PSEUDORANDOM FUNCTIONS AND CPA SECURITY

BURTON ROSENBERG
UNIVERSITY OF MIAMI

Contents

1. Eavesdropping and Pseudorandomness

The Vernam cipher as proved to be a good starting point for a discussion of encryption. It provides perfect security as long as the key is a never-repeating string of truly random bits. The Shannon definition of perfect security was shown equivalent to an Adversarial Indistinguishability Game.

In Adversarial Indistinguishability Game, a key is chosen secretly and the adversary provides two equal length messages, m_0 and m_1, and receives $c = \mathcal{E}_k(m_i)$, where $i \in_R \{ 0, 1 \}$. The adversary attempts to discover the value i, but in the case of a Vernam cipher, the adversary can do no better than ignore c and guess at i randomly.

Perfect secrecy, however, is not a practical solution. The notion we have of encryption is of a short, reusable secret, the key, that can be used long term to encode multiple secrets. We can achieve this by going from perfect secrecy to computational secrecy. Although the adversary game can be won, it cannot be won by a computationally bounded adversary with advantage beyond brute force trying all keys, or, equivalently, blind luck at guessing the key.

Date: September 5, 2023.

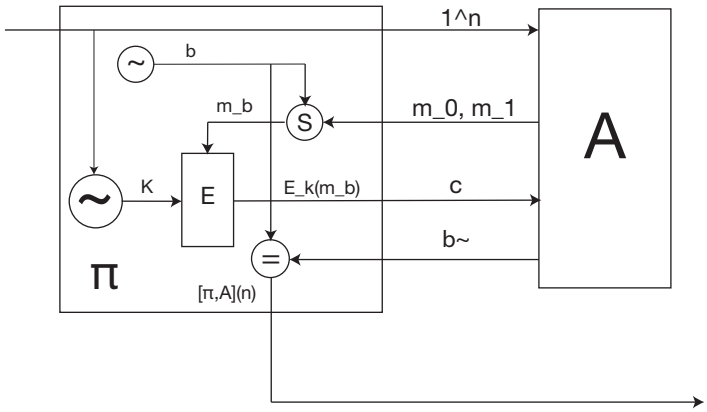

FIGURE 1. The Adversarial Indistinguishability Game.

The adversary then is replaced by a PPT algorithm (Probabilistic Polynomial Time), the key size becomes the input to an asymptotic condition, that states,

$$P([\,\Pi, \mathcal{A}\,](n)) \leq 1/2 + \mathit{negl}(n)$$

where $\mathit{negl}(n)$ is the inverse of any polynomial in n. The probability is taken over the choice if keys and any randomness in $\mathcal{A}$ or Π.

Given that the adversary if a PPT, the source of randomness for the cipher need not be actually random, it only needs to be indistinguishable from from by any PPT algorithm. This is the concept of pseudorandom — a sequence of bits, generated deterministically from a short random seed that looks random to any PPT algorithm, including the adversary that is trying to find and exploit any defect in the sequence to with its adversarial game.

2. Attack models and their solutions

While we have a solution now for a practical cipher, it is only secure in an unrealistic adversarial model. Classical cryptanalysis had recognized various forms of attack, For one, even the powerful Vernon cipher is breakable if the key is ever reused.

Known plaintext attack: The codebreaker might still be a passive eaves dropper, but they manage to have access to cipher text of a known plain text. Even when plaintext is not certain, it can be guessed at. Turing called these *cribs*, and were useful in breaking the enigma. The cribs consisted of such things are the standard open greetings of letters.

Chosen plaintext attack: It has been shown useful to codebreakers if they are able to inject plaintext messages to be encrypted. This attack can be adaptive as the analyst gets more information to determine the most useful plaintext to feed the encryption.

Chosen ciphertext attack: It is further useful if the codebreaker can decide on a ciphertext to be decrypted and be given the decryption.

Given these real-world examples, a series of adversarial models have been proposed the elucidate why each of these practical codebreaking techniques work, and how to thwart them (supposing that that is what you want, or to enable them, if you are a double agent).

The series of attacks actually form a single ranking of attacks in which defending against one, defends against all lower ranking attacks. Also, to show the necessity of each step in the defense, one can provide examples at each level of an encryption that is strong for all lower levels but not for this level.

The purpose of this is to make clear what each defense technique does, and why it is necessary.

2.1. **Multiple messages.** The eavesdropping attack is modified to allow for key reuse. The adversary chooses two lists of messages,

$$m_0 = \langle\, m_{0,0}, m_{0,1}, \ldots, m_{0,k-1} \,\rangle \text{ and } m_1 = \langle\, m_{1,0}, m_{1,1}, \ldots, m_{1,k-1} \,\rangle$$

where $|m_{0,i}| = |m_{1,i}|$ and receives an encryption of one list or the other.

$$\mathcal{E}_k(m_b) = \langle\, \mathcal{E}_k(b,0), \mathcal{E}_k(b,1), \ldots \mathcal{E}_k(b,k-1) \,\rangle.$$

The adversary my decide which list was encrypted.

