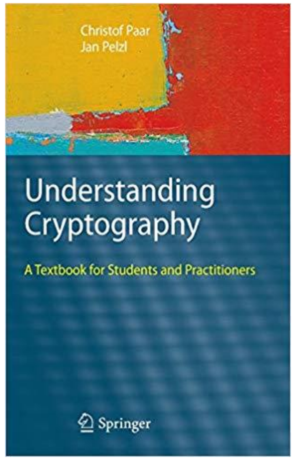
COM/BLM 426
Information Security

Asst. Prof. Dr. Bulent TUGRUL

btugrul@eng.ankara.edu.tr

Slides are mainly based on:

Understanding Cryptography: A Textbook for Students and Practitioners



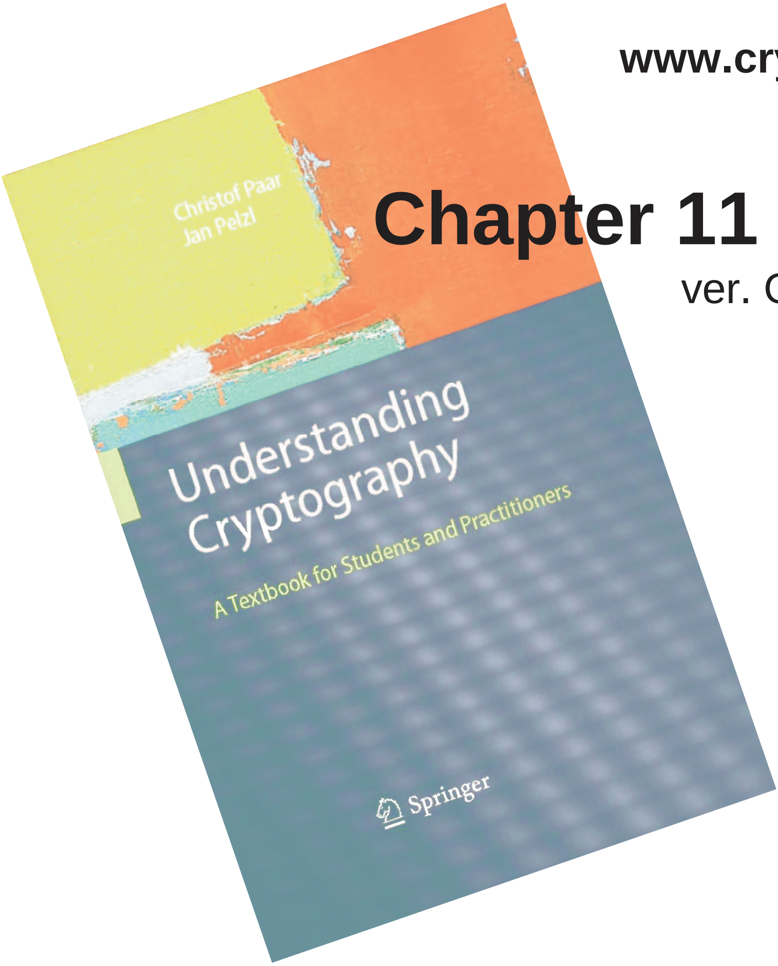
by Christof Paar and Jan Pelzl

Springer, 1st Edition, 2010

Understanding Cryptography – A Textbook for Students and Practitioners

by Christof Paar and Jan Pelzl

www.crypto-textbook.com



Chapter 11 – Hash Functions

ver. October 29, 2009

These slides were prepared by Stefan Heyse and Christof Paar and Jan Pelzl

Some legal stuff (sorry): Terms of Use

Content of this Chapter

- Why we need hash functions
- How does it work
- Security properties
- Algorithms
- Example: The Secure Hash Algorithm SHA-1

