classmate
Date 5/06/18
Page

Introduction to Cryptology — Bimal Roy

- Encryption and Decryption :-

→ $\mathcal{M}$: message space

M : message $\in \mathcal{M}$

$\mathcal{C}$: cipher space

C : cipher $\in \mathcal{C}$

We convert these to binary. The cipher C is a binary string.

→

Encryption $E_K(M) = C$

We have to find the encryption function

→

Decryption $D_{K'}(C) = M$

K and K' may be same or different

Same → Symmetric Key

Different → Asymmetric Key / Public Key

→ E_K and $D_{K'}$ should be easy to compute

→ If someone gets C, E and D, but not K and K', it is going to be difficult to compute M.

- Shannon's Notion of Perfect Secrecy :-

Unconditional probability: $P(M=m)$

Conditional probability : $P(M=m / C=c)$

If $P(M=m/C=c) = P(M=m)$ for all m, c, then the notion of perfect secrecy is defined.

m is the bits of the message which we have to send.

- A simple cryptosystem : (example of Shannon's criteria)

A —one bit→ B (Two people communicating with 1 bit)

$M = \begin{cases} 0 & \text{probability} = 0.6 \\ 1 & \text{probability} = 0.4 \end{cases}$

$K = \begin{cases} 0 & \text{if H} \\ 1 & \text{if T} \end{cases}$

We have tossed a coin. If the result is H, then the key is 0, else if it is tail, the key is 1

Encryption: $C = M \oplus K$

↑ addition modulo 2.

Decryption: $M = C \oplus K$

Now,

m can be 0 or 1

k can be 0 or 1

Thus, we can have 4 combinations between m and c.

$P(M=0/C=0) = 0.6$

$P(M=0/C=1) = 0.6$

$P(M=1/C=0) = 0.6$

$P(M=1/C=1) = 0.6$

We apply Bayes' Theorem to the get the above four probabilities.

$$\therefore P(M=0/C=0) = \frac{P(C=0/M=0)}{P(M=0/C=0) + P(M=1/C=0)}$$ [∵ we know the cipher]

$$= \frac{0.5 \times 0.6}{(0.5 \times 0.6) + (0.5 \times 0.4)}$$

$$= \frac{0.6}{1}$$

$$= 0.6$$

Problem: Size of the key = Size of the message, which is impossible to manage.

- Diffie-Hellman Key Exchange (1975) —

→ Difficult Computational Problem :-

① Factoring large integers

② Knapsack

③ Discrete Logarithm (we generally have to approximate the logarithm result using Taylor Series)

Example: {0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10} modulo 11 = G.

Discrete logarithm problem

2 $2^2=4$ $2^3=8$ $2^4=5$ $2^5=10$

$2^6=9$ $2^7=7$ $2^8=3$ $2^9=6$ $2^{10}=1$

$\therefore 2^9 = 6$ Here, generator is 2

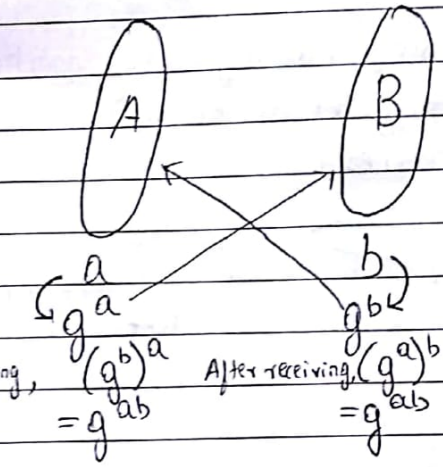
$\therefore \log_2 6 = 9$

If we can express all numbers in the set as a power of another number, then it is called a generator.

g: generator if every element in G can be expresed as some power of g

Given p, we have to find the generator. In our example p=11. g depends on p.

→



g: generator

People has g, g^a, g^b. From here we have to compute g^{ab}. It is like the discrete logarithm problem, and hence not possible to solve this problem till data.

The security depends on a and b only.

Diffie-Hellman is only used for authentication

a and b are supposed to be random numbers which are to be generated at run-time.

Problem: Man in the middle attack

** The Diffie-Hellman paper is said to be the birth of modern cryptography.

This key exchange is meant for private key cryptography, not for public key cryptography.

Date 5/6/18
Page

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

- Buildings - B

Residences - Numbers

B_0 - 00, 01, ... , 09

B_1 - 10, 11, ... , 19

⋮

B_9 - 90, 91, ... , 99

There is a committee of 10 members, one from each building.

This is done using hashing (to make the delete, search, insert functions easier). The hash table has 10 entries.

A_0 - 4

A_1 - 7

⋮

A_9 - 5

From which residence the member is selected.

$h: \{00, 01, \ldots 99\} \rightarrow \{0, 1, \ldots 9\}$

$\therefore h(17) = 7$

Now, we are removing the restriction of one member per building. Here, collision will occur.

