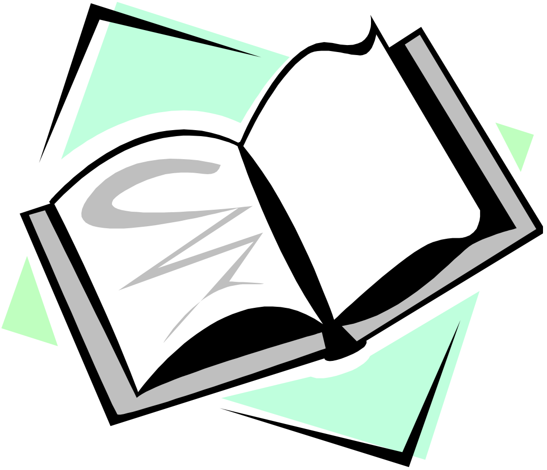
Computer Security
CS 526
Topic 4

Cryptography: Semantic Security, Block Ciphers and Encryption Modes

Readings for This Lecture

- Required reading from wikipedia
  - Block Cipher
  - Ciphertext Indistinguishability
  - Block cipher modes of operation



Notation for Symmetric-key Encryption

- A symmetric-key encryption scheme is comprised of three algorithms
  - **Gen** the key generation algorithm
    - The algorithm must be probabilistic/randomized
    - Output: a key *k*
  - **Enc** the encryption algorithm
    - Input: key *k*, plaintext *m*
    - Output: ciphertext c := $\textbf{Enc}_k(m)$
  - **Dec** the decryption algorithm
    - Input: key *k*, ciphertext *c*
    - Output: plaintext m := $\textbf{Dec}_k(m)$

Requirement: $\forall k \, \forall m \; [\, \textbf{Dec}_k(\textbf{Enc}_k(m)) = \mathrm{m} \,]$

Randomized vs. Deterministic Encryption

- Encryption can be randomized,
  - i.e., same message, same key, run encryption algorithm twice, obtains two different ciphertexts
  - E.g, $\mathbf{Enc}_k[m] = (r, \text{PRNG}[k||r] \oplus m)$, i.e., the ciphertext includes two parts, a randomly generated r, and a second part
  - Ciphertext space can be arbitrarily large
- Decryption is determinstic in the sense that
  - For the same ciphertext and same key, running decryption algorithm twice always result in the same plaintext
- Each key induces a one-to-many mapping from plaintext space to ciphertext space
  - Corollary: ciphertext space must be equal to or larger than plaintext space

Towards Computational Security

- Perfect secrecy is too difficult to achieve.
- The computational approach uses two relaxations:
  - Security is preserved only against **efficient** (computationally bounded) adversaries
    - Adversary can only run in feasible amount of time
  - Adversaries can potentially succeed with some **very small probability** (that we can ignore the case it actually happens)
- Two approaches to formalize computational security: concrete and asymptotic

The Concrete Approach

- Quantifies the security by explicitly bounding the maximum success probability of adversary running with certain time:
  - "A scheme is (t,ε)-secure if **every** adversary running for time at most t succeeds in breaking the scheme with probability at most ε"
  - Example: a strong encryption scheme with n-bit keys may be expected to be (t, $t/2^n$)-secure.
    - N=128, $t=2^{60}$, then $\varepsilon = 2^{-68}$. (# of seconds since big bang is 2^{58})
- Makes more sense with symmetric encryption schemes because they use fixed key lengths.

The Asymptotic Approach

- A cryptosystem has a security parameter
  - E.g., number of bits in the RSA algorithm (1024,2048,…)
- Typically, the key length depends on the security parameter
  - The bigger the security parameter, the longer the key, the more time it takes to use the cryptosystem, and the more difficult it is to break the scheme
- The crypto system must be efficient, i.e., runs in time polynomial in the security parameter
- "A scheme is secure if every Probabilistic Polynomial Time (PPT) algorithm succeeds in breaking the scheme with only negligible probability"
  - "negligible" roughly means exponentially small as security parameter increases