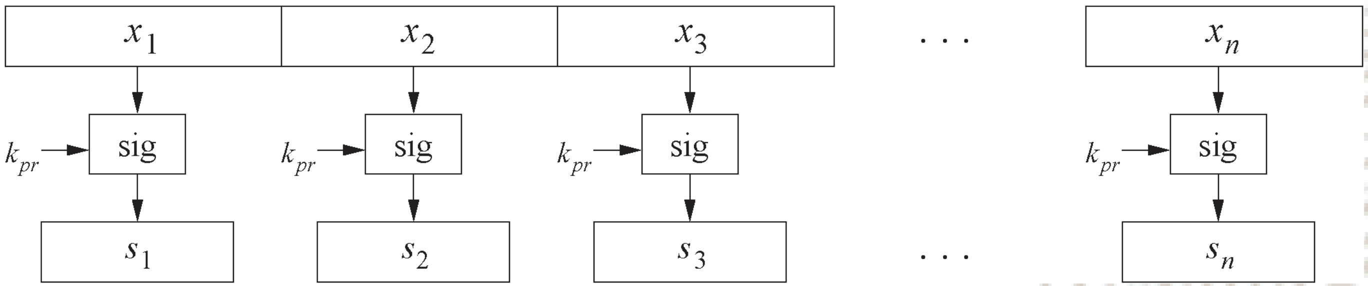
Content of this Chapter

- **Why we need hash functions**
- How does it work
- Security properties
- Algorithms
- Example: The Secure Hash Algorithm SHA-1

Motivation

Problem:

Naive signing of long messages generates a signature of same length.



- Three Problems
- Computational overhead
- Message overhead
- Security limitations
- For more info see Section 11.1 in "*Understanding Cryptography*".

Solution:

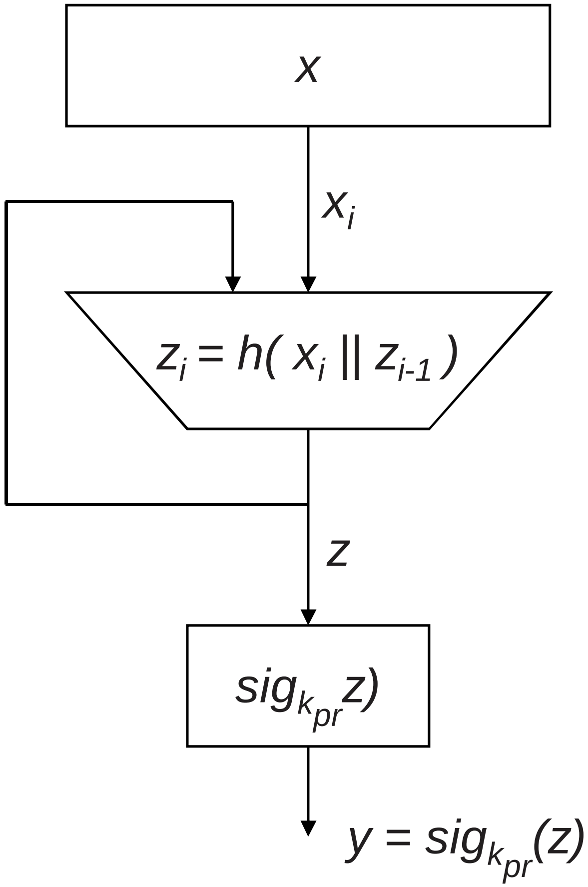
Instead of signing the whole message, sign only a digest (=hash)

Also secure, but much faster

Needed:

Hash Functions

Digital Signature with a Hash Function



Notes:

- x has fixed length
- z, y have fixed length
- z, x do not have equal length in general
- $h(x)$ does not require a key.
- $h(x)$ is public.

