[illegible]

The "AES IRREDUCIBLE POLYNOMIAL" is $P(x) = x^8 + x^4 + x^3 + x + 1$

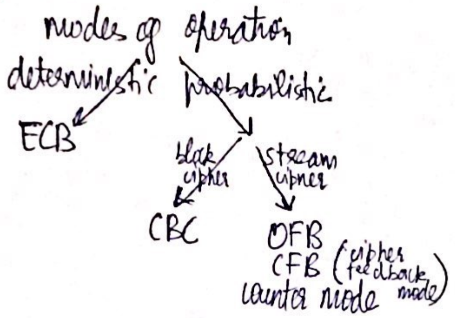
(d) Inversion in $GF(2^m)$

Again, the inverse $A^{-1}(x)$ of an element $A(x) \in GF(2^m)$ must satisfy
$A(x) \times A^{-1}(x) = 1 \mod P(x)$
↳ to find this we need extended Euclidean Algorithm

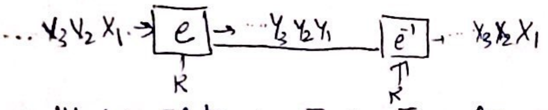
MODES OF OPERATIONS FOR BLOCK CIPHERS.

→ ways of using a block cipher for encryption.

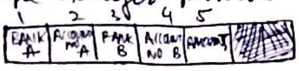
→



→ ELECTRONIC CODE BOOK MODE (ECB)



Attack: Electronic Funds Transfer
simple transfer protocol



(1) Assumption : Each field is exactly n bits wide.

(2) Assumption: Key K_{AB} is fixed for some time (ie no changing the key for some no. of transfers)

OSCAR is an active attacker (listen as well modify).

(a) OSCAR opens one account at bank A and one at bank B.

(b) OSCAR transfers repeatedly €1 from his A account to his B account.

(c) OSCAR wiretaps and checks for messages with identical ciphertext blocks. BL1||BL2||..||BL5 and he stores encrypted block BL4

(d) In all future transfers with BL1 and BL3 replace ~~[illegible]~~ 4th block to BL4. all transfers A→B are redirected to OSCAR'S ACCOUNT.

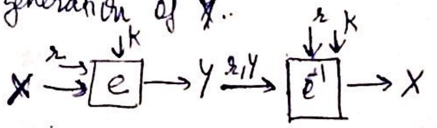
Note: (1) OSCAR does not break e( ).
(2) similar to the letter frequency attack against substitution cipher.

→ CIPHER BLOCK CHAINING MODE (CBC)

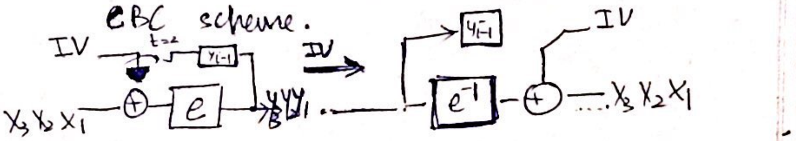
we want to solve 2 problems:
① make encryption probabilistic
② combine encryption of all blocks.

An encryption scheme is "DETERMINISTIC" if a particular PT is mapped to a fixed CT if the key is unchanged.

A "PROBABILISTIC" encryption scheme uses randomness to achieve a non deterministic generation of y..



CBC scheme.



$$\text{enc} = y_1 = e_K(x_1 \oplus IV)$$

$$y_i = e_K(x_i \oplus y_{i-1}); \; i \geq 2$$

$$\text{dec} = x_1 = e_K^{-1}(y_1) \oplus IV$$

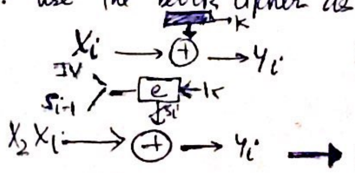
$$x_i = e_K^{-1}(y_i) \oplus y_{i-1}, \; i \geq 2$$

IV → Initial Vector.
→ does not have to be secret
→ should be "NONCE" or "NUMBER USED ONLY ONCE".

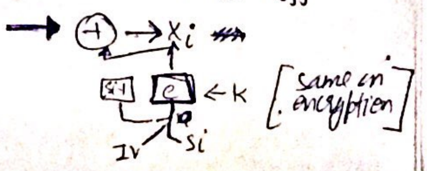
eg: for IV generation
1) TRUE RANDOM NUMBER
2) COUNTER VALUE. (must be stored by ~~[illegible]~~ ALICE)
3) $ID_A || ID_B ||$ TIME

→ OUTPUT FEEDBACK MODE (OFB)

Idea: use the block cipher as a keystream generator:



XOR bit by bit
here it from e_j with ~~[illegible]~~ $x_{i,j}$



[same in encryption]

(2) BRUTE FORCE ATTACK

given: (x_0, y_0)

$$DES^{-1}_{k_i}(y_0) \overset{?}{=} x_0 \; ; \; i = 0, 1, \dots 2^{56}-1$$

Deepcrack (1998) → special purpose DES hardware cracker → $250,000 machine proved DES is no longer secure.

