ICS 268: Cryptography and Communication Security September 28, 2004

Lecture 1: Crypto Overview, Perfect Secrecy, One-time Pad

Lecturer: Stanislaw Jarecki

(These notes incorporate material from Tal Malkin's Lecture 1-2 and Yevgeni Dodis's Lecture 1)

1 Lecture Summary

We overview the aims and the philosophy of modern cryptography. We exemplify this approach with the first shot at a definition of encryption scheme security, which we will develop later on in this class. We then give a classic definition given by Claude Shannon of *perfect secrecy* for an encryption. We show that various classic ciphers fail to satisfy this definition, but we also show a cipher called *One-Time Pad* which does satisfy it. However, this cipher has very limited applicability because the communicating parties must share a pre-agreed key which is as long as the message, i.e. as all the communication they will be able to secretly exchange between them. We show, moreover, that this is a fundamental limitation of every *perfectly secure* cipher. In other words, we show that no perfectly secure cipher can have keys shorter than the message. This motivates the need to relax Shannon's information-theoretic *perfect secrecy* requirement on encryption schemes with a *computational* secrecy property instead. We'll develop such computational secrecy property in the next lecture.

2 Modern Cryptography: Short Overview

The aim of modern cryptography is to design communication schemes (encryption schemes, identification schemes, message authentication schemes, etc) whose security properties can be **proven**, usually based on some computational hardness assumptions, e.g. an assumption that factoring, or computing discrete logs, is *hard*.[1]

However, to create a scheme whose security is provable, we first need to **define** the security property which we need to prove. The security property will be usually defined as a requirement that no efficient algorithm can *win in some communication game* with some *significant* probability (again, we postpone the precise definition of *significant* to the next lecture).

2.1 Example: Secret Communication and Secure Encryption

Let's exemplify the provable security approach with an example of encryption. How to define what a secure encryption is?

Assume there are three agents, Bob, Alice, and Eve. Alice wants to send Bob a private message that only Bob can read. Eve, which is an abbreviation for an eavesdropper, is an adversary who

[1] We'll see in the next lecture how to define this hardness precisely, but the intuition is that a problem is hard if no *efficient*, i.e. polynomial-time, algorithm can solve that problem with probability higher than some *negligible* factor. (We postpone the definition of *negligible* to the next lecture.)

may intercept Alice's communication, but reading it should not enable her to reconstruct Alice's message to Bob. This is the essence of the problem of secure communication.

To communicate the message secretly, Alice must transform her plaintext message somehow. We call this transformation an *encryption*, while the reverse transformation done by the receiver Bob is called *decryption*. The original message is called a *plaintext* while the output of the encryption procedure applied to the plaintext is called a *ciphertext*. Thus the above secure communication problem can be solved by a (private key) encryption scheme. We call an encryption scheme $\mathcal{E}\Sigma$ a triple of (efficient) algorithms $(Kgen, Enc, Dec)$ and a message space $\mathcal{M}$ s.t.

$KGen$	is a probabilistic algorithm which picks key k
	(for simplicity assume for now that k is picked at random in some keyspace $\mathcal{K}$)
Enc	is an encryption algorithm, $Enc : \mathcal{K} \times \mathcal{M} \to \mathcal{C}$
$\mathcal{C}$	is a ciphertext space (defined by Enc, $\mathcal{K}$, and $\mathcal{M}$)
Dec	is a decryption algorithm, $Dec : \mathcal{K} \times \mathcal{C} \to \mathcal{M}$

Apart from requiring that all these algorithms are efficient, we also require the *completeness* property[2], which says that:

$$\forall_{k \leftarrow \mathcal{K}}, \forall_{m \in \mathcal{M}}, \text{ if } c \leftarrow Enc(k, m) \text{ then } m \leftarrow Dec(k, c)$$

Now we are ready for the first attempt at defining what we mean by a *secure encryption*:

