Review:

- Shift Cipher → Broken
- Mono-alphabetic Cipher - Broken
- Vingenere Cipher - Broken

(Q) what to do next? [Board]

Today

→ Shannons answer to the Question
→ Define & Design
  ↳ unbreakable Ciphers
    [One-time Pad].
  ↳ One-time pad is impractical.

→ Shannon's Theorem
→ Every unbreakable cipher is impractical.

People didn't know what was ponible.
↳ Frequency analysis & statistics were imp.

Will this ever end? We would make better locks & wait for the thief to crack it.

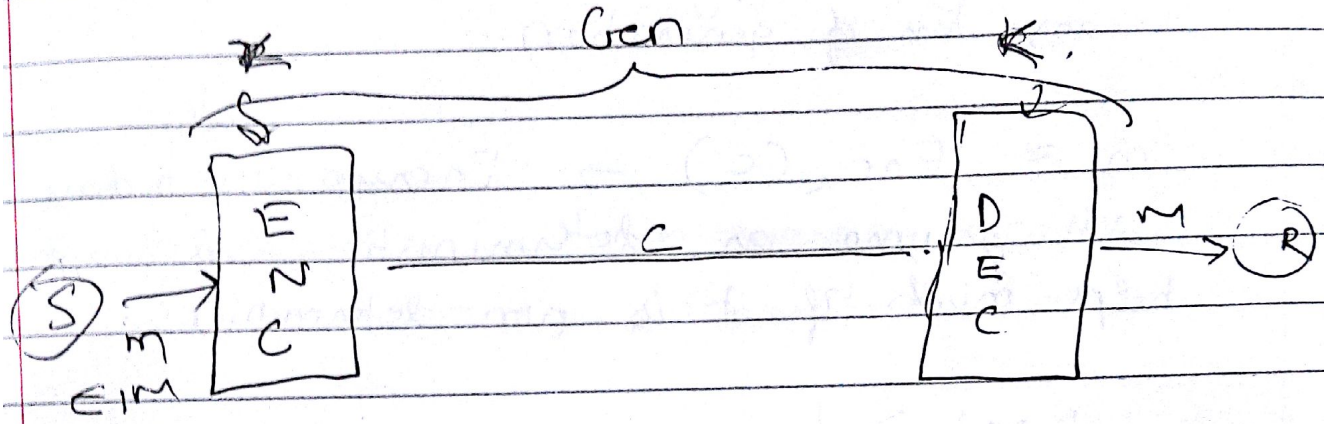
What do we want?
↳ An unbreakable cipher
  ↳ Just define it & design unbreakable ciphers!

Shannon proved that any ciphers which meets my definition of unbreakable is impractical.
unBreakable ∩ Practical = ∅.

Shannon's Answer

↳ Lets not keep on improving systems. Let's see what is the best possible system?

Perfect Secrecy:



Encryption scheme → Def^n → There is a key, there is a Encryption algorithm, there is a decryption algorithm.

We get a 3 tuple $< Enc, Dec, M >$
↳ message

We can have a key-generation algorithm too. It takes the length of key as input & returns a key from the key-space randomly.

$Gen(1^n)$ ← Security parameter

n is the input size and the key is generated of length 2^n.

$k \xleftarrow{R} \mathcal{K}$
↳ key space

$k \in \mathcal{K}$

$Enc_K(m) \rightarrow c$, i.e. $c \in \mathbb{C}$ (Ciphertext space)

$m \in \mathbb{M}$ (Message space)

$m = Dec_K(c)$, $m \in \mathbb{M}$

Decryption algorithm is deterministic without any loss of generation.

$m = Enc_K(c) \rightarrow$ Encryption algo, today we will assume as deterministic, although it helps much if it is non-deterministic.

$|\mathbb{M}| > 1$ → Message space should contain more than 1 message otherwise we don't need to communicate.

The probability of some key space is defined by the key-generation algorithm.
key generation algorithm is random!

The frequency of character in key is fixed by the key-generation algorithm.

We have $\mathbb{M}$ in the 4tuple as Message space is independent, unlike the key space or the cipher space as they are totally determined by the Algorithms.

We should always get a message back with probability 1. Decryption should necessarily deterministic.

Hence, $\exists$ f: $\mathcal{M} \to \mathcal{K}$, f should be a function which is atleast one-one.

Shannon's definition of a secret cipher:

An encryption scheme is said to be perfectly secret if :-

We know things about the plaintext beforehand, if looking at the Ciphertext, the knowledge about the plaintext doesn't increase. If reading or not reading the Ciphertext didn't change our life, then

$Pr[M = m \mid C = c] = Pr[M = m]$

for all $m \in \mathcal{M}$ and $\forall c \in \mathcal{C}$, where c actually occurs, i.e. $Pr[C = c] > 0$.

i.e. for all distributions over Message space, if we know the information about the message given the cipher text = information about the message before seeing m should be same.

We might not as well look at Ciphertext

Mono-alphabetic Cipher is not secure after 1 character in the perfectly ~~for~~ secret !!

So, as long as the key never repeats, it is perfectly secure.

There are infinite ways to define unbreakable ciphers. Shannon was the first one. This is not the only way & not suitable for other ways.

This is a designer way. It is an adversarial view in modern terms. The adversary should not even know anything from the message.

Alternative Definition of shannon's theorem.

An enc. scheme is perfectly secret if all distribution over M $\forall m \in M, \forall c \in K$.

Def 2 $Pr[C=c \mid M=m] = Pr(C=c)$.

This says that we should not know anything about the ciphertext if we know the plaintext.

Multiplying both sides with $Pr[M=m] / Pr[C=c]$

Hence, $Pr[C=c] \neq 0$, hence our assumption

$$Pr[C=c \mid M=m] \cdot Pr[M=m] = Pr[M=m]$$
$$Pr[C=c]$$

~~Multiplying this (P=H)~~

$$P(A \mid B) = \frac{P(A \cap B)}{P(B)} \quad \cancel{P(A) \cdot P(B)}$$

$$P(A \mid B) = \frac{P(B \mid A) \cdot P(A)}{P(B)}$$ [Bayes theorem

$\therefore$ LHS = $Pr[M=m \mid C=c] = Pr[M=m]$

Hence, they are equivalent.

$\forall m_0, m_1 \in M$ and for $\forall c \in C$.

$$P(C=c \mid M=m_0) = Pr[C=c \mid M=m_1].$$

→ Indistinguishability is its basis.

Both message space & cipher space both are infinite. In comp sci, there is no infinity, M space is finite as we define only the parts with non-zero probability space.

The message space is finite.
↳ But it is not ~~finite~~ limiting as

Encryption of 1 bit only exists. $M = \{0, 1\}$
Encrypting the 2nd bit is with only key is maybe encrypt the first bit, then xor it with 2nd bit then encrypt it again. Something on these lines.

We can however encrypt the message independently. Hence M can be finite, it should be big.

Take any 2 messages m_0 & m_1. The probability of m_0 being encrypted to C and m_1 being encrypted to C, if both are exactly same irrespective of M, then the scheme is perfectly secret.
(Remember it should hold for ANY m).

Proving the Alkunak Def implies 3rd definition is trivial

LHS = $Pr[C=c]$

RHS = $Pr[C=c]$

$\therefore$ LHS = RHS.

But the other way is non-trivial !!

Stenagogvaphy → We hid the fact that data is being sent. Steganography a good non-Kerchoff way of getting security.

No steganographic standard defined yet

$\forall m_0, m_1 \quad Pr[C=c \mid M=m_0] = Pr[C=c \mid M=m_1]$.

Given.

⇒ To prove :- $Pr[C=c \mid M=m_0] = Pr[C=c]$

$Pr[C=c] = \sum_{m \in M} Pr[C=c \mid M=m] \cdot Pr[M=m]$

$\sum_{m \in M} 1$

We know that $Pr[C=c \mid M=m]$ would be same for all m,

$Pr[C=c] = Pr[C=c \mid M=m] \cdot \sum_{m \in M} Pr[M=m]$, $Pr[M=m] = 1$

Hence, $Pr[C=c] = Pr[C=c \mid M=m]$

$\forall m \in M$.

Hence, all these 3 definitions of perfect secrecy ✓.

Are there other definitions? If this would be practical, then the other definitions would not even exist.

5] Semantic Security definition, very very faithful definition & it is workable. → Turing Award in 2012.

One-Time Pad (Vernam Cipher).

