Mid-term Exam

- 3/18 next Mon.
- 1 hour during class
- Written
- Open book
- No electronic devices
  - Considered as cheating
- No talking
  - 20% penalty each time

Double Transposition

- Plaintext: attackxatxdawnx
  - 5 x 3 matrix

	col 1	col 2	col 3
row 1	a	t	t
row 2	a	c	k
row 3	x	a	t
row 4	x	d	a
row 5	w	n	x

Permute rows $\Rightarrow$

	col 1	col 2	col 3
row 3	x	a	t
row 5	w	n	x
row 1	a	t	t
row 4	x	a	d
row 2	a	c	k

Permute cols $\Rightarrow$

	col 1	col 3	col 2
row 3	x	t	a
row 5	w	x	n
row 1	a	t	t
row 4	x	a	d
row 2	a	k	c

- Ciphertext: xtawxnattxadakc
- Key is matrix size and permutations: (3, 5, 1, 4, 2) and (1, 3, 2)

Double Transposition - Decryption

- Ciphertext: xtawxnattxadakc
  - 5 x 3 matrix

	col 1	col 3	col 2
row 3	x	t	a
row 5	w	x	n
row 1	a	t	t
row 4	x	a	d
row 2	a	k	c

Undo cols (1, 3, 2) ⇒

	col 1	col 2	col 3
row 3	x	a	t
row 5	w	n	x
row 1	a	t	t
row 4	x	a	d
row 2	a	c	k

Undo rows (3, 5, 1, 4, 2) ⇒

	col 1	col 2	col 3
row 1	a	t	t
row 2	a	c	k
row 3	x	a	t
row 4	x	d	a
row 5	w	n	x

- Plaintext: attackxatxdawnx
- Does not disguise the letters

1. Double Transposition

- Row permutation, then column permutation
- The key is matrix size and permutations
- The ciphertext is `lealethrawergtoe`

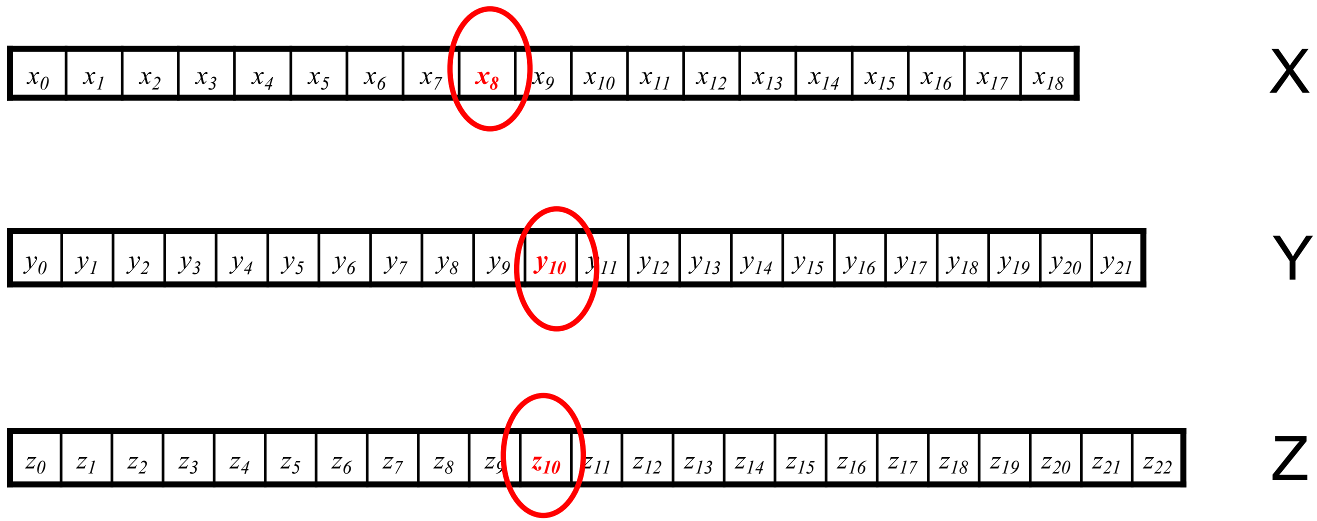
BINGHAMTON UNIVERSITY
STATE UNIVERSITY OF NEW YORK

2. Affine Cipher – Simple Substitution Cipher

- **$c = (a * p + b) \bmod 26$**
- **t -> H, o -> E**
- We can have $7 = 19a + b(\bmod 26)$ and $4 = 14a + b(\bmod 26)$
  - Subtract 2 equations
  - $3 = 5a$ mod (mod 26)
  - $3*5^{-1} = a \pmod{26}$
  - $3*21 = 11 \bmod 26 = a \pmod{26}$
  - $a = 11$
  - $b = 6$
- **$c = 11p + 6 \pmod{26}$**
- To decipher:
  - $ap = c – b \pmod{26}$
  - $p = a^{-1}*(c - b) = 11^{-1}(c-6) =$ **$19(c-6) \bmod 26$**
  - if you bow at all bow low