Our encryption secure for eavesdropping is not secure for multiple messages. The adversary provides these two message lists,

$$m_0 = \langle\, m, m \,\rangle \text{ and } m_1 = \langle\, m, m' \,\rangle$$

with $m \neq m'$. The returned encryption is $\langle\, c, c' \,\rangle$. If the encryption is stateless and deterministic than $c = c'$ when $b = 0$ and $c \neq c'$ otherwise.

This attack is very broad. It works because the encryption scheme is deterministic. If we are to have security for multiple messages, it must be that the encryption is a randomized algorithm that produces encrypts as different from each other for the same message input as for different message inputs.

Theorem 2.1. No deterministic encryption is secure against a Multiple Message attack.[1]

Theorem 2.2. There are encryption schemes secure against an eaves dropping attack insecure against a Multiple Message attack.[2]

[1]Theorem 3.21 in Katz and Lindell, 2nd Ed.

[2]Theorem 3.20 in Katz and Lindell, 2nd Ed.

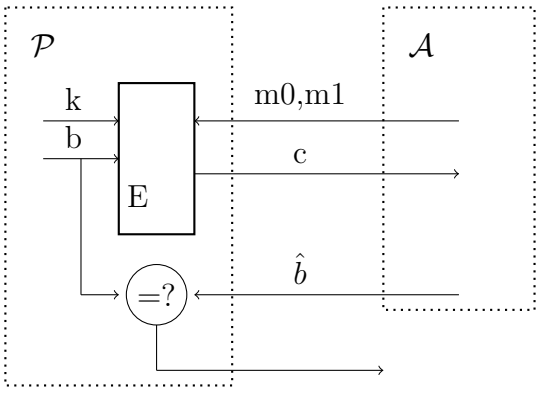

FIGURE 2. Indistinguisability game for a chosen-plaintext attack

2.2. **Chosen Plaintext Attacks.** The CPA game begins with the choice of a key, secret from the adversary. The adversary can present any number of messages at any time and get the resulting cipher text. (Note that the time bound on the adversary implies the number of queries is polynomial.)

At some point, the adversary presents and m_0 and and m_1 and receives an $\mathcal{E}_k(m_b)$. The adversary will guess the b. of this one query.

This is the classic model of CPA security, but there is an equivalent model where the bit b is chosen at the same time as the key, and the encryption queries are answered by an oracle,

$$\mathcal{O}_{k,b} : (m_0, m_1) \mapsto \mathcal{E}_k(m_b)$$

This model is generalization of the CPA model because $\mathcal{E}_k(m) = \mathcal{O}_{k,b}(m, m)$. It also generalizes Multiple Message security because a successful MM attack with

$$m_0 = \langle\, m_{0,j} \,\rangle \text{ and } m_1 = \langle m_{1,j} \,\rangle$$

can be carried in in this model by asking in sequence $\mathcal{O}_{k,b}(m_{0,j}, m_{1,j})$ and then deciding as the MM attack decides.

Theorem 2.3. A CPA secure encryption is Multiple Message Secure.[3]

The proof that LR-CPA and CPA equivalent is difficult, It would seem that the LR framework gives many datapoints useful at guessing b while CPA gives only one data point. Having dealt with this difficulty, it becomes very easy to show CPA implies MM. However MM does not imply CPA.

One can propose encryption schemes which are secure in the Multiple Message Model by not CPA secure. These take advantage of the adaptive capability of CPA. The proofs are often contrived encryption schemes that leak information in their

[3]Theorem 3.24 in Katz and Lindell, 2nd Ed.

answers, but a follow-up query is needed to exploit that leakage — something possible in a CPA attack but in an MM attack.

Theorem 2.4. There exist encryption schemes which are Multiple Message Secure but are not CPA secure.

3. Psuedorandom functions and permutations

A *random function* of the space of n-bits is a choice, uniformly at random from the space of all functions $f : \{0,1\}^n \to \{0,1\}^n$. This is an enormous space, as can be seen by the description of such a function. Baring some formula, we are allowed to assign to each of the 2^n and arbitrary n bit string, hence any string of length $n\,2^n$ gives a unique random function. Since there are $2^{n\,2^n}$ such strings this is the number of functions.

It is hard to conceptualize this. Once can take a sort of just-in-time-definition of a random function. The function f is a box with a list of values $x : f(x)$ assigned as the function is queried on $f(x)$. On x, the list is consulted and the previously decided on $f(x)$ is returned, if available. Else the box flips n coins resulting in an n bit string r, adds $x : r$ to the internal list and returns r as the value of $f(x)$.

