CSE 410 Fall 2025
Privacy-Enhancing Technologies

Marina Blanton

Department of Computer Science and Engineering
University at Buffalo

Lecture 3: Protecting Data at Rest

Data Protection

This is a part of a bigger plan

- Protecting data at rest
- Protecting data in transmission
- Protecting data during computation

Data Protection

The strongest way to protect data is by cryptographic means

- if you rely on access control, data is exposed upon compromise
- if you additionally rely on cryptographic tools, system compromise does not mean data compromise
- cryptographic tools are strong and cannot be broken if properly used
- key compromise may be possible, but will require additional vulnerabilities

Data Protection

When protecting data, you typically want both:

- data confidentiality – guarantees that data is remains private and unavailable to unauthorized parties
- data integrity – guarantees that data is authentic and has not be tampered with by unauthorized parties

This is typically achieved by means of symmetric cryptography

- symmetric encryption for confidentiality
- message authentication codes for integrity
- security objectives differ $\Rightarrow$ tools to realize them also differ

Symmetric Encryption

Symmetric (or secret-key) encryption means that the same key is used both for encryption and decryption

The key must be available at both times (and stored securely in between)

Such algorithms are:

- normally very fast
- can be used as primitives in more complex cryptographic protocols
- the key often has a short lifetime

Computational Security

Most of the cryptograhic techniques we'll discuss are computationally secure

- this means they are breakable in principle
- breaking them is very difficult for a computationally limited adversary
- algorithms are parameterized by a security parameter
- legitimate work has to be polynomial time in the security parameter
- breaking has to require super-polynomial effort
- alternatively, a polynomial-time adversary cannot succeed with reasonable probability

Symmetric Encryption Formally

More formally, a computationally secure symmetric key encryption scheme is defined as:

- a symmetric encryption scheme consists of polynomial-time algorithms (Gen, Enc, Dec) such that
  - Gen: on input the security parameter n, outputs key k
  - Enc: on input a key k and a message $m \in \{0,1\}^*$, outputs ciphertext c
  - Dec: on input a key k and ciphertext c, outputs plaintext m
- we write $k \leftarrow \mathsf{Gen}(1^n)$, $c \leftarrow \mathsf{Enc}_k(m)$, and $m := \mathsf{Dec}_k(c)$
  - this notation means that Gen and Enc are probabilistic and Dec is deterministic

Attacks Against Symmetric Encryption

- Encryption and decryption algorithms are assumed to be known to the adversary
- Types of attacks
  - ciphertext only attack: adversary knows a number of ciphertexts
  - known plaintext attack: adversary knows some pairs of ciphertexts and corresponding plaintexts
  - chosen plaintext attack: adversary knows ciphertexts for messages of its choice
  - chosen ciphertext attack: adversary knows plaintexts for ciphertexts of its choice
- We want a general-purpose algorithm to sustain all types of attacks

Security Against Chosen-Plaintext Attacks

- In chosen-plaintext attack (CPA), adversary $\mathcal{A}$ is allowed to ask for encryptions of messages of its choice
  - $\mathcal{A}$ is given black-box access to encryption oracle and can query it on different messages
- $\mathcal{A}$ is asked to distinguish between encryptions of messages of its choice
  - $\mathcal{A}$ chooses two messages m_0 and m_1 and one of them is encrypted
  - $\mathcal{A}$'s task is to determine which message was encrypted
  - $\mathcal{A}$ has only negligible chances of success with CPA-secure encryption

Symmetric Encryption

The above definition allows us to encrypt messages of any length

In practice, there are two types of symmetric key algorithms:

- block ciphers
  - the key has a fixed size
  - prior to encryption, the message is partitioned into blocks
  - each block is encrypted and decrypted separately
- stream ciphers
  - the message is processed as a stream
  - pseudo-random generator is used to produce a long key stream from a short key

Block Ciphers

- The algorithm maps an n-bit plaintext block to an n-bit ciphertext block
- Most modern block ciphers are product ciphers
  - we sequentially apply more than one operation to the message
- Often a sequence of permutations and substitutions is used
- A common design for an algorithm is to proceed in iterations
  - one iteration is called a round
  - each round consists of similar operations
  - ith round key k_i is derived from the secret key k using a fixed, public algorithm