BINGHAMTON UNIVERSITY
STATE UNIVERSITY OF NEW YORK

A5/1 Majority of three clocking bits



- At each cycle: $m = \mathrm{maj}(x_8, y_{10}, z_{10})$
  - Examples: $\mathrm{maj}(0,1,0) = 0$ and $\mathrm{maj}(1,1,0) = 1$
- For each register, if bit == maj, then step
- Then compute the keystream bit using $x_{18} \oplus y_{21} \oplus z_{22}$

BINGHAMTON UNIVERSITY
STATE UNIVERSITY OF NEW YORK

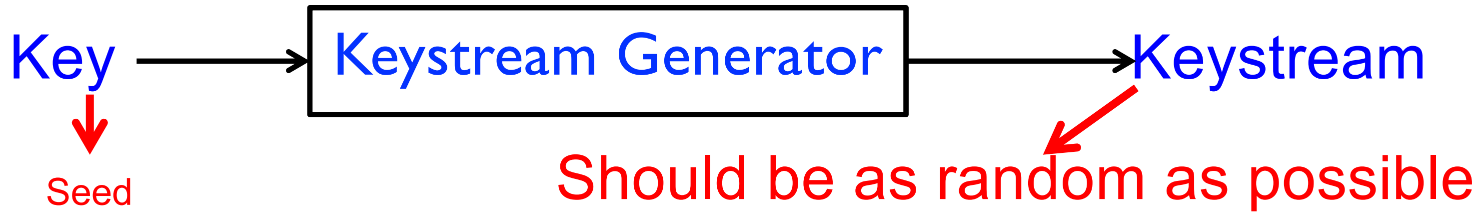
3. A5/1

- First round,
  - X = (x0, x1, ..., x18) = (10101010**1**0101010101)
  - Y = (y0, y1, ..., y21) = (1100110011**0**01100110011)
  - Z = (z0, z1, ..., z22) = (1110000111**1**000011110000)
  - maj = 1, x step, z step; last bits of x, y, z 0 1 0->key bit 1
- Next round,
  - X = (x0, x1, ..., x18) = (x1010101**0**1010101010)
  - Y = (y0, y1, ..., y21) = (1100110011**0**01100110011)
  - Z = (z0, z1, ..., z22) = (x111000011**1**100001111000)
  - maj = 0; x step, y step … last bits 1, 1, 0 -> key bit 0
- Repeat...
- Anwser: 10000

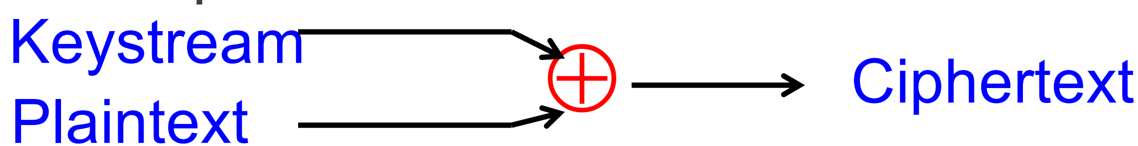
BINGHAMTON UNIVERSITY
STATE UNIVERSITY OF NEW YORK

Stream Cipher - Encryption

- A **keystream generator** takes a key K of n bits in length and stretches it into a long **keystream**

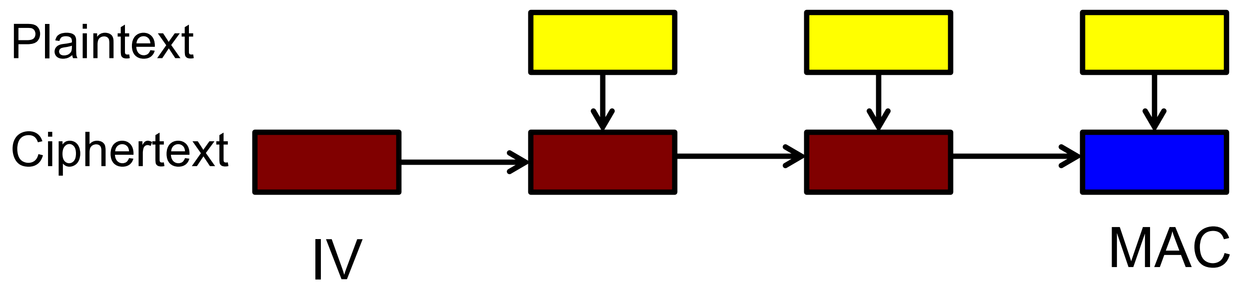


- **Encryption**: The keystream is XORed with the plaintext P to produce ciphertext C.



