

Cryptography Notes

Raffaele Castagna

Academic Year 2025-2026

Contents

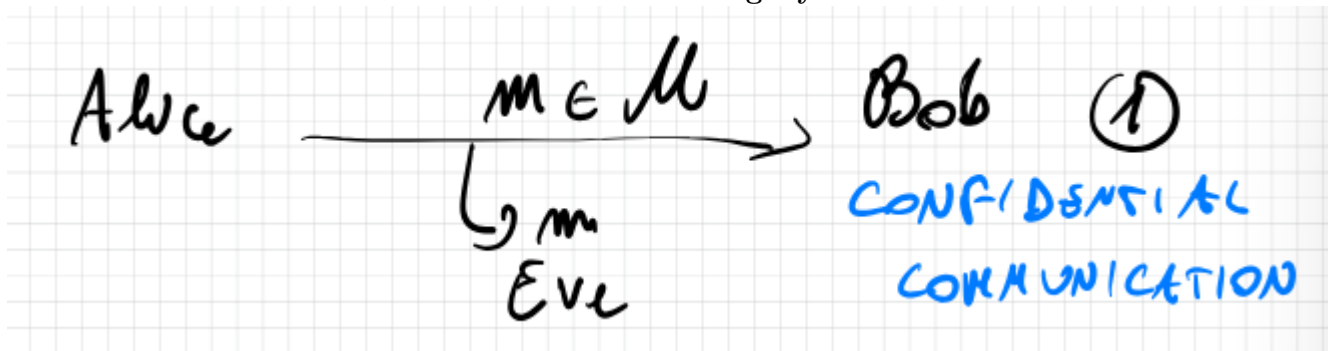
1	Intro to Cryptography	2
1.1	Secure Communication	2
1.2	Unconditional Security	2
1.3	Perfect Secrecy	3
1.4	OTP	3
1.5	Proof that the lemmas imply eachother	4
1.6	Message Authentication Codes	5
1.7	Randomness Extraction	6
2	Computational Security	9

1 Intro to Cryptography

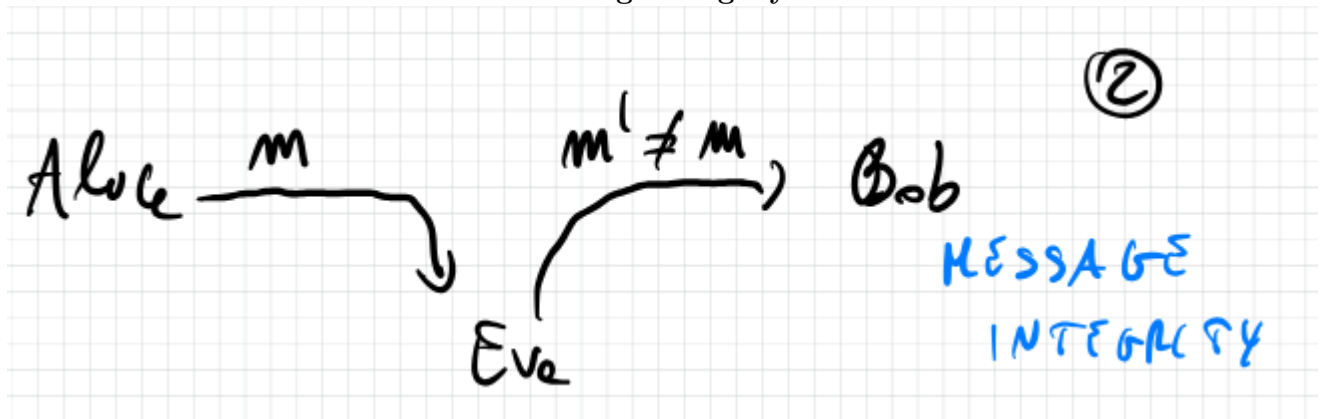
1.1 Secure Communication

We have multiple goals in cryptography, the most important ones being:

Confidential Integrity



Message Integrity



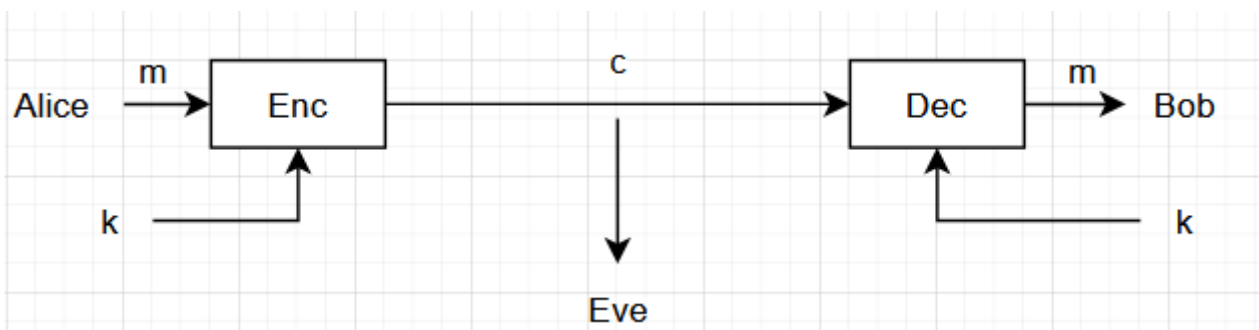
Basically we want our message to be both **confidential**, so no-one except the intended target sees it and we it to be unmodified, so that its **integrity** has not been compromised.

There are many different ways to do this, but in our case we only see two major ways:

- **Symmetric Cryptography:** Where Alice and Bob share a key $k \in \mathcal{K}$, the key is random and unknown to
- **Asymmetric Cryptography:** Where Alice and Bob do not share a key, but they have each their own key pair (p_k, s_k) where p_k is the public key and s_k is the secret/private key

1.2 Unconditional Security

To achieve confidential communication, we use symmetric cryptography.



With $m \in \mathcal{M}, c \in \mathcal{C}, k \in \mathcal{K}$

In this case we have Alice sending a message m which is then encrypted utilizing a randomly generate key k to generate the cyphertext c , after that to get back to the initial message m , Bob will then need to decrypt it utilizing his own key k on cyphertext c .

In a more formal way we can define Symmetric encryption (SKE) as $\Pi = (Enc, Dec)$ such that:

- $Enc : \mathcal{M} \times \mathcal{K} \rightarrow \mathcal{C}$
- $Dec : \mathcal{C} \times \mathcal{K} \rightarrow \mathcal{M}$
- k is uniform over \mathcal{K} (k is chosen according to some distribution)

An encryption scheme must satisfy the correctness requirement:

Definition 1. $\forall k \in \mathcal{K}, \forall m \in \mathcal{M}$ it holds that $Dec(k, Enc(k, m)) = m$

Kerchoff's Principle:

Definition 2. Security should not depend on the secrecy of the algorithm but on the secrecy of the key.

1.3 Perfect Secrecy

Definition 3. Let M be any distribution over \mathcal{M} and K be uniform over \mathcal{K} (Then observe $C = Enc(K, M)$ in a distribution over \mathcal{C}), we say that $(Enc, Dec) = \Pi$ is **perfectly secret** if $\forall M, \forall m \in \mathcal{M}, \forall c \in \mathcal{C} : Pr[M = m] = Pr[M = m | C = c]$ (The probability that M is m is equal to the probability that M is m knowing that C is c , so by knowing the cyphertext, we don't gain additional information).

Lemma 1. The following are equivalent:

- Perfect Secrecy
- M and C are independent
- $\forall m, m' \in \mathcal{M}, \forall c \in \mathcal{C} : Pr[Enc(k, m) = c] = Pr[Enc(k, m') = c]$ with k being uniform over \mathcal{K}

1.4 OTP

Let us see if OTP (One Time Pad) is perfectly secret

We know that the OTP uses \oplus to generate and later decypher the cyphertext, we have that $K = M = C = \{0, 1\}^N$ with N being the length of the string, we know that:

- $Enc(k, m) = k \oplus m$
- $Dec(k, c) = c \oplus k = (k \oplus m) \oplus k = m$

To prove that it is perfectly secret let us utilize the third lemma:

$$Pr[C = c | M = m'] = Pr[Enc_k(m') = c] = Pr[m' \oplus K = c] = Pr[K = m' \oplus c] = 2^{-N}$$

and therefore:

$$Pr[Enc(k, m') = c] = 2^{-N}$$

There seem to be some limitations, the key can only be used once and it must as long as the message, let's assume we encrypt m and m' : $c_1 = k \oplus m_1$ $c_2 = k \oplus m_2$ therefore $c_1 \oplus c_2 = m_1 \oplus m_2$, so if I know a pair (m_1, c_1) then I could compute m_2 , therefore we cannot encrypt two messages with the same key.