Input → $\{0,1\}^{\ell}$ → ℓ bit strings.
Output → $\{0,1\}^{\ell}$ → ℓ bit strings
Key → $\{0,1\}^{\ell}$ → ℓ bit string

when Generator algorithm uniformly chooses a key at random on the ℓ-bit strings.

$$Pr[K=k] = \frac{1}{2^{\ell}} \text{ uniformly.}$$

Enc = $c = m \oplus k$
Dec = $m = c \oplus k$

Let us show that it follows the 1st def^n →

$$Pr[C=c \mid M=m_0] = Pr[M \oplus K = c \mid M = m_0]$$

$$= Pr[m_0 \oplus K = c] \text{ by definition.}$$

$$= Pr[m_0 \oplus c = K] = \frac{1}{2^{\ell}}.$$

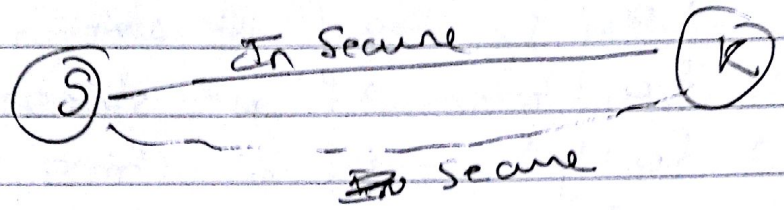
Hence, no matter what m_0 we choose the probability of it being mapped is $\frac{1}{2^{\ell}}$.

Since it satisfies this definition, we say it satisfies all definitions.

M → no here we replace the random variable.

We abuse notations to M being space of Messages M.

Why is it not suitable?



The length of key = length of message

We could actually just send the message instead. There is no point is sending the message.

The security bandwidth of the system = The bandwith of seaure channel.

One-time pad applications

↳ We can use one-time pad when we want to send info data.

↳ If secure channel is not available always. Deferred sending, i.e. secure channel is available on sundays, but data only on Mondays. So we can send key on sunday & then Monday send the secure data.

Q1) Can I speed up a fast-insecure channel to a fast-secure channel?
↳ Symmetric key cryptography.

Q2) Can I send a message securely if we don't have a secure channel?
↳ Public key Cryptography.

One option → we could have used Quantum Entanglement.

The theorem proves that Q1 = Q2. This was deep connections.

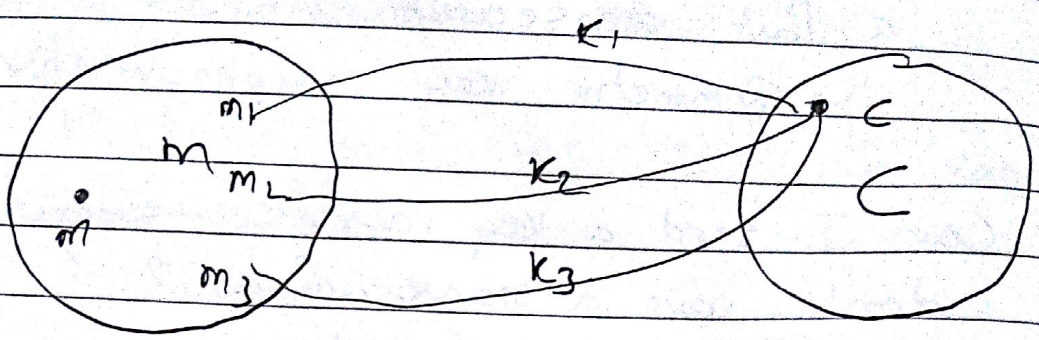
Shannon proved that one-time pad couldn't be speeded up. i.e. it will always be at the speed of secure channel in which case we would just send the message.

Thm
For any perfectly secret enc-scheme, if $|M| \leq |K|$.
i.e. the size of key space should be always greater than size of message space.

Proof :- Suppose $|M| > |K|$
what does it mean?
We assumed deterministic encryption
$Enc_K(m) = c$

So, if I take a ciphertext c and look for decryption, there will be a K which should never map to all M's, hence.

It is impossible.



There exists an m such that m can never be encrypted to c. [Pidgeon hole principle].
∴ There exists m $Pr[C=c \mid M=m] = 0$.
which violates our definition of perfect security.

as $Pr[C=c \mid M=m] = Pr[M=m]$.

Message space should necessarily be finite as otherwise keyspace should be infinite & we can never send the key !! due to this theorem.

Thm Shannon's theorem

Assume $|M| = |K| \leq |C|$
↳ This is the most efficient version of Cipherspace. We just assume now, we just assume the best case; Shannon's theorem gets very interesting.

An encryption scheme is perfectly secret if and only if
(a) Generator Chooses keys uniformly @ random $Pr[K=k] = \frac{1}{|K|}$ should be necessarily true.

(b) For every m and c, there is a unique key K s.t - $Enc_K(m) = c$.

This forms a bijective function.

So if (a) & (b) hold true then the schemes are perfectly.

What does this gives us ?

↳ If we perfectly secret schemes, the length of key should atleast be the size of message.

Hence, all perfectly secure schemes are impractical as we can just send the messages instead !!

We have to break the impossibilities !

The 2 ways : Public & Symmetric key.

The story so far ...

Review

Information Security → It is about circumventing impossibilities. We learn tools and techniques to help us circumvent impossibilities.

Cryptography is one such field of learning tools & techniques used to circumvent impossibilities.

↳ Instead of abracadabra, we say

(a) Adversary is bounded by PPTM

(b) Negligible probability of error is allowed.

(c) One-way functions exist.

Scientific Impossibilities → Perfect Secrecy, etc.
(Logical Impossibilities)

Rhetoric Impossibilities → Two names for the same concept, name1 & name2. We want name1 to exist & name2 to not exist. This is impossible!

Network Security

↳ One example. Thwarting attacks in the OSC layers of the network. What kind of impossibilities exist in network which are not scientific?

Names → Sniffing, Scanning, Spoofing, Flooding
↳ fundamental tools of a network hacker.

We argue these are all names given to a same concept. Sniffing → In a LAN our payload is visible to other people too.

Scanning → Everybody knows whether we are vulnerable or not from port scanning.

Flooding → we create congestion creating a DOS attack. We want to argue that these are good things.

Scanning - If I want to talk to you on a network? First we want to ask whether they know what all they know. Solution is that we all know all the languages that we know & understand. Hence, Scanning (

Handshake is allowed, but port scanning is not allowed.

This is really bad.

Identifying if the from address is correct or not. Verification protocols are difficult to identify at basically every time step. We cannot run back to trace origin, then the from address is important. Hence, the network cannot check it is true or not. Hence, efficient routing is allowing spoofing, i.e. not care if the from address was correct or not is impossible.

Hence, efficient routing ≡ spoofing.

Flooding - If we make having IP packets costly, we will stop flooding. The freedom of generating lot of IP packets, the ease & low cost of generating IP packets is ideal for flooding i.e. flooding ≡ low cost of transmitting packets.

Hence, impossibility.

Shifting → ~~Hence~~ We cannot have a unique IP address to every place. It is very inefficient! The protocols should not change by location. Instead all mails arrive at IIIT gate and we have to check our own packets and take them. If we have efficient 3D printers, we can replicate everyone's parcel & generate exact copies of everyone's packet and go away.

Hence, efficiency of IP ≡ Shifting.

Secure Communication → Secret "Confidentiality".

- Shannon
  ↳ But was impractical.

Relax the perfect secrecy conditions to get exp time security instead of infinite time security.

One-way functions exist ≡ Pseudo random generators exist.

If pseudo-random generated ~~key~~ key is indistinguishable from original channel then

This won't work for multiple encryptions.

Hence, we need probabilistic encryption.

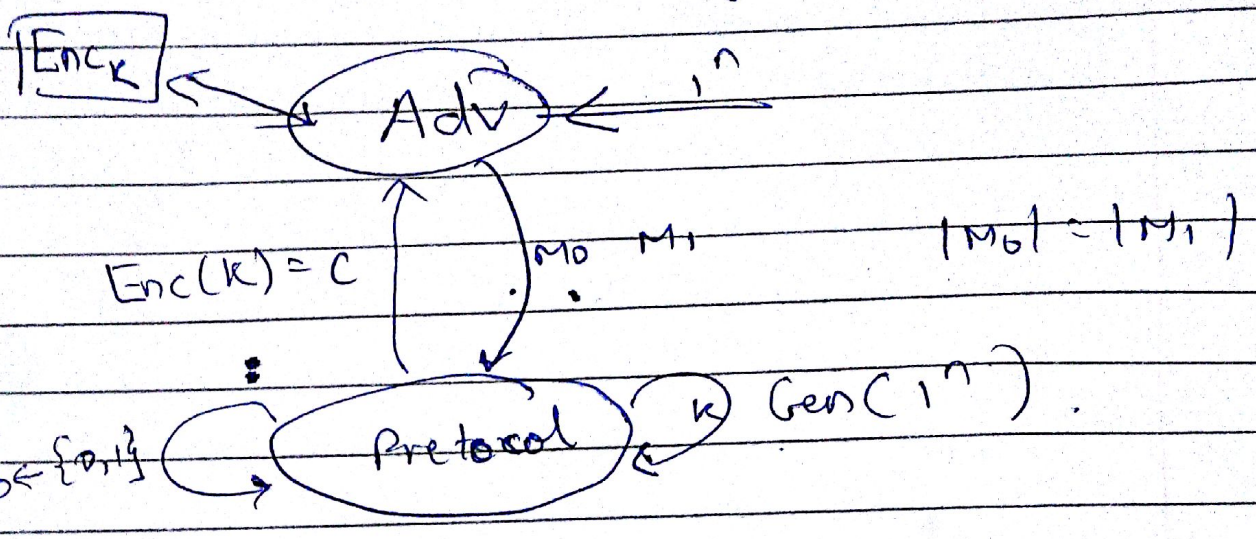
Hence, $\langle r, Enc, \langle r \rangle \oplus m \rangle$

$\langle r_1, G(k) \oplus r_1 \oplus M_1 \rangle$
$\langle r_2, G(k) \oplus r_2 \oplus M_2 \rangle$.

