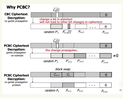
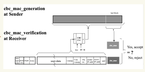
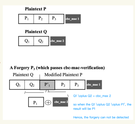
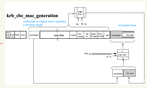
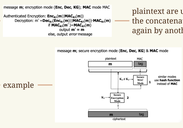
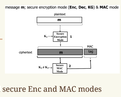
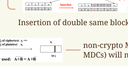
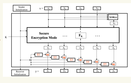
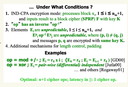
Encryption

- ECB
  - block by block encryption — Chosen Plaintext Attack — don't need to match the entire message, could guess part of the message
  - **should introduce probabilistic** — necessary for a secure encryption method but not sufficient
- Symmetric Encryption Modes
  - Chaining modes
    - block cipher
      - an encryption algorithm
      - if we input same input, we get same output
      - encryption functions are public
      - use a secret key to choose an encryption function to encrypt
      - $Y = F_K\{X\}$
    - CBC
      - XORed with previous encryption block (for the first block, XOR with an IV instead)
      - Encryption: $Y_i = F_K(Y_{n-1} \oplus X_i)$, where $Y_0 = IV$ — IV need to be random
      - derivative
    - CBC-Chain
      - random initial IV in message 1
      - initial IV's influence will propagate to the every following message encryption
    - CBCC - stateful CBC — Encryption: $Y_n = F_K(Y_{n-1} \oplus X_i)$, where $Y_0 = IV$ —
      - state initialization
    - CBC$ in kerberos v5
      - $C = F_K\{confounder||Cksum||m\}$
        - confounder is a random generated full block
        - cksum 校验和
      - IV is random for every message
    - PCBC
      - Encryption : $Y_n = F_K(Y_{n-1} \oplus X_i)$, where $Y_0 = IV$ — 
      - attack — swapping ciphertext blocks —
    - IGE
      - Encryption: $Y_i = F_K(Y_{i-1} \oplus X_i) \oplus X_{i-1}$, where $X_0 = IV_1, Y_0 = IV_2$
      - attack — swapping ciphertext blocks after specific CPA
  - counter modes
    - CTR
      - $Y_n = F_K\{ctr + idx\} \oplus X_n$, Ctr is random for each block — because the X_n does not participated in the encryption part, there is no need for padding in CTR
      - keep referring to the initial ctr as it keeps incrementing it in further encryption
    - CTR$
      - $Y_n = F_K\{ctr + idx\} \oplus X_n$, Ctr is from previous message
      - ctr is randomly selected for every block of any message
  - not secure, attack
    - modify one plaintext bit will leave other bits' encryption result unchanged
      - CPA, e.g. if x2 chosen to be y1, then the y2 will always be 0, if the noise of the next block encryption is fixed, adversary could exploit this
        - i need the PCBC
        - it could be applied to all CBC derivative modes that only use previous y as XOR noise
- Message Authentication Code (MAC) Modes
  - CBC MAC
    - for checking data integrity
    - 1
      - 
      - IV = 0, have padding in plaintext
      - secure for fixed-length messages
    - insecure for variable length — 
      - adversary could generate a M' (m1', m2', … , mn') whose XOR result equals cbc_mac, so it would not be discovered
    - 2 — 
  - or designing based on PCBC, IGE … (ppt say CTR is not very useful)
- Authenticated Encryption Modes
  - Encryption with Authenticity (and Integrity)
    - authenticity - determine origin of a message
    - integrity - Detect all message modifications (e.g., forgeries) with very high probability
  - requirements
    - for secrecy, authenticity, and integrity
    - need 2 pass over data (Encrypt + MAC), 1-2 crypto primitives
    - for efficiency, 1 pass over data, 1 crypto primitive is ok
    - protect against existential forgeries in CPA; or obtain ciphertext unforgeability
  - types
    - 2-pass
      - Encrypt & Authenticate (AuE)
        -
        - plaintext are used for encryption mode and MAC mode (the two modes using different key encryption), then concatenates together
        - mostly not secure for all secure Enc and MAC modes
          -
          - 1 bit changes in plaintext will not lead to other bits changed in ciphertext;
          - if the calculation of CBC-MAC equals XOR all the plaintext,
          - in this case, if we changes 1 bit in C', the CBC-MAC result is same as CBC-MAC(P2)
          - e.g.
      - Authenticate then Encrypt
        - example — 
        -
        - not secure for all secure Enc and MAC modes
        - secure for some
          - SSL
          - Kerberos v5 (hash)
      - Encrypt then Authenticate
        - example — 
        - secure for all secure Enc and MAC modes
    - 1-pass
      - designing approach
        - 1. Partition Message into Blocks - use padding if necessary;
        - 2. Compute Redundancy Block - use Manipulation Detection Code (MDC);
        - 3. Add redundancy block to message blocks;
        - 4. Encrypt message and redundancy block;
      - CBC-XOR (failed) — problems
        -
        - Truncation with CPA
        -
        - Swapping
        - 
        - Insertion of double same blocks
      - CTRC-XOR (failed) — — non-crypto Modification Detection Codes (MDCs) will not detect such attacks
      - XCBC-XOR
        - AuE type
        - 1 pass and 1 crypto primitive
        - 
        - 
    - weak or strong MAC definition —
- Parallel Encryption Mode — Randomizing Plaintext
  -
  -

Presented with xmind