Theorem 1 (Shannon). Let Π be any perfectly secret SKE then we have $|\mathcal{K}| \geq |\mathcal{M}|$.

Proof. Take \prod to be uniform over \mathcal{M} . Take any c s.t. $\Pr[C=c] > 0$.
Consider $\mathcal{M}' = \{Dec(k, c) : k \in \mathcal{K}\}$ and assume $|\mathcal{K}| < |\mathcal{M}|$ by contraddiction, then:

$$|\mathcal{M}'| \leq |\mathcal{K}| < |\mathcal{M}| \rightarrow |\mathcal{M}'| < |\mathcal{M}| \rightarrow \exists m \in \mathcal{M} \setminus \mathcal{M}'$$

Now:

$$\Pr[M = m] = |\mathcal{M}|^{-1} \text{ but } \Pr[M = m|C = c] = 0$$

□

1.5 Proof that the lemmas imply eachother

Let us prove that $1 \implies 2 \implies 3 \implies 1$

Let us start by proving that $1 \implies 2$:

Proof. We know that $\Pr[M = m] = \Pr[M = m|C = c] \rightarrow \frac{\Pr[M=m \wedge C=c]}{\Pr[C=c]} = \Pr[M = m \wedge C = c] = \Pr[M = m] * \Pr[C = c]$ and therefore we have proved their independence, so $I(M; C) = 0$

□

Let us prove that $2 \implies 3$

Proof. Let us fix an m from M and c from C :

$$\Pr[Enc(K, m) = c] = \Pr[Enc(K, M) = c|M = m] \implies \Pr[C = c|M = m] = \Pr[C = c]$$

Remember that $Enc(\dots)$ is c !

We do the same thing for m' and we get: $\Pr[C = c|M = m] = \Pr[C = c]$ for both of them.
Therefore: $\Pr[Enc(K, m') = c] = \Pr[C = c]$

□

And now $3 \implies 1$: Take any c from C :

$$\Pr[C = c] = \Pr[C = c|M = m] \text{ by 2 (we are claiming this)}$$

If the claim is true then:

$$\Pr[M = m|C = c] * \Pr[C = c] = \Pr[M = m \wedge C = c] = \Pr[C = c|M = m] * \Pr[M = m] \implies$$

$$\implies \Pr[M = m] = \frac{\Pr[M=m|C=c]*\Pr[C=c]}{\Pr[C=c|M=m]}$$

However we still need to prove the claim:

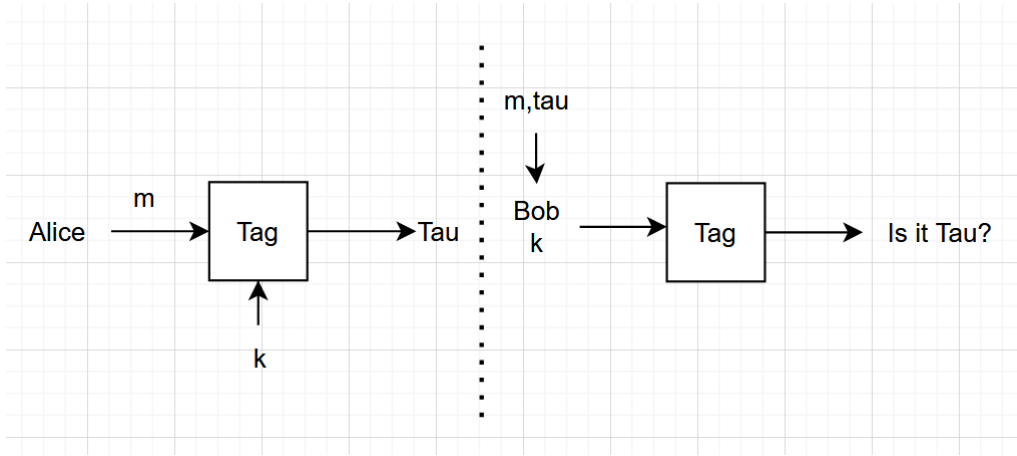
$$\Pr[C = c] = \sum_{m'} \Pr[C = c \wedge M = m'] = \sum_{m'} \Pr[C = c|M = m'] * \Pr[M = m'] =$$

