

Nota Prévia

Foreword

Parte destes acetatos foram adaptados de outros materiais pedagógicos, nomeadamente do material elaborado pelo Professor Simão Melo de Sousa (desousa@di.ubi.pt) da Universidade da Beira Interior, e pelos Professores J. Bacelar Almeida (jba@di.uminho.pt) e Manuel Bernardo Barbosa (mbb@di.uminho.pt) da Universidade do Minho.

Part of this slide set was adapted from other teaching material, namely of the slide sets elaborated by Professor Simão Melo de Sousa (desousa@di.ubi.pt) of the University of Beira Interior, and by Professors J. Bacelar Almeida (jba@di.uminho.pt) and Manuel Bernardo Barbosa (mbb@di.uminho.pt) of the University of Minho.

Secure Computing

Computação Segura

Engenharia Informática
Computação e Sistemas Inteligentes
Computer Science and Engineering
Computation and Intelligent Systems
2009 / 2010

Pedro R. M. Inácio

Departamento de Informática (DI)
Universidade da Beira Interior (UBI)
Department of Computer Science
University of Beira Interior

Aula 6

Lecture 6

Sumário

Códigos de Autenticação e de Manipulação de Mensagens. Evolução para a discussão das propriedades principais que as funções de Hash devem possuir de modo a poderem ser usadas em aplicações criptográficas.

Primeira abordagem ao tema da Assinatura Digital.

Summary

Message Authentication and Manipulation Codes (MACs and MDCs).

Discussion of the main properties that Hash functions should possess in order to be used in cryptographic applications.

First approach to the Digital Signature subject.

Data Integrity and Message Authentication

Authentication / Integrity

- **Data Integrity:** Accuracy and consistency of stored data, indicated by an absence of any alteration in data between two updates of a data record.
- **Authentication:** act of establishing or confirming something (or someone) as authentic.
- **Source Authentication:** ensuring that the source of the message is, in fact, the one claimed in the context of the communication.

Data Integrity and Message Authentication

Modification Detection Code or Message Authentication Code (I)

- Modification Detection Code (MDC), a.k.a Manipulation Detection Codes (and less commonly MICs), constitute a means to obtain a representative image or hash of a message with the final purpose of facilitating data integrity assurances as required by specific applications. MDCs are a subclass of **unkeyed** Hash Functions.
- The purpose of a MAC is to facilitate, **without the use of any additional mechanisms**, assurances regarding **both** the source of a message and its integrity. MAC have **two** functionally distinct parameters, a message input and a secret key.

Data Integrity and Message Authentication

Modification Detection Code or Message Authentication Code (II)

- When do I use MDCs alone?
 - For example, when the data travels through a channel where errors occur, but one is sure that potential errors are not due to malicious intents.
- When do I need MACs?
 - For example, when sensitive data travels through a means where an attacker may intercept and modify it along the path.
- MACs were thought in such a way that only the parties possessing the secret key can alter the message and correctly calculate its MAC.