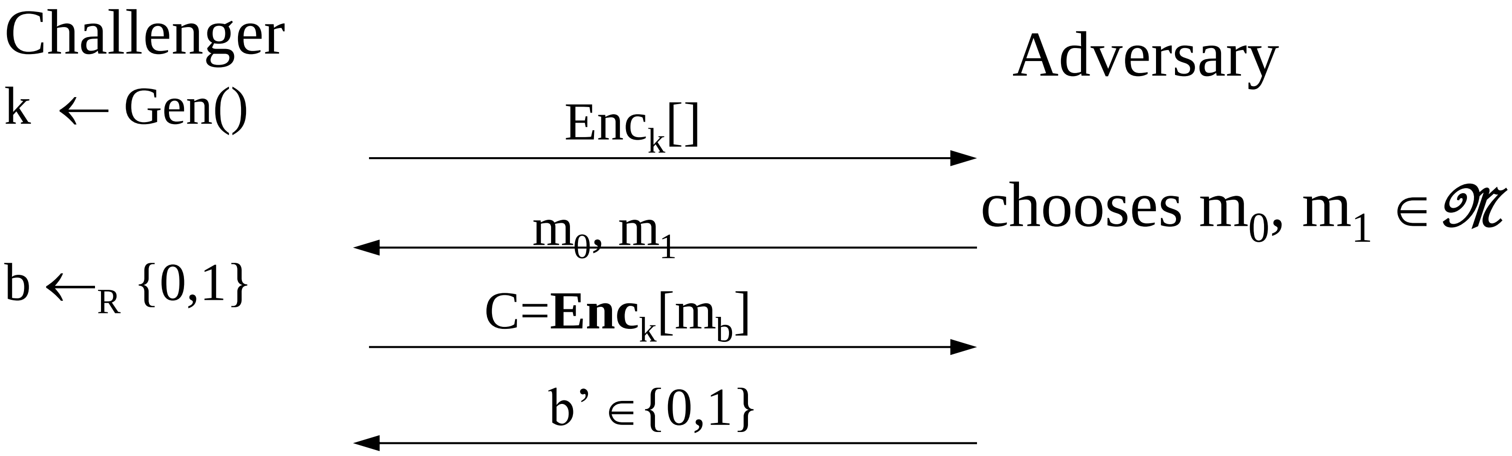
Defining Security

- Desire “semantic security”, i.e., having access to the ciphertext does not help adversary to compute any function of the plaintext.
  - Difficult to use

- Equivalent notion: Adversary cannot distinguish between the ciphertexts of two plaintexts

Towards IND-CPA Security:

- Ciphertext Indistinguishability under a Chosen-Plaintext Attack: Define the following IND-CPA experiment :
  - Involving an Adversary and a Challenger
  - Instantiated with an Adversary algorithm *A*, and an encryption scheme Π = (Gen, Enc, Dec)



Challenger

Adversary

$k \leftarrow \mathrm{Gen}()$

$\mathrm{Enc}_k[]$

chooses $m_0, m_1 \in \mathcal{M}$

m_0, m_1

$b \leftarrow_R \{0,1\}$

$C=\mathbf{Enc}_k[m_b]$

$b' \in \{0,1\}$

Adversary wins if b=b'

The IND-CPA Experiment Explained

- A k is generated by Gen(1^n)
- Adversary is given oracle access to $\mathrm{Enc}_k(\cdot)$, and outputs a pair of equal-length messages m_0 and m_1
  - Oracle access: one gets its question answered without knowing any additional information
- A random bit b is chosen, and adversary is given $\mathrm{Enc}_k(m_b)$
  - Called the challenge ciphertext
- Adversary still has oracle access to $\mathrm{Enc}_k(\cdot)$, and (after some time) outputs b’
- Adversary wins if b=b’

CPA-secure (aka IND-CPA security)

- A encryption scheme Π = (Gen, Enc, Dec) has indistinguishable encryption under a chosen-plaintext attack (i.e., is IND-CPA secure) iff. for all PPT adversary *A*, there exists a negligible function negl such that
  - Pr[***A*** *wins* in IND-CPA experiment] $\leq$ ½ + negl(n)

- No deterministic encryption scheme is CPA-secure. Why?