$$\sum_{m'} \Pr[Enc(K, m') = c|M = m'] * \Pr[M = m'] = \sum_{m'} \Pr[Enc(K, m') = c] * \Pr[M = m']$$

$$\sum_{m'} \Pr[Enc(k, m) = c] * \Pr[M = m'] = \Pr[Enc(k, m) = c] * \sum_{m'} \Pr[M = m'] \Leftarrow 1$$

$$\Pr[Enc(k, m) = c] = \Pr[Enc(K, M) = c|M = m] \rightarrow \Pr[C = c|M = m]$$

1.6 Message Authentication Codes



In case it is τ then we accept it, else no.

There is no need to prove correctness as τ is deterministic, so if we had the same k and m , we should get the same τ

Unforgeability

It should be hard to forge τ' such on msg m' and it should be hard to produce (m, τ) as long as $m' \neq m$

Definition 4. Statistical secure MAC We say that $\Pi = \text{Tag}$ has ϵ -statistical security (unforgeability) if $\forall m, m' \in \mathcal{M}$ with $m \neq m' \forall \tau, \tau' \in \mathcal{T}$:

$$\Pr[\text{Tag}(K, m') = \tau' \mid \text{Tag}(K, m) = \tau] \leq \epsilon$$

TLDR: Fix any m, m' with $m' \neq m$ take τ, τ' on the condition that τ is tag of m and given τ' , it is always less than or equal to ϵ

Here ϵ is a parameter e.g. 2^{-80}

Exercise Let us prove that it is impossible to get $\epsilon = 0$

Because a random $\tau' \in \mathcal{T}$ has probability $\geq \frac{1}{|\mathcal{T}|}$ to be correct it is impossible.

Note that the definition is valid for One-Time!

We will show:

- The notion is Achievable
- It's inefficient, in fact:

Theorem 2. Any t -time $2^{-\lambda}$ statistically secure Tag has a key of some $(t+1)^*\lambda$

We will now show that any form of hash function with a particular property satisfies the definition.

Definition 5. Pairwise independence A family $\mathcal{H} = \{h_k : \mathcal{M} \rightarrow \mathcal{T}\}_{k \in \mathcal{K}}$ is pairwise independent if: $\forall m, m' \in \mathcal{M}$ s.t. $m \neq m'$ then: $(h(K, m), h(K, m'))$ is uniform over $\mathcal{T}^2 = \mathcal{T} \times \mathcal{T}$ for K uniform over \mathcal{K}

Theorem 3. Any family \mathcal{H} of pairwise independent functions directly gives a $\epsilon = \frac{1}{|\mathcal{T}|}$ - statistically secure MAC.

Proof. Fix any $m \in \mathcal{M}, \tau \in \mathcal{T}$:

$$\Pr[\text{Tag}(K, m) = \tau] =$$

$$Pr_k[h(K, m) = \tau] = \frac{1}{|\mathcal{T}|} \text{ by pairwise independence}$$

Similiarly, for any m, m' s.t. $m \neq m', \tau, \tau' \in \mathcal{T}$.

$$\begin{aligned} Pr_k[Tag(K, m) = \tau \wedge Tag(K, m') = \tau'] &= \\ Pr_k[h(K, m) = \tau \wedge h(K, m') = \tau'] &= \frac{1}{|\mathcal{T}|^2} \end{aligned}$$

By Bayes:

$$\begin{aligned} Pr[Tag(K, m') = \tau' | Tag(K, m) = \tau] &= \\ \frac{Pr[h(K, m') = \tau' \wedge h(K, m) = \tau]}{Pr[h(K, m) = \tau]} &= \\ \frac{\frac{1}{|\mathcal{T}|^2}}{\frac{1}{|\mathcal{T}|}} &= \frac{1}{|\mathcal{T}|} \end{aligned}$$

□

Now we need to instantiate it, here is a construction, Let p be a prime:

$$\begin{aligned} h_{a,b}(m) &= am + b \pmod{p} \\ k = (a, b) &\in \mathbb{Z}_p^2 = \mathcal{K} \\ \mathbb{Z}_p &= \mathcal{M} = \mathcal{T} \end{aligned}$$