We still cannot meet the multiple encryption schemes yet. Today, we will see CPA ⇒, a very powerful attack.

CPA

→ He has free access to M_0 & M_1, both plaintexts. We encrypt one of them with only ~~one of them~~ given key. He can himself encrypt both the messages, etc. Still he should not able to distinguish our ~~attacks~~ messages, i.e. should not be able to decipher the message.



Our codes have to be CPA-secure. USA broke the Enigma code ~~as~~ by a CPA as most encryption

$$Priv^{CPA}_{A,\Pi} = 1 \text{ if } b = b'$$

$$= 0 \quad \text{otherwise.}$$

Hence, deterministic security would fail. It has to be CPA secure, i.e. ~~given the key~~ given ciphertext & plaintexts & encryption algorithm $G(K)$, he can just encrypt the message and check the output to see if it matches.

An encryption scheme → CPA-Secure if $\forall$ PPTM A.

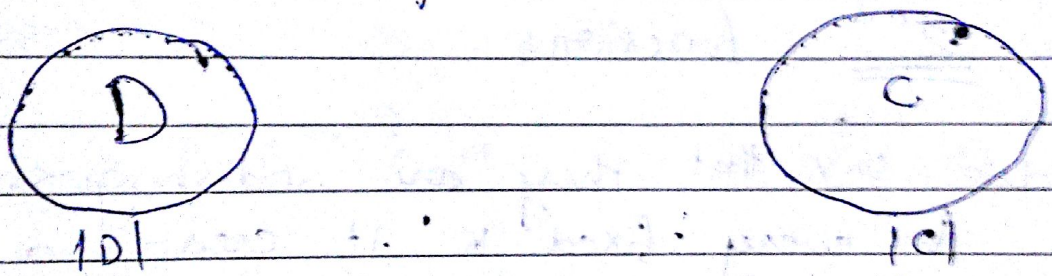
Pseudo-random generators exist, then we can prove pseudo-random functions.

One way functions exist → lead to
Pseudo-random generators exist → lead to
Pseudo-random functions exist

Instead of choosing a string uniformly at random, we choose a function uniformly at random instead, using a short key.

Then we prove given pseudo-random functions, CPA secure schemes exist.

$f: D \to C$:



→ $|C|^{|D|}$ functions.

index all these functions in $|D| \log |C|$ bits.

and then

we have to index functions.

Let $f: \{0,1\}^n \to \{0,1\}^n$

bitstrings of length n. → bitstrings of length n.

No of functions → $(2^n)^{2^n} = 2^{n \cdot 2^n}$ functions.

Pseudo-random function → If an adversary cannot distinguish b/w our picked function and a one of these, it is indistinguishable.

$F \rightarrow \{0,1\}^n \times \{0,1\}^n \rightarrow \{0,1\}^n$

$F(k,x) = y$

Instead we write this as follows :-

$F_k(x) = y$.

∴ If we fix the first operand, then the function is equivalent. we can vary k to give us different function

In our method, we only vary k, then we get only 2^n functions among the 2^{n2^n} functions.

we say that they are indistinguishable, and for every fixed k, it creates a function

$F_k : \{0,1\}^n \rightarrow \{0,1\}^n$.

we vary only k & for each k, we get a fixed function.

We cannot distinguish the 2 worlds with polynomial amount of samples, we ought to have exponential amount of samples.

Standard method distinguishing functions using oracle machines.

The adversary actually does not input functions.
Oracle turing machines → It is like a program can query to a server named as Oracles (functions itself).

If A is a TM & B is a TM, A^B is called as an Oracle Turing machine - B gives answer in O(1)

A function F_K is said to be pseudorandom $\{0,1\}^n \to \{0,1\}^n$ (it can be m to n) but we'll assume $n \to n$ for this course.

A function F_K is said to be pseudorandom function (PRF) by a PPTM D :-

$$Pr\left[D^{F_K(\cdot)}(1^n) = 1\right] - Pr\left[D^{f(\cdot)}(1^n) = 1\right] = neg$$

D will output 1 when given access to an oracle machine v/s D will output 1 when given access to a random machine can distinguish b/w the 2 worlds.

We focus on the finish line.
The hare & tortoise story, we have to learn how they ran the race, our textbook just says tortoise crossed it before hare; and we exhalt success and remove all failure stories.

W.r.t business & profitability → success story is only important. In education, failures are much more important than successes.

Say in LKG, we did not ~~allow~~ know ABCD, we ~~or wait~~ tell the principle that I want to be detained in Class 1 to learn it again. It is ~~impossible~~ condemned. Society says that you've failed ~~in~~ in Class 1! It is said as if the last thing that we want to do is to fail.

What we are endowed currently is not sufficient for my happiness. We have to fail, react positively and we have to fail.

When we fail, ~~and~~ then only we need reinforcement telling us we failed. Our muscles ~~would~~ have to break down. We have to fool our muscles into believing that we have to do it for survival. We would get bigger muscles. We have to fail again. Similarly, with our brain. We have to try lifting things we are not capable of.

So, changes should happen. We would take up challenges where we fail. We have to take up new challenges everyday.

Understand the symptoms of IQ challenge. Deep Frustration: Go take rest, come back and take steps to reduce the IQ change.

If we do probabilistic encryption,

$C = \langle r, F_K(r) \oplus m \rangle$ is CPA-Secure Encryption

PRF.

What are drawbacks?

↳ Length doubling

↳ Find r.

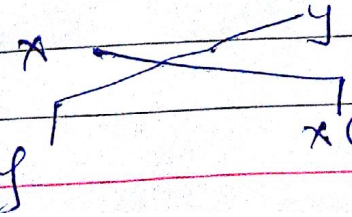
Block-Ciphers

Many methods to given 1^r, encrypt many many blocks.

Heuristic Block Ciphers → DES, AES, etc.

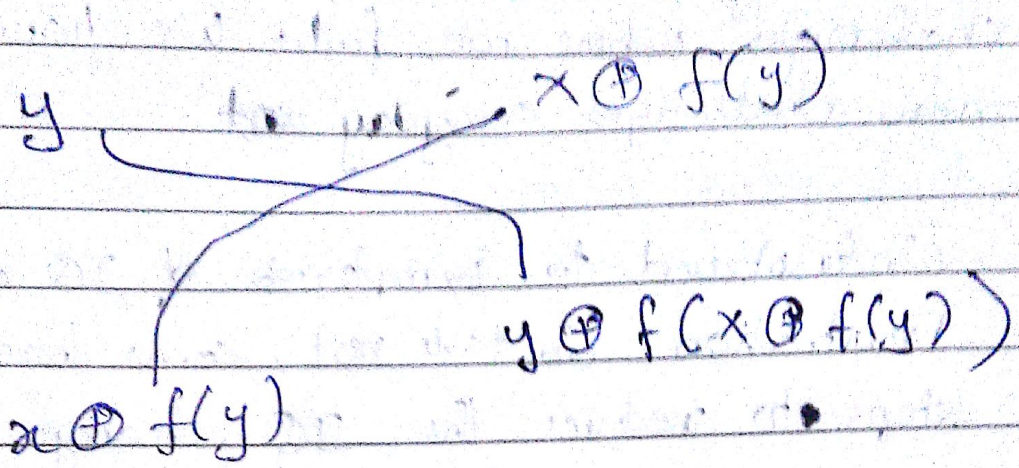
Provable Block Ciphers →

Fiestal Network → Use r again & again to make something non-pseudo-random to something pseudo-random



This is invertible even if f is not.