Another (Equivalent) Explanation of IND-CPA Security

- Ciphertext indistinguishability under chosen plaintext attack (IND-CPA)
  - Challenger chooses a random key K
  - Adversary chooses a number of messages and obtains their ciphertexts under key K
  - Adversary chooses two equal-length messages m_0 and m_1, sends them to a Challenger
  - Challenger generates $C=E_K[m_b]$, where b is a uniformly randomly chosen bit, and sends C to the adversary
  - Adversary outputs b' and wins if b=b'
  - Adversary advantage is | Pr[Adv wins] – ½ |
  - Adversary should not have a non-negligible advantage
    - E.g, Less than, e.g., $1/2^{80}$ when the adversary is limited to certain amount of computation;
    - decreases exponentially with the security parameter (typically length of the key)

Intuition of IND-CPA security

- Perfect secrecy means that any plaintext is encrypted to a given ciphertext with the same probability, i.e., given any pair of M_0 and M_1, the probabilities that they are encrypted into a ciphertext C are the same
  - Hence no adversary can tell whether C is ciphertext of M_0 or M_1.
- IND-CPA means
  - With bounded computational resources, the adversary cannot tell which of M_0 and M_1 is encrypted in C
- Stream ciphers can be used to achieve IND-CPA security when the underlying PRNG is cryptographically strong
  - (i.e., generating sequences that cannot be distinguished from random, even when related seeds are used)

Computational Security vs. Information Theoretic Security

- If only having computational security, then can be broken by a brute force attack, e.g., enumerating all possible keys
  - Weak algorithms can be broken with much less time
- How to prove computational security?
  - Assume that some problems are hard (requires a lot of computational resources to solve), then show that breaking security means solving the problem
- Computational security is foundation of modern cryptography.

Why Block Ciphers?

- One thread of defeating frequency analysis
  - Use different keys in different locations
  - Example: one-time pad, stream ciphers
- Another way to defeat frequency analysis
  - Make the unit of transformation larger, rather than encrypting letter by letter, encrypting block by block
  - Example: block cipher

Block Ciphers

- An n-bit plaintext is encrypted to an n-bit ciphertext
  - $\mathcal{P}$: $\{0,1\}^n$
  - $\mathcal{C}$: $\{0,1\}^n$
  - $\mathcal{K}$: $\{0,1\}^s$
  - **E**: $\mathcal{K} \times \mathcal{P} \rightarrow \mathcal{C}$: E_k: a permutation on $\{0,1\}^n$
  - **D**: $\mathcal{K} \times \mathcal{C} \rightarrow \mathcal{P}$: D_k is E_k^{-1}
  - Block size: n
  - Key size: s

Data Encryption Standard (DES)

- Designed by IBM, with modifications proposed by the National Security Agency
- US national standard from 1977 to 2001
- De facto standard
- Block size is 64 bits;
- Key size is 56 bits
- Has 16 rounds
- Designed mostly for hardware implementations
  - Software implementation is somewhat slow
- Considered insecure now
  - vulnerable to brute-force attacks

Attacking Block Ciphers

- Types of attacks to consider
  - known plaintext: given several pairs of plaintexts and ciphertexts, recover the key (or decrypt another block encrypted under the same key)
  - how would chosen plaintext and chosen ciphertext be defined?
- Standard attacks
  - exhaustive key search
  - dictionary attack
  - differential cryptanalysis, linear cryptanalysis
- Side channel attacks.

DES's main vulnerability is short key size.

Chosen-Plaintext Dictionary Attacks Against Block Ciphers

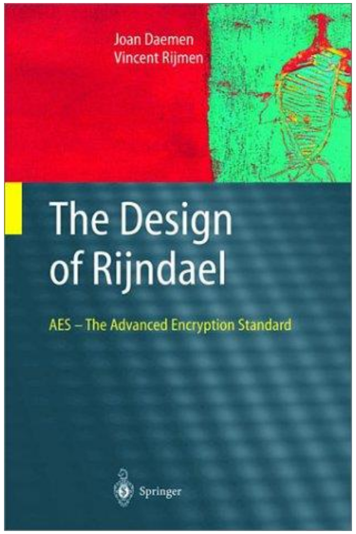
- Construct a table with the following entries
  - (K, $E_K[0]$) for all possible key K
  - Sort based on the second field (ciphertext)
  - How much time does this take?
- To attack a new key K (under chosen message attacks)
  - Choose 0, obtain the ciphertext C, looks up in the table, and finds the corresponding key
  - How much time does this step take?
- Trade off space for time

