Date 6/6/18

Block Cipher, Hash Function, MAC, AEAD - Srimanta Bhattacharya

- Collision-Resistant Hash Functions :

① Definition $h: D \rightarrow R \quad |D| >> |R| = N$

② Security Notions :

Collision Resistance → Secondary Primary Resistance
Primary Resistance ←

③ Birthday Paradox

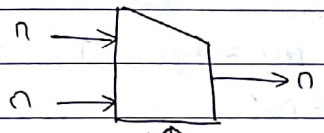
④ Birthday Attack

$$T = O(\sqrt{N}), \quad \varepsilon \approx 1/2$$

$$\varepsilon \geq \frac{q(q-1)}{4N} = \frac{q^2}{4N} \quad \therefore N \geq q^2/4$$

$$\Rightarrow N \geq q$$

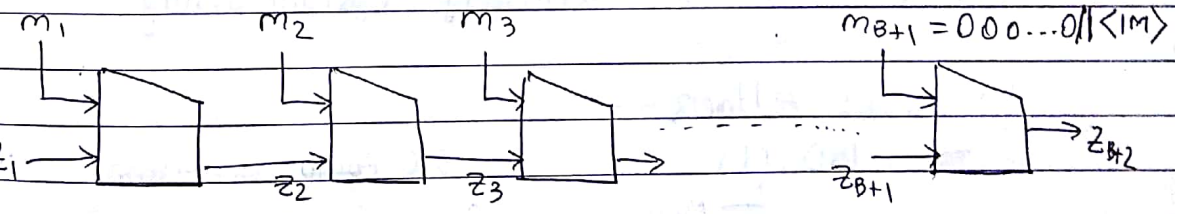
- Markle-Dangard Construction —



Compression
2n bits → n bits

We consider a hash function that compresses 2n bits to n bits.

Our goal is to construct a hash function which compresses an arbitrary length input to an output of n bits, generally 2^n bits to n bits.



Arbitrary string input to first block

We divide the message into m bit blocks.

The hash function in this construct concatenates mi and zi.

Number of Blocks → $B = \lceil |M| / n \rceil$

M: Total message
n: Length of blocks

We divide the message M into B blocks of length n. If there is space in the last block, we pad it with zeros.

$m_{B+1} = 000\ldots0||<M>$ → Length of the message padded with zeroes

Claim: If a single block is collision free, then the

entire construct is collision free.
→ We suppose there is a collision of two messages M and M', with $M \neq M'$.
There will arise two cases:
① M and M' are of equal length
② M and M' are of different length

Case 1: $|M| \neq |M'|$
∴ The last block for both messages will be different, i.e. $m_{B+1} \neq m'_{B+1}$ ($m_{B+1} = 00\ldots0 \| \langle |M| \rangle$), But there is a collision for $M \times M'$.
Hence, our original claim that if a single block is collision free, the entire construct is collision free is wrong.

Case 2: $|M| = |M'|$
∴ The last blocks for both messages will be same, i.e. $m_{B+1} = m'_{B+1}$.
We will see that $m_i = m'_i$, as z_i is same for all, Hence, finally $M = M'$.
Hence, from these two cases we have shown that if there is collision, either a single block is not collision free or the messages are equal.
→ The SHA-256 hash function is build based on this construct where each block size is 256 bits. Also z_i is some arbitrary constant string.

• Joux's Attack —
→ MD(M) — Function for Markle-Dangard Construction; M — Message
We concatenate MD(M) and MD(M). If MD(M) is of size n bits, we get 2n bits message.

$\underbrace{MD(M)}_{n} \| \underbrace{MD(M)}_{n} \rightarrow 2n$

According to birthday paradox to get a collision from $\{0,1\}^n$ we need to have $N = 2^{2n}$
$\therefore T = O(\sqrt{2^{2n}}) = O(2^n)$

$f: \{0,1\}^r \rightarrow \{0,1\}^n$

According to birthday paradox, the upper bound for getting collision is $\frac{q(q-1)}{4N}$. Here, $N = 2^{2n}$. So the collision is

roughly 1/4. if $q = 2^n$

Now, $O(n2^{n/2}) << O(2^n)$

We can show that we need approximately $n.2^{n/2}$ queries to show there is a collision.