MAC for integrity

- Message Authentication Code (MAC)
  - Used for data **integrity**
  - Integrity **not** the same as confidentiality
- MAC is computed as **CBC residue**
- That is, compute CBC encryption, saving only final ciphertext block, the MAC



How does MAC work?

- Suppose Alice has 4 plaintext blocks
- Alice computes

  $\mathbf{C_0} = E(IV \oplus P_0, K)$, $\mathbf{C_1} = E(\mathbf{C_0} \oplus P_1, K)$,

  $\mathbf{C_2} = E(\mathbf{C_1} \oplus P_2, K)$, $\mathbf{C_3} = E(\mathbf{C_2} \oplus P_3, K) = \mathbf{MAC}$
- Alice sends IV, P_0, P_1, P_2, P_3 and **MAC** to Bob
- Suppose Trudy changes P_1 to X
- Bob computes

  $\mathbf{C_0} = E(IV \oplus P_0, K)$, $\boldsymbol{C_1} = E(\mathbf{C_0} \oplus X, K)$,

  $\boldsymbol{C_2} = E(\boldsymbol{C_1} \oplus P_2, K)$, $\boldsymbol{C_3} = E(\boldsymbol{C_2} \oplus P_3, K) = \boldsymbol{MAC} \neq \mathbf{MAC}$
- That is, error propagates into MAC

4. RC4

- Based on $ci = pi \oplus ki$
- $k_0 = c_0 \oplus p_0$

- Replace c_0 with $c_0' = p_0' \oplus k_0 = p_0' \oplus (c_0 \oplus p_0)$
- Trudy knows c_0 , $p_{0,}$ So she can forge this c_0'

- No. Any change in ciphertext can be propagated into the MAC.

BINGHAMTON UNIVERSITY
STATE UNIVERSITY OF NEW YORK

Block Cipher Notation

- P = plaintext block
- C = ciphertext block

- Encrypt P with key K to get ciphertext C
  - $C = E(P, K)$
- Decrypt C with key K to get plaintext P
  - $P = D(C, K)$

Triple DES or 3DES

- Today, 56 bit DES key is too small
  - Exhaustive key search is feasible
- But DES is everywhere, so what to do?
- **Triple DES** or **3DES** (112 bit key)
  - $C = E(D(E(P, K_1), K_2), K_1)$
  - $P = D(E(D(C, K_1), K_2), K_1)$

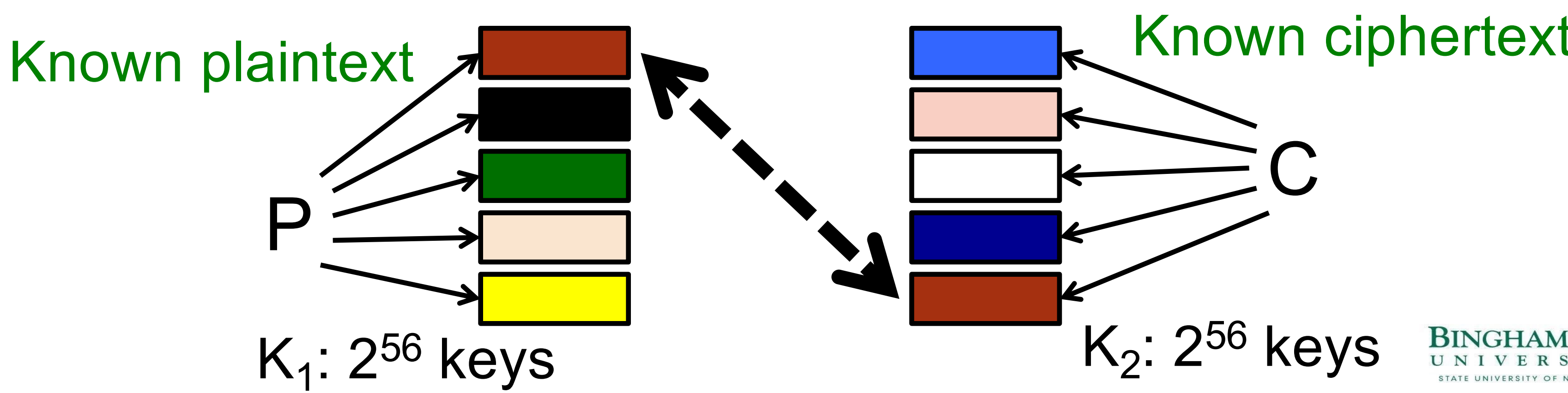
Only two keys!