Design Principles of Block Ciphers

Confusion-diffusion paradigm

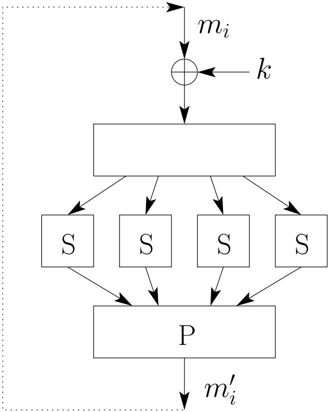
- split a block into small chunks
- define a permutation on each chunk separately (confusion)
- mix outputs from different chunks by rearranging bits (diffusion)
- repeat to strengthen the result

Design Principles of Block Ciphers

Substitution-permutation networks

- since a permutation on a block can be specified as a lookup table, this is called substitution
- instead of having substitutions defined by the key, such functions are fixed and applied to messages and keys
- mixing algorithm is called mixing permutation

Design Principles of Block Ciphers



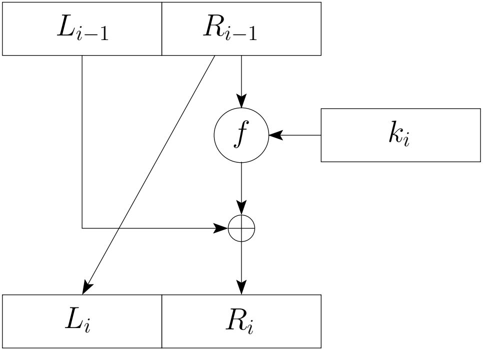
- for this type of algorithm to be reversible, each operation needs to be invertible

Design Principles of Block Ciphers

Another way to realize confusion-diffusion paradigm is through Feistel network

- in Feistel network each state is divided into halves of the same length: L_i and R_i
- in one round:
  - $L_i = R_{i-1}$
  - $R_i = L_{i-1} \oplus f(k_i, R_{i-1})$
- the main advantage is that operations don't need to be reversible (the inverse of the algorithm is not used)
- this expands possible design options

Design Principles of Block Ciphers



- reverse one round's computation as $R_{i-1} = L_i$ and $L_{i-1} = R_i \oplus f(k_i, R_{i-1})$

Design Principles of Block Ciphers

Substitution and permutation algorithms must be carefully designed

- choosing random substitution/permutation strategies leads to significantly weaker ciphers
- each bit difference in S-box input creates at least 2-bit difference in its output
- mixing permutation ensures that difference in one S-box propagates to at least 2 S-boxes in next round

Block Ciphers

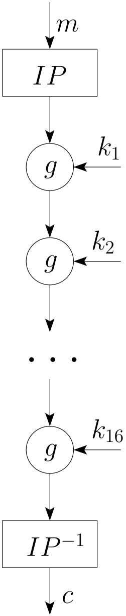
- Larger key size means greater security
  - for n-bit keys, brute force search takes $2^n/2$ time on average
- More rounds often provide better protection
  - the number of rounds must be large enough for proper mixing
- Larger block size offers increased security
  - security of a cipher also depends on the block length

Data Encryption Standard (DES)

- In 1973 National Institute of Standards and Technology (NIST) published a solicitation for cryptosystems
- DES was developed by IBM and adopted as a standard in 1977
- It was expected to be used as a standard for 10–15 years
- Was replaced only in 2001 with AES (Advanced Encryption Standard)
- DES characteristics:
  - key size is 56 bits
  - block size is 64 bits
  - number of rounds is 16

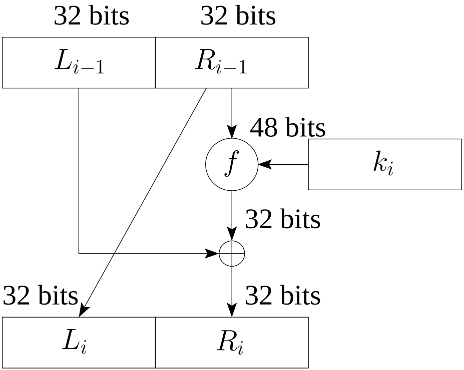
DES

- DES has a fixed initial permutation IP prior to 16 rounds of encryption
- The inverse permutation IP^{-1} is applied at the end