→ $MD_1(M) \,||\, MD_2(M)$

We suppose there are $2^{n/2}$ messages that has MD_1 hash.

Two different Markle-Dangard Constructs → same message

$\therefore MD_1(M_1) = MD_1(M_2) = \ldots\ldots = MD_1(M_{2^{n/2}})$

For all these $2^{n/2}$ cases, the first part of the messages map to the same output.

To ensure collision occurs or not, we have to consider $MD_2(M)$

From Birthday Paradox, we know that if we try $2^{n/2}$ messages for $MD_2(M)$, then the chances of collision is high, greater than 1/2.

Hence, we might write $MD_1(M_i) = MD_2(M_j)$.

Our job is to find $2^{n/2}$ messages that will collide in the Markle Dangard construct. By birthday paradox out of these $2^{n/2}$, two messages will collide. We provide $2^{n/2}$ messages as input to each stage each and generate $2^{n/2}$ output messages. At each each stage, there are two messages which might collide Let for the first stage, the colliding messages be m_1^1 and m_1^2 and for the second stage be m_2^1 and m_2^2 and so on till $m_{n/2}^1$ and $m_{n/2}^2$ respectively.

$\therefore M_1 = m_1^1 \,||\, m_2^1 \,||\, \ldots\ldots \,||\, m_{n/2}^1$

$\therefore M_2 = m_1^2 \,||\, m_2^2 \,||\, \ldots\ldots m_{n/2}^2$

$m_1^1 \quad m_1^2$

$m_2^1 \quad m_2^2$

$\vdots \quad \vdots$

$m_{n/2}^1 \quad m_{n/2}^2$

Now, if we take m_i^1 and m_j^2, we will see that collision is bound to happen.

Hence, we have queried $n/2 * 2^{n/2}$ times to show that there is a collision among $2^{n/2}$ messages.

∴ Query complexity: $O(n/2 * 2^{n/2})$

- Block Cipher —

→ Given a string, x of length n bit

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

We randomly pick a string k of length n

$$K \xleftarrow{\$} \{0,1\}^n$$

Encryption: $x \oplus K$

→ We use one-time pad to encrypt a 1 bit message m

one bit message — m ⊕ $Pr[m=0] = p$ $Pr[m=1] = 1-p$ $m = \{0,1\}$

one bit key — b $Pr[b=1] = 1/2$ $Pr[b=0] = 1/2$ $b \xleftarrow{\$} \{0,1\}$

$C = m \oplus b$

$Pr[C=0] = 1/2$

$Pr[C=1] = 1/2$

Now,

$Pr[M=0 \mid C=0] = p$

$Pr[M=0 \mid C=1] = p$

$Pr[M=1 \mid C=1] = 1-p$

$Pr[M=0 \mid C=0] = 1-p$

The distribution in the message space does not change even after knowing the cipher text

Hence, this is a perfectly secret system.

→ Random Permutation RP:

$RP: \{0,1\}^n \longrightarrow \{0,1\}^n$

$RP(a \in \{0,1\}^n) = b$

$RP(a') = b' \in \{0,1\}^n \setminus \{b\}$

$a \neq a' \Rightarrow b \neq b'$ (Random Permutation, not random function)

The probability that we get 'a' the first time is $1/2^n$, i.e. $Pr\{RP(a') = b'\} = 1/2^n$

Number of permutations available in the domain is $(2^n)!$

→ we have a message of size n bits

$m \leftarrow \{0,1\}^n$

The key needs to identify the permutation of the message m. For random permutation, we have $(2^n)!$ permutations present.

$\therefore$ Size of key $= \log_2 (2^n!)$

$= \Theta(n \cdot 2^n)$

Hence, for perfect secrecy we need to have a key size of 64×2^{64}, assuming $n = 64$; which is very large.

To avoid this, we select a set of random permutations of cardinality 2^{64}. Hence the key is of 64 bit. The permutation generated from the 2^{64} set of elements should behave similar to the random permutation where the set size was $2^{64}!$

Hence, we use Pseudo-random sequences in block cipher, which can be broken if enough computation power is provided.