- Collision resistant hash functions -

$h: D \rightarrow R$, D: Domain

$|D| >> |R|$ R: Range

↳ This implies that collision is bound to occur. We know this from the Pigeon Hole Principle.

We represent elements of the domain by $\{0,1\}^d$ and elements of range by $\{0,1\}^n$.

If $x \in D$, $h(x)$ should be bounded by a polynomial time algorithm in n. This is for

Efficiency Purpose.

** Read Only Memory: we can write only once and then read. If someone tries to modify, then it should be detected. Here we do not have the knowledge of the data. The function is made public and then the data is chosen.

→ In cryptography, we do not assume any data for the hash functions. The function is open and we have to design it in that fashion.

In cryptography,

$h: D \longrightarrow R$ (D: $\{0,1\}^d$, R: $\{0,1\}^n$)

$2^{64}-1 \rightarrow |D| >> |R| \leftarrow 256$ bit

Sender ($x, h(x)$)	$x, h(x)$ / $x', h(x')$ →	Receiver

If x is changed then $h(x')$ is received.

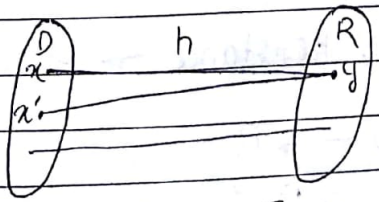
- Definition: A hash function $h: D \rightarrow R$ is an efficiently computable function.

The security measures are:

① Collision Resistance
② Second - Preimage Resistance
③ Preimage Resistance

(SPR superset of CR; PR superset of SPR)

→ Preimage: For $y \in R$ its preimage under h is any $x \in D$ such that $h(x) = y$.



If $|D| > |R|$, atleast one element in R will have two preimages in D

→ Preimage resistance: Given y it should be difficult to compute x, given $h(x) = y$.

→ Second Preimage resistance: Given x, it should be

difficult to compute $x' \neq x$, such that $h(x') = y$.

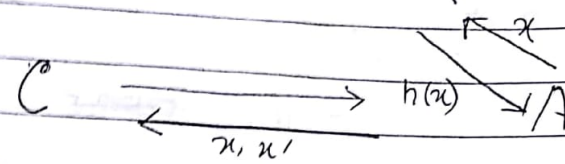
→ Collision Resistant: Given x and y, we have to find x', such that $h(x') = y$.

- Collision - Resistance —

We define it in terms of a game

Players: Challenger C, Adversary A.

$$h : D \longrightarrow R$$



A wins if $x \neq x'$ and $h(x) = h(x')$

Here adversaries are algorithms; more specifically probabilistic algorithms. It probes everytime to get a different x and x'.

$$CR_{adv}[A^h, q] = Pr\{x \neq x', h(x) = h(x')\}$$

↑ Collision Resolution Advantage

↑ Implies Adversary works on the hash function

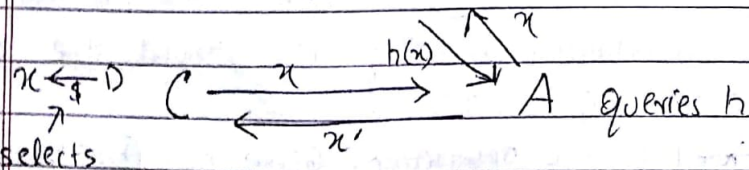
$$CR_{adv}[h, q] = \max_A CR_{adv}[A^h, q]$$

h is (q, ϵ) - collision - resistant, if

$$CR_{adv}[h, q] \leq \epsilon$$

- Second - Preimage Resistance —

$$h : D \longrightarrow R$$



x selects randomly an element from the domain

A wins ... the game if $x \neq x'$ and $h(x)=h(x')$

$\therefore$ $SPR_{adv}[A^h, q] = Pr\{x \neq x', h(x)=h(x')\}$

↑ Secondary Preimage Resistance Advantage

$\therefore$ $SPR_{adv}[h, q] = \max_A SPR_{adv}[A^h, q]$

- Preimage Resistance —

$h: D \rightarrow R$

$x \xleftarrow{\$} D$ C $\xrightarrow{h(x)}$ $\xleftarrow{x'}$ A queries h q times

C picks x at random from the domain and sends h(x) to A.

A wins if $x = x'$ and $h(x) = h(x')$

$\therefore$ $PR_{adv}[A^h, q] = Pr\{x = x', h(x) = h(x')\}$

↑ Preimage Resistance Advantage

$\therefore$ $PR_{adv}[h, q] = \max_A PR_{adv}[A^h, q]$

** A (q, ε) - Collision-Resistant Hash Function is a (q, ε) - Second - Preimage Resistance Hash Function.

[We have to show that if $h \notin (q, \varepsilon)$ SPR then $h \notin (q, \varepsilon)$ CRHF].