Decryption



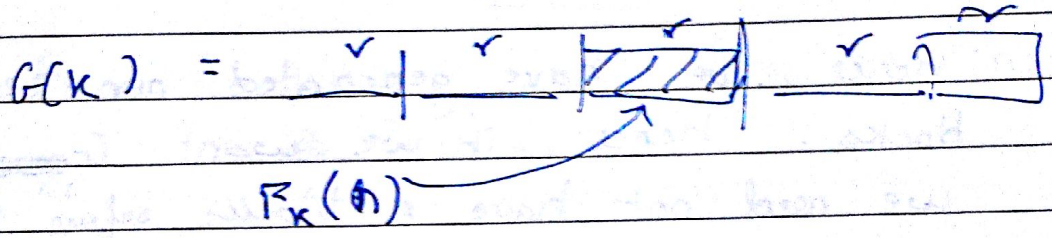
Repeat this n times, so that f was not pseudo-random, we hope it will become pseudo-random after 16 times say (DES) it will become pseudo-random.

Pseudo-random functions (PRF).

→ Using PRF to design a PFA-secure Enc.
  ↳ Proving the security.
→ Theoritical construction of PRF's from PRG's.
→ Practical block ciphers (PRF's from PRG)
  AES & DES.

From PRGs to PRF's.

Take seed k, apply g(k). we get a big random looking scheme.
We make blocks of n bits from them.



Suppose we want to access the parts of PRG randomly, then we have a PRF

PRG's with random access = PRF.

Let us say G is just length w

G is length-doubling

$$G(\{0,1\}^n) \rightarrow \{0,1\}^{2n}.$$

$$G(k) = \boxed{G_0(k) \,|\, G_1(k)}$$

$$F_k(b) = G_b(k)$$

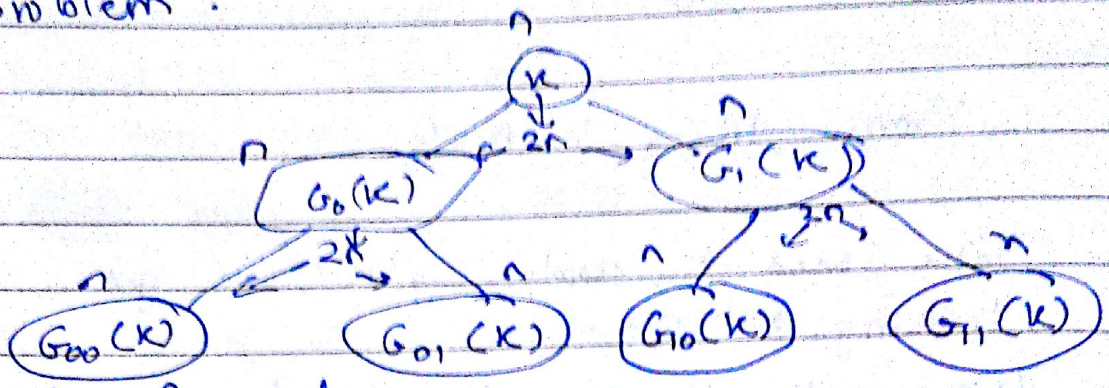
r = 1 bit always.

$F_k(b)$ $\quad F_k(0) \rightsquigarrow G_0(k)$ ∴ Random.

$\quad F_k(1) \rightsquigarrow G_1(k)$ ∴ Random.

So, we have a PRF from for 1 bit.
But, for 2 bits → 4 blocks
3 bits → 8 blocks
n bits → 2^n blocks.

Exp growth. Hence, PRF creation is an exp problem.



Total should be 2^n and

Hence, we have generated non-sequencial blocks. Hence, if we want $G_{000}(k)$ we need not have n blocks before it, we need only log(n) blocks.

Hence, for n bit strings having 2^n blocks, we need only $\phi(n)$ time to get a block, hence random access problem is solved and we have a prf.

$F_k(r_{n-1}\, r_{n-2}\, r_{n-3} \ldots\ldots r_1\, r_0)$

↙

$G_{(r_{n-1} r_{n-2} \ldots r_1 r_0)}(k)$

↙

$G_{(r_{n-1} r_{n-2} \ldots r_1)}(G_{r_0}(k))$

↘

$G_{(r_{n-1} r_{n-2} \ldots r_2)}(G_{r_1}(G_{r_0}(k)))$

log linearly.

Hence, $G_{k+1} = G_k$

$G_{r+1}(k) = G_r$

$G_{\Pi(r+1)}(k) = G_{\Pi r}(G_{(r+1)}(k))$

G_0 is given.

G is length doubling

$G(\{0,1\}^n) \rightarrow \{0,1\}^{2n}$

Practical block ciphers (one-to-one PRF's)
↳ DES & AES

Whenever things are inherently linear, linearity leads to an exp problem, use trees

How to use PRF's such that schemes are CPA-secure encryption scheme.

$key(r) = F_k(r) \| F_k(F_k(r)) \| F_k(F_k(F_k(r))) \| \ldots$

$key(r) = F_k(r) \| F_k(r+1) \| F_k(r+2)$

Both are pseudo random.

Proving CPA-Security

PRG exists { Gen (1^n) which has input $k \in K$ given $\{0,1\}^n$

$K \rightarrow \{0,1\}^n$.

Enc ⇒ $r \xleftarrow{R} \{0,1\}^n$

$c = \langle r, F_k(r) \oplus m \rangle$

Dec ⇒ $\langle r, c \rangle$

$m = c \oplus F_k(r)$.

We have to prove that $\forall$ PPTM D.

(1) — $Pr\left[D^{F_k()}(1^n) = 1 \right] - Pr\left[D^{f()}(1^n) = 1 \right] \leq$ negligible

$\forall$ PPTM A

(2) — $Pr\left[PrivK^{CPA}_{A,\Pi} = 1 \right] \leq \frac{1}{2} + negl(n)$.

If there is an A which does satisfy (2) then there exists a D which breaks (1).

Let us assume a scheme which uses actually random string π_R instead our pseudorandom π.

For random function π_R

$Pr\left[PrivK^{CPA}_{A,\pi_R} = 1 \right] = ?$

As if we replace random → pseudo random it should work [Def^n of pseudo randomness breaks if we are able to differentiate].

Say we are given encryption of m_0 & m_1.

$$C = \langle r, f(r) \oplus m \rangle$$

If it repeats an r, then we can guess the plaintext as we have the ~~probabilities of~~ all encrypted ones till now.

∴ we want our prob be

$$Pr\left[Priv^{cpa}_{A, \Pi_K} = 1 \right] \leq \frac{1}{2} + \frac{q(n)}{2^n}$$

Since our r is truly random, and $q(n)$ is bounded by PPTM, we win.

Say r is not truly random.
Let A 'break' the encryption scheme as $q(n)/2^n$ is not negligible.

$$Pr\left[Priv^{cpa}_{A, \Pi} = 1 \right] \geq \frac{1}{2} + \varepsilon(n).$$

where $\varepsilon(n)$ is non-negligible.

D : On input/oracle $\phi : \{0,1\}^n \to \{0,1\}^n$
D will use A to try to distinguish.

- For any query made, A 'breakes' the server to get M_0 & M_1
D doesn't have access to encryption server. A has. How to simulate that?

For any query A makes to encryption server. D makes an r randomly.
Send r to the oracle and get s as the answer

$F_k(r)$

ie. send $\langle r, s \oplus m \rangle$ to A.

A thinks it is the encryption server and breaks it

- Run A and get $M_0 M_1$.
- Create the C as some.

$$C = \langle r, s, s(r) \oplus m_b \rangle$$

$$b \xleftarrow{R} \{0,1\}^n.$$

Send C to A, A returns to b'.

If the function is random

$$Pr\left[D^{f(\cdot)}(1^n) = 1 \right] = \frac{1}{2} + neg$$

$$Pr\left[D^{F_k(\cdot)}(1^n) = 1 \right] = \frac{1}{2} + \frac{\varepsilon(n)}{2^n}$$

Subtract them RHS $= \frac{\varepsilon(n)}{2^n} - neg$

$$\approx \frac{\varepsilon(n)}{2^n}.$$

we are simulating the server using a truly random function.

Protocol Π which runs.

So, whenever A gives b' = b. D outputs 1 when $Pr\left[PrivK^{cpa}_{A,\Pi} = 1 \right]$ as then it outputs b = b'.

$$\Pr\left[D^{F_k()}(1^n)=1\right] \equiv \Pr\left[Priv^{cpa}_{A,\Pi}=1\right]$$

$$\Pr\left[D^{f()}(1^n)=1\right] \equiv \Pr\left[Priv^{cpa}_{A,\Pi_k}=1\right] \leq \frac{1}{2} + \frac{q(n)}{2^n}.$$

Negligibility is closed under addition.
under a PPTM.