** How many queries do we need to distinguish between random function and random permutation

Block Cipher, Hash Function, MAC, AEAD — Mridul Nandi

Different Candidates for Message Authentication Code

- Error Detecting —

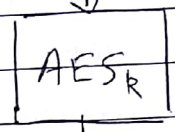
$m_0 m_1 \ldots\ldots m_q \quad \sum_{i=0}^{q} m_i$ ← XOR of all bits

m_i : detects if there is error or not

This is applicable only for statistical noise.

- 128-d

$M \| 0^d$



Message size is less than 128 we pad extra bits to get 128 bits as AES only recognizes 128 bits.

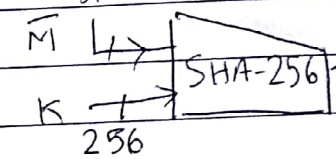
T

$T = AES(\overrightarrow{10\ldots0}^{M1})$

$= AES(\overrightarrow{10\ldots0}^{M2})$

We have to use injective padding, such that two different messages should not be same after padding is applied.

- 512



$\overline{M}$: Padded Message

K : Key

T : Tag

- MAC can be only implemented if Key is used.

• Broad Categories of MACs (arbitrary domain) -

→ Universal Hash-based:

These can be with or without Nonce

~ Poly 1305, UMAC, MMH etc

→ Block Cipher based:

~ Sequential (CBC type) -

ECBC, XCBC, TMAC, OMAC etc.

~ Parallel -

PMAC, XOR, DAG-based PRF etc.

→ Hash function based:

Here the hash function is used to compress the message, i.e. it is a compression type function.

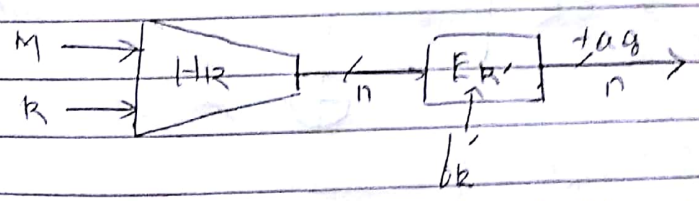
~ HMAC, NMAC, EMD, NI, Sandwich-MD, variants of cascade etc.

• Universal Hash Based MAC —

→ A universal hash function is used. Let H_K be a keyed hash function.

$$P_K[H_K(M) = H_K(M')] \leq \epsilon$$, which is very small, for $M \neq M'$

$$Tag = f_{K'}(H_K(M))$$



Let F be a finite field. Message $m = (m_1, \ldots, m_d) \in F^d$, where d depends on the size of the message.

Now, $K \xleftarrow{\in_{unif}} F$, i.e. K is uniformly randomly chosen from F

We now consider the Polyhash function $PH_K(m)$

$$PH_K(m) = Km_1 + K^2m_2 + \ldots + K^d m_d$$

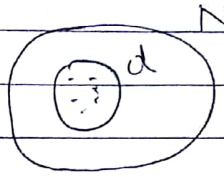
$$PH_K(m') = Km_1' + K^2m_2' + \ldots + K^d m_d'$$

$$Pr_K[PH_K(m) = PH_K(m')] = Pr_K[K(m_1 - m_1') + K^2(m_2 - m_2') + \ldots + K^d(m_d - m_d')]$$

$\therefore Pr\left[H_R(m) = H_R(m')\right] = 0 \leq \frac{d}{2^n}$ or $\frac{d}{N}$

where $|F| = N \approx 2^n$

and d = Message size.



From N things, d satisfies this property. The probability that d will be from N is given by $\leq \frac{d}{N}$

To compute the polynomial $Km_1 + K^2m_2 + \cdots + K^d m_d$ we normally need 2d multiplications. It can be computed within d multiplications, if the polynomial can be written as $K(\cdots K(Km_d + m_{d-1}) + m_{d-2} + \cdots + m_1)$

The advantage of this technique is that no IV is needed.