** CR $\subseteq$ SPR $\subseteq$ PR.

- $f = x^2 \bmod N \rightarrow N$ is multiplication of two big primes

This is a one way function and a PR function. If y and N is known, it is difficult to compute x. If x is known we can find x' by replacing x by $-x$ or $(x+N)$

** (Q, ε)-CRHF is a (Q, ε)-SPRHF

A (Q, ε) secondary preimage resistant hash function is a $(Q, \ldots)$ preimage resistant hash function.

Now, $(Q, \ldots)$

$\uparrow$

$f(\varepsilon, |D|, |R|)$ with $|D| \geq 2|R|$

- $h: D \rightarrow R$, $|D| \geq |R|$

No. of queries to h required to guarantee collision = $|R|$

Any hash function here will be $(|R|+1, 1/2)$ collision resistant function.

- q balls N bins

If $q > N \rightarrow$ collision guaranteed

If $q < N \rightarrow$ one bin will be empty.

we have to find the probability that one bin contains two balls if $q < N$

~ Probability that the bin number i has the ball $i = \frac{1}{N} \times \frac{1}{N} \times \frac{1}{N} \times \ldots \times \frac{1}{N}$ (q number of balls, q number of bins)

$= \frac{1}{N^q}$

We now have to choose the bins from all bins present, which will be an ordered choice. We choose q bins from all the N bins.

$\therefore$ Total number of choices $= (N)_q$

Now, $(N)_q = N(N-1)\ldots(N-q+1)$.

Hence, Probability that any bin contains no more than one ball is $(N)_q \times \frac{1}{N^q}$ (or atmost 1 ball)

$\sim$ Pr[Bin 1 has Ball 1, Bin 2 has Ball 2, ... Bin q has Ball q] =

Pr[Bin 1 has Ball 1] x Pr[Bin 2 has Ball 2] x ... x Pr[Bin q has ball q]

$= \frac{1}{N} \times \frac{1}{N} \times \ldots \times \frac{1}{N} = \frac{1}{N^q}$

We choose q bins from N in $(N)_q$ ways

Hence, the probability that each of the N bins contains at most one ball is given by.

Pr[Each of the N bins contains at most 1 ball]

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

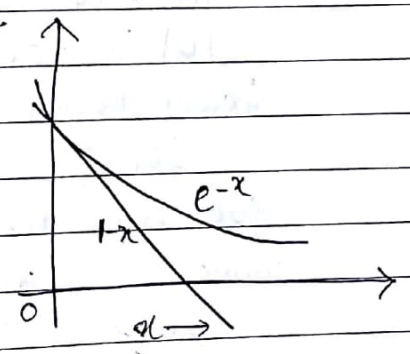
$= R_{N,q}$

$\therefore R_{N,q} = \prod_{i=1}^{q-1} (1 - i/N)$

Now, we know $1-x \leq e^{-x}$ for $0 \leq x \leq 1$

Also $e^{-x} \leq 1 - x/2$, for $0 \leq x \leq 1$

We write the above two as $1-x \leq e^{-x} \leq 1-x/2$ for $0 \leq x \leq 1$



$\therefore R_{N,q} = \prod_{i=1}^{q-1} (1 - i/N)$

$\leq e^{-1/N} \times e^{-2/N} \times e^{-3/N} \times \ldots \times e^{-\frac{q-1}{N}}$

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

Now, we have to find the probability of collision, i.e. one bin contains more than one ball.

$$\therefore Pr[\text{collision}] = 1 - R_{N,q}$$

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

$$\geq 1 - e^{\frac{-q(q-1)}{4N}} \simeq \frac{q^2}{4N}$$

If $q = 23$, $N = 365$, $P[\text{collision}] = 1/2 \rightarrow$ Birthday Paradox

If $q \simeq \sqrt{N}$, then chances of collision are high.

We also can write $q^2/4N$ is the lower bound for collision

Also,

$$Pr[\text{collision}] = 1 - R_{N,q}$$

$$\leq \frac{q(q-1)}{2N}, \text{ where } R_{N,q} \geq 1 - \frac{q(q-1)}{2N}$$

We thus write $\frac{q(q-1)}{2N}$ is the upper bound for collision.

- $h : D \longrightarrow R$

  $|R| = N$

  $|D| \geq 500N$

Lower Bound on Collision Probability: $1 - e^{\frac{-q(q-1)}{2N}}$

Let $q = 2\sqrt{N} + 1$ (Trial and Error)

We pick, $x_1, x_2, \ldots, x_q$ randomly from the domain. $x_1, x_2, \ldots, x_q \xleftarrow{\$} D$

We than apply h to $(x_1, x_2, \ldots, x_q)$ and the distribution $h(x_1), h(x_2) \ldots, h(x_2)$ may not be uniform.