Advanced Encryption Standard

- In 1997, NIST made a formal call for algorithms stipulating that the AES would specify an unclassified, publicly disclosed encryption algorithm, available royalty-free, worldwide.
- Goal: replace DES for both government and private-sector encryption.
- The algorithm must implement symmetric key cryptography as a block cipher and (at a minimum) support block sizes of 128-bits and key sizes of 128-, 192-, and 256-bits.
- In 1998, NIST selected 15 AES candidate algorithms.
- On October 2, 2000, NIST selected **Rijndael** (invented by Joan Daemen and Vincent Rijmen) to as the AES.

AES Features

- Designed to be efficient in both hardware and software across a variety of platforms.
- Block size: 128 bits
- Variable key size: **128, 192, or 256 bits.**
- No known weaknesses


Joan Daemen
Vincent Rijmen
The Design of Rijndael
AES – The Advanced Encryption Standard
Springer

Need for Encryption Modes

- A block cipher encrypts only one block
- Needs a way to extend it to encrypt an arbitrarily long message
- Want to ensure that if the block cipher is secure, then the encryption is secure
- Aims at providing Semantic Security (**IND-CPA)** assuming that the underlying block ciphers are strong

Block Cipher Encryption Modes: ECB

- Message is broken into independent blocks;
- Electronic Code Book (ECB): each block encrypted separately.
- **Encryption: $c_i = E_k(x_i)$**
- **Decrytion: $x_i = D_k(c_i)$**

Properties of ECB

- Deterministic:
  - – the same data block gets encrypted the same way,
    - reveals patterns of data when a data block repeats
  - – when the same key is used, the same message is encrypted the same way
- Usage: not recommended to encrypt more than one block of data

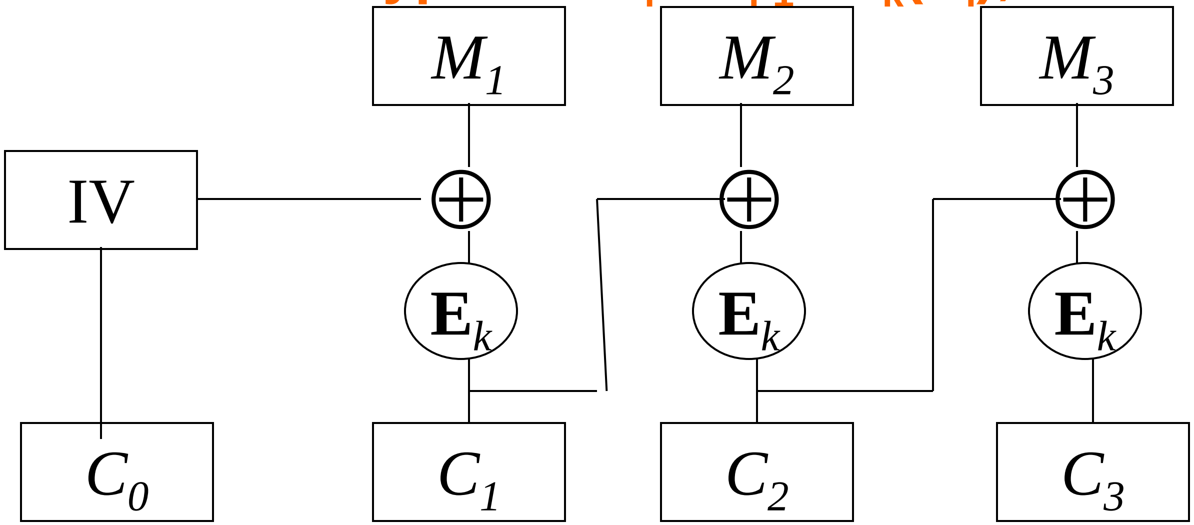
- How to break the semantic security (IND-CPA) of a block cipher with ECB?