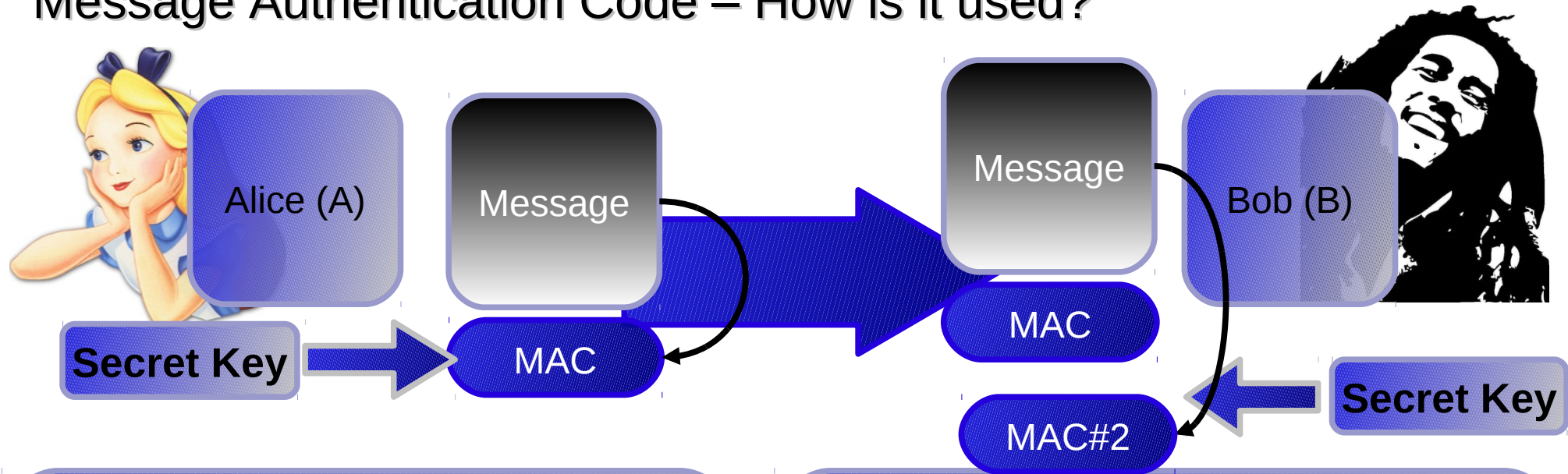
Data Integrity and Message Authentication

Message Authentication Code

- MACs were thought in such a way that only the parties possessing the secret key can alter the message and correctly calculate its MAC:
 - The party that created the message has the secret key and can calculate the MAC;
 - The party that receives the message has the secret key and can calculate the MAC;
 - A third party receives the message:
 - Either it possesses the not so secret key and can modify the message;
 - It does not possess the secret key, and cannot calculate the MAC of modified messages.

Data Integrity and Message Authentication

Message Authentication Code – How is it used?



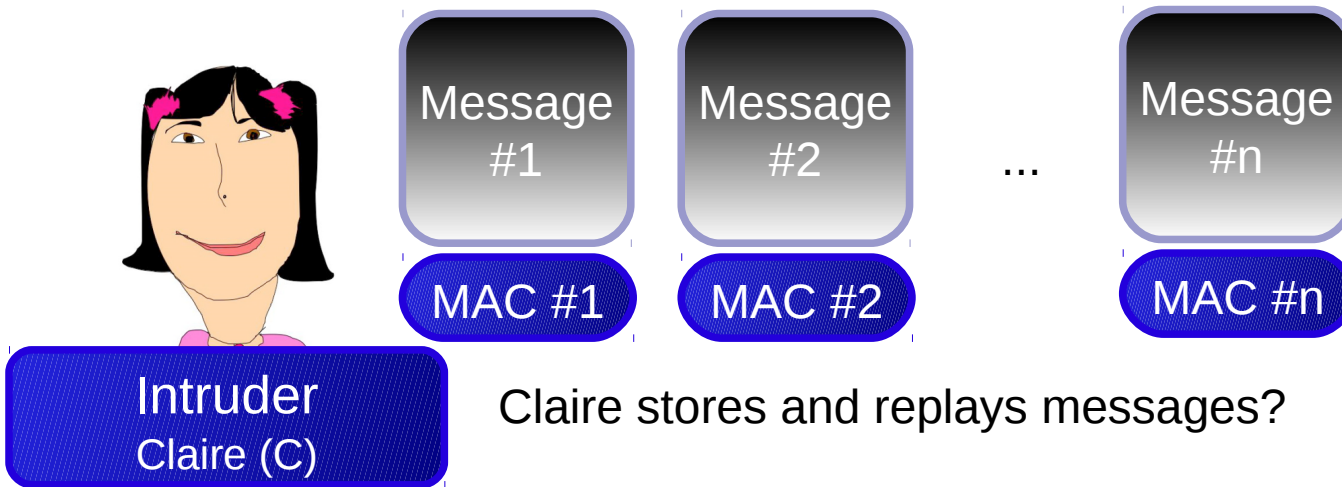
- Creates / alters the message;
- The MAC algorithm digests the message, after being initialized with a Secret Key.
- Sends the message **and** the MAC.