** We have to show that even if the choice is not uniformly random, the probability of collision is bounded by the birthday paradox.

** We also find the probability of collision after subtracting the probability where $(x_a = x_b)$ and show that: the probability is $\geq 3/4$.

Date 5/6/18
Page

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

Slide: [Understanding Symmetric Key Cryptography]

- What we want to Acheive in Cryptography:
  → Privacy (Encryption)
  → Integrity (Message Authentication Code - MAC)
  → Identity / Authenticity (MAC)
  Combination of 2+3 is implemented using MAC
  Combination of 1+3 → Authenticated Integrity

- Symmetric Key Cryptography :-
  → Key K, Enc(K, M) = C and Dec(K, C) = M
  → Minimum Condition : Dec(K, Enc(K, M)) = M
  Here we assume that both sender and receiver have the same Key.
  $Dec(K_i) = Enc^{-1}(K_i)$.
  → AES, DES : small domain algorithms.
  → ---- counter mode encryption, CBC encryption ~~are~~ can encrypt any length messages.
  → AES, DES are Block Ciphers where as Counter mode and CBC are modes of encryption.

Symmetric Key Encryption

One Time Padding : Classical encryption algorithm
→ Two simple ways to encrypt :
① $M \oplus K = C$
② $C = M + K \bmod N$ for some predetermined large N.
→ There are certain issues:
① Performance Issue: Key size is as large as message size
② Security Issue: Key recovery. It leaks information of messages while encrypting more than once.

→ Sender $\xrightarrow[C_2 = M_2 \oplus K]{C_1 = M_1 \oplus K}$ → Receiver

$\therefore C_1 \oplus C_2 = M_1 \oplus M_2$

Hence, in symmetric key cryptography key remains same. Adversary knows $C_1 \oplus C_2 = M_1 \oplus M_2$. Hence the adversary knows $M_1 \oplus M_2$, which is leaking information, leading to security issues.

- Stream Cipher Encryption :-

Classical and efficient encryption for arbitrary sized message.

$$C = G(K, |M|) \oplus M$$

$G : \{0,1\}^k \times N \rightarrow \{0,1\}^*$ such that for all positive integer l and $K \in \{0,1\}^k$, $G(K,l) \in \{0,1\}^l$

This system stretches the message.

- Pseudorandom Bit Generator (PRBG) :-

→ U_n is a random, uniformly distributed n-bit string.

→ For all l, $G(U_k, l)$ should be close to U_l.

→ Definition: $G : \{0,1\}^k \times N \rightarrow \{0,1\}^*$ is called (t, ε, l)-PRBG if for all algorithm D runs in time t,

$$|Pr(D(U_l) = 1) - Pr(D(G(U_k, l)) = 1)| \leq \varepsilon$$

PRBG would solve large key issue. It still cannot use more than once, as it leaks information of messages.

- Classical Streamcipher :-

→ It generates key stream in online manner

→ We need to hold current internal secret state to make use of multiple times

→ If random position key-stream can be generated efficientey, we can still use

• IV- Based Stream Cipher:-

→ IV standards for Initial Value.

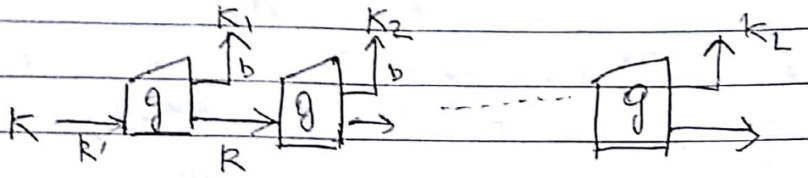
$(C, IV) = PRBG(IV \| K) \oplus$

Here, some salt is generated and added to the message, which is known by sender and receiver.

The lifetime of the key is bounded.

If salt is of size k, the lifetime is bounded by 2^k.

• Generating key stream in online manner:-



g is like a pseudo random series generator.

• RC4 Stream Cipher :-

→ Algorithm:

$i = 0, j = 0$

while generating output:

$i = (i+1) \bmod 256$

$j = (j + S[i]) \bmod 256$ // S is the stream

swap $S[i], S[j]$

$K = (S[i] + S[j]) \bmod 256$

output K

endwhile

K = 256 bytes.

1 byte is generated after each run.

The key stream is public.

In symmetric key only algorithm is public.