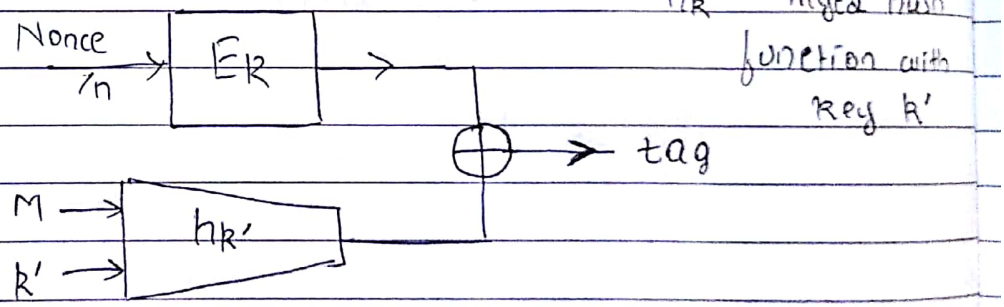
→ Now we consider a technique with a nonce

Secret key : (K, K')

27. Let $h_{K'}()$ be a universal hash function and N is the nonce.

$$Tag = f_K(N) + h_{K'}(M)$$

$h_{K'}$ → Keyed hash function with key K'



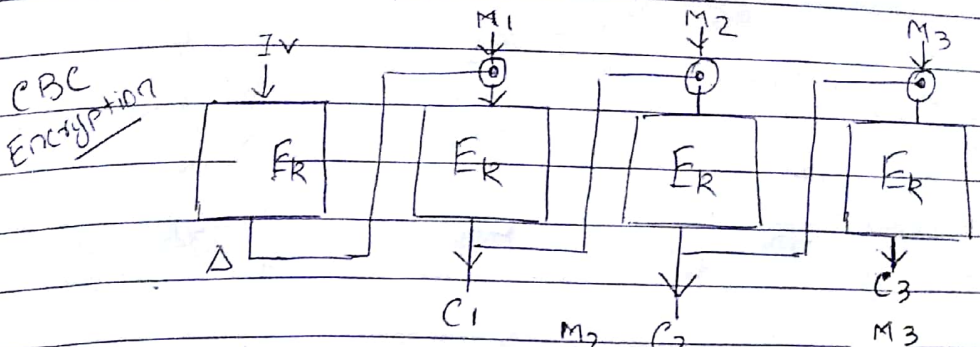
Here we use a δ-universal hash function instead of an universal hash function.

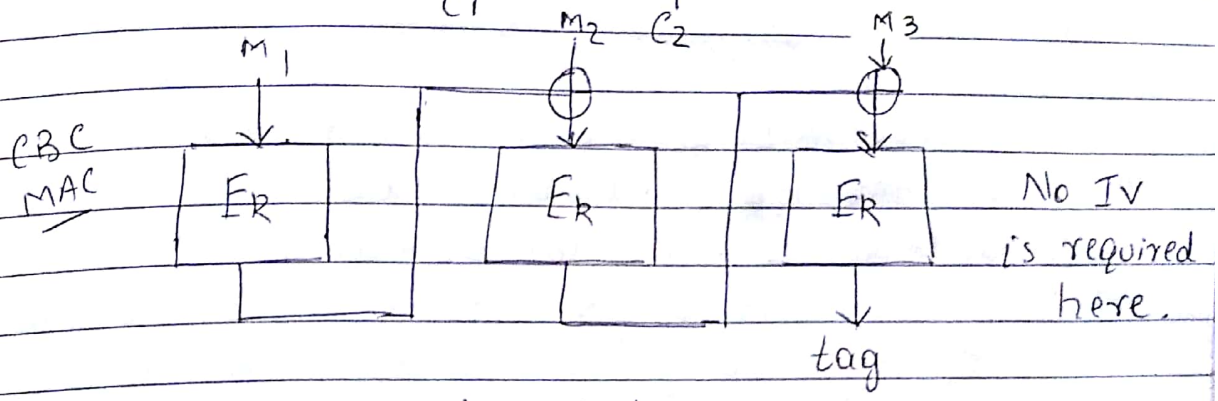
Polyhash is an example of δ-universal hash function

A δ-universal hash function is an universal hash function, but not the other way round.

