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 $(\mathrm{Gen}, \mathrm{Enc}, \mathrm{Dec})$ 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 $\mathrm{Enc}_k(m)$ taking values in $\mathcal{C}$.
- For each $k \in \mathcal{K}$, we have a map $\mathrm{Dec}_k : \mathcal{C} \to \mathcal{M}$ satisfying
$$\mathrm{Dec}_k(\mathrm{Enc}_k(m)) = m \quad \text{for all } m \in \mathcal{M}$$
.

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

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 $\mathrm{Enc}_k(m) \in \mathcal{C}$ to be the word obtained by adding $k \pmod{26}$ to each letter of m. So $\mathrm{Enc}_3(zac) = cdf$. So $\mathrm{Dec}_k = \mathrm{Enc}_{-k}$ 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
$$Pr(\mathrm{Enc}_k(m) = c) = Pr(\mathrm{Enc}_k(m') = c).$$
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 $\mathrm{Enc}_k(m)$ 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 $\mathrm{Enc}_k(m)$ to produce $c \in \mathcal{C}$. If Enc_k is deterministic ($\mathrm{Enc}_k(m)$ 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 $Pr(\mathrm{Enc}_k(m) = c) = Pr(k = 0) = \frac{1}{26}$, i.e., the probability that aa gets encoded into ff happens only if $k = 5$, so with probability $\frac{1}{26}$. However $Pr(\mathrm{Enc}_k(m') = c) = 0$ 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 a tuple $(\mathcal{A}, m_0, m_1)$ where $m_0, m_1 \in \mathcal{M}$ and for each $c \in \mathcal{C}$ $\mathcal{A}(c)$ is a random variable taking values in $\{0, 1\}$. Given this adversary, 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 $\mathcal{A}(c_b)$.
(4) Now define $\mathrm{Exp}_b \in \{0, 1\}$ to be 1 if $b = b'$ and 0 if $b \neq b'$.

We now 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 pass (m_0, m_1) to a *challenger* who randomly choose $b \in \{0, 1\}$ and passes m_b into the encryption scheme and gets a corresponding c_b.

The challenger then sends back c_b to the adversary (but it doesn't send back 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 challenger does). If they can succeed (i.e., $(b == b')$) with probability greater than $\frac{1}{2}$ (better than just a random guess), then the Proposition says that the scheme is not perfect.

To avoid confusing the different probabilitities, we let

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

denote the probability that $\mathcal{A}(c) = b'$ 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

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

likewise, the second term is

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

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

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

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:

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

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 $Pr_{\mathcal{A}}(\mathcal{A}(c) = \mathbb{1}_{\{c_0\}}(c)$ 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. $\square$

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 $\Pi = (\mathrm{Gen}, \mathrm{Enc}, \mathrm{Dec})$ such that

- For each non-negative integer n, we have that $\mathrm{Gen}(n)$ is a random variable with values in a set $\mathcal{K}$ called the keyspace. We assume $\mathrm{Gen}(n)$ 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 security parameter n, we have a subset of $\mathcal{M}$ that we call the set of *n-encryptable* messages.
- For each $k \in \mathcal{K}$ and n-encryptable $m \in \mathcal{M}$, the random variable $\mathrm{Enc}_k(m)$ 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 $\ell(n)$*. Basically it means that if m is n-encryptable then it must have 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 $\Pi = (\mathrm{Gen}, \mathrm{Enc}, \mathrm{Dec})$ conists of, for each non-negative integer n, a tuple $(m_0(n), m_1(n), \mathcal{A}(n))$ satisfying the following.

- Each $m_0(n), m_1(n)$ are messages in $\mathcal{M}$ of the same length and this length is polynomial in n.
- For each $c \in \mathcal{C}$ we have that a random variable $\mathcal{A}(n)(c)$ that returns a random output in $\{0, 1\}$ and it runs in polynomial time (it is polynomial in $\max\{n, |c|\}$).
- If Π is a fixed-length private-key encryption scheme of length $\ell(n)$, then $|m_0(n)| = |m_1(n)| = \ell(n)$.