- **Basic Protocol for Digital Signatures with a Hash Function:**

Alice — Bob

Bob → Alice: K_{pub}

Bob:

$z = h(x)$

$s = sig_{K_{pr}}(z)$

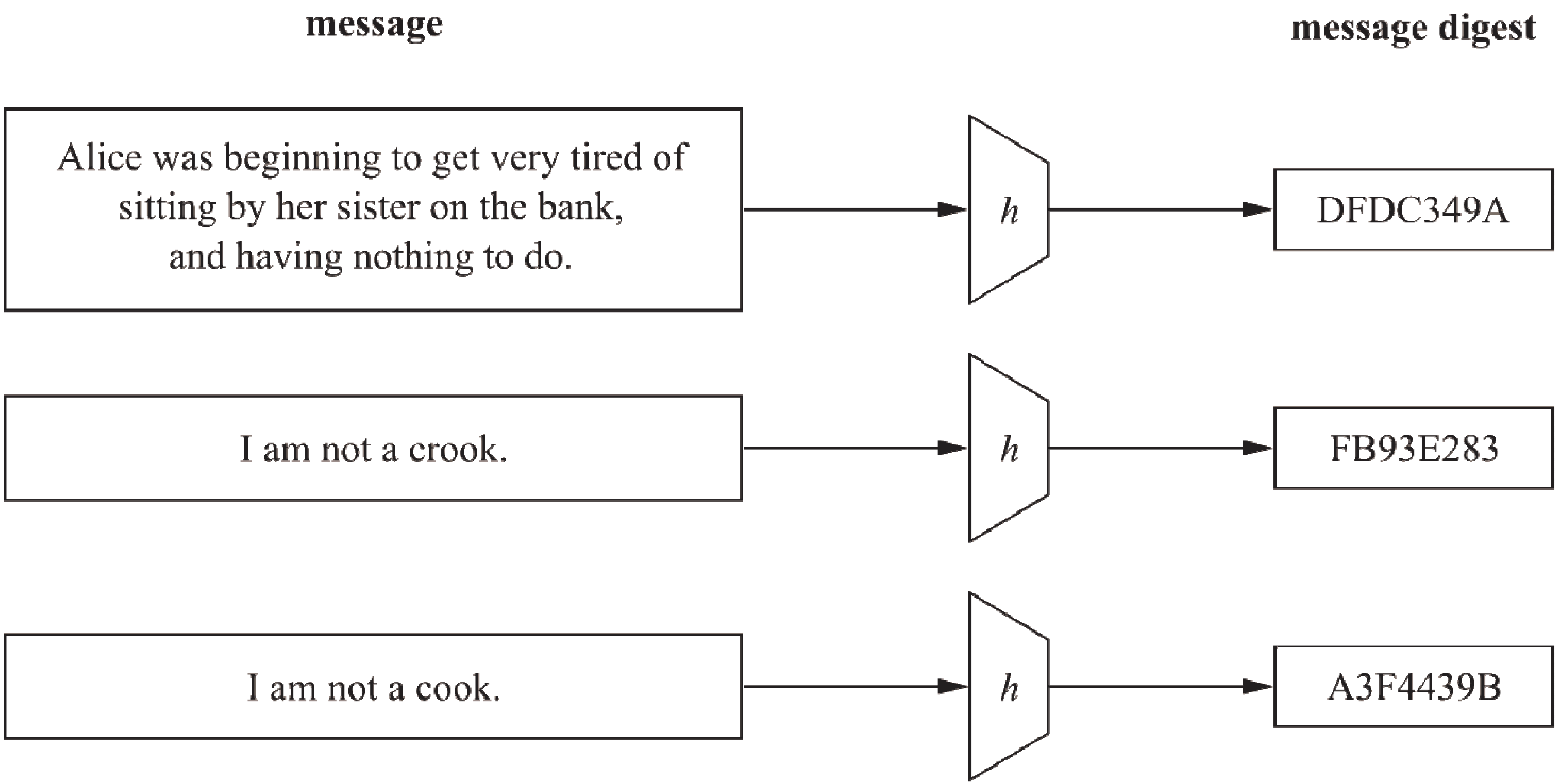
Bob → Alice: (x, s)

Alice:

$z' = h(x)$

$ver_{K_{pub}}(s,z')=true/false$

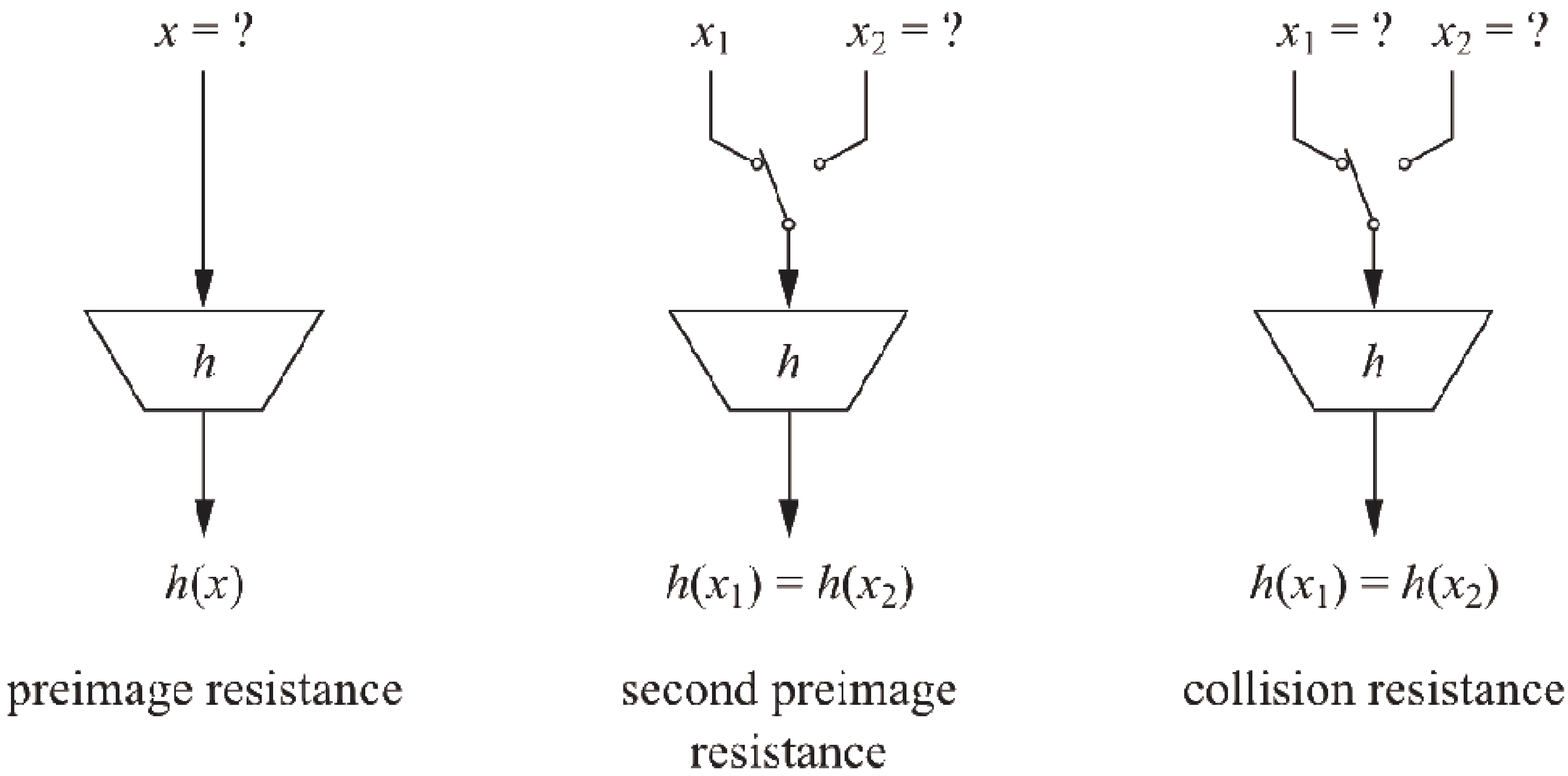
Principal input–output behavior of hash functions



Content of this Chapter

- Why we need hash functions
- How does it work
- **Security properties**
- Algorithms
- Example: The Secure Hash Algorithm SHA-1

The three security properties of hash functions



Hash Funktionen: Security Properties

- **Preimage resistance:** For a given output z, it is impossible to find any input x such that h(x) = z, i.e., h(x) is one-way.
- **Second preimage resistance:** Given x_1, and thus $h(x_1)$, it is computationally infeasible to find any x_2 such that $h(x_1) = h(x_2)$.
- **Collision resistance:** It is computationally infeasible to find any pairs $x_1 \neq x_2$ such that $h(x_1) = h(x_2)$.