Lemma 2. *The above \mathcal{H} is pairwise independant.*

Proof. For all $m, m' \in \mathbb{Z}_p, \tau, \tau' \in \mathbb{Z}_p$ with $m \neq m'$

$$\begin{aligned} Pr_{(a,b) \in \mathbb{Z}_p^2} [h_{a,b}(m) = \tau \wedge h_{a,b}(m') = \tau'] &= \\ Pr_{(a,b) \in \mathbb{Z}_p^2} \left[\begin{pmatrix} m & 1 \\ m' & 1 \end{pmatrix} \begin{pmatrix} a \\ b \end{pmatrix} = \begin{pmatrix} \tau \\ \tau' \end{pmatrix} \right] &= \\ Pr_{(a,b) \in \mathbb{Z}_p^2} \left[\begin{pmatrix} a \\ b \end{pmatrix} = \begin{pmatrix} m & 1 \\ m' & 1 \end{pmatrix} \begin{pmatrix} \tau \\ \tau' \end{pmatrix} \right] &= \\ \frac{1}{p^2} = \frac{1}{|\mathbb{Z}_p|^2} = \frac{1}{|\mathcal{T}|^2} \end{aligned}$$

□

1.7 Randomness Extraction

Alice and Bob need a **random** key, how can they generate it?

Randomness is crucial for crypto, and two components are necessary in any RNG (e.g. Fortuna, /dev/rand):

- Randomness extraction: By measuring physical quantities we can get an **unpredictable** sequence of bits (Not necessarily uniform or for cheap!)
From this we extract a **random** Y which is short (e.g. 256 bits)
- Expand it to any amount (polynomial) using a pseudorandom generator (PRG) - but this requires computational assumptions.

We want to understand how to extract from an unpredictable source X .

Example Von Neumann Extractor Assume $B \in \{0, 1\}$ s.t. $\Pr[B = 0] = p < \frac{1}{2}$.

- Sample $b_1 \in B, b_2 \in B$
- if $b_1 = b_2$ then Resample
- Else output 1 $\iff b_1 = 0, b_2 = 1$, or 0 if $b_1 = 1, b_2 = 0$

Assuming it outputs something, this will be s.t.

$$\Pr[\text{Output } 0] = \Pr[\text{Output } 1] = p * (1 - p)$$

$$\Pr[\text{No output after } N \text{ tries}] = (1 - 2p(1 - p))^N \text{ which becomes small for large enough } N$$

We want to generalize this question, ideally we want to design a function Ext that takes a random variable X and outputs an uniform $\text{Ext}(X)$, but this is impossible as the source must be unpredictable and Ext is deterministic

Definition 6 (Min-Entropy). The min-entropy of X is: $H_\infty = -\log_2 \max \Pr[X = x]$

Example: Let $X \equiv U_m$ Uniform over $\{0, 1\}^N$. $H_\infty(X) = N$

If X is a constant we have $H_\infty(X) = 0$

Here's the next best thing:

Design Ext that extracts UNIFORM $Y = \text{Ext}(X)$ for every X s.t. $H_\infty(X) \geq k$

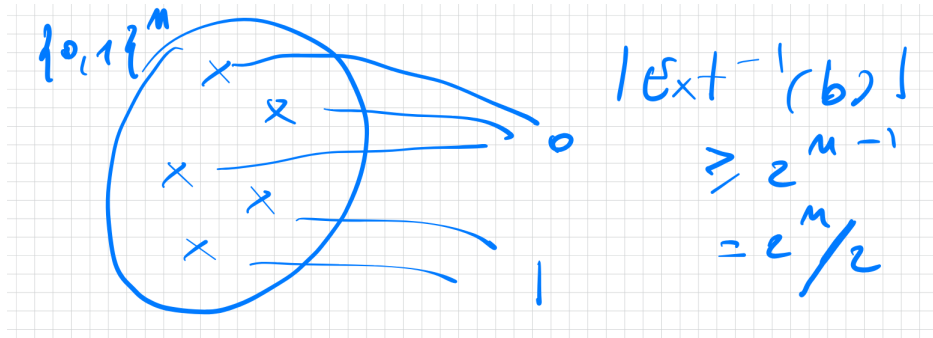
But this is also impossible, even if

$$\text{Ext}(X) = b \in \{0, 1\}$$

$$k = n - 1$$

$$x \in \{0, 1\}^n$$

And here's why: fix any $\text{Ext}: \{0, 1\}^n \rightarrow \{0, 1\}$ and let $b \in \{0, 1\}$ be the output of maximizing $|\text{Ext}^{-1}(b)|$



The bad X : Define X to be Uniform over $\text{Ext}^{-1}(b)$. Since it is uniform: $H_\infty(X) \geq n - 1$ but $\text{Ext}(X) = b$ so not uniform.