DES Encryption Modes: CBC

- Cipher Block Chaining (CBC):
  - Uses a random Initial Vector (IV)
  - Next input depends upon previous output

Encryption: $C_i = \mathbf{E}_k(M_i \oplus C_{i-1})$, **with** $C_0 = \mathrm{IV}$

Decryption: $M_i = C_{i-1} \oplus \mathbf{D}_k(C_i)$, **with** $C_0 = \mathrm{IV}$

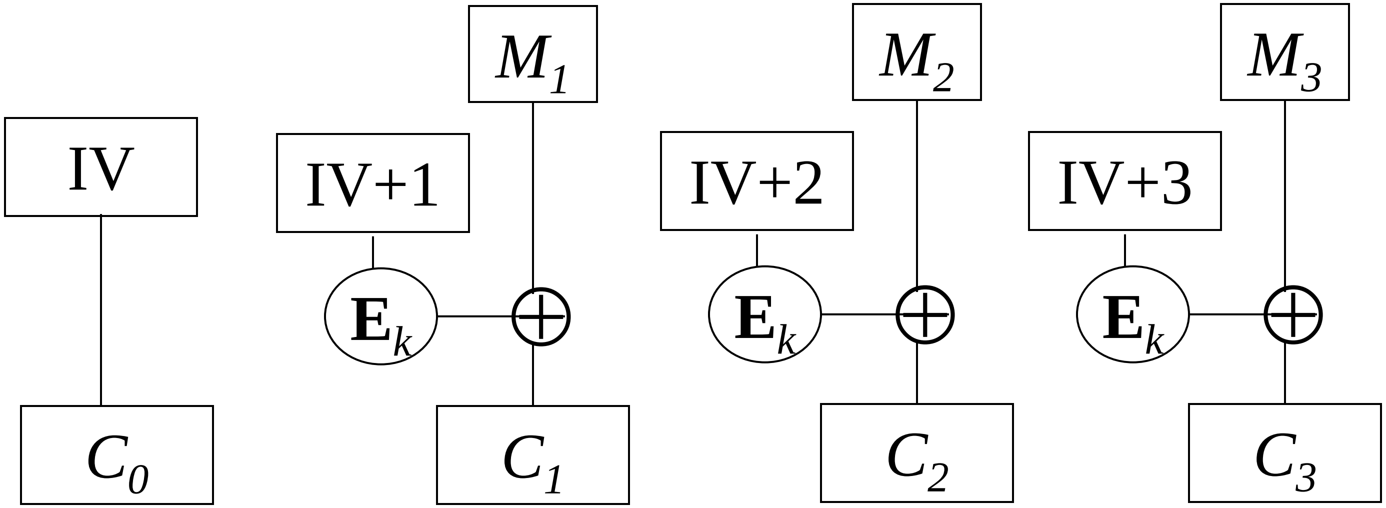


Properties of CBC

- Randomized encryption: repeated text gets mapped to different encrypted data.
  - can be proven to provide IND-CPA assuming that the block cipher is secure (i.e., it is a Pseudo Random Permutation (PRP)) and that IV's are randomly chosen and the IV space is large enough (at least 64 bits)
- Each ciphertext block depends on all preceding plaintext blocks.
- Usage: chooses **random** IV and protects the **integrity** of IV
  - The IV is not secret (it is part of ciphertext)
  - The adversary cannot control the IV

Encryption Modes: CTR

- Counter Mode (CTR): Defines a stream cipher using a block cipher
  - Uses a random IV, known as the counter
  - Encryption: C_0=IV, $C_i = M_i \oplus E_k[IV+i]$
  - Decryption: IV=C_0, $M_i = C_i \oplus E_k[IV+i]$



Properties of CTR

- Gives a stream cipher from a block cipher

- Randomized encryption:
  - when starting counter is chosen randomly

- Random Access: encryption and decryption of a block can be done in random order, very useful for hard-disk encryption.
  - E.g., when one block changes, re-encryption only needs to encrypt that block. In CBC, all later blocks also need to change

Coming Attractions …

- Cryptography: Cryptographic Hash Functions and Message Authentication