Hash Funktionen: Security

It turns out that collison resistance causes most problems

- How hard is it to find a collision with a probability of 0.5 ?
- Related Problem: How many people are needed such that two of them have the same birthday with a probability of 0.5 ?
- No! Not 365/2=183.   23 are enough ! This is called the birthday paradoxon (Search takes $\approx\sqrt{2^n}$ steps) .
- For more info see Chapter 11.2.3 in *Understanding Cryptography*.
- To deal with this paradox, hash functions need a output size of at least 160 bits.

Content of this Chapter

- Why we need hash functions
- How does it work
- Security properties
- **Algorithms**
- Example: The Secure Hash Algorithm SHA-1

Hash Funktionen: Algorithms

Hash Algorithms

Special Algorithms,
e.g. MD5 - family

based on
block ciphers

- **MD5 -** family
- **SHA-1**: output - 160 Bit; input - 512 bit chunks of message *x;*
  operations - bitwise AND, OR, XOR, complement und cyclic shifts.
- **RIPE-MD 160:** output - 160 Bit; input - 512 bit chunks of message *x*; operations – like in SHA-1, but two in parallel and combinations of them after each round.

Content of this Chapter

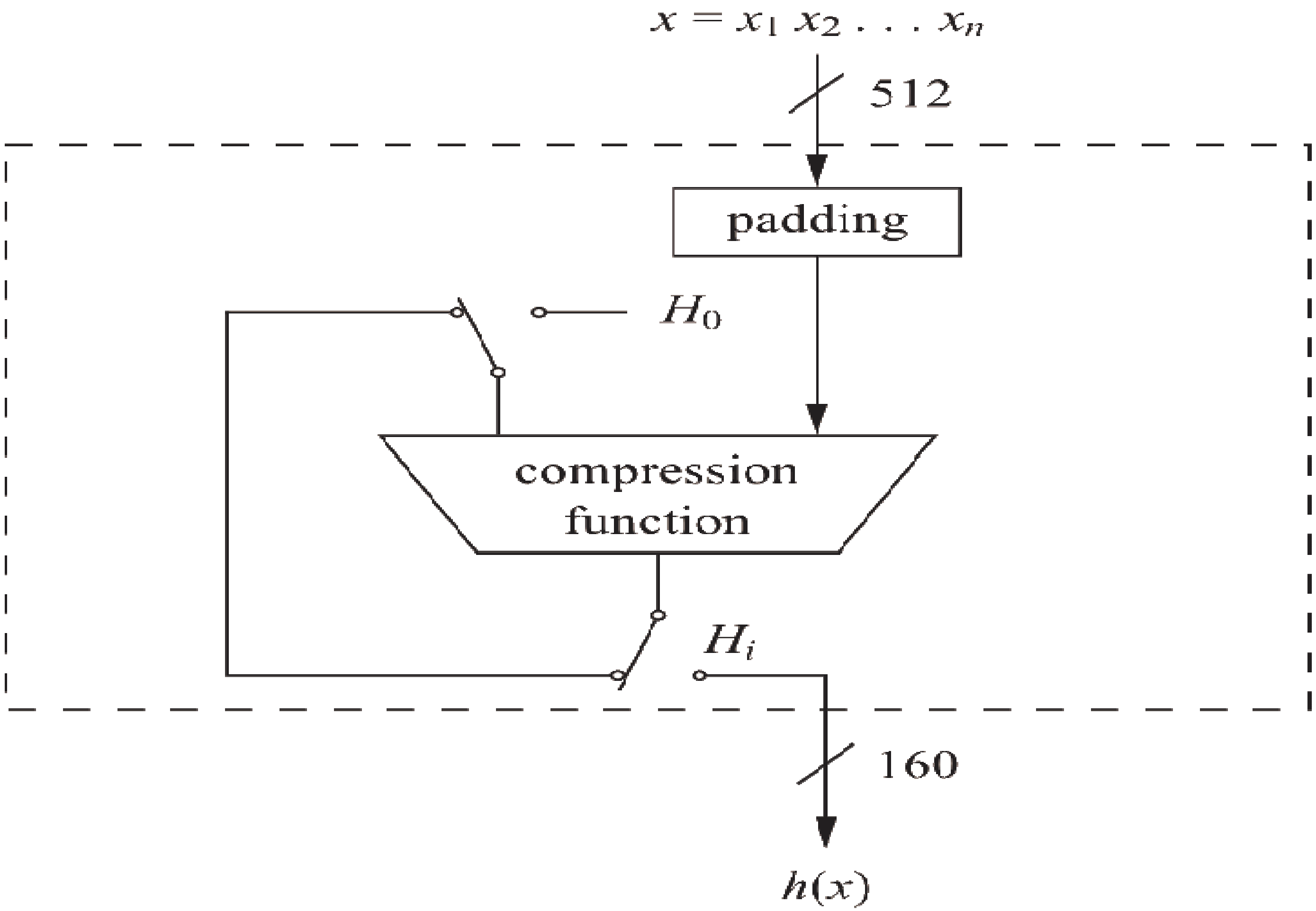
- Why we need hash functions
- How does it work
- Security properties
- Algorithms
- **Example: The Secure Hash Algorithm SHA-1**