Warning! An upcomming abuse of language: For the same of simplicity of notation, we supress the parameter n and denote $m_b(n)$ by m_0 (for $b = 0, 1$) etc. We also abuse the language by calling $\mathcal{A}$ the adversary (even though it also depends on m_0, m_1, again this is implicit suppressed data... it's just like when we say that X is a topological space rather than a tuple (X, τ)).

We now modify the experiments Exp_0 and Exp_1 as follows.

Definition 2.4. Let $(m_0, m_1, \mathcal{A})$ be an efficient adversary to an encryption scheme $\Pi = (\mathrm{Gen}, \mathrm{Enc}, \mathrm{Dec})$. For each fixed security parameter n, the adversary chooses two messages $m_0 = m_0(n)$ and $m_1 = m_1(n)$ 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 $\mathrm{Gen}(n)$.
(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 $\mathcal{A}(c_b)$.
(4) Now define $\mathrm{Exp}_b \in \{0, 1\}$ 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 $\mathrm{PrivK}_{\mathcal{A},\Pi}$ 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.

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 $\mathcal{A}$ there exists a negligible function $negl : \mathbb{Z}_{\geq 0} \to \mathbb{R}_{>0}$ such that

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

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

In other words, we now do not require the adversary to be no better than a random guess, we just require it to be at most negligibly better than a random guess when running a polynomial time guessing algorithm. We denote the quantity above by

$$SSAdv^*(\mathcal{A}, \Pi) := \left| Pr(\mathrm{PrivK}_{\mathcal{A},\Pi} = 1) - \frac{1}{2} \right|.$$

This notation is used by Boneh-Shoup and the $SSAdv^*$ refers to the term *Semantic Security Advantage* of the adversary $\mathcal{A}$ for the encryption scheme Π. Semantic Security is another term for EAV security. We define

$$SSAdv(\mathcal{A},\Pi) = \left| Pr_{\mathrm{Exp}_0}(\mathcal{A}=1) - Pr_{\mathrm{Exp}_1}(\mathcal{A}=1) \right|.$$

It is easy to show that

$$SSAdv^*(\mathcal{A},\Pi) = \frac{1}{2} \cdot SSAdv(\mathcal{A},\Pi),$$

thus we could use $SSAdv$ instead of $SSAdv^*$ in the definition of EAV-secure.

3. Some consequences of EAV-security

3.1. **Message recovery.** Suppose that we have a computational encryption scheme $\Pi = (Gen, Enc, Dec)$ that is EAV-secure. Let us show that it cannot be hacked by an efficient hacker as follows. Suppose that $\mathcal{H}$ is the hacker's function that attempts to discover the original message m based purely on the encrypted ciphertext c . More formally, for each $c \in \mathcal{C}$ and security parameter n we have a random variable $\mathcal{H}_n(c)$ that returns in polynomial time (polynomial in n and $|c|$) an element in $\mathcal{M}$. Thus $\mathcal{H}_n(c)$ is what the hacker thinks is m, based on the encrypted ciphertext c (the security parameter n is public knowledege?). We can't expect $\mathcal{H}$ to never succeed, it could be a constant function $\mathcal{H}(c) =$ "*hello*" independent of c and thus it is a broken clock that is somtimes right, especially if "*hello*" is a popular word. What we want is the probability of success to be very small for a random message $\mathcal{M}$ for any probability distribution on $\mathcal{M}$ that does not have very popular words (e.g., uniform across a large subset of $\mathcal{M}$).

