption process. Let (n, e) be a public key for the RSA cryptosystem. Then, for a block type BT, a padding string PS, and the message D, the encryption block format EB is defined by the standard as

$$EB = 00 \parallel BT \parallel PS \parallel 00 \parallel D.$$

The operator '$\parallel$' is used for the concatenation between two strings. A block type would be '02' in the case of public key operation, which we will focus on throughout this paper, and a padding string should be a non-zero random $k-3-|D|$-byte integer assuming the modulus n is a k-byte integer. After the encryption block EB is converted to an integer, the leading 00-byte ensures that the whole block is less than the modulus n. Moreover, the random padding string should be at least 8-bytes long in order to prevent an adversary from trying every possible encryption block in the reasonable time.

Then, how secure is the RSA encryption with the PKCS#1 v1.5 padding scheme? If we carefully construct the message D[1], it is generally believed to be semantically secure, although to date, there is no proof of this fact. It is not known whether the scheme is secure against CPA.

3.3 Beyond Semantic Security

Figuring out how high the level of security should be guaranteed for secure encryption is not an easy task. In fact, semantic security seems to be fairly difficult to achieve. Then, is it enough to set semantic security as a practical standard of public key cryptosystem security? Intuitively, the CCA model may appear too impractical to be considered a real threat. For example, it is very unlikely to imagine a real scenario where an adversary uses the CCA model

[1] [5] showed that the PKCS#1 v1.5 encryption is vulnerable to the chosen plaintext attack if the plaintext ends with sufficiently many zeroes.

to attack the system. How could an adversary have the ability to access the decryption oracle? Even if he could, why would he not use this oracle to decrypt the challenge ciphertext? Because of these doubts, the CCA model was considered a theoretical concept that would never become a potential threat. However, it has been proven that achieving CCA security is actually a very important matter.

In 1998, Daniel Bleichenbacher [4] showed that the PKCS#1 v1.5 can be attacked by a CCA level adversary if we do not take the definition of the CCA model too literally. Instead of accessing the decryption oracle, the attack adaptively collects the partial information returned from the oracle whether the ciphertext is a valid PKCS#1 encryption or not. Since an adversary would not get a full plaintext from the oracle, this attack is also called a partial chosen ciphertext attack.

We first outline how the Bleichenbacher's attack works by using a hypothetical storyline. In the story, your elder sister Alice asked you to deliver a sealed bottle to Bob, her boyfriend. You knew that there was some kind of liquid inside of the bottle, but since the bottle was completely opaque, you never knew what liquid was in it just by looking at the bottle. Instead, you somehow found out a way to inject another fluid through the cork into the container. Then, you handed over the bottle containing the mixed fluid to Bob. After Bob broke the seal, he could verify the color of the liquid and decide whether the bottle was actually sent by Alice. Since you manipulated this gift, Bob may or may not be upset with you for delivering the wrong bottle. Depending on Bob's attitude, you may be able to reduce the range of possible candidates for this secret liquid. For example, if you decided to inject some black ink into the bottle but Bob did not notice this, then you could exclude light-colored liquids such as water or milk. You could also use an indicator which would cause a chemical reaction to intentionally change the color of the liquid. If you repeat this experiment several times, it would be possible to guess the right answer at some point.

Now we explain the attack in detail. Let (n, e) be a public key for the RSA cryptosystem, where n is a k-byte modulus and e is an encryption exponent. The ciphertext c is m^e mod n, where m is a plaintext. We say the ciphertext c is PKCS comforming if m, a decryption

of c, has the PKCS#1 format. This means that the first two bytes of m are '00' and '02' respectively. Therefore, if the plaintext m is PKCS conforming, then

$$2B \leq m \leq 3B - 1$$

where $B = 256^{k-2}$. Assume that the adversary tries to find the plaintext m by using the corresponding ciphertext and the decryption oracle. The oracle does not provide a full decryption of a query ciphertext but returns the boolean value whether the ciphertext is PKCS conforming or not. Essentially, the adversary chooses integers s, and send $c' \equiv cs^e \pmod{n}$ to the oracle until he finds sufficiently many PKCS conforming ciphertexts c'. According to [4], nearly 2^{20} chosen ciphertexts will be required to derive m, which can be done practically. The process of the attack can be divided into three steps. We define a set of intervals M_i such that m is included in one of these intervals after a proper s_i has been found at the i'th stage.