If the function is furthermore a permutation, it is called a random permutation. The probability assigned to each permutation is different than to a function, for there are fewer of them, and the distribution is a. uniform distribution.

Just as a random number generator is a building block for crypto systems, and are made practical against PPT bound adversaries, so are random functions and random permutations.

A pseudo-random function is a deterministic function $F : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}^n$ for which no PPT can distinguish from a random function. To provide more detail, the function F is written $F_k(x)$, where the first parameter to the function is a key, and the second the input to the function. Drawing a random f is drawing from that super large space, and drawing from F is choosing a key, that is, drawing from a space of 2^n different functions. This is similar to the pseudo-random generator, where the seed for a long string of random bits is of limited diversity compared the possible outcomes.

The discussion above also applies to F being a permutation, in which case we consider a pseudo-random permutation as being indistinguishable by a PPT from a random permutation.

The formal definition is made with the help of a *distinguisher*, a PPT of any sort that exams the output of samples from F and f to looking for anything that can distinguish them. The notation we introduce for a distinguisher uses the notion of an *oracle*. Formally, we cannot hand over to $\mathcal{D}$ the description of F or f, because

the description is too long. Neither polynomial space nor time contain it. Rather we present F and f as a box that $\mathcal{D}$ will query in one time step.

Definition 3.1. The function F is pseudo-random if for all PPT adversaries $\mathcal{D}$,

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

where the first probability is over the choice of k and the second is over the choice of f, as well as any randomness in $\mathcal{D}$.

4. Block ciphers

Just as pseudo-random generators are used to create a sort of cipher called a stream cipher, pseudo-random functions are permutations are used to create a sort of cipher call a *block cipher*.

In practice, the difference between stream and block ciphers can become less clear. A pseudo-random function can be used to create a pseudo-random generator, a vice-a-versa. Also, none of our constructions that are clearly based on one or the other primitives are used in practice. Practical considerations add to the constructions that made them less clearly of one sort or the other.

That said, here we have our CPA resistant block cipher based on a pseudo-random function.

Theorem 4.1. Let F be a pseudo-random function. Choose k, an n bit key, and encrypt m, an n bit message by choosing uniformly $r \in_R \{ 0, 1 \}$ and calculating the cipher text,

$$\langle r, m \oplus F_k(r) \rangle.$$

Decryption of cipher text $\langle r, s \rangle$ is $m = s \oplus F_k(s)$. The cipher is CPA secure.

As was observed above, however, this cipher is not CCA secure. For that we continue on future lecture notes.

5. Chosen Ciphertext Attacks

An additional power for the adversary is to be able to request not just encryptions of chosen plaintext, but decryptions of chosen cipher text. This model is important because several reasonable CPA resistant systems are not CCA resistant.

One issue is that of *malleability*. Given $c = \mathcal{E}_k(m_b)$, it is clear that the adversary allowed a CCA attack cannot ask for the decryption of c. However, it can ask for decryptions of a modification of c. Even for CPA resistant encryptions, this modified c can decrypt to something that is close enough to one of the m_b to reveal which, that is, to reveal b.

Another issue is some real-world attacks take a $c = \mathcal{E}_k(m)$ and probe the message m by sending modifications of c. In many cases, the encryption of m includes padding

that makes some modifications of c invalid cipher texts, and the attacker feels around the edge to extract information about m by what modifications of c are rejected by the decryption and which are not.

Theorem 5.1. There are encryption schemes that are CPA secure but are not CCA secure.

Proof: The encryption,

$$\mathcal{E}_k : m \mapsto \langle\, r, m \oplus F_k(r) \,\rangle$$

where r is a random string and F_k is a pseudorandom permutation, is not CCA secure. Given $\mathcal{E}_k(m) = \langle\, r, c \,\rangle$, query the decryption of $\langle\, r, \bar{c} \,\rangle$ is $\bar{m}$, where the bar is bit inverse. This observation is certainly enough to win the indistinguishability game. However the encryption is CPA secure.

5.1. **Authenticated Encryption.** In practice, confidentially is not enough for the expectations we have for secure messages. We would like to know that the message is as sent by the counter-party, and not otherwise tampered with or forged. The possession of a shared secret is used for the sending party to calculate a message tag, which the receiving party verifies. The security guarantee is that is be computationally infeasible to calculate a tag without knowledge of the key, even after the attacker having sampled tags, even of messages of it choosing.

The requirement of CCA security can be satisfied by adding such a tag to a CPA secure encryption scheme (if done properly). The combination is called *authenticated encryption*. The full discussion is for another note.