• Block cipher based MAC —

→ CBC : Block Cipher based MAC :-

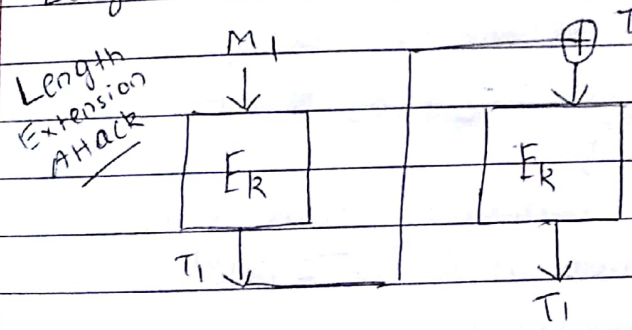




No IV is required here.

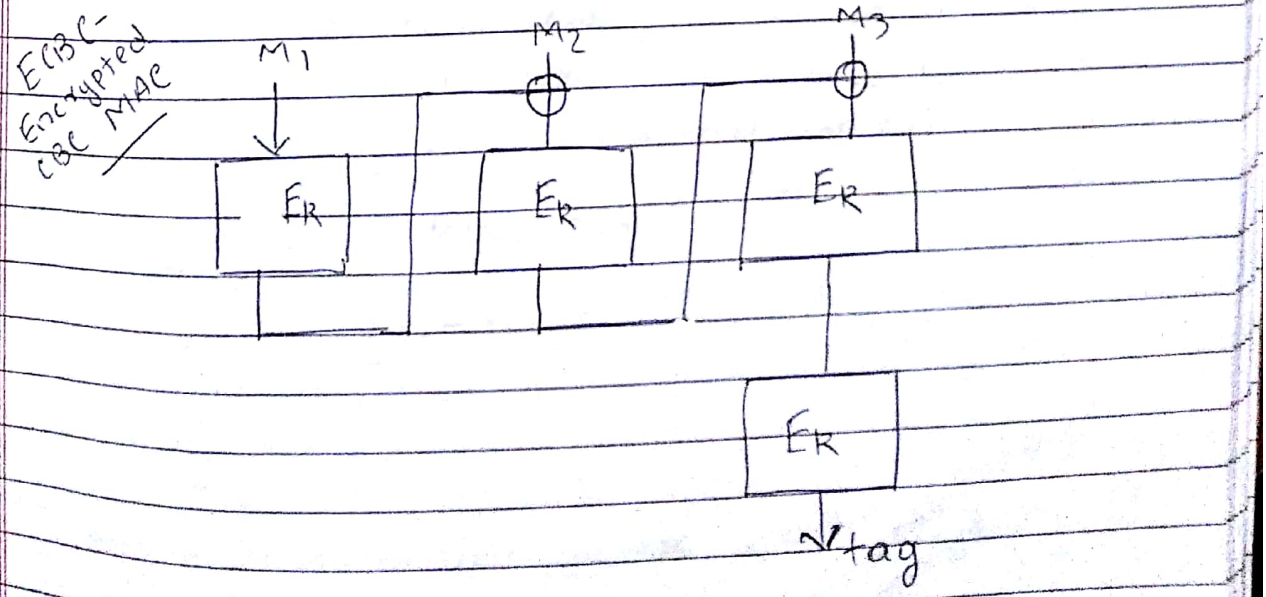
CBC MAC is secure for prefix-free message space and for fixed length.

CBC MAC is not secured for arbitrary domain : Length Extension Attack.



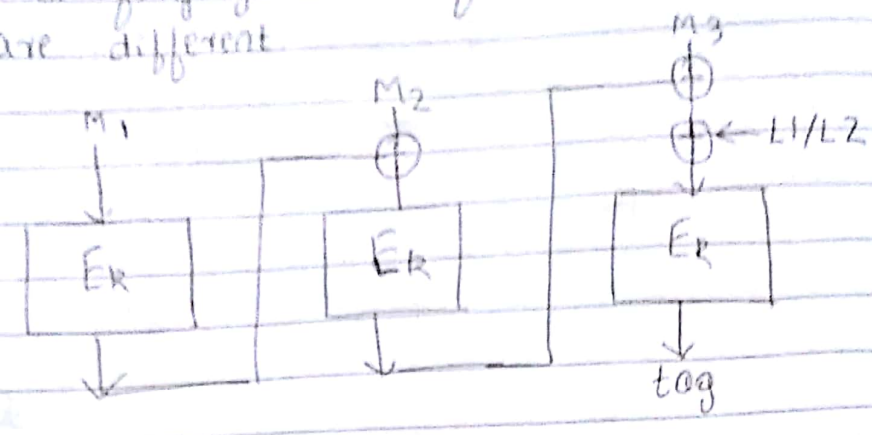
This is the Length Extension Attack.