** Martin-Shamir Attack on RC4.

- AHACK on WEP based RC4 stream cipher.
- Trivium Stream Cipher.
- Counter Mode Encryption :-
  → Need not hold current internal secret state, moreover we can directly compute the Key-stream.
  → Random position Key-stream can be generated efficiently.
  → Pseudorandom Function:

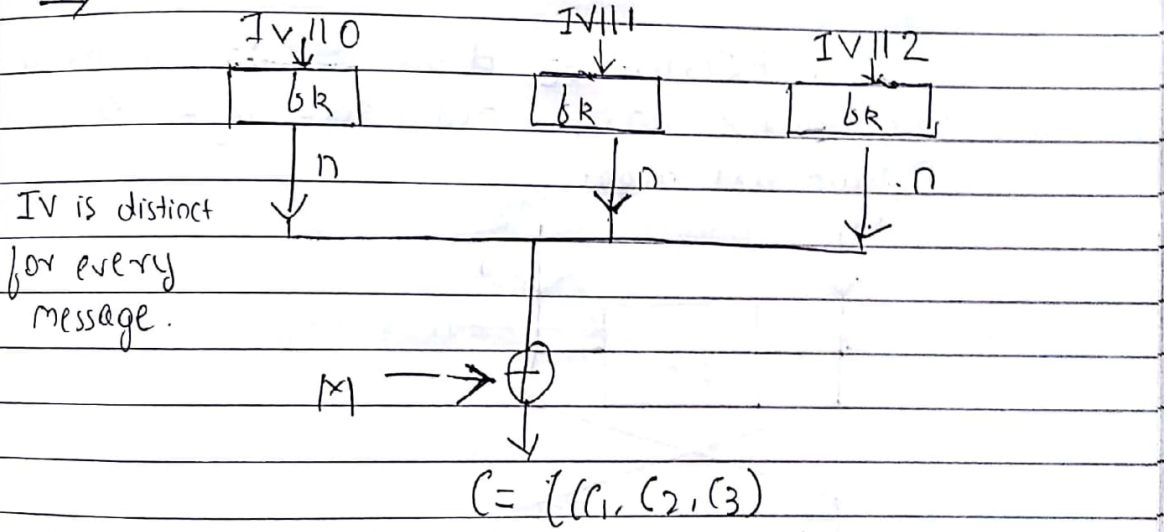
$$f : \{0,1\}^k \times \{0,1\}^m \rightarrow \{0,1\}^n$$

s.t. for all distinct m-bit strings $x_1, \ldots, x_s$

$f(U_K, x_1), \ldots, f(U_K, x_m)$

This is computationally close over the uniform distribution over $\{0,1\}^{ns}$

→



On the fly decrypting can be done. If C_3 is received first ... then C_3 can be XOR-ed with IV||2 to get m, provided we get the IV.

→ Let log L = 0 (maximum message size is n bits)

Let $K \in \{0,1\}^k$ random secret key.

Choose $IV \in \{0,1\}^{n-l}$ which is distinct for each encryption.

We can define key-stream (like stream cipher) of counter mode as

$f_k(IV||0) || \ldots f_k(IV||l'-1)$

where $(l'-1)n < |M| \leq l'n$.

Instead of keyed function, we use a keyed permutation (also called a block cipher) is more popular. Here $m=n$.

→ Designing the function f_k is not easy.

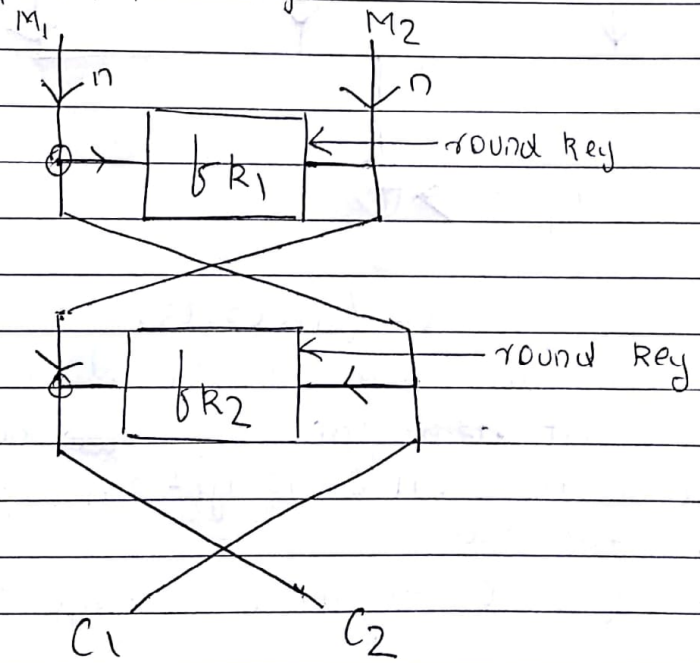
- Block Cipher :-

→ Two ways of defining block cipher :-

1. SPN (eg : AES)
2. Feistel (eg : DES)

→ In SPN, we compute multiple confusion and diffusion stages. In the confusion stage we perform addition of bits and in diffusion we perform swapping of positions.

→ In Feistel — we divide the message into two parts – plain text left and plain text right



From C_1 and C_2, we must be able to fetch M_1 and M_2.