So, we now know if an adversary can break the CPA secure scheme, he can break pseudo-randomness. This is the thin-thread.

Block Ciphers.

→ Strong pseudo-random permutations.
Block ciphers are one-one PRF's, hence they have to be pseudorandom.

Strong pseudo-random ⇒ The adversary is not only given access to encryption, but also decryption ~~secure~~ server. Still the scheme has to be secure.

$$\Pr\left[D^{F_k(),F_k^{-1}()}(1^n)=1\right] - \Pr\left[D^{f()f^{-1}()}(1^n)=1\right] \leq negl(n).$$

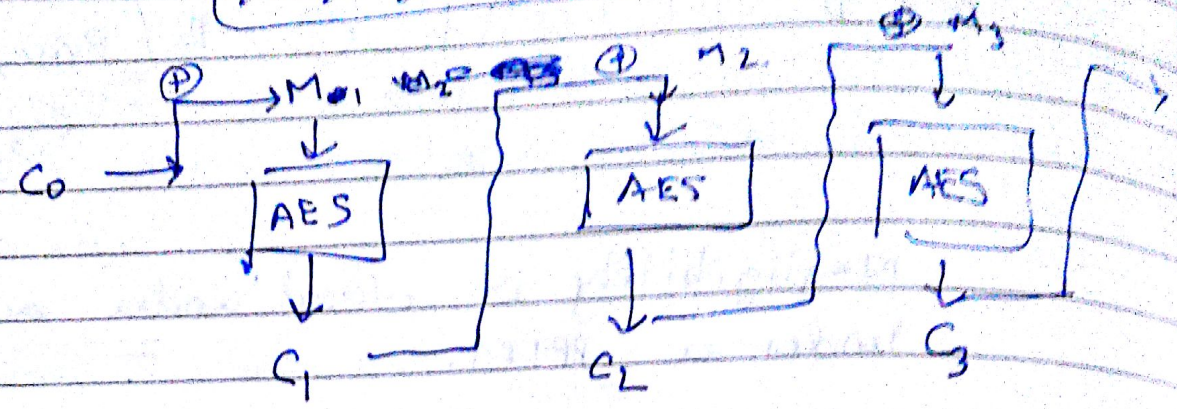
DES (Data Encryption Standard).
AES (Advanced Encryption Standard).

→ Cipher-Block-Chaining (CBC).
→ $c_i = \langle r_i, AES_k(r_i) \oplus m_i \rangle$.

$C_i = AES_K(C_{i-1} \oplus m_i)$.

C_0

$r_1 = C_{i-1} \oplus m_i$.



Initialy $r = C_0$.

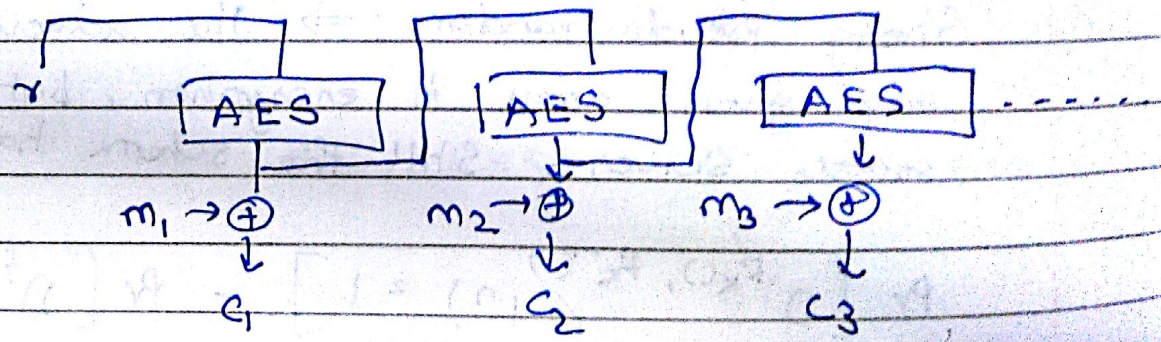
We basically xor the ciphertexts linearly

M has q blocks → $\langle m_1, m_2, \ldots, m_q \rangle$

C has $q+1$ blocks → $\langle C_0, C_1, C_2, \ldots, C_q \rangle$.

Output Feedback Mode (OFB).

This can convert block ciphers to stream ciphers & is much faster.



One major difference if C_i is dropped, then decrypting C_{i+1}, C_{i+2}, etc is not possible.

Only thing needed in OFB is just C_0 (r)

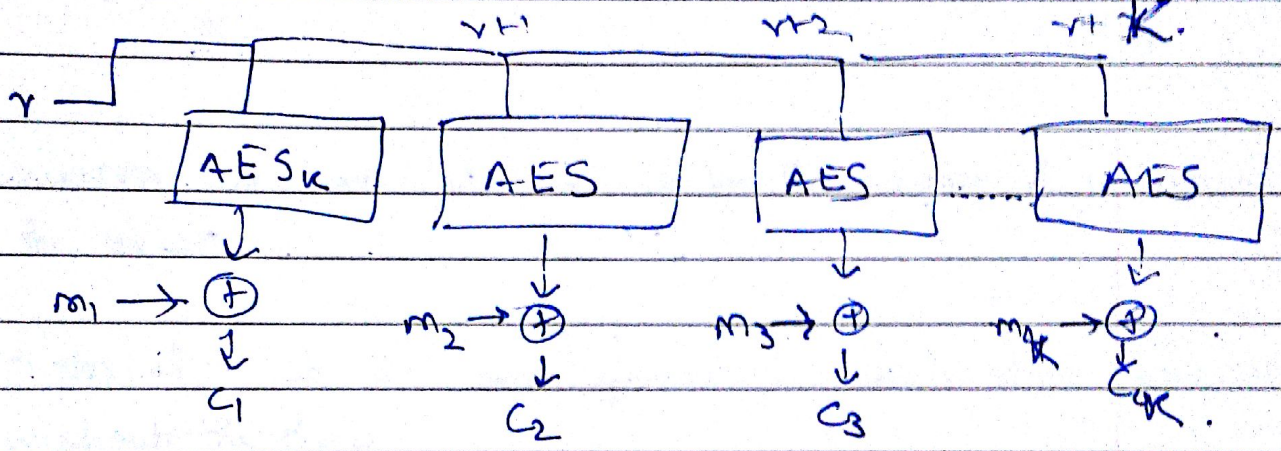
Hence, the OFB mode is much better as it removes much sequencial nature.

IP layer it's not FIFO, hence, sequencial thing is not provided. IP security, i.e. in network layers, the CBC is bad. OFB is much better.

Block Cipher in OFB, then we can generate a PRG by the r as input, and we can use that output for a stream cipher.

Counter Mode (Randomized counter mode)
✓ V-popular as no feedback.

We have our r



We proved $\langle r_i, F_K(r_i) \oplus m_i \rangle$ all the schemes here are CPA-secure.

One r can generate all r's by adding 1 & encrypt steff directly.

r is called as start vector which has to be random & key is random; hence not only key K is used (K & r) are used for making the key remaining same.

✓ Randomized Counter mode is much popular.

Use discrete log to get a PRG.