- Receives the message **and** the MAC;
- The MAC algorithm digests the message, after being initialized with a Secret Key.
- Check if the calculated MAC is equal to the received one. If so, both integrity and authentication are assured.

What if an MDC was used instead? (the MDC is not keyed, anyone can calculate it)

Data Integrity and Message Authentication

Message Authentication Code – What can Claire do about it?



- Claire may be storing messages for latter use. She may try to replay messages if the same key is being used for several messages (active attack):
 - Notice that MAC assures that the message was originally created by Alice, but not that this was the message she sent right now (it may be a replay).
- Claire may attempt to modify both a message and associated MAC by trying to guess the effects that a modification may have on the MAC, even without knowing the secret key.
- The last remark leads this presentation to the identification of the requirements that the MAC should fulfill, and to the notions of hash functions and cryptographic hash functions.

Data Integrity and Message Authentication

Definition of Hash Function

A hash function (in the unrestricted sense) is a function h which has, as a minimum, the following two properties:

- 1. compression** — h maps an input bits sequence x with an arbitrary finite length, to an output bits sequence $h(x)$ with a fixed length n .
- 2. ease of computation** — given h and an input bit sequence x , $h(x)$ is easy to compute.

By definition, hash functions are not injective (several input bits sequences may produce the same hash value). To provide strict integrity assurances under the possibility of tampering, these functions should possess additional properties (see below).

Data Integrity and Message Authentication

Definition of Cryptographic Hash Function

Besides exhibiting the previous two properties, a Cryptographic Hash Function possesses also the following ones:

1. **preimage resistance** — for essentially all pre-specified outputs, it is computationally infeasible to find any input which hashes to that output, i.e., to find any preimage x such that $h(x) = y$ when given any y for which a corresponding input is not known.
2. **2nd-preimage resistance** — it is computationally infeasible to find any second input which has the same output as any specified input, i.e., given x , to find a 2nd-preimage $x' \neq x$ such that $h(x) = h(x')$.
3. **collision resistance** — it is computationally infeasible to find any two distinct inputs x, x' which hash to the same output, i.e., such that $h(x) = h(x')$. (Note that here there is free choice of both inputs.)

A cryptographic hash function provides a conditional assurance: it is very unlikely that that a given hash value results from a message different from the one with which it is associated.

Data Integrity and Message Authentication

Meaning of Preimage and Collision Resistance

Suppose that the length of the hash output is H .

- Preimage Resistance means that it would take a Brute Force Attack, i.e. an average of 2^{H1} composite operations to find a a an image colliding with a pre-selected one.