Definition 1 (One-Way Secure Encryption (first draft)) *We call a (private key) encryption scheme* $\mathcal{E}\Sigma$ one-way secure *if for every* efficient *algorithm* E*, the following holds:*[3]

$$Pr[E(c) = m \mid k \leftarrow \mathcal{K}, m \leftarrow \mathcal{M}, c \leftarrow Enc(k, m)] \leq \mathsf{negl} \tag{1}$$

(where by negl *we denote a* negligible *probability)*

In other words, if k and m are picked at random and $c = Enc(k, m)$ then E is unlikely to output back m given the ciphertext c. (We postpone the exact definition of an *efficient* algorithm and *negligible* probability to the next lecture.)

Intuitively, what the "one-way security" property of encryption implies is that it is *computationally hard* to decrypt the message from a ciphertext of a random message. We call it "one-way" because it says that encrypting is like a one-way process: It is easy to encode a message in a ciphertext (given the key), but it is hard to reverse this encoding and get the message back (without the key).

Limitations of One-Way Security. While this "one-way security" property must hold for an encryption scheme to be secure in any sense at all, this property is actually pretty weak:

What about a-priori knowledge on messages space? If the message is chosen not at random from the whole (big) space $\mathcal{M}$, but say the message is always some fixed m_0 or m_1, for example m_0 ="buy IBM stock" and m_1 ="sell IBM stock", then the fact that the encryption is one-way secure as specified above does not imply that an efficient algorithm E cannot guess the message m, if E knows that m is equal to either m_0 or to m_1.

[2] "Completeness" properties always say that everything works as it should if the parties use the protocol as it is intended

[3] With notation $Pr[A \mid B]$ we denote a *conditional probability* that event A holds *given* that event B occurs.

What if all security depends on only few bits of m? Even if E does not have any a-priori information about the messages exchanged by A and B, one-way security is hardly enough. Because what if E is interested not in the whole m but only in some particular bits of it? An encryption scheme can be indeed one-way secure while revealing some bits of m, and if these bits are exactly what E wants most to learn about m, the encryption scheme is not satisfactory.

2.2 Other Security Properties and Other Cryptographic Tasks

One moral from the above is that the above natural-sounding security property of encryption is in fact pretty weak. The second moral is that it can be strengthened:

Semantic Security Rather than merely requiring that E cannot decrypt the whole m given $c = Enc(k, m)$ for randomly chosen k and m, we can pose a stronger requirement, called "semantic security". We'll define this notion precisely later on, but at first approximation, we call an encryption scheme semantically secure if anything that can be computed by some efficient algorithm E about message m given the ciphertext $c = Enc(k, m)$, for random k and m chosen according to any, potentially very restricted, probability distribution $\mathcal{M}$, can be also efficiently computed *without* knowing this ciphertext. In other words, semantic security captures the notion that the ciphertext is *useless* for deciphering any information about message chosen from any a-priori known distribution.[4]

Active Attacks In both communication games above, the adversary E is *passive* in the sense that E only gets to see one ciphertext c of message $m \in \mathcal{M}$ for some probability distribution $\mathcal{M}$. However, an adversary can be active too, so the security can be defined in terms of (non-existence of) an efficient algorithm E which ***first*** has access to either the encryption algorithm of Alice or the decryption algorithm of Bob, for example E can ask Alice to see encryptions of some (test) messages $m_1, m_2, ...$ of E's choice, and *only then* E faces either the one-way encryption or the semantic-security test.

Another point I want you to draw from the above example is that this was just one example of defining some security property of a cryptographic communication scheme. In the above case we defined "one-way security" of a private key encryption scheme. But there are other "secure communication tasks" we will want to achieve in this class:

Public key encryption allows Alice and Bob to have a private conversation without the necessity of first exchanging a secret key. One party, say Bob, can have a public and a private key pair. Alice, knowing Bob's public key, can encrypt messages so that only Bob, the owner of the corresponding private key, can decrypt it. Unlike the private-key encryption discussed above, this way everyone, for example by grabbing Bob's public key from his webpage, can send messages secretly to Bob. But also unlike in the case of private-key encryption, when we define the security property of public-key encryption, the attacker E should also learn Bob's public key. Thus the communication game between A, B, and C changes quite a bit.