Solution: Swap the quantifiers.

Definition 7 (Seeded Extractor). A function $\text{Ext} : \{0, 1\}^n \times \{0, 1\}^d \rightarrow \{0, 1\}$ is a (k, ϵ) -seeded extractor if for every X over s.t. $H_\infty(X) \geq k$:

$$(S, \text{Ext}(S, X)) \approx_\epsilon (S, U_e)$$

for $S \equiv U_d$ (uniform over $\{0, 1\}^d$). (Note that $(S, U_e) \equiv U_{d+\epsilon}$).

What does this mean? There is a standard way to measure distance between distributions:

$$Z \equiv_{\epsilon} Z' \iff SD(Z, Z') \leq \epsilon$$

$$SD(Z, Z') = \frac{1}{2} \sum_z |Pr[Z = z] - Pr[Z' = z]|$$

This is equivalent: \forall Unbounded adversary A :

$$|Pr[A(z) = 1 : z \in Z] - Pr[A(z) = 1 : z \in Z']| \leq \epsilon$$

Theorem 4 (Leftover Hash Lemma). *Let $\mathcal{H} = \{h_s : \{0, 1\}^n \rightarrow \{0, 1\}^l\}_{s \in \{0, 1\}^d}$ be a family of pairwise independent hash functions. Then $Ext(x, s) = h_s(x)$ is a (k, ϵ) -seeded extractor for $k \geq l + 2 \log_2(\frac{1}{\epsilon}) - 2$.*

Lemma 3. *Let Y be a RV over \mathcal{Y} . Such that:*

$$Col(Y) = \sum_{y \in \mathcal{Y}} Pr[Y = y]^2 \leq \frac{1}{|\mathcal{Y}|} * (1 + 4\epsilon^2)$$

Then, $SD(Y, U) \leq \epsilon$

Proof.

$$SD(Y, U) = \frac{1}{2} \sum_{y \in \mathcal{Y}} |Pr[Y = y] - Pr[U = y]|$$

$$\frac{1}{2} \sum_{y \in \mathcal{Y}} |Pr[Y = y] - \frac{1}{|\mathcal{Y}|}|$$

$$\text{Let } q_y = Pr[Y = y] - \frac{1}{|\mathcal{Y}|}$$

$$\text{Let } s_y = \begin{cases} 1 & \text{if } q_y \geq 0 \\ -1 & \text{else} \end{cases}$$

$$\text{Hence } SD(Y, U) = \frac{1}{2} \sum_{y \in \mathcal{Y}} s_y q_y$$

$$\begin{aligned} &= \frac{1}{2} \langle s, q \rangle \leq \frac{1}{2} \sqrt{\langle \vec{q}, \vec{q} \rangle * \langle \vec{s}, \vec{s} \rangle} \text{ by Cauchy-Schwarz} \\ &= \frac{1}{2} \sqrt{\sum_{y \in \mathcal{Y}} q_y^2 * |\mathcal{Y}|} \end{aligned}$$

Now, We analyze the term $\sum_{y \in \mathcal{Y}} q_y^2$:

$$\begin{aligned} \sum_{y \in \mathcal{Y}} q_y^2 &= \sum_{y \in \mathcal{Y}} (Pr[Y = y] - \frac{1}{|\mathcal{Y}|})^2 = \\ &= \sum_{y \in \mathcal{Y}} Pr[Y = y]^2 + \frac{1}{|\mathcal{Y}|^2} - 2 \frac{Pr[Y = y]}{|\mathcal{Y}|} = \\ &= \underbrace{\sum_{y \in \mathcal{Y}} Pr[Y = y]^2}_{Col(Y)} + \frac{1}{|\mathcal{Y}|} - 2 \frac{1}{|\mathcal{Y}|} = \\ &= Col(Y) - \frac{1}{|\mathcal{Y}|} \leq \frac{4\epsilon^2}{|\mathcal{Y}|} \end{aligned}$$