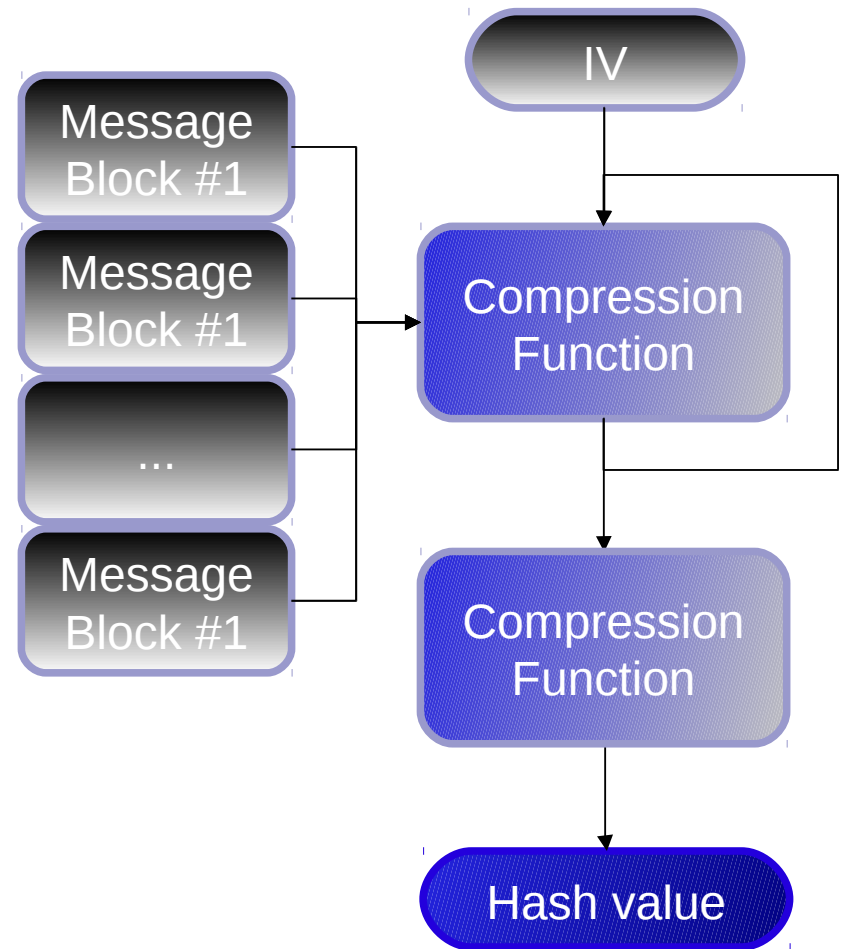
Collision Resistance means that one would have to generate an average of $1.17 * 2^{H2}$ pairs to get find two messages colliding with a probability of 0,5.

- An attack for finding two arbitrary messages with a colliding hash value is called a ***Birthday Attack***:
 - One generates and stores several pairs $(x, H(x))$ until he or she finds two colliding hash values.
 - The Birthday Attack is inspired in the famous *paradox* with the same name (*Birthday Paradox*). A point of confusion of this paradox is that it states that it only takes 23 people to have a probability of 50% for two of them having a birthday on the same day of the year.

Data Integrity and Message Authentication

Construction of Cryptographic Hash Functions

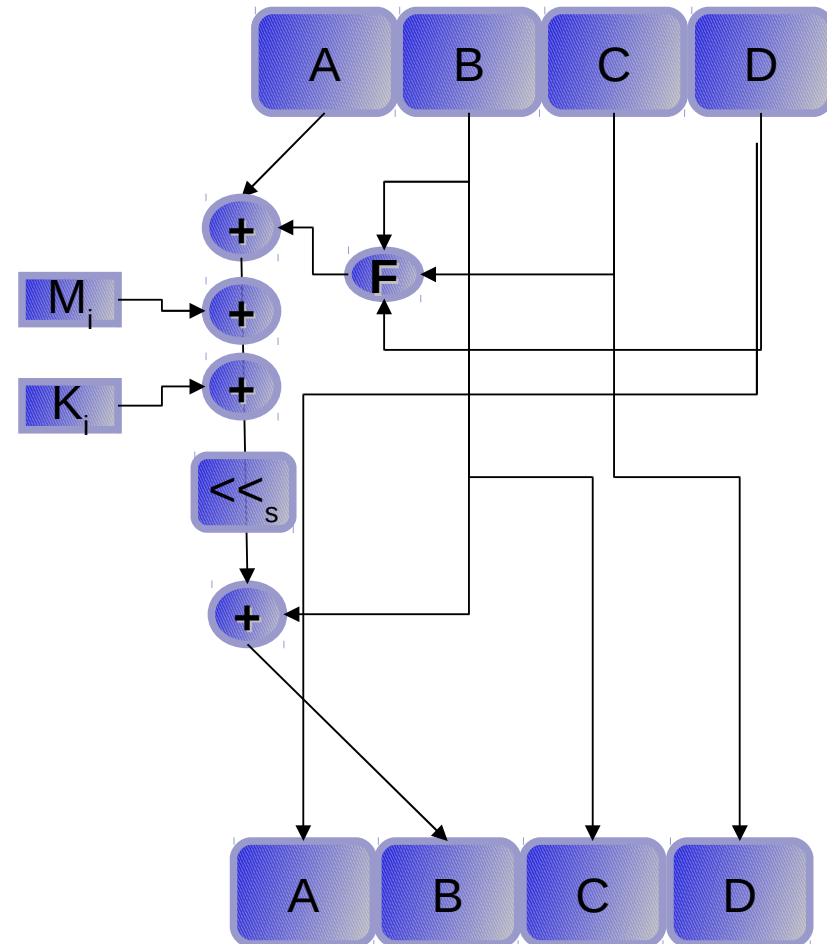
- The Merkle–Damgård construction comprises a popular means to build cryptographic hash functions, though recent proposals make use of other types of constructions (why?):
 - The advantage of using this construction is that, if the compression function is collision-resistant, then the hash function will also exhibit that property.
 - Nonetheless, it is not possible to parallelize the construction.