For one query we know (M_1, T_1) Then we forge $(M_1 \| (M_1 \oplus T_1) , T_1)$



** Find a forgery attack for ECBC, when the keys are different

Remedies for ECBC



1. XCBC: K, L1, L2 are independent keys
2. TMAC: K, L1 independent, L2 = a·L1
3. OMAC: L1 = a EK(0), L2 = a·L1

→ To prevent length extension attack we rotate C_i before doing the XOR operation

• Probability —

→ Random Experiment:

X: Outcome ⟶ .... Sample Space Ω

Outcome comes from the sample space

We have to define $Pr[X = x] \; \forall \; x \in \Omega$

→ $F = \{f : S \longrightarrow T\}$

F is a set of functions from S to T.

Number of functions in F, $|F| = |T|^{|S|}$

If f is a random function, it is denoted as $f \xleftarrow{\$} F$

$x_1, x_2, \ldots, x_q$ are distinct

$y_1, y_2, \ldots, y_q$ may not be distinct

For a particular function f_0,

$$Pr[f = f_0] = \frac{1}{|T|^{|S|}}$$

$$\therefore Pr[f(x_1) = y_1, \ldots, f(x_q) = y_q] = \frac{|T|^{|S|-q}}{|T|^{|S|}}$$

$$= \frac{1}{|T|^q}$$

→ Let P be set of all permutations from T to T.

$P = \{\pi : T \rightarrow T\}$

Number of function permutations in P, $|P| = |T|!$

We select a random ... and uniform permutation from P.

$\pi \xleftarrow{\$} P$

$x_1, x_2, \ldots, x_q \leftarrow$ unique

$y_1, y_2, \ldots, y_q \leftarrow$ may not be unique

$$\therefore \Pr[\pi(x_1) = y_1, \ldots, \pi(x_q) = y_q] = \frac{(|T|-q)!}{|T|!}$$

$$= \frac{1}{N(N-1)\cdots(N-q+1)}$$

$\rightarrow X_1, X_2, \ldots, X_q \xleftarrow{\$} T$

$$\Pr[\exists i \neq j, X_i = X_j] = 1 - \prod_{i=1}^{q}(1 - i/N)$$

↑ This independently is the probability for no collision

Now, $1 - x \geq (1-\epsilon)e^{-x}$

Hence, we can write

$$1 - \prod_{i=1}^{q}(1 - i/N) \simeq 1 - \prod_{i=1}^{q} e^{-i/N}$$

$$= 1 - e^{\frac{-q(q-1)}{2N}}$$

If we want the probability to be $1/2$, we may write

$$e^{-q^2/2N} = 1/2$$

$$\Rightarrow q = O(\sqrt{N})$$

To distinguish between random function and random permutation we need atleast $\sqrt{N}$ queries

We can write $RF \simeq RP$ as long as $q << \sqrt{N}$

Random Permutation does not have collisions.

Random Functions have collisions and to get queries, we need to make atleast $\sqrt{N}$ queries

PRP-PRF Switching Lemma: $RF \simeq RP$ as long as $q << \sqrt{N}$.

Distinguishing Advantage $Adv^{dist}(RF, RP) \leq \frac{q^2}{2N}$

* Wegman Carter
* We assume a Δxu Hash function H_k ϵ-universal Δxu Hash fn

$$\Pr_k[H_k(M) \oplus H_k(M') = \delta] \leq \epsilon$$

k is chosen uniformly and randomly from $\mathcal{K}$

$k \leftarrow \mathcal{K}$ (key space)

$\# \{ k : H_k(M) \oplus H_k(M') = \delta \} \leq \epsilon \cdot |\mathcal{K}|$ — ①

($\#$ = cardinality)   If ① is true, we also say that H_k is ϵ-Δxu Hash function.

The best ϵ for security purposes is $\epsilon = \frac{1}{2^n}$ as $H_k : \{0,1\}^* \to \{0,1\}^n$.