Digital signatures enable Bob to verify that the message he receives is from Alice, and that it has not been modified by anyone else. Here E's task will be to create a signature *forgery*, i.e. a

[4] If this sounds too complex, just skip it for now. It'll be clearer later on.

message which will convince Bob that it came from Alice, while in fact Alice has never signed this message. As in the case of encryption, we can and should consider active attackers E who first ask Alice to sign any number of (test) messages $m_1, m_2, ...$ of their choice, and still are unable to produce a forgery on some m s.t. for all i, $m \neq m_i$.

Identification Schemes enable Alice to authenticate herself to Bob. Eve's task is to spoof Alice to Bob, i.e. to convince Bob that she, Eve, is Alice too...

Secure function evaluation allows Alice and Bob (and possibly more parties) to compute some function of their input without revealing any extra information to other parties or eavesdroppers. One example of a function one would want to compute securely are various database problems. For example, Alice and Bob each have a database containing a list of some police suspects. Can they compare the lists securely in the sense that they will identify all suspects who appear on both lists, but no other information about their databases will be exchanged in the process of this computation?

2.3 PRINCIPLES OF CRYPTOGRAPHIC GAMES

The encryption example exemplifies some basic principles we rely on when defining security properties of cryptographic schemes:

Algorithms are public We preclude the "security by obscurity" approach. We assume that the adversary knows what scheme he is attacking, and hence knows what Encryption and Decryption algorithms Alice and Bob use. This assumption is implicit in definition 1 when we say "for every" algorithm E, which implies that we include algorithms E which are aware of algorithms $KGen$, Enc, and Dec.

Secrets are secret Note that attacker E does not get to see the key k. This is almost always the case. If E knows everything that A and B know, it's hard to think of any notion of a secure communication task that A and B can perform while E is somehow precluded from participation.

Adversary is bounded Usually, the adversary E has some limitations. In definition 1, he is required to be an efficient, i.e. probabilistic polynomial-time, algorithm. This limits the amount of time and space he has to compute. But there are notions of security for *unbounded* adversary too. This will be the *perfect security* notion we'll develop below.

Computational hardness assumptions We will usually assume that there are some problems which are computationally hard, like the factoring problem.

2.4 PROVABLE SECURITY, LAYERS OF CRYPTOGRAPHIC TOOLS

The security proof is usually an argument which shows that if there exists an efficient attacker E, which, for example, breaks the one-wayness property for some encryption scheme $\mathcal{E}\Sigma$, i.e. it contradicts equation (1), then one can construct another efficient algorithm, say E', which breaks the underlying hardness assumption, for example E' factors large composite numbers, or E' computes discrete logarithms. By taking a counterpositive of the above statement, the statement implies that

if the underlying hardness assumption holds (i.e. factoring or discrete log problem are indeed computationally hard) then the encryption scheme $\mathcal{E}\Sigma$ is one-way secure.

In fact, the big picture of cryptography is that we can build in the above fashion, from relatively simple assumptions (like hardness of factoring), to more complicated ones (like one-wayness of encryption). In the process we can identify some cryptographic objects, like one-way functions, pseudorandom number generators, collision-resistant functions, and more, whose security properties are "natural" in the sense that we can find multiple uses for them, to construct efficient signatures, authentication schemes, encryption schemes, and other secure communication schemes. At the same time, they are well-defined and therefore we can attempt to construct them using various number-theoretic assumptions (like hardness of factoring or discrete-log) and we can improve and/or better understand the efficiency and security of these constructions.

3 PERFECT SECRECY AND ITS LIMITATIONS

We'll start exploring provable security of encryption by recalling Shannon's classic definition of what it means for an encryption to be secure. We'll see the limitation of Shannon's approach, and we'll how we can overcome this limitation with the modern cryptography paradigm.

