NOTES ON CRYPTOGRAPHY

Based on Katz-Lindell. All random variables considered discrete?

1. Introduction and perfect secrecy

Encryption Scheme: An encryption scheme consists of a tuple $(\text{Gen}, \text{Enc}, \text{Dec})$ where

- Gen is a random variable with values in a set $\mathcal{K}$, called the *keyspace*.
- There is a set $\mathcal{M}$, called the *message space*.
- There is a set $\mathcal{C}$, each element is called a *ciphertext*.
- For each $k \in \mathcal{K}$ and $m \in \mathcal{M}$ we have a random variable $\text{Enc}_k(m)$ taking values in $\mathcal{C}$.
- For each $k \in \mathcal{K}$, we have a map $\text{Dec}_k : \mathcal{C} \to \mathcal{M}$ satisfying

$$\text{Dec}_k(\text{Enc}_k(m)) = m \quad \text{for all } m \in \mathcal{M}$$

.

More formally, $\text{Enc}_k(m) : \Omega \to \mathcal{C}$ on some sample space Ω. The last point means that $\text{Dec}_k(\text{Enc}_k(m)(\omega)) = m$ for all $\omega \in \Omega$. If $\text{Enc}_k(m)$ is a constant map, we simply consider it to be a deterministic function $\text{Enc}_k : \mathcal{M} \to \mathcal{C}$.

Example 1.1 (Caeser cipher)**.** In this example, we identify naturally the lowercase letters $\{a, b, \ldots, z\}$ with $\mathbb{Z}/26\mathbb{Z}$. We let $\mathcal{K} = \mathbb{Z}/26\mathbb{Z}$ and we let $\mathcal{M} = \mathcal{C}$ be the set of all words in the lowercase letters. We now define $\text{Enc}_k(m) \in \mathcal{C}$ to be the word obtained by adding k (mod 26) to each letter of m. So $\text{Enc}_3(zac) = cdf$. So $\text{Dec}_k = \text{Enc}_{-k}$ is the inverse. The distribution on $\mathcal{K}$ is uniform (i.e., Gen takes values uniformly in $\mathcal{K}$).

Definition 1.2 (Perfect secrecy)**.** An encryption scheme is said to be *perfectly secret* if for all $m, m' \in \mathcal{M}$ and $c \in \mathcal{C}$ we have that

$$Pr(\text{Enc}_k(m) = c) = Pr(\text{Enc}_k(m') = c).$$

More precisely, if $P_{\mathcal{K}}$ is the distribution on $\mathcal{K}$ and $P_{\mathcal{C},m,k}$ is the distribution on $\mathcal{C}$ induced by $\text{Enc}_k(m)$ then

$$\sum_{k \in \mathcal{K}} P_{\mathcal{K}}(k) P_{\mathcal{C},k,m}(c) = \sum_{k \in \mathcal{K}} P_{\mathcal{K}}(k) P_{\mathcal{C},k,m'}(c).$$

That is, Pr refers to the probability distribution where one chooses $k \in \mathcal{K}$ randomly (according to the random variable Gen part of the encryption scheme) and then one runs the random variable $\text{Enc}_k(m)$ to produce $c \in \mathcal{C}$. If Enc_k is deterministic ($\text{Enc}_k(m)$ is constant for all k, m) then obviously Pr is just $P_{\mathcal{K}}$.

Example 1.3 (Caeser cipher is not perfectly secret)**.** Let $m = aa$ and $m' = ab$ and let $c = ff$. Then $Pr(\text{Enc}_k(m) = c) = Pr(k = 0) = \frac{1}{26}$, i.e., the probability that aa gets encoded into ff happens only if $k = 5$, so with probability $\frac{1}{26}$. However $Pr(\text{Enc}_k(m') = c) = 0$ because ab can only be encoded into one of $ab, bc, ce, \ldots, za$.