DES

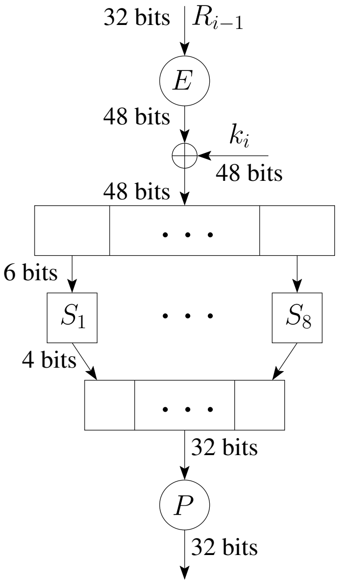
- DES uses the Feistel network



DES f function

The f function $f(k_i, R_{i-1})$

- expand R_{i-1}
- XOR expanded R_{i-1} with k_i
- appliy S-boxes substitution
- permutes the result



DES S-boxes

- There are 8 S-boxes
  - S-boxes are the only non-linear elements in DES design
  - they are crucial for the security of the cipher
- Example: S_1

14	4	13	1	2	15	11	8	3	10	6	12	5	9	0	7
0	15	7	4	14	2	13	1	10	6	12	11	9	5	3	8
4	1	14	8	13	6	2	11	15	12	9	7	3	10	5	0
15	12	8	2	4	9	1	7	5	11	3	14	10	0	6	13

  - input to each S-box is 6 bits $b_1 b_2 b_3 b_4 b_5 b_6$
  - row $= b_1 b_6$, column $= b_2 b_3 b_4 b_5$
  - output is 4 bits

DES S-boxes

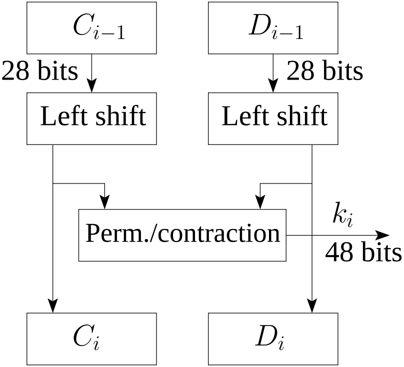
More about S-boxes..

- a modified version of IBM's proposal was accepted as the standard
- some of the design choices of S-boxes weren't public, which triggered criticism
- in late 1980s – early 1990s differential cryptanalysis techniques were discovered
- it was then revealed that DES S-boxes were designed to prevent such attacks
- such cryptanalysis techniques were known almost 20 years before they were discovered by others

DES Key Schedule

Key computation consists of:

- circular shift
- permutation
- contraction



Attacks on DES

- Brute force attack: try all possible 2^{56} keys
  - time-consuming, but no storage requirements
- Differential cryptanalysis: traces the difference of two messages through each round of the algorithm
  - was discovered in early 90s
  - not effective against DES
- Linear cryptanalysis: tries to find linear approximations to describe DES transformations
  - was discovered in 1993
  - has no practical implication

Brute Force Search Attacks on DES

- It was conjectured in 1970s that a cracker machine could be built for \$20 million
- In 1990s several DES challenges were solved
  - Challenge II-2 was solved in 1998 by EFF using a DES cracker machine in 56 hours
    - a DES Cracker machine was built for less than \$250,000
  - Challenge III was solved in 1999 by the DES Cracker and a network of 100,000 computers in 22 hours
    - http://www.distributed.net/des
- During the transition, triple DES was commonly used

Summary of Attacks on DES

- DES
  - best attack: brute force search
  - 2^{55} work on average
  - no other requirements
- Double DES
  - best attack: meet-in-the-middle
  - requires 2 plaintext-ciphertext pairs
  - requires 2^{56} space and about 2^{56} work
- Triple DES
  - key space is 112 or 168 bits
  - best practical attack: brute force search