Data Integrity and Message Authentication

Example of Cryptographic Hash Functions: Message Digest 5

- MD5 (and MD4) produces a 128-bit digest from a message with a maximum length of $2^{64} - 1$ bits (due to a design option).
- Each input message is padded to a multiple of 512 bits (the last 64 bits contain the size of the message).
- The algorithm operates over a 128-bit state, divided into four 32-bit words, denoted A, B, C and D, which are initialized with constants:
 - Curiosity: the empty string has an MD5("") value:
D41d8cd98f00b204e9800998ecf8427e
- The message is divided in chunks of 512 bits, which are further subdivided in pieces of 32 bits. These pieces are combined with A, B, C and D in 64 operations, represented by the scheme on the left.
- F is one of four different functions:
 - $F(X, Y, Z) = X \wedge Y \vee \neg X \wedge Z$
 - $G(X, Y, Z) = X \wedge Z \vee Y \wedge \neg Z$
 - $H(X, Y, Z) = X \oplus Y \oplus Z$
 - $I(X, Y, Z) = Y \oplus (X \vee \neg Z)$



Data Integrity and Message Authentication

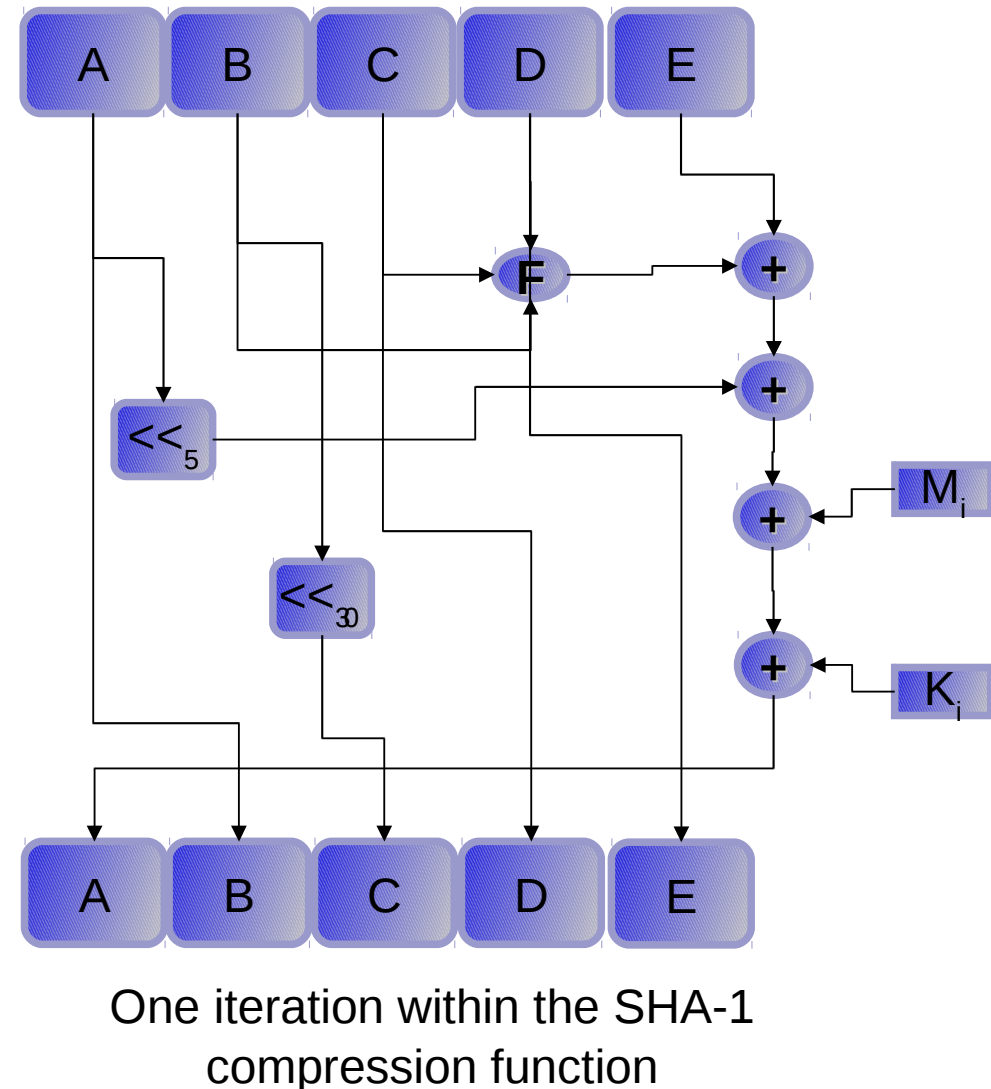