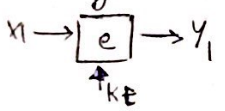
COPACOBANA (2007) → did it in $10,000

DES Alternatives:

cipher	comment
AES	de facto world standard
3 DES	still very secure
AES - Finalists	4 ciphers all very secure RC6, serpent etc

BRUTE-FORCE ATTACK'S REVISITED:

Exhaustive key searches can give FALSE POSITIVE results



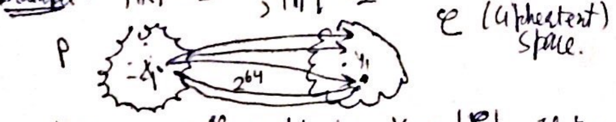
k_t → Real key.

i.e. $e_{k^{(1)}}(x_1) = y_1$ [found by OSCAR]

but $k^{(1)} \neq k_t$

Likelihood that this happens depends on the relative size of the key space $|K|$ and plaintent space $|P|$.

Example: $|K| = 2^{80}$, $|P| = 2^{64}$

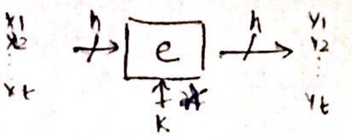


there are 2^{80} mappings $x_1 \rightarrow |ℭ|$. If the mappings select random elements from ℭ.

key candidates $= \frac{2^{80}}{2^{64}} = 2^{16}$

Note: ONE key candidate is the target key.

Idea: use second pair PT/CT

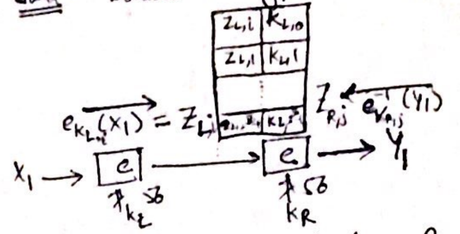


Given a block ciphere with a key length λ and block size n, and t pairs of PT/CT the expected number of False keys is

$$2^{\lambda - tn}$$

DOUBLE ENCRYPTION:
(meet in the middle attack).

Idea: Double encrypt



complexity for brute force?

naiive: $x_1 \overset{?}{=} e^{-1}_{k_i}(e^{-1}_{k_j}(y_1))$

$2^{56} \times 2^{56} = 2^{112}$ key tests.

⇒ Lifetime of UNIVERSE etc.

Q: Can we search for k_L and k_R separately?

Ans → than $\underbrace{2^{56}}_{\text{search for } k_L} + \underbrace{2^{56}}_{\text{search for } k_R} \rightarrow 2^{57}$ hardly more secure than single encryption

⇓

MEET IN THE MIDDLE ATTACK:

Phase I: Search through all k_L → store intermediate values z_L.
complexity → $2^{56} + 2^{56}$ storage locations

Phase II: compute $z_{R,j} = e^{-1}_{k_{R,j}}(y_1)$ $k_{R,j} = 0, 1, \dots, 2^{56}-1$
For every $z_{R,j}$ check for collision, i.e. is there a value $z_{L,i}$ s.t. $z_{R,j} \overset{?}{=} z_{L,i}$. If so, $(k_{L,i}, k_{R,j})$ that were used in the collission are possible keys (k_L, k_R).

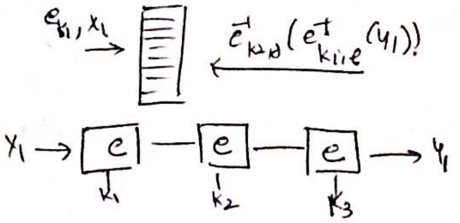
Note: sometimes we have to use a second pair (x_2, y_2). $x_2 \overset{?}{=} e^{-1}_{k_{L,i}}(e^{-1}_{k_{R,j}}(y_2))$

TOTAL COMPLEXITY = 2^{56} enc + 2^{56} storage
+ 2^{56} enc

2^{57} enc + 2^{56} storage.