Now,

$$\max_{\delta} \Pr_k[H_k(M) \oplus H_k(M') = \delta] \leq \epsilon$$

We consider that δ for which the probability is previous probability is maximum

→ Alice   Bob

$h, k_1, k_2, \ldots, k_l$ $\xrightarrow{M_1,\ T_1 = H_h(M_1) \oplus k_1}$ $k_1, k_2, \ldots, k_l, h$

$k_i \in \{0,1\}^n$ $\xrightarrow{M_1, T_1'}$

$H_h : \{0,1\}^* \to \{0,1\}^n$

$\xrightarrow{M_2,\ T_2 = H_h(M_2) \oplus k_2}$

Here we have considered large key as per Wegman Carter

→ Brassard suggested to used small key and then we apply the same method

Alice and Bob shares a small key K and then stretches it

Alice   Bob

h, K h, K ** Check BBS Psudo Random Bit Generator

↓ $G \leftarrow$ PRBG   ↓

$k_1, k_2, \ldots$ $k_1, k_2, \ldots$

$\xrightarrow{(M, T, c)}$

We use thus on the fly computation as small key

→ Shoup then suggested to use a Random permutation. Thus, $k_2 = \Pi_k(c)$

∴ Tag $T_c = \Pi_k(c) \oplus H_h(M_c)$

We can also use ~~Pseudo~~ Random Function.

$$T_i = f_K(i) \oplus H_h(M_i)$$

We can show that Random Function is secure using its independence property. We then show that Random Permutation is secure as it is close to Random Function using PRP-PRF switching lemma.

Tutorial Classes
[Probability]

- Deterministic Phenomenon:
  Before occurence we know the exact result

- Random Experiment:
  We know all the possible outcomes, but for a particular instant we do not know the exact result

- Laplace (Position + momentum : everything is known)
  ↳ Heiseberg (Randomness exists after a certain bound)

- Sample Space

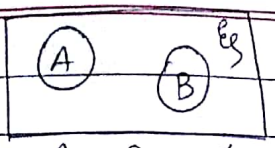
- Set has n elements, total number of subsets is 2^n.

- Any subset of Sample Space can be termed as an event in this case.

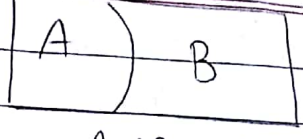
- Mutually exclusive events:
  Both events do not have any elements in common.

- Mutually exhaustive events:
  Both events combine to create the sample space.

- If we consider both mutually exclusive and mutually exhaustive events, they together form a partition.



$A \cap B = \phi$
Mutually Exclusive



$A \cup B = \phi$
Mutually Exhaustive

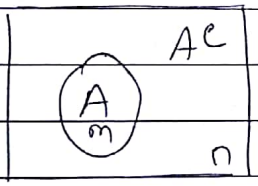
Both mutually exclusive and exhaustive → Partition created.

- Probability —
  → Sample Space — S
  A — Is an event
  $|S| = n \quad |A| = m$

$$Pr(A) = \frac{m}{n}$$

As $m <= n$, $0 \leq P(A) \leq 1$

→



Venn-Diagram Showing Complement

$$P(A^c) = \frac{n-m}{n} = 1 - \frac{m}{n}$$

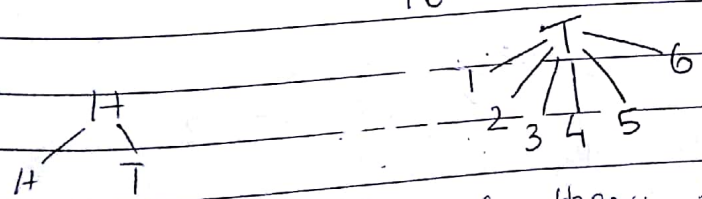
$$= 1 - P(A)$$

$$\therefore P(A^c) = 1 - P(A)$$

** We throw a coin. If head occurs, we again throw a coin. If tail occurs, we throw a die.

Sample Space: HH T1
HT T2
:
T6

we can visualize this this in terms of a tree



We cannot use classical theory to find this probability. Every occurence has a weightage Hence, concept of independent Events.