3.1 SHANNON'S SECRECY AND PERFECT SECRECY

In 1950's, Claude Shannon, the creator of information theory, proposed an information theoretic definition of what it means for an encryption scheme to be secure. Although Shannon's original definition of security (see Appendix A) is not easiest to work with, it turns out that Shannon's secrecy is equivalent to the following notion (also formulated by Shannon):

Definition 2 (Perfect Secrecy) *An encryption scheme satisfies perfect secrecy if for all messages m_1, m_2 in message space $\mathcal{M}$ and all ciphertexts $c \in \mathcal{C}$, we have*

$$\underset{K\leftarrow\mathcal{K}}{Prob}[Enc(K, m_1) = c] = \underset{K\leftarrow\mathcal{K}}{Prob}[Enc(K, m_2) = c] \tag{2}$$

where both probabilities are taken over the choice of K in $\mathcal{K}$ and over the coin tosses of the (possibly) probabilistic algorithm Enc.

Note that another way of stating the perfect secrecy requirement is that for any $c \in \mathcal{C}$ there exists a constant $p_c \in [0, 1]$ s.t. for all $m \in \mathcal{M}$, we have $\underset{K\leftarrow\mathcal{K}}{Prob}[Enc(K, m) = c] = p_c$.

In other words, what perfect secrecy requires is that, given a ciphertext, every message in the message space is exactly as likely to be the underlying plaintext. To put it yet another way, the plaintext is independent of the ciphertext. Thus the perfect secrecy requirement implies that the eavesdropper truly learns nothing at all about the underlying plaintext.

Given this strong definition of secrecy, let's look how some classic ciphers do with regards to it.

3.2 SHIFT CIPHER

$\mathcal{M} : \{A, ..., Z\}^\ell$, where ℓ is the message length, but we'll encode each letter m_i of $m = [m_1, ..., m_\ell]$ as an integer in $\{0, ..., 25\}$

$\mathcal{K} : \{0, ..., 25\}$ defines the shift (Historical note: $k = 3$ is known as "Caesar's cipher")

$c = End(k, [m_1, ..., m_\ell]) = [(m_1 + k \bmod 26), ..., (m_\ell + k \bmod 26)]$

Example 1 $m =$*"HELLO"; $k = 1$; $c =$"IFMMP"*

Weakness of this cipher are easy to see:

- Keyspace is too small, only 26 possible keys; succumbs to brute force (exhaustive search) attack easily.
- Vulnerable to frequency analysis.
- Very easy to break:
  - e.g. Repeats in m are repeats in c.
  - e.g. If Eve knows the first word of a message is "HELLO", it is trivial to derive the key.

However, let's see how this cipher fails the perfect secrecy definition.

Claim 3 *Shift cipher does not satisfy the perfect secrecy property for $\ell \geq 2$.*

Proof: Take m_1 = "AB", m_2 = "AZ", and c = "BC". Then there exists a key $k \in \mathcal{K}$ s.t. $Enc(k, m_1) = c$, namely $k = 1$. However, for all $k \in \mathcal{K}$ we have $Enc(k, m_2) \neq c$, and hence $\underset{K \leftarrow \mathcal{K}}{Prob}[Enc(K, m_1) = c] = 1/26$, while $\underset{K \leftarrow \mathcal{K}}{Prob}[Enc(K, m_2) = c] = 0$, and so the perfect secrecy requirement is violated. □

Claim 4 *For $\ell = 1$, i.e. for one-letter long messages, shift cipher actually does satisfy the perfect secrecy property.*

Proof: Note that for every letter m and every $c \in \mathcal{C}$ where $\mathcal{C} = \mathcal{M} = \{A, ..., Z\}$, there is a unique key $k = c - m \bmod 26$ s.t. $Enc(k, m) = (m + k \bmod 26) = c$. And therefore for every m, c we have $\underset{k \leftarrow \mathcal{K}}{Prob}[Enc(k, m) = c] = 1/26$, and so the perfect secrecy requirement is satisfied. □