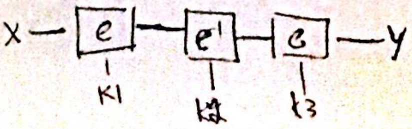
DOUBLE ENCRYPTION IS ONLY MARGINALLY MORE SECURE THAN SINGLE ENCRYPTION.

TRIPLE ENCRYPTION:



total: $2^{56} + 2^{56} + 2^{56} \cdot 2^{56}$.
≈ 2^{112} (BF → 2^{168})

⇒ 3 DES has an effective key length of 112 BITS



example:

Alice

$x = 4$

Bob

(1) $p = 3, q = 11$

(2) $n = pq$
$= 33$

(3) $\phi(n) = 2 \times 10$
$= 20$

(4) Choose $e = 3$
$\gcd(3, 20) = 1$ ✓

(5) $d = e^{-1} = 7 \mod 20$

$\xleftarrow{K_{pub}(33,3)}$

$y = 4^3$
$= 64 \mod 33$
$= 31$

$\longrightarrow$ $x = y^d = 31^7$
$= 4 \mod 33$

TRICK: $31^7 \equiv (-2)^7 \mod 33$
$\equiv -128 \mod 33$
$\equiv -4 \cdot 33 + 4 \mod 33$
$\equiv 4 \mod 33$ (dd).

$$\frac{x^{2^{1024}-1} \cdot x = x^{2^{1024}}}{2^{1024}-1 \text{ MULTIPLICATIONS}} \qquad \frac{x^{2^{1023}} \cdot x^{2^{1023}} = x^{2^{1024}}}{1024 \text{ MULTIPLICATIONS}}$$

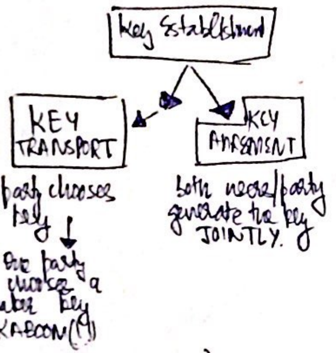
SQUARE and MULTIPLY ALGORITHM
(binary method / left to Right exponiation)

Ex: x^{26} $(26)_{10} = (11010)_2$

SQ	$x \cdot x = x^2$	$(x^1)^2 =$	$x^{(10)_2}$
MUL	$x \cdot x^2 = x^3$	$x^1 \cdot x^{(10)_2} =$	$x^{(11)_2}$
SQ	$x^3 \cdot x^3 = x^6$	$(x^{11})^2 =$	$x^{(110)_2}$
SQ	$x^6 \cdot x^6 = x^{12}$	$(x^{110})^2 =$	$x^{(1100)_2}$
MUL	$x \cdot x^{12} = x^{13}$	$x^{(1100)_2} \cdot x^{1_2} =$	$x^{(1101)_2}$
SQ	$x^{13} \cdot x^{13} = x^{26}$	$(x^{(1101)_2})^2 =$	$x^{(11010)_2}$

working scheme: scan the exponent bits left to Right:
1) in every iteration we SQUARE
2) if current bit is 1: MULTIPLY by x

SYMMETRIC KEY ESTABLISHMENT

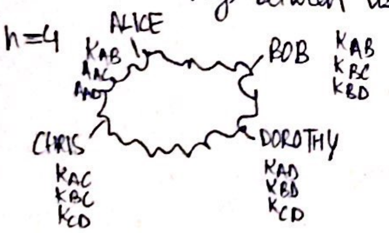


one party chooses the key ↓ if one party chooses a weaker key KABOOM(!)

both users/party generate the key JOINTLY.

Naive Approach: The n^2 Key Distribution Problem of setup.

Assumption → Establish pairwise secret keys between users



for n users:
# keys → $n(n-1) \approx n^2$
# key pairs → $n(n-1)/2$

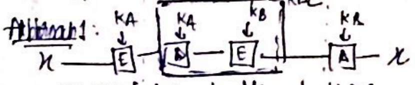
Drawbacks: (1) Large \# of keys ($O(n^2)$).
(2) Adding new users is COMPLEX.
(need to update all other users also))

KEY DISTRIBUTION CENTER (KDC) BASED PROTOCOLS

Idea: Central "trusted authority (KDC)" that shares one key with every user.