Example of Cryptographic Hash Functions: Secure Hash

- SHA-1 (as well as SHA-0) produces a 160-bit digest from a message with a maximum length of $2^{64} - 1$ bits (see padding scheme used in MD5).
- SHA-1 is based on principles similar to those used in the design of MD4 and MD5, but has a more conservative design.
- SHA-1 is widely used in cryptographic applications:
 - It is used in TLS and SSL, PGP, SSH, S/MIME, and IPsec.
 - A prime motivation for the publication of the Secure Hash Algorithm was the Digital Signature Standard, in which it is incorporated.
- The empty string produces the following hash value:
 - $\text{SHA1}("") = \text{da39a3ee5e6b4b0d3255bfef95601890afd80709}$

Data Integrity and Message Authentication

Example of Cryptographic Hash Functions: Secure Hash 2 and 3?

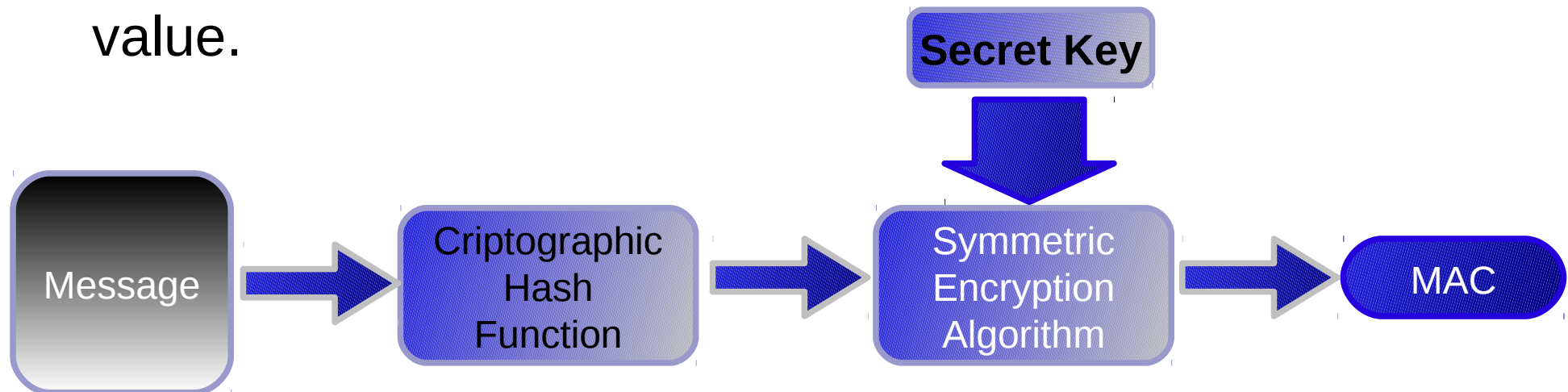
- The SHA-2 family uses an identical algorithm with a variable digest size which is distinguished as SHA-224, SHA-256, SHA-384, and SHA-512.
- An effort to find the next Secure Hash (SHA-3) is currently undergoing (started in 2007).