3.3 SUBSTITUTION CIPHER

$\mathcal{M} : \{A, ..., Z\}^\ell$, where ℓ is the message length

$\mathcal{K}$: permutations on $\{0, ..., 25\}$; i.e. each $k \in \mathcal{K}$ is chosen at random and is a 1:1 mapping from $\{0, ..., 25\}$ to $\{0, ..., 25\}$

$c = Enc(k, [m_1, ..., m_\ell]) = [k(m_1), k(m_2), ..., k(m_\ell)]$, for $m = [m_1, ..., m_\ell] \in \mathcal{M}$ and $k \in \mathcal{K}$

Example 2 *Key k is the mapping from Plaintext Alphabet to Ciphertext Alphabet.*
So if m="HELLO"; k is mapping in Table 1; c="JREEM"

a|b|c|d|e|f|g|h|i|j|k|l|m|n|o|p|q|r|s|t|u|v|w|x|y|z|←Plaintext Alphabet
a|b|g|w|r|o|v|j|z|l|s|e|x|q|m|c|t|i|h|k|p|y|u|d|f|n|←Ciphertext Alphabet

Table 1: An Example Permutation

Comments on security:

- Vulnerable to brute force attack: keyspace is 26! (not so bad for a computer).
- Vulnerable to frequency analysis (i.e. language patterns), monographs, bigraphs, trigraphs...

Claim 5 *Permutation cipher also is perfectly secure for $\ell = 1$ but fails to satisfy the perfect secrecy property for $\ell \geq 2$.*

(We leave the proof as an exercise.)

3.4 ADDITIVE CIPHER

$\mathcal{M} : \{A, ..., Z\}^{\ell}$, where ℓ is the message length.

$\mathcal{K} : \{A, ..., Z\}^{n}$.

$c = Enc(k, [m_1, ..., m_\ell])$ s.t. $c = [c_1, ..., c_\ell]$ where $c_i = (m_i + k_{i(\bmod n)}) \bmod 26$ for $m = [m_1, ..., m_\ell] \in \mathcal{M}$ and $k = [k_0, ..., k_{n-1}] \in \mathcal{K}$.

i.e. a key k can be thought of as the double (n-character keyword, effective key). If $n < \ell$, concatenate k to itself $\lfloor \ell/k \rfloor$ times to get the effective key.

Example 3 $m =$*"HELLOBOB"*
$k =$*(keyword="SECRET",effective key="SECRETSE")*
Using $\{A, ..., Z\} \equiv \{0, ..., 25\}$*, we get*
$c =$*[(H+S),(E+E),(L+C),(L+R),(O+E),(B+T),(O+S),(B+E)] mod 26 = "ZINCSUGF"*

Comments on security:

- Although this withstands simple frequency analysis, the scheme breaks once you figure the keyword length (n). Once n is known (or guessed), one simply does frequency analysis on n blocks of ciphertext, each block formed by concatenating every nth letter of the original c.
- A brute force attack requires a search of 26^n, which leaks information if $n < \ell$.

Claim 6 *Additive cipher is perfectly secure for $\ell = n$ but fails to satisfy the perfect secrecy property for $\ell \geq n$.*

3.5 ONE TIME PAD [OTP] (VERNAM CIPHER)

$\mathcal{M} : \{0,1\}^\ell$, where ℓ is the message length.

$\mathcal{K} : \{0,1\}^\ell$.

$c = Enc(k, m) = m \oplus k$, for $m \in M, k \in K$, where "$\oplus$" stands for a bit-wise xor

$m = Dec(k, c) = c \oplus k$, for $m \in M, k \in K$.