ALICE	KDC	BOB
K_A	K_A, K_B	K_B

Approach 1:



communication traffic at KDC becomes a BOTTLENECK

Approach 2:

REQUEST(ID_A, ID_B) →

KDC generates random $K_{session}$
$y_A = e_{K_A}(K_{session})$
[we encrypt the keys]
$y_B = e_{K_B}(K_{session})$

← y_A, y_B

$K_{session} = e^{-1}_{K_A}(y_A)$

$y = e_{K_{session}}(x)$

$\xrightarrow{(y, y_B)}$ $K_{session} = e^{-1}_{K_B}(y_B)$

$x = e^{-1}_{K_{session}}(y)$

for n users:
# key pairs → n $(O(n))$
# keys → $2n$ $(O(n))$

Advantage:
just add to the KDC and we are done. Ex
Adding a ~~user~~ new user only requires secure channel KDC ↔ user at initialisation time.

Remark: $K_A, K_B, K_C \ldots \ldots$ → "KEY ENCRYPTIONS KEYS" or KEKs.

WEAKNESSES:

(1) KDC is a single point of failure.

(2) NO "PERFECT FORWARD SECRECY" ie if the KEKs are compromised, then all past communication can be Decrypted.

Remark:
KDC is the basis for KERBEROS.

(3) REPLAY ATTACK & KEY CONFIRMATION ATTACK

↓ neither Alice or Bob know whether $K_{session}$ is actually a new one.
↳ use (?)

↓ Alice doesn't know that the key she receives from KDC is actually for a session b/w her and BOB.

ASSYMETRIC KEY ESTABLISHMENT:

2 PK approaches:

(1) Key agreement eg DH

(2) key transport

MAN IN THE MIDDLE (MITM) ATTACK:

DH revisted, but with active attacker.

ALICE

$K_{pr} = a, K_{pub} = \alpha^a = A$

$\xrightarrow{A}$

OSCAR.

BOB

$K_{pr} = b, B = \alpha^b = K_{pub}$

$\xleftarrow{B}$

$k_{AB} = B^a$

$K_{AB} = A^b$

$\tilde{A} = \alpha^{\theta_1}$ & $\tilde{B} = \alpha^{\theta_2}$

$\xrightarrow{\tilde{A}}$

$\xleftarrow{\tilde{B}}$

~~k_{AB}~~

$K = (\tilde{B})^a$

$K_{AO} = (\alpha^{\theta_2})^a$

$K_{AO} = A^{\theta_2}$

$K_{BO} = B^{\theta^{\theta_1}}$

~~K_{AB}~~

$K = (\tilde{A})^b$

$K_{BO} = (\alpha^{\theta_1})^b$

OSCAR shares ~~a~~ one session key with Alice and one with Bob. however, Alice and Bob still think they are talking to each other

∴ OSCAR has now full control over the communication over ALICE and BOB.

ALICE | OSCAR | BOB

$Y = AES_{K_{AO}}(x)$

$\xrightarrow{Y}$

$x = AES^{-1}_{K_{AO}}(Y)$

$\xrightarrow{Y}$ $AES^{-1}_{K_{BO}}(Y) \neq x$ (dd)

Bob might figure out that something wrong ∴ OSCAR DOES THIS.

$Y' = AES_{K_{BO}}(x)$

$\xrightarrow{Y'}$

$x = AES^{-1}_{K_{BO}}(Y)$

& then BOB sends back the REPLY etc etc.

→ The MITM attack works against ALL Public key (PK) schemes. (dd)

Q→ what is the basis of the attack??

Ans → The public key's are not authenticated.

Every time Oscar can replaice the Public key with his own one in every ASSYMETRIC protocol

CERTIFICATES:

Idea: use a crypto "tool" that provides AUTHENTICATION.

Digital signature — also asymmetric, OLIVER can still replace it.

MAC → symmetric securing asymmetric defeats the whole purpose

→ needs a CENTRALLY TRUSTED AUTHORITY ("certifying authority" or CA)

~~[illegible]~~

→ $cert_A = [(K_{pubA}, ID_A),$ $sign_{K_{pr,CA}}(K_{pubA}, ID_A)]$ ← computed by the CA

DH with certificates:

ALICE | OSCAR | BOB

$a = K_{pr}, A = K_{pub}$

$b = K_{pr}, B = K_{pub}$

$\xrightarrow{\quad}$ $cert_A =$ ~~[illegible]~~ $= [(A, ID_A), S_A]$

$\xleftarrow{\quad}$ $cert_B = [(B, ID_B), S_B]$

$ver_{K_{pub,CA}}(cert_B)$

$ver_{K_{pub,CA}}(cert_A)$

$K_{AB} = B^a = (\alpha^b)^a$

$K_{AB} = A^b = \alpha^{ab}$

OSCAR now needs to compute $sig_{K_{pr,CA}}(\tilde{A}, ID_A)$ → he does not have the Private key of CA

Scanned with CamScanner