Example 1.4 (one time pad)**.** Let G be a finite group. We define an encryption scheme where the keyspace and namespace is G. The encryption is $Enc_k(m) = km$. Decryption is given by $Dec_k(m) = k^{-1}m$. The distribution on keyspace is uniform. This scheme is perfectly secret as

$$Pr(Enc_k(m) = c) = Pr(km = c) = Pr(k = cm^{-1}) = \frac{1}{|G|},$$

and this expression is clearly independent of m for each fixed c. In the literature, the one-time pad is actually defined only for the case $G = (\mathbb{Z}/2\mathbb{Z})^\ell$. It is called the one-time pad because one needs to generate a new key for each message, i.e., if we send two differnet messages m and m' then the eavesdroper can compute $Enc_k(m) + Enc_k(m') = k + m + k + m' = m + m'$. Thus the eavesdroper can compute the XOR of two different secret messages, which can be bad. Another drawback is the lack of efficiency in that the keyspace is as large as the message space.

Definition 1.5. Consider an encryption scheme as above. Let $\mathcal{P}_\mathcal{M}$ be any distribution on the message space $\mathcal{M}$. We define the induced distribution Pr on $\mathcal{K} \times \mathcal{M} \times \mathcal{C}$ to be the distribution where

$$Pr(K = k, M = m, C = c) = \mathcal{P}_\mathcal{K}(k)\mathcal{P}_\mathcal{M}(m)P_{\mathcal{C},k,m}(c).$$

In other words we choose $k \in \mathcal{K}$ randomly and then $m \in \mathcal{M}$ independently and then $c \in \mathcal{C}$ is drawn according to the random variable $\mathrm{Enc}_k(m)$.

Proposition 1.6. An encryption scheme is perfectly secret if and only if for any distribution $\mathcal{P}_\mathcal{M}$ the induced distribution Pr satisfies the property that

$$Pr(M = m \mid C = c) = Pr(M = m) \quad \text{for all } m \in M \text{ and } c \in C \text{ with} Pr(C = c) > 0.$$

Example 1.7. Suppose we have a distribution where $\mathcal{P}_M(ab) = 0.4$ and $\mathcal{P}_M(aa) = 0.1$ and $\mathcal{P}_M(bb) = 0.5$. If we are using the Caeser cipher, then $Pr(M = ab` \mid C = dd) = 0 \neq 0.4$ but clearly $P(C = dd) > 0$. Thus the Caeser cipher is not perfectly secret.

Definition 1.8 (Adversary)**.** An adversary to a given encryption scheme $\Pi = (\mathrm{Gen}, \mathrm{Enc}, \mathrm{Dec})$ consists of, for each $m_0, m_1 \in \mathcal{M}$ and $c \in \mathcal{C}$, a random variable $\mathcal{A}(m_0, m_1, c)$ taking values in $\{0, 1\}$. For each fixed m_0, m_1, we define two experiments, Exp_0 and Exp_1 as follows. For fixed $b \in \{0, 1\}$ we define the random variable Exp_b by the following experiment:

(1) Choose $k \in \mathcal{K}$ randomly according to Gen.
(2) Choose $c_b \in \mathcal{C}$ randomly according to $\mathrm{Enc}_k(m_b)$.
(3) Now send c_b to the adversary. They then choose $b' \in \{0, 1\}$ randomly according to $\mathcal{A}(m_0, m_1, c_b)$.
(4) Now define $\mathrm{Exp}_b \in \{0, 1\}$ to be 1 if $b = b'$ and 0 if $b \neq b'$.

For each fixed m_0, m_1, we define a random variable $\mathrm{PrivK}_{\mathcal{A},\Pi}$ as follows:

(1) Choose $b \in \{0, 1\}$ uniformly randomly.
(2) Run Exp_b as above.
(3) Now define $\mathrm{PrivK}_{\mathcal{A},\Pi} \in \{0, 1\}$ to be the result of Exp_b.