SHA-1

- Part of the MD-4 family.
- Based on a Merkle-Dåmgard construction.
- 160-bit output from a message of maximum length 2^{64} bit.
- Widely used (even tough some weaknesses are known)

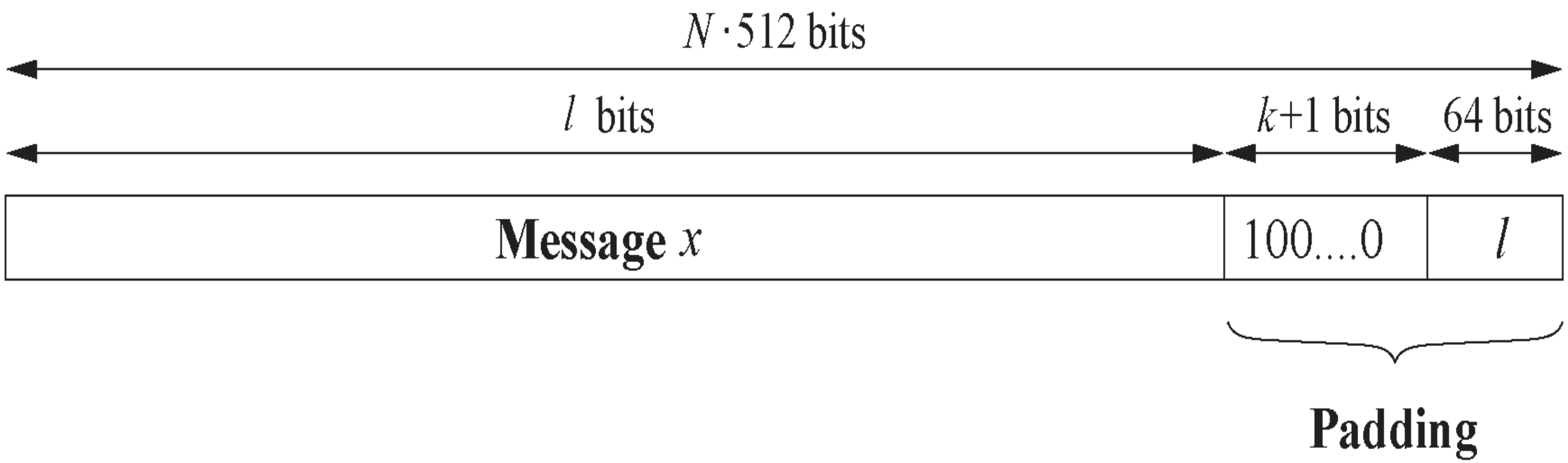
SHA-1 High Level Diagramm

- Compression Function consists of 80 rounds which are divided into four stages of 20 rounds each



SHA-1: Padding

- Message x has to be padded to fit a size of a multiple of 512 bit.
- $k \equiv 512 - 64 - 1 - l = 448 - (l + 1) \bmod 512$.