More on 3DES

- Why Encrypt-Decrypt-Encrypt with 2 keys?
  - Backward compatible: $E(D(E(P, K), K), K) = E(P, K)$
  - And 112 bits is enough
- Why not $\mathbf{C = E(E(P, K_1), K_2)}$?
  - A (semi-practical) **known plaintext** attack

Meet-in-the-middle attack

- Pre-compute table of $E(P, K_1)$ for every possible key K_1 (resulting table has 2^{56} entries) used for search
- Then for each possible K_2 compute $D(C, K_2)$ until a match in table is found (*2^{56}*)
- When match is found, have $\mathbf{E(P, K_1) = D(C, K_2)}$
- Result gives us keys: $\mathbf{C = E(E(P, K_1), K_2)}$



5. Double DES

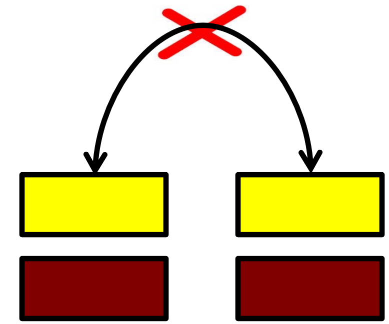
- $C = D(E(P, K_1), K_2)$

- Use K_2 to encrypt both sides, we get
- $E(C, K_2) = E(P, K_1)$
  - Try to find K_1 and K_2 make the above equation work

- Still suffer from meet in the middle attack

BINGHAMTON UNIVERSITY
STATE UNIVERSITY OF NEW YORK

ECB Mode



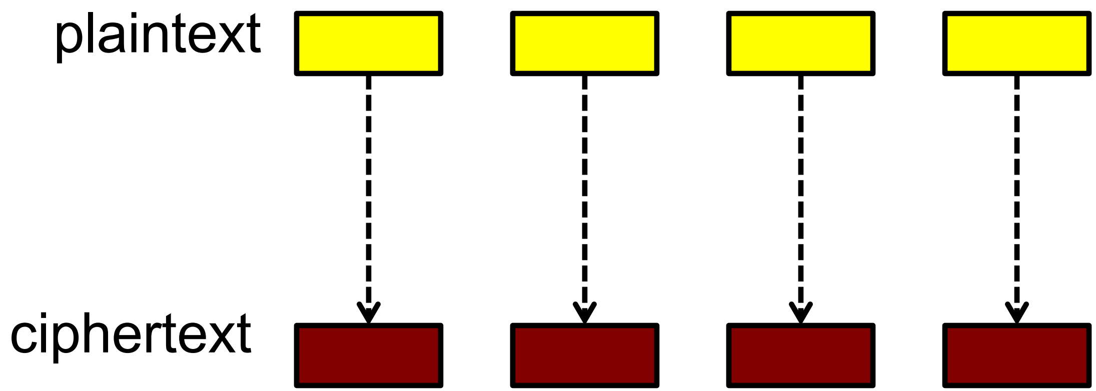
- Notation: $C = E(P, K)$
- Given plaintext $P_0, P_1, \ldots, P_m, \ldots$
- Most obvious way to use a block cipher:

Encrypt	**Decrypt**
$C_0 = E(P_0, K)$	$P_0 = D(C_0, K)$
$C_1 = E(P_1, K)$	$P_1 = D(C_1, K)$
$C_2 = E(P_2, K)$ …	$P_2 = D(C_2, K)$ …

- For fixed key K, this is "electronic" version of a codebook cipher (without additive)
  - With a different codebook for each key

ECB Mode

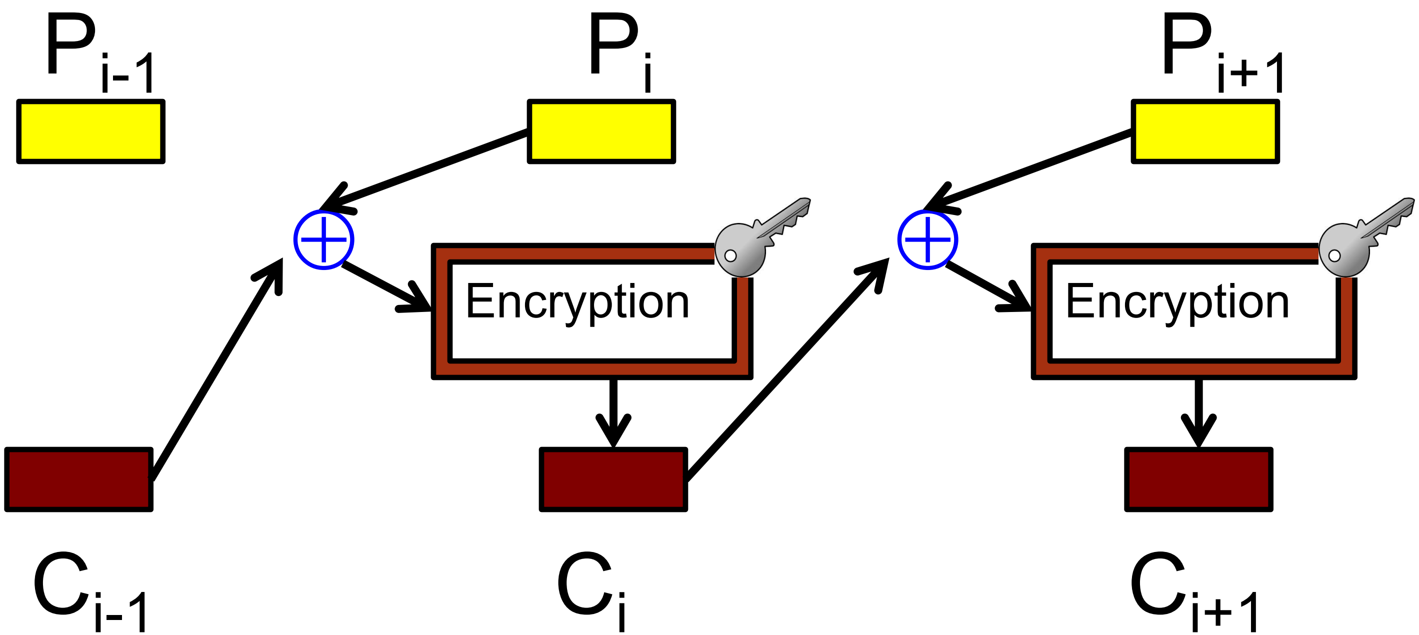
- Good: both encryption and decryption can be done in parallel