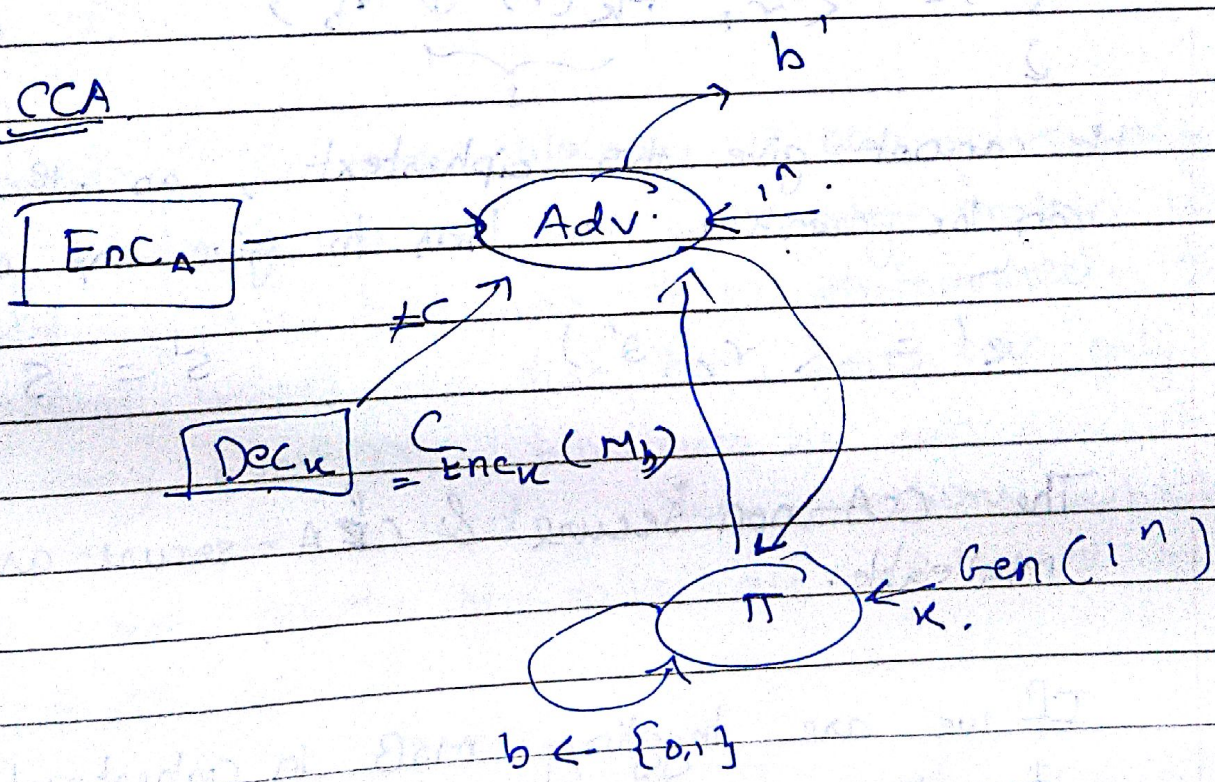
Review

→ Confidentiality of deformations.
→ Perfect Secrecy
→ Ciphertext-only attack.
→ Known-Plaintext Attack.
→ Chosen-plaintext attack. (CPA-Secure).

Today. → Chosen-Ciphertext attack (CCA-Secure)
↳ State-of-the-art.
↳ Message Authentication.

Till now, we studied how the incomplete nature of the adversary (PPTM) and then
Can the adversary not distinguish access to M_0 & M_1 even with chosen plaintexts. without the key inserted but not known. He can encrypt/decrypt any no of messages with the key inserted.

CCA



Why is access to decryption server necessary?

We know that encryption is necessary as we can freely access the server.

Now, the adversary can give many bits in the ciphertext and if they turn out to be cool indeed, then we can manipulate the text and check for the validity.

Hence, we introduce that the adversary has free access to decryption server too & hence can give many message & check their plaintext.

$C = G(K) \oplus m$ ✗ Not Secure under this obviously as we can change bits.

$\langle r, F_K(r) \oplus m \rangle$

↳ $M_0 = 0^n$

↳ $M_1 = 1^n$

$C = \langle r, F_K(r) \oplus m_b \rangle$

↓

He cannot give this ciphertext, as otherwise anything can be cracked. He has to give any other ciphertext.

$c' = \langle r, s' \rangle$ $s' = s$ with MSB toggled.

The CCA-non secure & CPA-secure are called as Malleable.

If we are toggling MSB in ciphertext, we are toggling MSB in plaintext.

$m' = F_k(r) \oplus s'$

$m = F_k(r) \oplus s$

$m' \oplus m = s' \oplus s$

Non-malleable Encryption → Any change adversary does in the ciphertext should not be able to know what change here causes what change there. He must be absolutely clueless.

We will see a different thing from Confidentiality altogether (move away from Crypto)

Data Integrity → The next impossibilities in our way.

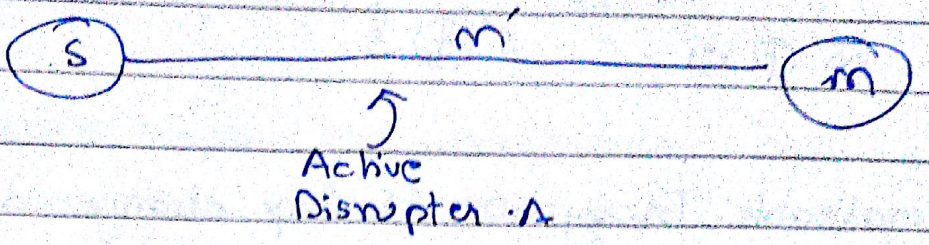
This course is a collection of examples of circumventing impossibilities using our magic wand of the 2.

Example: In comp. biology → ~~Our~~ adversary is the guy who gives us the ~~integrity~~ protein whose folding structure has to be determined. This approach would say that the poses can only give me easy questions. Whatever he can generate easily can be easily solved, the questions which are NP-hard for me to solve are NP-hard to generate for him.

This techniques are actually good enough !!

PRP's, PRF's, PRG's, one-way functions, the same stuff solves many other problems.

Data Integrity



Active disrupter may be as benign as noise or an active attacker b/w you & bank.

So, our target is make a very small subset in a very big set which makes sense & hence majority of the encrypted messages wont make sense.

We cannot have message checking stuff.

Now, we have some codewords by which stuff makes sense. We bound the disruptes, say A such that the codewords & scheme is such that given a message m & ciphertext c, he cannot generate m' & c' such that c' = enc(m'). He should not probably be even able to check if m' has a tag t (something appended).

MAC$_k$ is there in this world.

Message Authentication Codes.

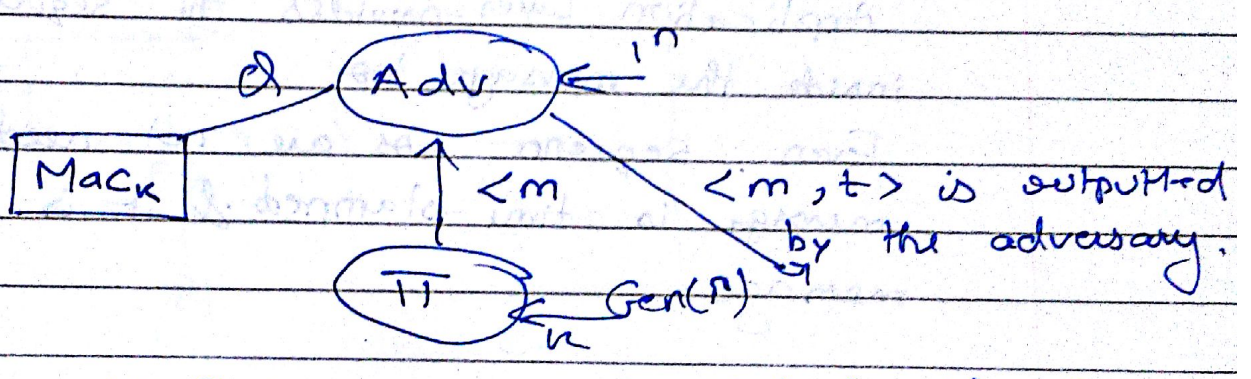
(Gen, Mac, Vrfy)

$t \leftarrow Mac_k(m)$.

Yes/No $\leftarrow Vrfy_k(m, t)$

It should have YES if the $mac_k(m)$ & m if given for verification algorithm gives YES mac is probabilistic, but Verify is deterministic as it has to return YES with prob 1 if the inputs are correct.

A message authentication scheme is secure if: for any message except m should be given with their tags if asked (~~chosen~~ CCA) and the adversary should not be able to forge the tag t of m.



(a) If $M \notin Q$ and YES = Vrfy(m, t) ⇒ Break.
(Verify MAC = 1 condition)

i.e. if M doesn't belong to a scheme ~~as m~~ given but Verify(m, $mac_k(m)$) returns yes should not happen

$\therefore$ if $Pr(MAC = 1) < \text{negligible}$.
A, Π

The main role of the protocol is adversary has to generate a valid tag for any selected unseen message m.
He has access to servers for which he can passes in a message to the sever which tells him the tag.

Looking at $\langle m, t \rangle$?
$\langle m', t' \rangle$ → Verify says YES.

Replay Attack may/may not be a valid security breach depending on the specific application layer.
↳ Can be easily thwarted by including a sequence number in the message, so replaying the message, to pose it as a new instance, sequence number should vary otherwise it would be ignored, but sequence no

We are designers, our assumption is that the Application Layer provides the sequence numbering inside the message too.
Even sequence nos are not needed. Every message is time-stamped & it is included in the message.

Solutions to solve Data-Integrity problem

→ CBC-MAC → Mid 80's - 90's..
→ H-MAC. → used now

$$t \leftarrow F_k(m)$$

if $F_k(m)$ was random, then for any new message m regardless of previous seen messages, the output would be randomly generated in $\frac{1}{2^n}$ for an n-bit key.

$t \leftarrow F_k(\oplus m_i)$ is the tag fails!