Proposition 1.9. An encryption scheme $\Pi = (\mathrm{Gen}, \mathrm{Enc}, \mathrm{Dec})$ is perfectly secret if and only if for all adversaries $\mathcal{A}$, we have that $Pr(\mathrm{PrivK}_{\mathcal{A},\Pi} = 1) = \frac{1}{2}$.

Intuitively, the adversary constructs any m_0, m_1 that they wish, they then get an assistant to randomly choose $b \in \{0, 1\}$ and pass m_b into the encryption scheme and get a corresponding c_b. The adversary then

has to guess, based on this triple m_0, m_1, c_b a $b' \in \{0,1\}$ and try to get $b' = b$ (they do not know b, only the assistant does). If they can succeed (i.e., $(b == b')$) with probability greater than $\frac{1}{2}$, then the Proposition says that the scheme is not perfect.

To avoid confusing the different probabilitities, we let

$$Pr_{\mathcal{A}}(\mathcal{A}(m_0, m_1, c) = b')$$

denote the probability that $\mathcal{A}(m_0, m_1, c) = b'$ where m_0, m_1, c are **fixed** (so here randomness is purely dicated by the adversary $\mathcal{A}$ and not the experiment). While

$$Pr_{\mathrm{Exp}_b}(\mathcal{A} = b')$$

denotes the probability that $\mathcal{A}$ returns b' in Exp_b.

Proof of Proposition. Note that

$$Pr(\mathrm{PrivK}_{\mathcal{A},\Pi} = 1) = \frac{1}{2} Pr_{\mathrm{Exp}_0}(\mathcal{A} = 0) + \frac{1}{2} Pr_{\mathrm{Exp}_1}(\mathcal{A} = 1)$$

We compute the first term as

$$\frac{1}{2} Pr_{\mathrm{Exp}_0}(\mathcal{A} = 0) = \frac{1}{2} \sum_{k \in \mathcal{K}} P_{\mathcal{K}}(k) \sum_{c \in \mathcal{C}} P_{\mathcal{C},k,m_0}(c) Pr_{\mathcal{A}}(\mathcal{A}(m_0, m_1, c) = 0)$$

likewise, the second term is

$$\frac{1}{2} Pr_{\mathrm{Exp}_1}(\mathcal{A} = 1) = \frac{1}{2} \sum_{k \in \mathcal{K}} P_{\mathcal{K}}(k) \sum_{c \in \mathcal{C}} P_{\mathcal{C},k,m_1}(c) Pr_{\mathcal{A}}(\mathcal{A}(m_0, m_1, c) = 1)$$

now using the identity $Pr_{\mathcal{A}}(\mathcal{A}(m_0, m_1, c) = 1) = 1 - Pr_{\mathcal{A}}(\mathcal{A}(m_0, m_1, c) = 0)$ we can add these two terms to get

$$Pr(\mathrm{PrivK}_{\mathcal{A},\Pi} = 1) = \frac{1}{2} \sum_{k \in \mathcal{K}} P_{\mathcal{K}}(k) \sum_{c \in \mathcal{C}} P_{\mathcal{C},k,m_1}(c) + \frac{1}{2} \sum_{k \in \mathcal{K}} P_{\mathcal{K}}(k) \sum_{c \in \mathcal{C}} Pr_{\mathcal{A}}(\mathcal{A}(m_0, m_1, c) = 0)(P_{\mathcal{C},k,m_0}(c) - P_{\mathcal{C},k,m_1}(c)).$$

The first sum is

$$\frac{1}{2} \sum_{k \in \mathcal{K}} P_{\mathcal{K}}(k) \sum_{c \in \mathcal{C}} P_{\mathcal{C},k,m_1}(c) = \frac{1}{2}$$

as we are summing over the sample space of a probability measure. Now we have to show that the second sum vanishes for all $\mathcal{A}$ if and only if the scheme is perfectly secret. We rewrite this sum (ignoring the $\frac{1}{2}$ factor) as:

$$\sum_{c \in \mathcal{C}} Pr_{\mathcal{A}}(\mathcal{A}(m_0, m_1, c) = 0) \left(\sum_{k \in \mathcal{K}} P_{\mathcal{K}}(k) P_{\mathcal{C},k,m_0}(c) - \sum_{k \in \mathcal{K}} P_{\mathcal{K}}(k) P_{\mathcal{C},k,m_1}(c) \right)$$