Step 1: For initialization, let $[u, v] = [2B, 3B - 1]$. Since m itself is PKCS conforming, we have

$$M_0 = \{[u, v]\}$$

First, we set $i = 1$ and search for the smallest integer $s_1 > 1$, such that the encryption of $s_1 m \pmod{n}$ is PKCS conforming. Instead of searching for every integer, we can narrow down the search space with the lower bound on s_1. Since $s_1 m$ is bigger than v for any $s_1 > 1$, we have the following bound of $s_1 m$ for some positive integer t.

$$tn + u \leq s_1 m \leq tn + v$$

Thus, we can start searching from $s_1 \geq \lceil \frac{n+u}{v} \rceil \geq \lceil \frac{n}{3B} \rceil$. We have the lower bound of the search space

$$s_1 \geq \left\lceil \frac{n}{3B} \right\rceil. \tag{1}$$

After we find PKCS conforming $s_1 m$, we can update the set of intervals M_1 as

$$M_1 = \bigcup_t \left\{ [u, v] \cap \left[\frac{tn + u}{s_1}, \frac{tn + v}{s_1} \right] \right\} \tag{2}$$

for all t, such that $\frac{us_1 - v}{n} \leq t \leq \frac{vs_1 - u}{n}$.

Step 2: After the previous step, M_1 most likely contains multiple intervals. In this step, we narrow down these intervals and find the one which actually contains m.

Increase i by 1 as we move on to the next stage. Now we search for the smallest s_i, such that $s_i > s_{i-1}$ and the encryption of $s_i m$ is PKCS conforming. After finding s_i, the set M_i can be computed as

$$M_i = \bigcup_{a,b,t} \left\{ [a, b] \cap \left[\frac{tn + u}{s_i}, \frac{tn + v}{s_i} \right] \right\} \tag{3}$$

for all intervals $[a, b] \in M_{i-1}$ and $\frac{as_i - v}{n} \leq t \leq \frac{bs_i - u}{n}$.

We repeat this step until M_i contains only one interval.

Step 3: Step 3 will be repeated until M_i indicates a unique integer. We search for the new s_i, such that $s_i \approx 2s_{i-1}$ and the encryption of $s_i m \pmod{n}$ is PKCS conforming. Recall that the search space is $\left\lceil \frac{tn+u}{v} \right\rceil \leq s_i \leq \left\lfloor \frac{tn+v}{u} \right\rfloor$ for some integer t. However, if there is no such t for s_i, we discard s_i and find a new one. The length of M_i is less than $\frac{v-u}{s_i} \approx \frac{B}{s_i}$, and the expected magnitude of s_i is about double the value of s_{i-1}. Therefore, there will be an end point to this step with sufficiently many iterations.

3.4 A Watered Down Bleichenbacher's Attack

In this section, we demonstrate a real example of Beleichenbacher's attack by using a simplified version of PKCS#1 v1.5. Let P be the probability that a randomly chosen integer m is PKCS conforming. According to [4], we have

$$0.18 \cdot 2^{-16} < P < 0.97 \cdot 2^{-8}$$

if we assume that a modulus n is a 512-bit integer. Basically, we need to find a PKCS conforming integer s_i at random in order to perform Bleichenbacher's attack against RSA encryption with PKCS#1 v1.5 padding scheme. It seems that demonstrating or examining this attack by hand is almost impossible. Thus, by increasing the probability P, we can actually execute the attack and see how this attack finds the plaintext m.

3.4.1 A Simplified Version of PKCS#1 v1.5

To increase the probability P, we shall use the following block format EB which is very similar to the original PKCS#1 v1.5:

$$EB = 02 \parallel PS \parallel D.$$

Let the modulus n be a k-byte integer. Then, we need to use the modulus $n \geq 3B$ where $B = 256^{k-1}$. Moreover, since there is no 00-byte between the random padding string and the data, we assume that we only use a fixed size for the padding string. We call this format a simplified PKCS#1.

Let us say the plaintext m is PKCS conforming if the format of m is a simplified PKCS#1. In fact, every string in which the first byte is equal to '02' is PKCS conforming, Therefore, P becomes much larger than before. To be more specific, the probability that the first byte is '02' is $\frac{2^{8(k-1)}}{n}$. Thus, for a random integer $0 \leq m < n$, P is larger then $\frac{1}{256}$.

3.4.2 Exercise

- Plaintext(1 byte): 55_{16}

- Random padding(1 byte): AC_{16}

- Simplified PKCS#1 m(3 byte): $02AC55_{16} = 175189$

Assume that the above information is secret.

Let $n = 13102589$, $e = 13$, and a ciphertext $c = m^e \% n = 3537984$. We want to reveal m by using the Bleichenbacher's attak. We shall define a set of intervals M_i as we did in the previous

section.