As he can get a XOR outside will not work, because prefix, suffix nothing should work

$m_1, m_2, m_3 \ldots, m_{q-1} \rightarrow t_1 \Rightarrow \bigoplus_{i=1}^{q-1} F(m_i)$

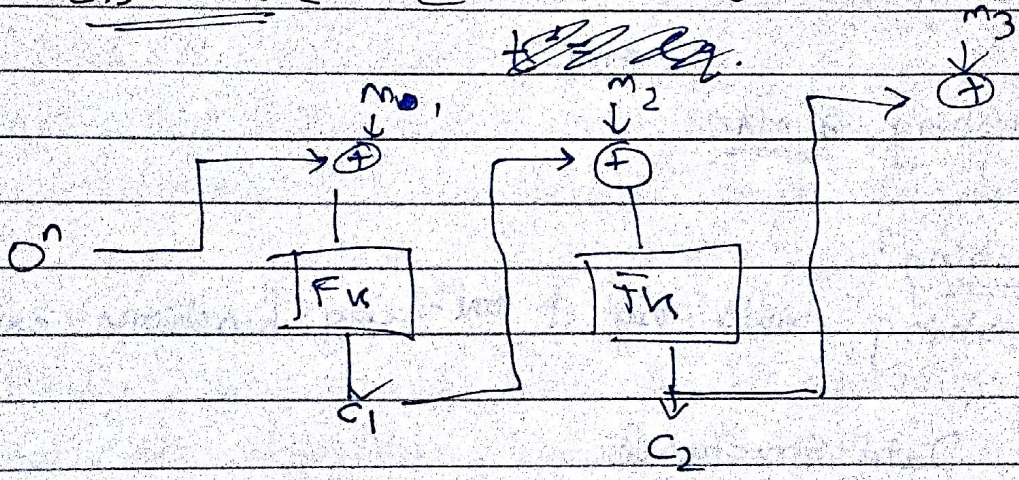
$m_q \rightarrow t_2 \Rightarrow F(m_q)$

Answer for Σm_q is $t_1 \oplus t_2$.

Here, known.

~~$C_{n-1} \oplus m_n$~~.

CBC-MAC (Fixed-length MAC).



In crypto O^n had to be random, otherwise it would fail, here it cannot be random otherwise this one would break.

$m_1 \rightarrow C_1$

$m_1 m_2 \rightarrow C_2$

$(m_1) \| (m_1 \oplus C_1)$

$C_1 \oplus (m_1 \oplus C_1) \equiv m_1 \rightarrow F_K(m_1) = C_2$

$\therefore C_2 = C_1$ we know

even though we never queried it.

How to handle variable length

① Prepend length → $l_{m_1, \ldots, m_q}$

② K ; $K_1 \leftarrow F_K(l)$

↓

Fixed length cbc

③ K_1 , K_2 → CBC-MAC$_K$ $-t_1$

$t \leftarrow F_{K_2}(t_1)$

H-MAC.

Hashing ⇒ MAC.

↓

Hashing?

→ Hashing exists only if one-way functions exist.

Data Structures

↳ Hash Table

Attributes: i) Collision → $x \neq y$ & $h(x) = h(y)$.

How to handle collisions?

↳ In cryptography, handling collisions is a problem that should never arise.

Imagine :- Password is stored publicly after hashing in the same file.

Collision ⟺ Breaching the collision scheme.

Sc. collision - resistant functions should be present.

Collision-resistant hash function.

$\langle Gen, H \rangle$

Generator outputs a key $s \xleftarrow{R} \{0,1\}^n$

$$H^s : \{0,1\}^* \longrightarrow \{0,1\}^{\{m\}}.$$

For all PPTM A,

$$Pr\left[A(\sigma, H^s) \text{ finds } x \neq y,\ H^s(x) = H^s(y)\right] < \text{negligible}.$$

We have to have a key-based hash unlike SHA-1 where finding a collision is easily as it encrypts the files in 128 bits

Just the difference is that here key is public, it is an indexer. It is not a secret key.

$$H^s : \{0,1\}^* \longrightarrow \{0,1\}^{p(m)}.$$

Hence, we should say for a randomly chosen s, such an algorithm should not exist.

We ~~don't~~ know ~~whether~~ that AGI exists ~~or not~~, as we don't know whether we have capability of finding it, → SHA1.

↳ We don't even know whether AGI exists or not → Our cryptographic hashes.

MD Transform (Merkel Damgard Transform)

↳ we want a hash function which can compress slightly. Then, we can apply it many many times & get an output. We don't know how to break it.

By having a key, we have ~~infinite~~ many such function making it much collision-resistant.

Birthday Attack

Assuming How many people should be there in the room, so that 2 people have the same birthday. 50% of the time $\rightarrow O(\sqrt{n})$ time. Hence, the bit key is double the size so that it is impossible for this attack to work.

(Q) $S = \{a_1, a_2, a_3, \ldots\ldots, a_n\}$.

$y = \{y_1, y_2, y_3, \ldots\ldots, y_q\} \rightarrow y_i \in_R S$.

What's the chance that $\exists y_i, y_i = y_j$?

$$1 - x \leq e^{-x} \quad \forall x \in R.$$

In the range $0 \leq x \leq 1$

$$\boxed{1 - x \leq e^{-x} \leq 1 - \frac{x}{2}}$$

Thm: $Pr[\text{Coll}(N, q)]$ is used for $Pr(y_i = y_j, \exists y$

Thm $\boxed{\frac{(q)(q-1)}{4N} \leq Pr[\text{Coll}(N, q)] \leq \frac{q^2}{2N}}$

As $q \rightarrow \sqrt{N}$, both the bounds are constant.

Proof :- $Pr[\text{Coll}(N, q)] \leq \frac{q^2}{2N} \leq \sum_{i \in q} \frac{1}{N} = \frac{(2)(q)}{2N}$

(This)

→ Hence, we directly derive it from the Union-bound condition.

Prob(Collision) = 1 − Prob(No collision).

We can get Prob(No collision) by induction.

At length 1 → 1 − Prob(collision(N, 1))

Length 2 → 1 − Prob(collision(N, 1)) . (Prob(No collision(N, 1)) | collision(N, 2)).

Hence, length K,

→ $1 - \prod_{j=1}^{K-1} \text{Prob}(\text{collision}(N, j) \cdot \text{Prob}(\text{No collision}(N, j) \mid \text{collision}(N, j+1))$

$\cdot \text{Prob collision}(N, K)$.

Prob(NoCollision(N,0)) →

Prob(No collision(N,2)) ⇒ $\frac{1}{N} \cdot \left(\left(1 - \frac{1}{N}\right) \cdot \left(\frac{2}{N}\right)\right)$

Prob(No collision N,3) ⇒ $\frac{1}{N} \cdot \left(\left(1 - \frac{1}{N}\right) \cdot \left(\frac{2}{N}\right)\right) \cdot \left(\left(1 - \frac{2}{N}\right) \cdot \left(\frac{3}{N}\right)\right)$

Prob(collision(N,K)) ⇒ $1 - \prod_{i=0}^{2-1} \left(1 - \frac{i}{N}\right) \leq 1 - \prod_{i=0}^{q-1} e^{\left(1 - \frac{i}{N}\right)}$

$\leq 1 - \prod_{i=0}^{q-1} e^{\sum_{i=0}^{N} \left(1 - \frac{i}{N}\right)}$

$\leq 1 - \prod_{i=0}^{2-1} e^{\left(\frac{-1}{N}\right)\left(\frac{(q)(q-1)}{2}\right)}$

$\leq 1 - \prod_{i=0}^{q-1} e^{\left(\frac{-1}{N}\right)\left(\frac{(q)\left(\frac{q-1}{2}\right)}{2N}\right)}$

$\leq 1 - \underbrace{e^{h}}_{\leq 1 - \frac{x}{2}}$.

$$\leq 1 - \left(1 \cdot \frac{(q)(q-1)}{2N}\right)$$

$$\leq \frac{(q)(q-1)}{2N}$$ → Only this part is failing as $e^{-x} \leq 1 - \frac{x}{2}$

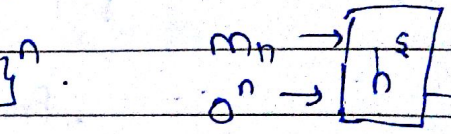
This analysis is correct till $q \to \sqrt{N}$. fails after $n \geq 1$.

Hence, birthday attack works.