→ Attacks on Fiestel :-
- If we use same key in each round.
- Chosen plaintext attack on 2 rounds.
- Chosen ciphertext on 3 rounds.

Security: 3 round is secure against chosen plaintext and 4 round is secure against chosen plaintext and ciphertext adversaries.

• Security Notions of Symmetric Key Encryption—
→ Need to describe: ① Power
② Goal of adversary
→ Power: only ciphertext, both plaintext and ciphertext. Number of such texts. Access of encryption/decryption function etc.
→ Goal: key recovery, message recovery, some information about message, distinguishing from ideal encryption etc.
→ Distinguishing: Adversary wants to differentiate between all the messages. This is the minimum goal of the adversary. We do not consider a encryption algorithm which does not provide this minimum amount of security.

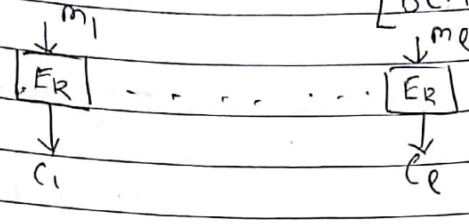
• Symmetric Key Primitives—
Distinguishing Game: Distinguishing a real keyed construction from an ideal object.
→ Pseudorandom Function or PRF:
Indistinguishable from (uniform) random function
→ Pseudorandom Permutation or PRP:
Indistinguishable from (uniform) random permutation by only making forward queries.
→ Strong Pseudorandom Permutation or SPRP):
Indistinguishable from (uniform) random permutation by only making forward and backward (i.e. inverse) queries.

• ECB Encryptions —

Electronic code-book Encryption (ECB) is used for larger messages.

$C_1 = BC(K, M_1), \ldots, C_\ell = BC(K, M_\ell)$

[BC: Block Cipher]



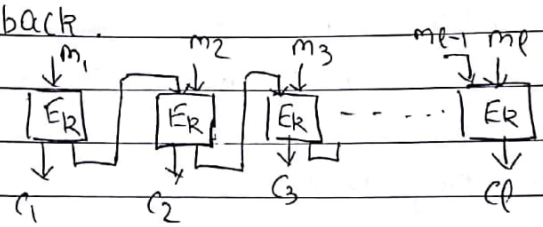
We need the inverse of E_K to get the decrypted message.

There are ℓ blocks, i.e. the message is divided into ℓ blocks.

ECB encryption may not be that good; as we are encrypting each block or word separately.

Drawback: If $m_1 = m_2$, then $c_1 = c_2$. So we get an idea about the message is received.

We can correct the drawback by using a feedback.



• OCB and CBC Encryption —

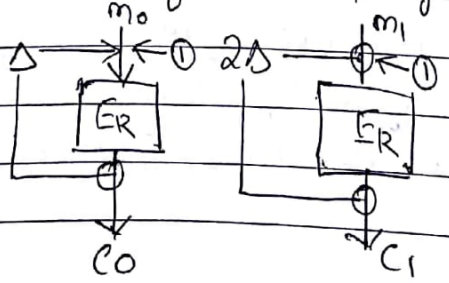
OCB resolves the drawbacks of ECB

$C_1 = BC(K, M_1 \oplus \Delta) \oplus \Delta$

$\vdots$

$C_\ell = BC(K, M_\ell \oplus 2^{\ell-1}\Delta) \oplus 2^{\ell-1}\Delta$

Δ is generated from IV, eg. $BC(IV) = \Delta$



We change Δ for every block.

We have to find the probability, that the adversary can guess Δ.

Here, $m_0 \oplus \Delta = m_1 \oplus 2\Delta$ — (1) points in the diagram

$\therefore \Delta = m_0 \oplus m_1$

As Δ is random, the probability of finding $m_0 \oplus m_1$ has very low probability.

IV is not secret, but Δ is secret

Here, all the cipher texts are assumed to be random.

CBC uses a sequential approach, where the message of the ith stage is XORed with a feedback from (i-1)th stage.