Then:

$$SD(Y, U) \leq \frac{1}{2} \sqrt{\frac{4\epsilon^2}{|\mathcal{Y}|} * |\mathcal{Y}|} = \epsilon$$

□

Next we apply the lemma to prove the Leftover Hash Lemma:

Proof.

$$Y = (S, \text{Ext}(X, S)) = (S, h(S, X))$$

and compute $\text{Col}(Y)$:

$$\begin{aligned} \text{Col}(Y) &= \sum_{y \in \mathcal{Y}} \frac{1}{|\mathcal{Y}|} \Pr[Y=y] = \Pr[Y = Y'] \\ &= \Pr[S = S' \wedge h(S, X) = h(S', X')] \\ &= \Pr[S = S' \wedge h(S, X) = h(S, X')] \\ &= \Pr[S = S'] * \Pr[h(S, X) = h(S, X')] \\ &= \frac{1}{2^d} * \Pr[h(S, X) = h(S, X')] \\ &= \frac{1}{2^d} * (\Pr[X = X'] + \Pr[h(S, X) = h(S, X') \wedge X \neq X']) \\ &\leq \frac{1}{2^d} * \left(\frac{1}{2^k} + \frac{1}{2^l}\right) \text{ by pairwise independence and } H_\infty(X) \geq k \\ &= \frac{1}{2^{d+l}} (1 + 2^{l-k}) \leq \frac{1}{2^{d+l}} (2^{2-2\log_2(\frac{1}{\epsilon})} + 1) \\ &= \frac{1}{|\mathcal{Y}|} * (1 + 4\epsilon^2) \end{aligned}$$

□

2 Computational Security

We know that without any assumptions we can do Symmetric crypto and randomness generation, with some strong limitations.

- Privacy: $|msg| = |key|$ and one-time use
- Integrity: same as above.
- Randomness We can't extract more than k from $p_y k$

We want to overcome all these limitations. We'll do so off of the base of some assumptions

- Adversary is Computationally Bounded
- Hard Problems exist

We will make conditional statements:

Theorem 5. *If Problem X is hard (against efficient solvers), Then cryptosystem Π is secure (against efficient adversaries)*

Consequence: if Π is insecure, \exists efficient solver for X !

Depending on what X is, the above could be **Groundbreaking**.

Examples:

$X = "P \neq NP"$ $X = "Factoring \text{ is hard}"$

$X = "Discrete Log \text{ is hard}"$

We are not able to just assume $P \neq NP$, we need a stronger assumption: **One-Way Functions:**
These are functions that are easy to compute but hard to invert.

Clearly $OWF \implies P \neq NP$, why?

Because if $P = NP$, OWF do not exist as checking if $f(x) = y$ is efficient and this it's in $NP=P$. We cannot exclude that $P \neq NP$ but still, OWF do not exist.

To better demonstrate this, we can refer to the following worlds created by Russel Impagliazzo:

- Algorithmica: $P=NP$
- Heuristica: $P \neq NP$ but no "average-hard" problems
- Pessiland: $P \neq NP$ and "average-hard" problems exist, but no OWF
- Minicrypt: OWFs exist
- Cryptomania: OWF exist + Public-key crypto exist

First we must start by fixing a model of computation: Turing Machines
efficient computation = polynomial time TMs.

Let's be generous: Adversaries can use any amount (polynomial) of randomness: Probabilistic Polynomial Time (PPT) TMs.

In what comes next we could define two approaches:

- **Concrete Security** Security holds w.r.t. t -time TMs except w.p. $\leq \epsilon$ (e.g. $t = 2^{80}$ steps, $\epsilon = 2^{-80}$)
- **Asymptotic Security** Let λ be a security parameter. Adversaries are $\text{poly}(\lambda)$ -time PPT TMs ($\epsilon = \text{negligible} = \text{negl}(\lambda)$)

Definition 8 (Negligible). $\epsilon : \mathbb{N} \rightarrow \mathbb{R}$ is negligible if $\forall p(\lambda) = \text{poly}(\lambda) \exists \lambda_0 \in \mathbb{N} \text{ s.t. } \forall \lambda > \lambda_0 : \epsilon(\lambda) \leq \frac{1}{p(\lambda)}$

(In other words, $\epsilon(\lambda) \leq O(\frac{1}{p(\lambda)}) \forall p(\lambda) = \text{poly}(\lambda)$)