Decryption works because $(c \oplus k) = ((m \oplus k) \oplus k) = m \oplus (k \oplus k) = m$

Example 4

$$\begin{aligned} m &= [001110010000101101011011 10] \\ k &= [111010011010100011000111 1] \\ c &= [110100001011111001110000 1] \end{aligned}$$

Remark: Comments on security

- Here an exhaustive search gives no additional information since the keyspace is as large as the message space, and every ciphertext could correspond to any message. Indeed, this cipher satisfies perfect security!
- Using the same k of some length $n < \ell$ more than once breaks the scheme because it leaks some information about the plaintext $m = [m_1, ..., m_\ell]$. For example, if $\ell = 2n$ then note that $c_{[1,...,n]} = m_{[1,...,n]} \oplus k$ (where $k = k_{[1,...,n]}$) and similarly $c_{[n+1,...,2n]} = m_{[n+1,...,2n]} \oplus k$. And therefore
$$m_{[1,...,n]} \oplus m_{[n+1,...,2n]} = c_{[1,...,n]} \oplus c_{[n+1,...,2n]} \tag{3}$$
which means that $c = c_{[1,...,2n]}$ gives some information about $m_{[1,...,2n]}$. This information in particular precludes some choices of $m \in \mathcal{M}$, namely those that violate equation (3).

Theorem 7 *OTP encryption satisfies the* perfect secrecy *requirement.*

Proof: Take any $m \in \mathcal{M}$ and $c \in \mathcal{C}$, and let $k^* = m \oplus c$. Note that:

$$\begin{aligned} \underset{k \leftarrow \mathcal{K}}{Prob}[Enc(k, m) = c] &= \underset{k \leftarrow \mathcal{K}}{Prob}[(k \oplus m) = c] \\ &= \underset{k \leftarrow \mathcal{K}}{Prob}[k = c \oplus m] \\ &= \underset{k \leftarrow \mathcal{K}}{Prob}[k = k^*] \\ &= \frac{1}{2^\ell} \end{aligned}$$

Since the equation holds for every $m \in \mathcal{M}$, it follows that for every $m_1, m_2 \in \mathcal{M}$ we have $\underset{k \leftarrow \mathcal{K}}{Prob}[Enc(k, m_1) = c] = \frac{1}{2^\ell}$ as well as $\underset{k \leftarrow \mathcal{K}}{Prob}[Enc(k, m_2) = c] = \frac{1}{2^\ell}$, which implies that

$$\underset{k \leftarrow \mathcal{K}}{Prob}[Enc(k, m_1) = c] = \underset{k \leftarrow \mathcal{K}}{Prob}[Enc(k, m_2) = c]$$

which establishes perfect security of OTP. □

Even though the OTP method is secure, it has a big flaw which makes it impractical: The key needs to be as long as the total length of all communication that can ever be encrypted using this key. So if two communication components (computers, cell phones, whatever) establish a shared key of length, say, $\ell = 1K$ bytes, and plan to use it to communicate secretly using the OTP cipher, they can only send at most $1K$ of traffic between each other. There might be some exotic cases where this is enough, but it usually will not be.

Note also that there does not seem to be a way to bootstrap this initial key so that to send some traffic, say $0.5K$, using $0.5K$ of the key material, and then have one party create and secretly send to the other a new $1K$ secret key using the remaining $0.5K$ of the old secret key material.

In fact, we can easily show that no such trick can possibly work and that the above impracticality of the OTP cipher is *inherent* to any cipher which satisfies Shannon's perfect secrecy requirement.

Theorem 8 (Optimality of OTP) *For* **any** *encryption scheme $\mathcal{E}\Sigma$ which satisfies Shannon's perfect secrecy requirement, it must be that the keyspace $\mathcal{K}$ has the same size as the message space $\mathcal{M}$.*

If keys and messages are binary string, $\mathcal{M} = \{0,1\}^\ell$ and $\mathcal{K} = \{0,1\}^n$, this means that an encryption scheme can be perfectly secure only if $n = \ell$. Thus the OTP cipher is optimal with respect to the key length.