→ $\Delta \in \{0,1\}^n$

we view Δ as $\Delta \in \{0, 1, \ldots, p-1\}$ where p is prime

We want $2\Delta \bmod p$

$2^2\Delta \bmod p$ Multiplying by 2 means left shift

$\vdots$

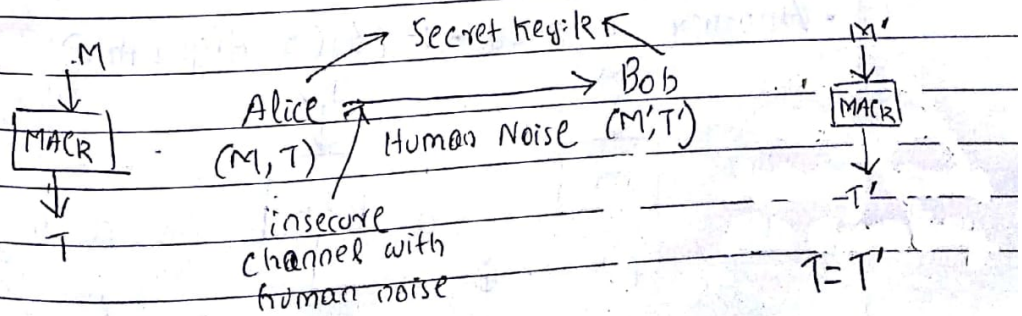
$2^i\Delta \bmod p$ to be distinct

The multiplicative group in Z_p is cyclic, i.e it is a generator. Hence, after running n times, we get 0, and 2 is the primitive of the finite field.

Slide: [Different Candidates for Message Authentication Codes]

- Alice → Bob
  M — Statistical Noise — M'
  Here, we use Error correcting Code.

22.



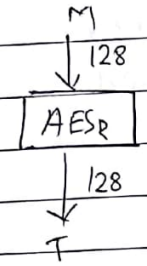
Statistical Noise can be corrected using error correcting codes. But if there is human noise, it cannot be corrected. Hence, we need a mechanism to understand if error has occured or not.

→ The goal of the adversary is:
Find M' and T'
$\Rightarrow MAC_K(M') = T' \;\&\; (M';T') \neq (M_i;T_i)\ \forall i$

If attacker changes a message to previous message then this attack cannot be prevented by timestamp.

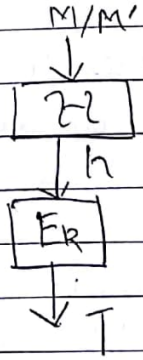
Power → Adversary knows MAC_K

• A simple example :- (Hash-Then MAC) (Universal Hash Based MAC)



We Apply an injective padding (eg. 10*)
Then, we compute
$T = AES_K(M^*)$, M^* is the padded message.

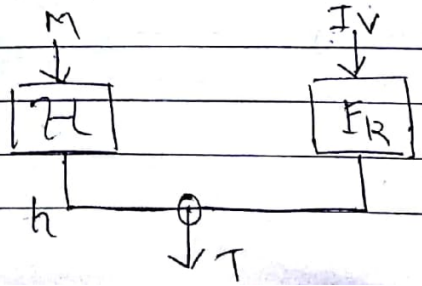
Forgery-security depends on the corresponding security for $AES_K()$.
For large messages, we use a hash function.



(M, T) ⇓ (M', T)

This is known as Hash Then MAC approach

• Another approach :- (WCS Algorithm)



This is used as a tag value, not for encryption

Here, for two messages M and M', $H(M) \oplus H(M') = \delta$, where δ is very small, and is the difference between the messages.

We perform the following queries:

$(M, IV) \Rightarrow T$

$(M', IV) \rightarrow T' = T \oplus \delta$

If we use H_{k_2} instead of H_{k_1}, then probability of finding δ is tough.

→ Almost XOR Universal Hash Function (AXU):

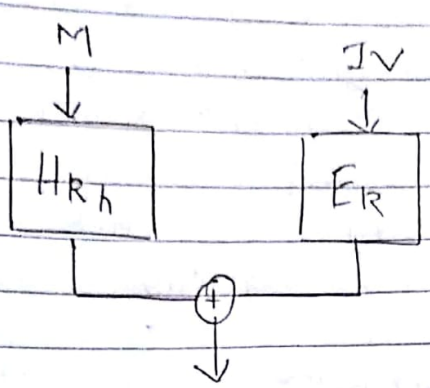
$$Pr[H_{k_2}(M) \oplus H_{k_2}(M')] = \delta \leq \frac{1}{2^n}$$

[$M \neq M'$ and k is random]

classmate
Date 5/6/18
Page

Tutorial Class

WCS



H_{K_h} is a Keyed ... hash function.
Polyhash is an AXU hash function.

$|K_h| = n$ bits

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

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

$|M| = \ell n$

$Poly_{K_H}(M) = M_\ell K_h + M_{\ell-1} K_h^2 + \dots + M_i K_h^\ell$

$M = (M_1, M_2, \dots, M_\ell)$,
where each $M_i \in \{0,1\}^n$

If H_{K_h} is a Polyhash function, if we repeat IV even once, then the system will break and the secret key can be recognized.

$T = Poly_{K_H}(M) \oplus E_K(IV)$ — ①

$T' = Poly_{K_H}(M') \oplus E_K(IV)$ — ②

Our objective is to find K_H to show that the MAC fails.

We choose M and M' in such a way that if we subtract ① and ②, we get K_h, given the fact that the higher order terms get cancelled.