Proposition 3.1. Suppose that we have, for each security parameter n, a probability $P_{\mathcal{M}}$ on $\mathcal{M}$ that is supported on the n-encryptable messages and is samplable in time that is polynomial in n (there is a polynomial time random algorithm generating this distribution). Let $\epsilon > 0$ such that $P_{\mathcal{M}}(m) < \epsilon$ for all $m \in \mathcal{M}$. Assuming that our encryption scheme is EAV-secure, we must have

$$Pr(\mathcal{H}_n(Enc_k(m)) = m) < \epsilon + negl(n)$$

for some negligible function $negl$, where n is the security parameter of the encryption scheme.

More formally, for each security parameter n the Pr is a distribution on $\mathcal{M} \times \mathcal{K} \times \mathcal{C} \times \mathcal{M}$. That is, we choose randomly according to $\mathcal{M}$ the message m, then randomly choose a key k (using the encryption scheme's $Gen(n)$) followed by choosing $c = Enc_k(m)$ (the $Enc_k(m)$ could have a random output c) and then having the hacker choose $m' = \mathcal{H}_n(c)$ (also randomly). In this case $Pr((m,k,c,m'))$ is the probability for getting those particular m,k,c,m' in this process.

Proof of Proposition. We use the hacker $\mathcal{H}_n$ to construct an adversary as defined in EAV-security and then use that to derive the bound. Given n-encryptable messages m_0 and m_1 we define an adversary $\mathcal{A}(c)$ which outputs the bit $b' \in \{0,1\}$ equal to 1 if $\mathcal{H}_n(c) = m_1$ and it outputs 0 otherwise. We now compute the $SSAdv$ of this adversary. By definition

$$Pr_{Exp_1}(\mathcal{A}=1) = Pr(\mathcal{H}(Enc_k(m_1)) = m_1)$$

where in this Pr the m_1 is fixed.

On the other hand,

$$Pr_{Exp_0}(\mathcal{A}=1) = Pr(\mathcal{H}(Enc_k(m_0)) = m_1).$$

This means that

$$SSAdv(\mathcal{A}_{m_0,m_1}, \Pi) = |Pr(\mathcal{H}(Enc_k(m_1)) = m_1) - Pr(\mathcal{H}(Enc_k(m_0)) = m_1)|\,.$$

Thus we have a bound

$$Pr(\mathcal{H}(Enc_k(m_1)) = m_1) \le SSAdv + Pr(\mathcal{H}(Enc_k(m_0)) = m_1).$$

We now integrate this bound over $m_1 \in \mathcal{M}$ to get

$$\begin{aligned}\sum_{m_1} P_{\mathcal{M}}(m_1) Pr(\mathcal{H}(Enc_k(m_1)) = m_1) &\le \max_{m_1 \in \mathcal{M}} SSAdv + \sum_{m_1} P_{\mathcal{M}}(m_1) Pr(\mathcal{H}(Enc_k(m_0)) = m_1) \\ &\le \max_{m_1 \in \mathcal{M}} SSAdv + \epsilon \sum_{m_1} Pr(\mathcal{H}(Enc_k(m_0)) = m_1) \\ &= \max_{m_1 \in \mathcal{M}} SSAdv + \epsilon.\end{aligned}$$

But $\max_{m_1 \in \mathcal{M}} SSAdv$ is a negligible function by the EAV assumption, thus completing the proof. Note there is a subtelty: a maximum of infinitely many negligible functions is not negligible, however here for each fixed security parameter n, we choose one such pair of messages $m_0(n), m_1(n)$ in polynomial time that maximizes this quantity, thus constructing a single adversary (remember $m_0(n)$ and $m_1(n)$ is part of the data of an adversary). □

3.2. **Leaking information of a message.** Let us suppose more generally now that our hacker does not want to recover the message, but just wants to know some properties of this message, which we define to be a function $f : \mathcal{M} \to \{0, 1\}$. For example, can the hacker find the sum of bits mod 2? Of course the hacker can always guess and he may be right with probabilitiy $\frac{1}{2}$, but that's not really hacking. The next result shows that for an EAV-secure a computationally efficient (polynomial time) hacker cannot do much better than random guessing.