If A and B are two events, for independent events $P(AB) = P(A)P(B)$

Here, in our case $P(HH) = P(H)P(H) = 1/4$

$P(TT) = P(T)P(T) = 1/2$

- Frequency Definition:

$$P(A) = \lim_{n \to \infty} \frac{m}{n}$$

Eg: We toss a coin n number of times, we tend n to infinity and then find the number of times head occurs or tail occurs

- Random variable:

For every outcome of an event to a real number. The function which maps this is known as a Random variable.

$$X : S \longrightarrow \mathbb{R}$$

X: Random variable; S: sample space; $\mathbb{R}$: Real Numbers

Eg: ① $S = \{H, T\}$ $X(H) = 0$ $X(T) = 1$

$P(X=0) = P(H) = 1/2$

$P(X=1) = P(T) = 1/2$

② Two coins are tossed

$HH \to 2$

$HT \to 1$

$TH \to 1$

$TT \to 0$

X is the number of heads

$\therefore P(1) = P(\{HT\}, \{TH\})$

$= P(HT) + P(TH)$

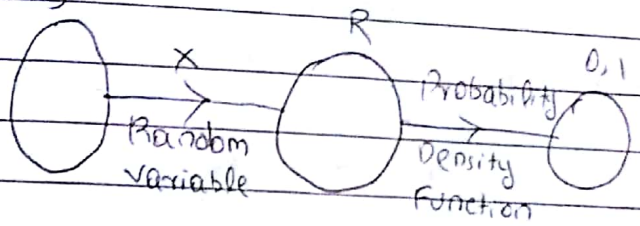
$= 1/4 + 1/4 = 1/2$

Probability distribution

$P(X=0) = 1/4$	
$P(X=1) = 1/2$	$\sum P(x) = 1$
$P(X=2) = 1/4$	

Probability Distribution tells us how the probability is distributed over the sample space

We get a discrete distribution in this case, i.e. the image set of X is discrete, technically a density function.S



** Distribution Function: Cumulative probability

Discrete Distribution:- Binomial Distribution
Poisson Distribution

Continuous Distribution: Gaussian / Normal Distribution

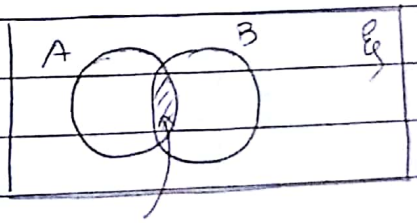
For density functions:

→ all values are positive

→ $\sum f(x)$ or $\int f(x) = 1$

↑ Discrete ↑ Continuous

• Conditional Probability —



We first assume that A has occured and then find B for given A has occured

$$\therefore P(B/A) = \frac{|A \cap B|}{|A|} = \frac{\frac{|A \cap B|}{n}}{\frac{|A|}{n}}$$

$$= \frac{P(A \cap B)}{P(A)}$$

$$\therefore P(A/B) = \frac{P(A \cap B)}{P(B)}$$

If A and B are independent,

$$P(A/B) = P(A)$$
$$P(B/A) = P(B)$$

- $P(A \cup B) = P(A) + P(B)$, if A and B are mutually exclusive
  $= P(A) + P(B) - P(A \cap B)$, general case

- Total Probability —
  We consider a sample space which is partitioned

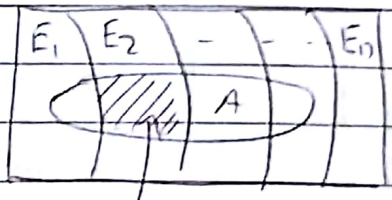
we are given probabilities of $E_1, E_2, \ldots E_n$

We have to find probability of A wrt $E_1, E_2 \ldots E_n$

$P(A) = P\{(A \cap E_1) + (A \cap E_2) + \ldots + (A \cap E_n)\}$

↑ Total Probability

- Bayes Theorem —
  We do the reverse of Total Probability. We know A has occured and we have to find from which partition it is selected



we have to find this

$$\therefore P(E_i/A) = \frac{P(A/E_i)}{P(A)}$$

$$= \frac{P(A/E_i)}{P(A \cap E_1) + P(A \cap E_2) + \ldots + P(A \cap E_n)}$$