Now,

$T' \oplus T = Poly_{K_H}(M) \oplus Poly_{K_H}(M')$

Let $|M| = 2n$ bits

$(M = M_1, M_2, IV)\ T$

$(M = M_1', M_2', IV)\ T'$

$\therefore T \oplus T' = M_2 K_h \oplus M_2' K_h \oplus M_1 K_h' \oplus M_1' K_h'$

$= M_2 K_h \oplus M_2' K_h \oplus 0$ [$M_1 K_h'$ and $M_1 K_h'$ are same]

$= M_2 K_h \oplus M_2' K_h$

$= (M_2 \oplus M_2') K_h$

We know $T \oplus T'$ and $M_2 \oplus M_2'$

we can write $T \oplus T' = \delta$ and $M_2 \oplus M_2' = \delta_T$

$\therefore K_h = \delta_T \cdot \delta^{-1}$

— (As we are working in the finite field, δ has an inverse)

Our goal is to come up with a new message (M^*, IV^*, T^*) such that

$Poly_{Kh}(M^*) \oplus E_k(IV^*) = T^*$

We choose IV^* same as IV

From (1), $E_k(IV) = T \oplus Poly_{Kh}(M)$

we know,

$Poly_{Kh}(M^*) \oplus E_k(IV^*) = T^*$

or, $Poly_{Kh}(M^*) \oplus T \oplus Poly_{Kh}(M) = T^*$

We choose any M^* other than M and M', and compute T^*.

- **PRF**

$F : K \times D \longrightarrow R$

two valued function; K: key; D: Domain; R: Range

Assume $k = 2^n$

$\{F(k_1, \_), F(k_2, \_), \ldots, F(k_{2^n}, \_)\}$

$F(k_1, \_)$: a function where first input is fixed

Set of Functions or Family of Functions

$$Func(D,R) = \{ f : D \to R \}$$

$|D| \to$ finite

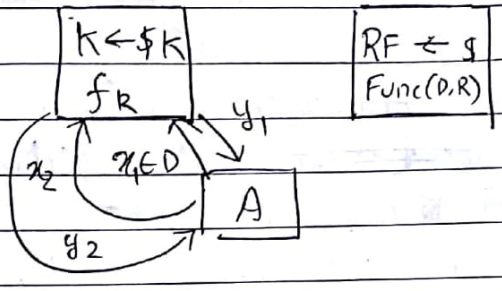
$|R| \to$ finite

$$|Func(D,R)| \to |R|^{|D|}$$

$$Pr\left[RF = f\right] = \frac{1}{|R|^{|D|}}$$

Advantage of PRF:

$$Adv_F^{PRF}(A) = Pr\left[A^{F_K} = 1\right] - Pr\left[A^{RF} = 1\right]$$

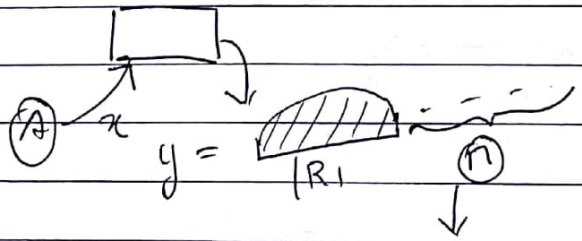


Insecure PRF construction:-

(i) $F_K'(x) = F_K(x) \,||\, 0^n$

$F_K : D \to R$

$F_K' : D \to R \times \{0,1\}^n$



The adversary will check whether it is all 0. If it is 0, the adversary sends 1.

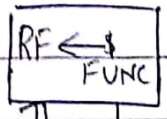
$$Pr\left[A^{F_K} = 1\right] = 1$$

A^{F_K} : K is a random variable. This implies for $K \in K$: A interacts with F_K' to give output 1.

$$\therefore Pr[A^{F_K} = 1] = \frac{|\{K \in K : A \text{ interacts with } F_K' \text{ to give } 1\}|}{|K|}$$

$$= \frac{|K|}{|K|} = 1$$

$$Pr[A^{RF} = 1] = \frac{|R|}{|R|2^n} = \frac{1}{2^n}$$



x → Func(D, $R \times \{0,1\}^n$) → y

Now, $Pr[Y = y] = \frac{|R|}{|R| \times 2^n} = \frac{1}{2^n}$ ← $Pr[Y = y] = \frac{1}{|R| \times 2^n}$

where Y is a random ~~variable~~ such that the last n bits are ~~zero~~

Taking a random Y from $R \times \{0,1\}^n$

$$\therefore Adv_F^{PRF}(A) = \left(1 - \frac{1}{2^n}\right)$$

If $n = 128$, $1/2^n$ is very small, i.e. $Adv_F^{PRF}(A)$ is close to 1.

Hence, this construct is not secure.

(ii) $F_K'(x, y) = F_K(x) \oplus F_K(y)$