Question: Anything bad about it?

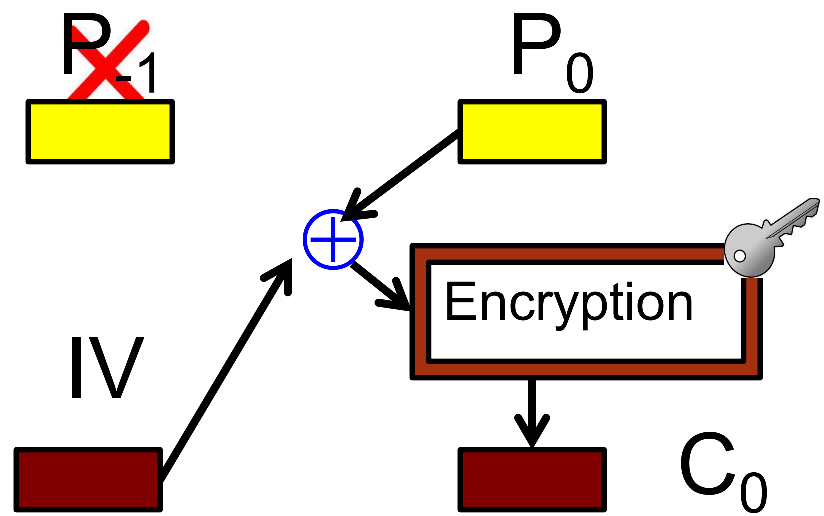
Cipher Block Chaining (CBC) Mode -- (Encryption)

- Blocks are "chained" together: $C_{i+1} = E(C_i \oplus P_{i+1}, K)$



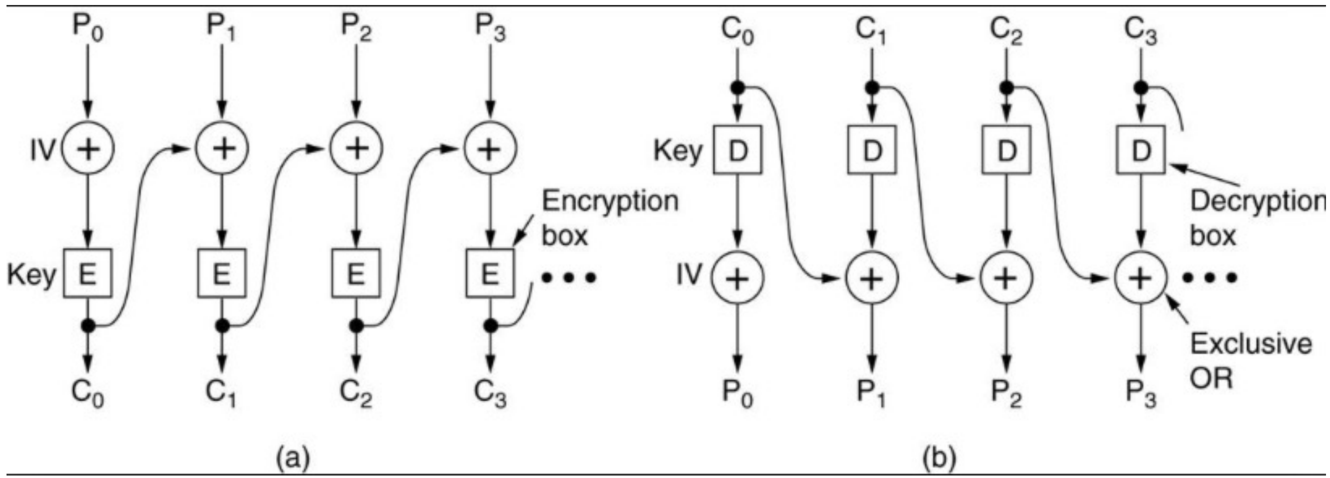
Initialization vector

Question: how does the receiver knows IV for decryption?