Proof: Assume for contradiction that an encryption scheme $\mathcal{E}\Sigma$ has $|\mathcal{K}| < |\mathcal{M}|$. Take any ciphertext $c\mathcal{C}$ s.t. there exists some $m^* \in \mathcal{M}$ and $k^* \in \mathcal{K}$ s.t. $Enc(k^*, m^*) = c$. Let us count the number of messages m that could result from the decryption of c under *some* valid secret key $k \in \mathcal{K}$. I.e., let $S \in \mathcal{M}$ be the set of messages $S = \{m \mid \exists_{k \in \mathcal{K}}\ Enc(k, m) = c\}$. Note that $S = \{m = Dec(k, c)\}_{k \in \mathcal{K}}$, and since to every $k \in \mathcal{K}$ there corresponds at most one unique $m = Dec(k, c)$, the size of set S is at most the size of $\mathcal{K}$. Therefore, there exists a non-empty set $S' = \mathcal{M} \setminus S$ of messages s.t. for each $m \in S'$ there exists no key $k\mathcal{K}$ s.t. $c = Enc(k, m)$. In other words, for all $m \in S'$ we have

$$\underset{K \leftarrow \mathcal{K}}{Prob}[Enc(K, m) = c] = 0$$

But since there exists m^*, k^* s.t. $c = Enc(k^*, m^*)$, then in particular

$$\underset{K \leftarrow \mathcal{K}}{Prob}[Enc(K, m^*) = c] \neq 0$$

And this violates the perfect secrecy requirement. □

4 WHAT'S AHEAD: PRACTICAL *and* PROVABLY SECURE CRYPTO

The above negative result does not mean that secure encryption has no hope of being practical. What it does show, however, is that Shannon's notion of perfect secrecy is too strong to be useful in practice. Fortunately, we can relax this notion from information-theoretic secrecy to *computational* secrecy, and achieve provably secure encryption schemes secure under this notion. While information-theoretic secrecy required that every given a ciphertext, every plaintexts are exactly as likely, the computational secrecy notion will ask only that no *efficient algorithm* can tell, given a ciphertext, and, say, any two messages that could potentially be plaintexts corresponding to this ciphertext, whether one of these messages is more likely than the other to be the actual plaintext.

This computational notion will be the subject of the next lecture (and indeed the subject of the rest of this class).

A Shannon's Original Definition of Secrecy

Definition 9 (Shannon Secrecy) *Let $\mathcal{D}$ be any probability distribution on messages. An encryption scheme satisfies Shannon secrecy w.r.t. the message distribution $\mathcal{D}$ if for all messages m (regardless of the probability distribution $\mathcal{D}$) and all $c \in \mathcal{C}$ we have*

$$\underset{M \leftarrow \mathcal{D}, K \leftarrow \mathcal{K}}{Prob}[M = m \mid Enc(K, M) = c] = \underset{M \leftarrow \mathcal{D}}{Prob}[M = m] \tag{4}$$

where the first probability is taken over M chosen according to distribution $\mathcal{D}$, over random keys K chosen in $\mathcal{K}$, and over the possible random choices of the (possibly) probabilistic encryption algorithm Enc, while the second probability is taken over $M \leftarrow \mathcal{D}$.

We say that encryption scheme satisfies Shannon secrecy if it satisfies Shannon secrecy w.r.t. all probability distributions $\mathcal{D}$ on messages.

What this requirement says is that, for any a-priori distribution of messages $\mathcal{D}$,[5] the probability that a communicated message $M\mathcal{D}$ is equal to any particular message m *does not change* even if we know the ciphertext $c = Enc(K, M)$, for randomly generated key $K \leftarrow \mathcal{K}$. In other words, seeing a ciphertext does not tell us any *more* about message M which the attacker does not already know from the a-priori message distribution $\mathcal{D}$.

[5]For example, we might know that A and B usually communicate in English sentences, in which case the a-priori probability distribution $\mathcal{D}$ would assign an extremely low probability to message $m =$"axrqe asdvas"...