By definition of perfect secrecy, the term inside is 0, i.e., because it is

$$Pr(C = c | M = m_0) - Pr(C = c | M = m_1).$$

Conversely, suppose that this whole expression is 0 for all $\mathcal{A}$. In particular, for each fixed $c_0 \in C$, if we choose $\mathcal{A}$ so that $Pr_{\mathcal{A}}(\mathcal{A}(m_0, m_1, c) = \mathbb{1}_{\{c_0\}}(c)$ then we see that this expression is

$$P(C = c_0 | M = m_0) - P(C = c_0 | M = m_1)$$

and is equal to 0. □

Proposition 1.10. In a perfectly secret scheme, we have $|\mathcal{K}| \geq |\mathcal{M}|$.

Proof. Suppose for contradiction that $|\mathcal{K}| < |\mathcal{M}|$. Choose $c \in C$ such that $Pr(Enc_k(m) = c) > 0$ for some $k, m \in \mathcal{K} \times \mathcal{M}$. Now let $\mathcal{M}(c) = \{Dec_k(c) \mid k \in \mathcal{K}\}$. Then clearly $|\mathcal{M}(c)| \leq |\mathcal{K}| < |\mathcal{M}|$ so we may choose $m' \notin \mathcal{M}(c)$. By perfect secrecy, we have that $Pr(Enc_k(m') = c) = Pr(Enc_k(m) = c) > 0$. This means that $Dec_k(c) = m'$ for some $k \in \mathcal{K}$. This means $m' \in \mathcal{M}(c)$, a contradiction. □

2. Computational Security

Proposition 1.10 shows that if we want perfect secrecy, then we need the keyspace $\mathcal{K}$ to be rather large, i.e., at least as large as $\mathcal{M}$. This is impractical computationally, e.g., we don't want to require a 1GB key to encrypt a 1GB file. To allow for a smaller keyspace, we will have to relax the definition of perfect secrecy to only *efficient* adversaries.

Definition 2.1. A *computational encryption scheme* is a tuple $\Pi = (\text{Gen}, \text{Enc}, \text{Dec})$ such that

- For each non-negative integer n, we have that $\text{Gen}(n)$ is a random variable with values in a set $\mathcal{K}$ called the keyspace. We assume $\text{Gen}(n)$ runs in polynomial-time in n (it is a probabilistic Turing machine running in polynomial time), that is, it outputs a random key in polynomial time in n. We call n the *security parameter*.
- The message space and ciphertext space is $\mathcal{M} = \mathcal{C} = \{0,1\}^*$.
- For each $k \in \mathcal{K}$ and $m \in \mathcal{M}$, the random variable $\text{Enc}_k(m)$ returns a ciphertext in $\mathcal{C}$ in *polynomial-time*, that is, polynomial in the security parameter n (and the length of $|m|$?). In particular, the length of the output is polynomial in n. (DOES THE POLYNOMIAL DEPEND ON the key k?)
- $Dec_k : \mathcal{C} \to \mathcal{M} \cup \{NULL\}$ is a mapping such that $Dec_k(c) = m$ for all k, m and $c \in \mathcal{C}$ such that $Enc_k(m)$ returns c with positive probability. It is also polynomial in its input length m. (It can return $NULL$ if for example c is invalid, not in the output of any encryption).
- We sometimes assume that $Enc_k(m)$ is only defined for messages of length $|m| = \ell(n)$ where $\ell(n)$ is a function of n (can't encrypt arbitrarily long messages with a fixed security parameter n). If this is the case we call this a *fixed-length private-key encryption scheme for messages of length* $\ell(n)$.