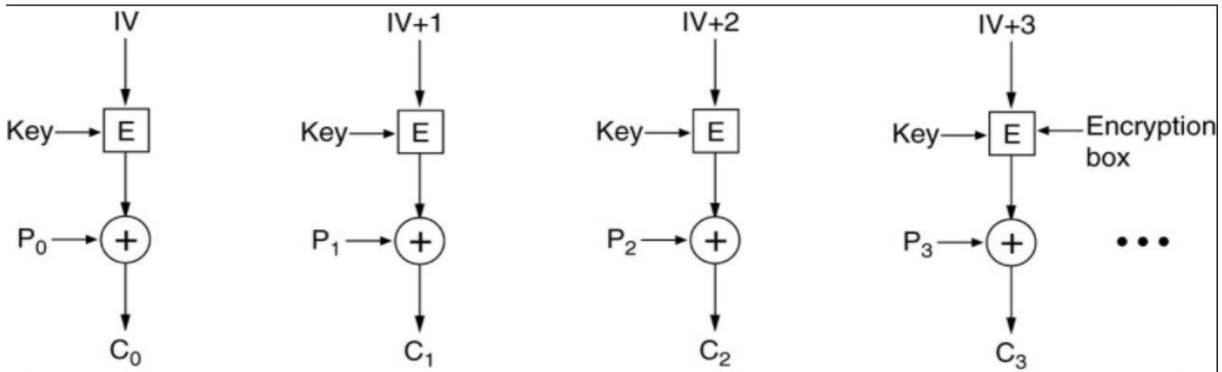


- A random initialization vector (IV) is used to initialize CBC
- IV is random, but not secret
- Analogous to classic codebook *with additive*

A better picture of CBC



Counter Mode (CTR)



Encryption

$C_0 = P_0 \oplus E(IV, K),$

$C_1 = P_1 \oplus E(IV{+}1, K),$

$C_2 = P_2 \oplus E(IV{+}2, K), \ldots$

Decryption

$P_0 = C_0 \oplus E(IV, K),$

$P_1 = C_1 \oplus E(IV{+}1, K),$

$P_2 = C_2 \oplus E(IV{+}2, K), \ldots$

6. Counter Mode

- $\mathbf{C_i = P_i \oplus E(IV + i, K)}$
- Instead we use
- $C_i = P_i \oplus E(K, \mathbf{IV + i})$
  - Use IV+i as key
  - IV is not secret
  - If Trudy can get a single block of known P, then she can get the K, then can decrypts all blocks

BINGHAMTON UNIVERSITY
STATE UNIVERSITY OF NEW YORK

RSA: Trapdoor key generation

- Let p and q be two large prime numbers
  - A **prime** number has no positive divisors other than 1 and itself
- Let $N = p \cdot q$ be the **modulus**
- Choose e **relatively prime** to $(p-1) \cdot (q-1)$
- Find d such that $\boldsymbol{d \cdot e = 1 \bmod (p-1) \cdot (q-1)}$
  - So d is the *multiplicative inverse* of e in the *ring* of integers modulo (p-1) ·(q-1). Recall d must exist!
- **Public key** is $\boldsymbol{(N, e)}$
- **Private key** is $\boldsymbol{d}$

RSA encryption and decryption

- Message M is treated as a *number* in [0, N)
- To encrypt M with public key we compute

  $$C = M^e \bmod N$$

- To decrypt ciphertext C with private key compute

  $$M = C^d \bmod N$$

Public key is ***(N, e)***, **private key** is ***d***

Public Key Notation

- **Sign** message M with Alice's **private key:** $[M]_{Alice}$
- **Encrypt** message M with Alice's **public key:** $\{M\}_{Alice}$
- Then

$$\{[M]_{Alice}\}_{Alice} = M$$

$$[\{M\}_{Alice}]_{Alice} = M$$

- Notations:
  - Square brackets: [] → Private key
  - Curly brackets: {} → Public key

7. RSA

- To encrypt: $C = M^e \bmod N$ = 19^3 = **28** mod 33.
- To decrypt: $M = C^d \bmod N$ = 28^7 = **19** mod 33

- To sign: $S = [M]_{Alice} = M^d \bmod N = 25^7 \bmod 33 = 31$.
- To verify: $M = \{[M]_{Alice}\}_{Alice} = S^e = 31^3 =$ **25** mod 33
  - Bob received message **25**, the signature is verified

BINGHAMTON UNIVERSITY
STATE UNIVERSITY OF NEW YORK

Overview of Diffie-Hellman

- Invented by Malcolm Williamson (GCHQ, British Equivalent of NSA) and, independently, by Diffie and Hellman (Stanford)
  - **Diffie and Hellman won ACM Turing award for this!**
- A **“key exchange” algorithm**
  - Used to establish a shared symmetric key
- ***Not* for encrypting or signing**

Based on the discrete logarithm algorithm

- Based on **discrete log** problem, which is believed to be difficult:
  - **Given:** $\mathbf{g}$, $\mathbf{p}$, and $\mathbf{g}^{k} \bmod \mathbf{p}$
  - **Find:** exponent $\mathbf{k}$
    - For example, in real numbers, log2(8)=3 because $\mathbf{2}^{3}$ =8
    - But for discrete log, finding the k is not feasible to do
- Example
  - Question 1: $g = 2$, $p = 17$, $(g^{k} \bmod p) = 13$. What is k?