Initial step: $i = 0$

Since m is PKCS conforming, $2 \cdot 256^2 \leq m \leq 3 \cdot 256^2 - 1$. Let $u = 2 \cdot 256^2 = 131072$ and $v = 3 \cdot 256^2 - 1 = 196607$, then we have

$$M_0 = \{[u, v]\}$$

Step 1: $i = 1$

We need to find $s_1 > 1$, such that the encryption of $s_1 m \pmod{n}$ is PKCS conforming. By (1), we start searching from the lower bound of $s_1 \geq \left\lceil \frac{n}{3 \cdot 256^2} \right\rceil = 67$.

During the exercise, we check PKCS conformability by simply computing $s_1 m \pmod{n}$ instead of using the decryption oracle. We start from the lower bound and increase by one for each attempt. This may be time-consuming to some degree, but still practical even by hand with a simple calculator. After several attempts, we found our first integer, $s_1 = 300$. A set of intervals M_1, therefore, can be updated by using the equation (2) as

$$M_1 = \bigcup_t \left\{ [u, v] \cap \left[\frac{u + tn}{300}, \frac{v + tn}{300} \right] \right\}$$

for all t, such that $\frac{300u - v}{n} \leq t \leq \frac{300v - u}{n}$. As both 3 and 4 are in t, we have two intervals

$$M_1 = \{[131463, 131681], [175139, 175356]\},$$

and this ends step 1.

Step 2: $i = 2$

The next smallest integer that we can find as s_2 is 375.

Since $|M_1| = 2$, we need to compute M_2 for each interval. We start with the first interval $[131463, 131681]$ in M_1. Let $a = 131463$ and $b = 131681$ to use them with the equation (3). However, the bound for t in this case indicates that t cannot be an integer: $3.75 \leq t \leq 3.76$. Thus, we discard this interval and move on to the next one. Let $a = 175139$ and $b = 175356$.

By (3), we can determine a proper integer $t = 5$ from the bound we have for t. We narrow down M_2 from M_1 to finish step 2.

$$M_2 = \left\{ [a, b] \cap \left[\frac{5n + u}{375}, \frac{5n + v}{375} \right] \right\} = \{[175139, 175225]\}$$

Step 3: $i = 3, 4, 5$

This step will be repeated until the set M_i contains a unique integer. The next s_i can be found somewhere in the neighborhood of $2s_{i-1}$. Therefore, 750 would be a good place to start searching for the next s_3 candidate. Fortunately, we found $s_3 = 749$ which narrows down M_3 to $[175139, 175196]$. By repeating this procedure, s_4 and s_5 can also be found as follows:

$$s_4 = 1347 \rightarrow M_4 = \{[175188, 175196]\}$$

$$s_5 = 2544 \rightarrow M_5 = \{[175188, 175190]\}$$

Our objective is to make the interval uniquely indicate $m = 175189$ at some point. Hopefully, M_6 or M_7 will achieve this goal. In order to narrow down the interval M_5, however, we need a much bigger s_6 than just twice the size of s_5. Since the interval M_5 very closely approaches m, it seems quite enough for the demonstration of Bleichenbacher's attack.

4 Provably Secure Encryption Schemes Against CCA2

Provably secure encryption simply means that it has a mathematical proof to guarantee its security against certain types of attacks. For instance, one-time pad has been proven to be a perfectly secure encryption based on the notion of perfect secrecy by Shannon, even though it is not usually considered as a practical encryption scheme in real life. Throughout the previous section, we focused on the fact that indistinguishability under chosen-plaintext attack, or semantic security, is not enough. After Daniel Bleichenbacher had shown that PKCS#1 v1.5, which was believed to be a secure encryption, was actually crackable by a CCA2 level adversary, the CCA security has increasingly become a practical issue. Thus, we need encryption schemes which are secure against CCA2, and it would be even better if these schemes are provably secure.

In fact, there are good encryption schemes that provide CCA2 security. One of the most popular schemes we use at present is the Optimal Asymmetric Encryption Protocol(OAEP). This protocol was first introduced by [3] in 1994 and was adopted as PKCS#1 v2.0 in 1998. OAEP is a random padding scheme similar to the one we discussed in the previous section. Unlike PKCS#1 v1.5, it provides resistance against CCA2 if it is used with the RSA encryption scheme. For many years, OAEP was believed to be secure not only with RSA but also with any kind of one-way permutation scheme. However, this belief was denied by Victor Shoup[15], as he found a significant gap in the OAEP security proof. This gap could not be filled, but RSA+OAEP has remained to be a provably secure encryption([15], [8]).