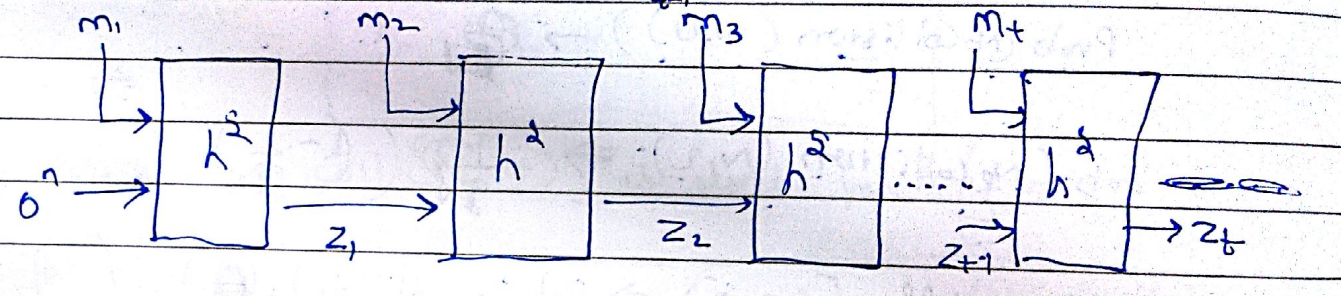
Merkle - Dangard Transform.

$h^s : \{0,1\}^{2n} \to \{0,1\}^n$

(?) $H^s : \{0,1\}^* \to \{0,1\}^n$.



$m = m_1 m_2 m_3 \ldots m_{t-1} m_t$.



We claim that H^s is secure if h^s is secure.

So, if given a collision for H^s, we should generate a collision for h^s.

i.e if $H^s(x) = H^s(y) \Rightarrow h^s(x) = h^s(y)$.

If length is variable then it gets complicated.

as this is not secure.

we prove it to be secure when $len(x) = len(y)$

z_t for both is same.

Since, $x \neq y$ but $z_{t_x} = z_{t_y}$.

→ Then, z_i should exist where they are not same as inputs are not same, but outputs are same. We can then say that that block h^s has a collision. Hence, $x \neq y$, some m_i would not be same & m_{i+1} is same, that h^s has a collision.

If $|x| \neq |y|$ (len) then no of iterations are not same and hence, then just append a layer which gives input the lengths,

the only way we could get a collision is to get a collision in the last block, but since both lengths are different, we

&

Z_p^* → given g & p, ~~finding~~ $g^x \pmod p$, is ~~hard~~.
finding x is hard.

$G, g \rightarrow h^s : Z_p^* \times Z_p^* \rightarrow Z_p^*$

Gen ~~s~~ $s \xleftarrow{R} Z_p^*$.

$$h^s(a, b) = g^a s^b \pmod p.$$

↳ Our idea is that if we can break this in PPTM, then we can break discrete log in PPTM.

$\langle a_1, b_1 \rangle$ and $\langle a_2, b_2 \rangle$ such that

$$\langle a_1, b_1 \rangle \neq \langle a_2, b_2 \rangle$$

$$g^{a_1} s^{b_1} = g^{a_2} s^{b_2} \pmod p.$$

If we get a_2, b_2, then we can get $\log(p)$

$$g^{a_1 - a_2} = s^{b_2 - b_1} \pmod{p}$$

Take to the power $1/(b_2 - b_1)$ on both sides

$$g^{\frac{a_1 - a_2}{b_2 - b_1}} = s \pmod{p}.$$

$$\boxed{\log_g s = \frac{a_1 - a_2}{b_2 - b_1} \pmod{p}}$$

Hence, we get the discrete log if $b_2 \neq b_1$.

$b_2 \neq b_1$ as if $b_1 = b_2$, then $a_1 = a_2$ should hold, but $\langle a_1, b_1 \rangle \neq \langle a_2, b_2 \rangle$

What if $s = 1$?

↳ we can easily find $a_1 = a_2$ and for any b.

What do we do now?

↳ when $s = 1$, then discrete log is not an NP hard problem as ans $:= 0$. Hence, $s \neq 1$ as then it is not valid that discrete log is NP-hard.

Advanced Encryption Standard [AES].

Designing CCA-secure encryption schemes using:
(a) CPA-Secure Encryption.
(b) MAC.

CCA-secure is the best we can ever do.

AES
↳ 128-bit version (data block & key) Encryption.
(Rijndael Cipher).

Block-size → 128 bits.

$AES_K : \{0,1\}^{128} \longrightarrow \{0,1\}^{128}$

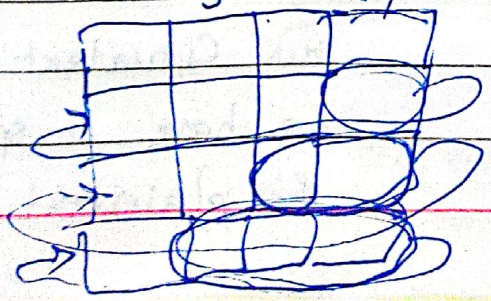
We have 192 bits & 256 bit versions too!

AES Encryption → Sub-bytes ShiftRows MixColumns AddRoundKey.

Without the AddRoundKey in the last step, we can invert it back to the nearest AddRoundKey.

SubBytes → we have a fixed S-Box, which basically says what bytes to substitute by what bytes.

ShiftRows → 1st Row shifted by 0, 2nd is by 1 bytes, 3rd is by 2 bytes, 4th is by 3 (bytes or cells)



Galvafield → 2^8. We represent elements of the field in $GF(2^8)$ which has 8 coefficients & the polynomial has degree 7, we do a bitwise XOR is the addition in the field.

Lookup $G(2^8)$ arithmetic.

SubBytes → Shift Rows → Matrix Multiplication → with a fixed matrix

↓ Lookup

↓ Shift rows

→ Add round key

we have to add in $G(2^8)$, which is basically doing a bitwise XOR the values & get the answer.

Then we have the Key Schedule.

To fill W_i given W_{i-1} column, we do a rotation to right and then SubBytes

Then we have a W_{i-4}, W_{i-1}, W_i & Rcon

↓

DO 2^{i-1} in $G(2^8)$

2^3 in $G(2^8) \equiv 01$.

$\langle r, F_k(r) \oplus M \rangle$

$M_0 = 0^n$.

$M_1 = 1$.

This decryption based scheme is very useful in the cases where we know how to change our Ciphertext such that we have a specified change in the plaintext.

Encrypt-then-Authenticate paradigm

→ Authenticate then encrypt, etc doesn't work.
Only encrypt, then Authenticate works.

Enc_K (CPA-secure).
MAC_{K_2} (MAC secure)

$c = \langle Enc_K(m), t \rangle$ is our Ciphertext

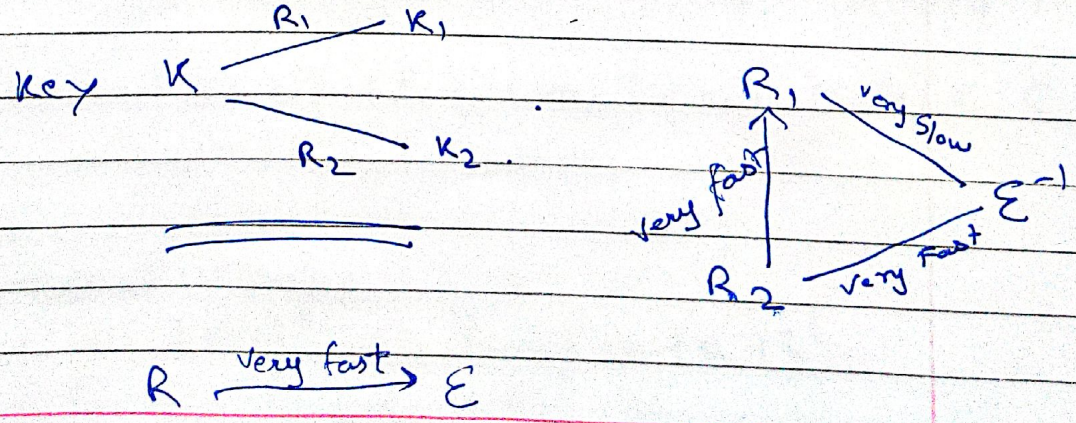
$t \leftarrow MAC_{K_2}(Enc_{K_1}(m))$.

The adversary cannot forge any message as since he cannot forge any document having the same tag.

K_1 → Generated using Gen Algo for Ciphertext
& K_2 by → Gen algo for MAC's.

How does the decryption algo work?
↳ Any modification which has a valid tag is allowed.

Addition is very slow if the data is represented in product of primes. ⇒ Different operations → amount of time representative.



PAGE NO.:

Publish K_1 ⟶ R's Public Key.

$E_K(M) = C$ but no-one knows the decryption.

Only I can apply $\vec{E_K}(M) \to M$

i.e. only I can decrypt the Ciphertexts.

Say Information is x in R_2.

conditionally hard.

$x \longrightarrow f(x)$.

This to be very fast

Polynomial (In P)

But, it should also have an additional information (trapdoor) without which it is very hard