Diffie-Hellman

- **Public:** $\mathbf{g}$ and $\mathbf{p}$
- **Private:** Alice's exponent $\mathbf{a}$, Bob's exponent $\mathbf{b}$

$$\text{Alice}, a \xrightarrow{\quad g^a \bmod p \quad} \text{Bob}, b$$

$$\text{Alice}, a \xleftarrow{\quad g^b \bmod p \quad} \text{Bob}, b$$

Alice, a

Bob, b

- Alice computes $(g^b)^a = g^{ba} = g^{ab} \bmod p$
- Bob computes $(g^a)^b = g^{ab} \bmod p$
- Use $K = g^{ab} \bmod p$ as symmetric key

8. Diffie-Hellman

- Bob got: g^a mod p
- Bob wants $(g^a)^b$ mod p = **X**
- But this require Bob to solve
  the **discrete log problem**, where the base is g^a mod p.
  - Solving discrete log problem is **difficult**

Knapsack Problem

- Given a set of n weights $W_0, W_1, \ldots, W_{n-1}$ and a sum S, is it possible to find $a_i \in \{0,1\}$ so that

  $$\mathbf{S = a_0 W_0 + a_1 W_1 + \ldots + a_{n-1} W_{n-1}}?$$

  Example

  - Weights $(62, 93, 26, 52, 166, 48, 91, 141)$
  - Problem: Find subset that sums to $S = 302$
  - Answer: $62 + 26 + 166 + 48 = 302$

Knapsack Problem

- **General knapsack** (GK) is hard to solve
- But **superincreasing knapsack** (SIK) is easy
- SIK: each weight is **greater than** the *sum of all previous weights*

$$W_0, W_1, \ldots, W_i, \ldots, W_{n-1}$$

Superincreasing: $W_0 + \ldots + W_{i-1} < W_i$

BINGHAMTON
UNIVERSITY
STATE UNIVERSITY OF NEW YORK

Knapsack Keys

- Start with (2, 3, 7, 14, 30, 57, 120, 251) as the SIK
- Choose $m = 41$ and $n = 491$
  - m, n *relatively prime*
  - n exceeds sum of elements in SIK
- Compute "general" knapsack: **modular multiplication**

  $2 \cdot 41 \bmod 491 = 82$

  $3 \cdot 41 \bmod 491 = 123$

  $7 \cdot 41 \bmod 491 = 287$

  $14 \cdot 41 \bmod 491 = 83$

  $30 \cdot 41 \bmod 491 = 248$

  $57 \cdot 41 \bmod 491 = 373$

  $120 \cdot 41 \bmod 491 = 10$

  $251 \cdot 41 \bmod 491 = 471$
- "General" knapsack: (82, 123, 287, 83, 248, 373, 10, 471)

BINGHAMTON UNIVERSITY
STATE UNIVERSITY OF NEW YORK

Knapsack Crypto Example

Encrypt with public key, decrypt with private key

Ciphertext: 548

Private key:

(2, 3, 7, 14, 30, 57, 120, 251)
Multiplier m = 41
Modulus n = 491

- To decrypt,
  - $548 \cdot 41^{-1} = 548 \cdot 12 = 193 \bmod 491$
  - Solve (easy) SIK with $S = \mathbf{193}$

$$(\mathbf{2}, 3, 7, \mathbf{14}, 30, \mathbf{57}, \mathbf{120}, 251)$$

$$\begin{array}{cccccccc} \uparrow & & & \uparrow & & \uparrow & \uparrow & \\ 1 & 0 & 0 & 1 & 0 & 1 & 1 & 0 \end{array}$$

9. Knapsack

- SIK is (3, 5, 10, 23), cipher 29
- $m^{-1}C = 6 * 29 = 174 = \mathbf{33} \bmod 47.$
- Using super-increasing knapsack in the private key, we find the plaintext is 0011
- Since $m^{-1} * m = 6m = 1 \bmod 47$, $\mathbf{m = 8}$
- Multiply each element in the super-increasing knapsack with m and reduce mod 47, we can get the general knapsack
  - (24, 40, 33, 43)

BINGHAMTON UNIVERSITY
STATE UNIVERSITY OF NEW YORK

Elliptic Curve Crypto (ECC)

- “Elliptic curve” is **not** a cryptosystem
- Elliptic curves are a different way to do the math in public key system
- Elliptic curve versions of DH, RSA, etc.
  - Compare to the exponential version
- Why would we want them if we already have DH and RSA?