Proposition 3.2. Let $\Pi = (Gen, Enc, Dec)$ be an EAV-secure encryption scheme and suppose that, for each fixed security parameter n, we have a probability distribution $P_{\mathcal{M}}$ supported on the n-encryptable messages. Let $f : \mathcal{M} \to \{0, 1\}$ be any function and suppose that, for each security parameter n and ciphertext $c \in \mathcal{C}$, we have a polynomial time random variable $\mathcal{H}_n(c)$ taking values in $\{0, 1\}$ (polynomial time in n and the length of the input from $\mathcal{C}$). Then there exists another polynomial time random variable $\mathcal{H}'_n$ (it does not depend on $c \in \mathcal{C}$) such that

$$|Pr(\mathcal{H}_n(Enc_k(m)) = f(m)) - Pr(\mathcal{H}'_n = f(m))| \le negl(n)$$

for some negligible function $negl$. Here the randomness in the first probability is in m (due to $P_{\mathcal{M}}$) and the choice of key k, randomness of Enc_k as well as the possible randomness of $\mathcal{H}_n(c)$ itself. In the second probability, all the randomness is in m (due to $P_{\mathcal{M}}$) as well as $\mathcal{H}'_n$.

Proof. Fix any (n-encryptable) message m_0 and define

$$\mathcal{H}'_n = \mathcal{H}_n(Enc_k(m_0)).$$

Thus $\mathcal{H}'_n$ is a random variable due to the randomness of the key k and the random function Enc_k. Now define, for each message m the adversary $(m_0, m_1, \mathcal{A}_{m_1})$ to be

$$\mathcal{A}_{m_1}(c) = 1 - |\mathcal{H}_n(c) - f(m_1)| \in \{0, 1\}.$$

Observe that

$$Pr_{Exp_1}(\mathcal{A}_{m_1} = 1) = Pr(\mathcal{H}_n(Enc_k(m_1)) - f(m_1) = 0) = Pr(\mathcal{H}_n(Enc_k(m_1)) = f(m_1)).$$

On the other hand

$$Pr_{Exp_0}(\mathcal{A}_{m_1} = 1) = Pr(\mathcal{H}_n(Enc_k(m_0)) = f(m_1)) = Pr(\mathcal{H}'_n = f(m_1)).$$

Thus we have that

$$SSAdv(\mathcal{A}_{m_1}, \Pi) = |Pr(\mathcal{H}_n(Enc_k(m_1)) = f(m_1)) - Pr(\mathcal{H}'_n = f(m_1))|$$

is a negligible function of n for each fixed m_1. We now integrate this inequality over m_1 with respect to the distribution $P_{\mathcal{M}}$ to get the desired result. $\square$

Example 3.3 (bit sum)**.** Let $f : \{0,1\}^* \to \{0,1\}$ denote the bit-sum of m mod 2. We apply the previous Proposition to this function. We suppose that our EAV scheme has fixed length n, i.e., all n-encryptable messages have length n. We suppose that $P_{\mathcal{M}}$ is the uniform distribution on $\{0,1\}^n$. Let $\mathcal{H}'_n$ be a random variable with values in $\{0,1\}$. Observe that if m and m' are two messages differing at exactly one bit, then they have different values for f and so

$$Pr(\mathcal{H}'_n = f(m)) + Pr(\mathcal{H}'_n = f(m')) = 1.$$

By pairing up all elements in $\{0,1\}^n$ into pairs that differ by just one bit, we see that

$$\frac{1}{2^n} \sum_{m \in \{0,1\}^n} Pr(\mathcal{H}'_n = f(m)) = \frac{1}{2}.$$

Thus the above proposition says that that any randomized polynomial time hacker $\mathcal{H} : \mathcal{C} \to \{0,1\}$ that tries to guess $f(m)$ based on $\mathcal{H}(Enc_k(m))$ will only succeed with probability $\frac{1}{2} + negl(n)$.