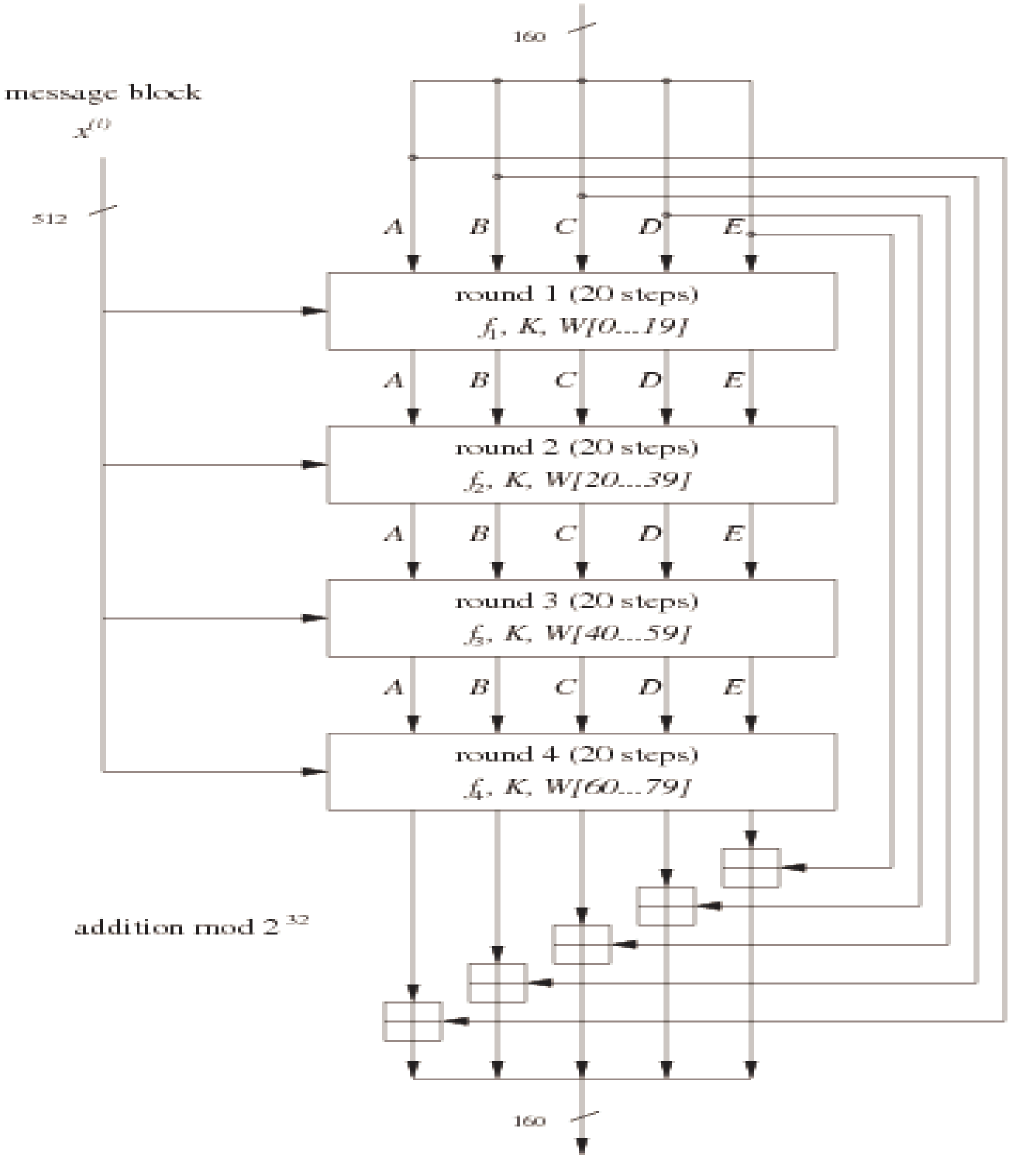
SHA-1: Hash Computation

- Each message block x_i is processed in four stages with 20 rounds each

SHA-1 uses:

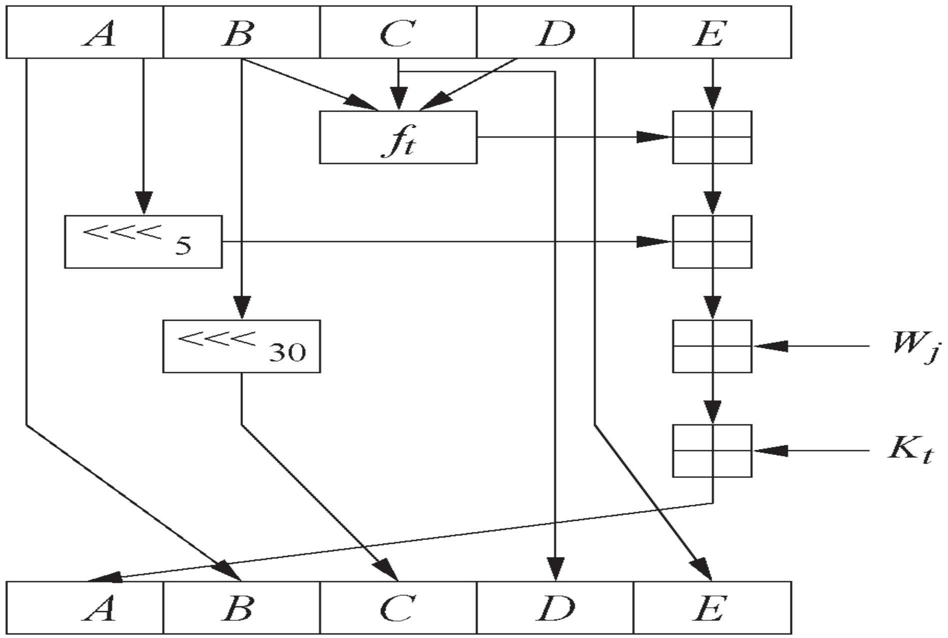
- A message schedule which computes a 32-bit word W0,W1,...,W79 for each of the 80 rounds
- Five working registers of size of 32 bits A,B,C,D,E
- A hash value H_i consisting of five 32-bit words $H_i^{(0)}$, $H_i^{(1)}$, $H_i^{(2)}$, $H_i^{(3)}$, $H_i^{(4)}$
- In the beginning, the hash value holds the initial value H_0, which is replaced by a new hash value after the processing of each single message block.
- The final hash value H_n is equal to the output h(x) of SHA-1.

SHA-1: All four stages



Chapter 11 of *Understanding Cryptography* by Christof Paar and Jan Pelzl

SHA-1: Internals of a Round



Stage t	Round j	Constant K_t	Function f_t
1	00…19	K=5A827999	f(B,C,D)=(B∧C)∨(¯B∧D)
2	20…39	K=6ED9EBA1	f(B,C,D)=B⊕C⊕D
3	40…59	K=8F1BBCDC	f(B,C,D)=(B⊕C)∨(B⊕D)∨(C⊕D)
4	60…79	K=CA62C1D6	f(B,C,D)=B⊕C⊕D

Lessons Learned: **Hash-Funktionen**

- Hash functions are keyless. The two most important applications of hash functions are their use in digital signatures and in message authentication codes such as HMAC.
- The three security requirements for hash functions are one-wayness, second preimage resistance and collision resistance.
- Hash functions should have at least 160-bit output length in order to withstand collision attacks; 256 bit or more is desirable for long-term security.
- MD5, which was widely used, is insecure. Serious security weaknesses have been found in SHA-1, and the hash function should be phased out. The SHA-2 algorithms all appear to be secure.
- The ongoing SHA-3 competition will result in new standardized hash functions in a few years.