In the OAEP security proof, the scheme is secure against CCA2 in the random oracle model [2]. In the random oracle model, the hash function or the random number generator used in the encryption scheme is treated as a real random oracle. Even though this model is often very useful in security proofs, one may think that this assumption is too strong since the implementation of randomness is impossible in the real world.

The Cramer-Shoup public key encryption scheme [6] is another provably secure cryptosystem against CCA2. This scheme is provably secure assuming that the Diffie-Hellman Decision Problem cannot be solved in polynomial time and that the hash function is Collision-Resistant[2]. One of the strong points of this scheme is that the security proof does not require the random oracle model. As the assumption used in Cramer-Shoup encryption is much weaker than the random oracle model, it makes the user think that the scheme is more reliable than the other.

Still, a number of encryption schemes are being designed to provide CCA2 security with or without the random oracle model. Furthermore, many researchers are presenting security proofs under lighter assumptions for existing encryption schemes. Ever since the emergence of practical attacks against semantic security, active research for higher level security is no longer just a theoretical matter.

[2]The hash function H is Collision-Resistant if it is infeasible to find x and y such that $x \neq y$ and $H(x) = H(y)$.

BIBLIOGRAPHY

[1] M. Bellare, A. Desai, D. Pointcheval and P. Rogaway. Relations among notions of security for public-key encryption schemes. In *Proceedings of Advances in Cryptology - Crypto '98*, Lecture Notes in Computer Science 1462, pp. 26-45, 1998.

[2] M. Bellare and P. Rogaway. Random oracles are practical: a paradigm for designing efficient protocols. In *Proceedings of the First ACM Conference on Computer and Communications Security*, pp. 62-73, 1993.

[3] M. Bellare and P. Rogaway. Optimal asymmetric encryption. In *Advances in Cryptology - Eurocrypt '94*, pp. 92-111, 1994.

[4] D. Bleichenbacher. Chosen ciphertext attacks against protocols based on the RSA encryption standard PKCS. In *Advances in Cryptology - Crypto '98*, Lecture Notes in Computer Science 1462, pp. 1-12, 1998.

[5] J.-S. Coron, M. Joye, D. Naccache and P. Paillier. New attacks on PKCS#1 v1.5 encryption, In *Advances in Cryptology - Eurorypt '00*, Lecture Notes in Computer Science 1807, pp. 369-381, 2000.

[6] R. Cramer and V. Shoup. A practical public key cryptosystem provably secure against adaptive chosen ciphertext attack. In *Advances in Cryptology - Crypto '98*, Lecture Notes in Computer Science 1462, pp. 13-25, 1998.

[7] H. Delfs and H. Knebl. Introduction to Cryptography - Principles and Applications. Springer, 2002.

[8] E. Fujisaki, T. Okamoto, D. Pointcheval and J. Stern. RSA-OAEP is secure under the RSA assumption. In *Advances in Cryptology - CRYPTO '2001*, Lecture Notes in Computer Science 2139, pp. 260-274, 2001.

[9] S. Goldwasser and S. Micali. Probabilistic encryption. *Journal of Computer and System Sciences 28*, pp. 270-299, 1984.

[10] J. Katz and Y. Lindell. Introduction to Modern Cryptography. Chapman & Hall/CRC, 2008.

[11] S. Micali, C. Rackoff and R. Sloan. The notion of security for probabilistic cryptosystems, *SIAM Journal on Computing 17*, pp. 412-426, 1988.

[12] M. Naor and M. Yung. Public-key cryptosystems provably secure against chosen ciphertext attacks, In *Proceedings of the 22nd annual ACM Symposium on Theory of Computing*, pp. 427-437, 1990.

[13] C. Rackoff and D. Simon. Non-interactive zero-knowledge proof of knowledge and chosen ciphertext attack, In *Proceedings of Advances in Cryptology - Crypto '91*, Lecture Notes in Computer Science 576, pp. 433-444, 1991.

[14] R. Rivest, A. Shamir, and L. Adleman. A method for obtaining digital signatures and public-key cryptosystems. In *Comm. of the ACM*, 21:2, pp. 120-126, 1978.

[15] V. Shoup. OAEP Reconsidered. In *Proceedings of Advances in Cryptology '2001*, Lecture Notes in Computer Science 2139, pp. 239-259, 2001.

[16] V. Shoup. Why chosen ciphtertext security matters. Research report RZ 3076 (#93122), IBM REsearch Zurich, 1998.

[17] Y. Watanabe, J. Shikata and H. Imai, Equivalence between semantic security and indistinguishability against chosen ciphertext attacks. In *Proceedings Public Key Cryptography - PKC 2003*, Lecture Notes in Computer Science 2567, pp. 71-84, 2003.