Example 2.2. The one time pad is a fixed length private-key encryption scheme with of length $\ell(n) = n$. For each security parameter n, we have that $Enc_k(m) = m + k \in (\mathbb{Z}/2\mathbb{Z})^n$ is defined for $m \in \{0,1\}^n = (\mathbb{Z}/2\mathbb{Z})^n$. The random variable $Gen(n)$ returns a random key $k \in (\mathbb{Z}/2\mathbb{Z})^n$ in polynomial time, in fact in linear time $O(n)$, as each bit can be randomly chosen in $O(1)$-time.

We assume that our adversaries are also randomized algorithms running in Polynomial time as follows.

Definition 2.3. An *efficient* adversary to a computational encyrption scheme $\Pi = (\text{Gen}, \text{Enc}, \text{Dec})$ is an adversary $\mathcal{A}$ as given in Definition 1.8 where for each m_0, m_1 of the same length and $c \in \mathcal{C}$ we have that the random variable $\mathcal{A}(m_0, m_1, c)$ returns a random output in $\{0,1\}$ and it runs in polynomial time (it is polynomial in $\max\{|m_0|, |m_1|, |c|\}$).

We now modify the experiments Exp_0 and Exp_1 as follows.

Definition 2.4. Let $\mathcal{A}$ be an efficient adversary to an encryption scheme $\Pi = (\text{Gen}, \text{Enc}, \text{Dec})$. For each fixed security parameter n, the adversary chooses two messages $m_0, m_1 \in \mathcal{M}$ of the same length and, in case the scheme is a fixed-length $\ell(n)$, we also require the messages to have length $|m_0| = |m_1| = \ell(n)$. We define two experiments, Exp_0 and Exp_1 as follows. For fixed $b \in \{0,1\}$ we define the random variable Exp_b by the following experiment:

(1) Choose $k \in \mathcal{K}$ randomly according to $\text{Gen}(n)$.
(2) Choose $c_b \in \mathcal{C}$ randomly according to $\text{Enc}_k(m_b)$.
(3) Now send c_b to the adversary. They then choose $b' \in \{0,1\}$ randomly according to $\mathcal{A}(m_0, m_1, c_b)$.
(4) Now define $\text{Exp}_b \in \{0,1\}$ to be 1 if $b = b'$ and 0 if $b \neq b'$.

For such fixed m_0, m_1 and n, we define a random variable $\text{PrivK}_{\mathcal{A},\Pi}$ as follows:

(1) Choose $b \in \{0,1\}$ uniformly randomly.
(2) Run Exp_b as above.
(3) Now define $\text{PrivK}_{\mathcal{A},\Pi} \in \{0,1\}$ to be the result of Exp_b.

Thus the difference now is that there are constraints on what messages m_0 and m_1 may choose and they depend on the (known) security parameter n. This is to ensure that the adversary can't trivially "win" just by looking at the text length of m_0, m_1 and c_b. That is, text-length is not securely hidden by the encryption scheme.

Definition 2.5. A function $f : \mathbb{Z}_{\geq 0} \to \mathbb{R}_{>0}$ is said to be negligible if for all $c > 0$ there exists $N > 0$ such that for all $n > N$ we have $f(n) < n^{-c}$.

For instance 2^{-n} is negligible.

Definition 2.6. We say that a computational encryption scheme is *EAV-secure* if for all efficient adversaries there exists a negligible function $negl : \mathbb{Z}_{\geq 0} \to \mathbb{R}_{>0}$ such that

$$\left| Pr(\text{PrivK}_{\mathcal{A},\Pi} = 1) - \frac{1}{2} \right| < negl(n),$$

where n denotes the security parameter (note that $\text{PrivK}_{\mathcal{A},\Pi}$ is a random variable for each fixed security parameter n).