Points on Elliptic Curve

- Discrete version: $y^2 = x^3 + ax + b \pmod{p}$
- Consider $y^2 = x^3 + 2x + 3 \pmod{5}$

$$x = 0 \Rightarrow y^2 = 3 \Rightarrow \text{no solution} \pmod{5}$$
$$x = 1 \Rightarrow y^2 = 6 = 1 \Rightarrow y = 1,4 \pmod{5}$$
$$x = 2 \Rightarrow y^2 = 15 = 0 \Rightarrow y = 0 \pmod{5}$$
$$x = 3 \Rightarrow y^2 = 36 = 1 \Rightarrow y = 1,4 \pmod{5}$$
$$x = 4 \Rightarrow y^2 = 75 = 0 \Rightarrow y = 0 \pmod{5}$$

- Then points on the elliptic curve are (1,1), (1,4), (2,0), (3,1), (3,4), (4,0), and the point at infinity: ∞

10. Elliptic Curve

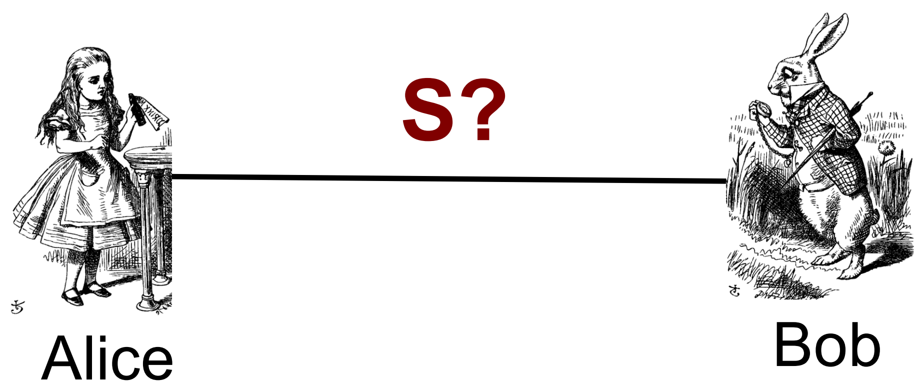
- $y^2 = x^3 + 11x + 19 (\mathrm{mod}\ 167)$

- Verify P (2, 7) on E
  - $7^2 = 2^3 + 11 * 2 + 19 \bmod 167$
  - 49 = 49
  - Points on the elliptic curve

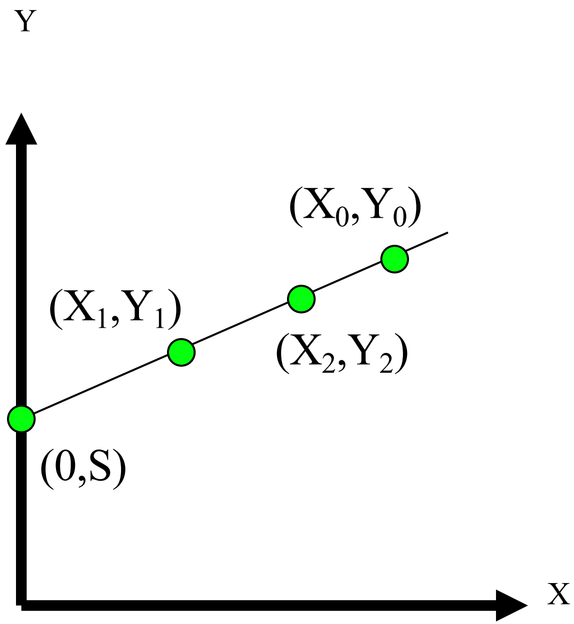
BINGHAMTON UNIVERSITY
STATE UNIVERSITY OF NEW YORK

What is secret sharing

- Goal: Alice and Bob want to share a secret S in the sense that:
  - Neither Alice nor Bob alone (nor anyone else) can determine S with a probability better than guessing
  - Alice and Bob together can easily determine S



Shamir's Secret Sharing



2 out of 3

- Give (X_0,Y_0) to Alice
- Give (X_1,Y_1) to Bob
- Give (X_2,Y_2) to Charlie
- Then any two can cooperate to find secret S
- But one can't find secret S
- A "2 out of 3" scheme

11. 2 out of 3

- We have $ax + by = 18$
- Use two values
  - $4a + 10/3 * b = 18$
  - $6a + 2 * b = 18$
- We can get $a=2$, $b=3$
- Then, $2x + 3y = 18$

- Make $x = 0$, get $S = 6$

BINGHAMTON UNIVERSITY
STATE UNIVERSITY OF NEW YORK