Data Integrity and Message Authentication

Implementation of Message Authentication Codes

1. Use a block cipher in a cipher feedback mode (CBC or CFB), encrypt the message and get the last block of ciphertext as MAC (perhaps encrypting it one more time with the cipher in the same mode).
2. Generate the hash value with a cryptographic hash function and use a symmetric cipher to encrypt the value.



Data Integrity and Message Authentication

Hash based Message Authentication Codes (HMACs)

An HMAC is a specific way of combining a cryptographic hash function with a secret key to construct a MAC:

$$\text{HMAC}(K,m) = H((K \oplus \text{opad}) \parallel H((K \oplus \text{ipad}) \parallel m)),$$

Where

- $H(\cdot)$ is a cryptographic hash function;
- K is a secret key padded to the right with extra zeros to the block size of the hash function;
- m is the message to be authenticated;
- \parallel denotes concatenation and \oplus exclusive or (XOR);
- opad and ipad are the outer ($0x5c5c5c\dots5c5c$) and inner padding ($0x363636\dots3636$), respectively.

Data Integrity and Message Authentication

Hash based Message Authentication Codes (HMACs)

- HMAC is specified in the Request for Comments (RFC) 2104: *HMAC: Keyed-Hashing for Message Authentication*.
- Nowadays, HMAC is one of the most commonly used MACs:
 - e.g. HMAC-SHA-1 and HMAC-MD5 are used within the IPsec and TLS protocols.

Digital Signatures

Informal definition of a Digital Signature

Is a Digital Signature just the digital version of a manuscript signature, or more than that?

On an introductory book to cryptography made by the PGP team, one may find the following excerpt:

*“A digital signature serves the same purpose as a handwritten signature. However, a handwritten signature is easy to counterfeit. A **digital signature is superior to a handwritten signature in that it is nearly impossible to counterfeit, plus it attests to the contents of the information as well as to the identity of the signer.**”*

Digital Signatures

The Goodies (main properties offered by digital signatures)

- A (manuscript) normal signature normally offers:
 - **Authenticity of the Information** (i.e. it assures the recipient of the message that the party that signed the message was aware of its contents e.g. he or she actually created the message);
 - **Non-repudiation assurances** (i.e. it provides the recipient with the means to prove that a given party issued the signed message, even if the party denies it).
 - **Data Integrity Assurances** (i.e. it assures that the message was not altered while in transit);
 - **Data Origin Authentication** (i.e. it assures that the message was originated by the signing party);
 - **Difficult to Forger** (i.e. it assures that no other party could have signed the message, except for the known sender)
 - In the case of a digital signature, this also means that the signature generated for a given message cannot be reused / adapted to other messages (i.e. a signature cannot be detached from the message it refers to).

Digital Signatures

Importance of Digital Signatures

- Nowadays, digital signatures may even be more used than privacy or confidentiality mechanisms.
 - E.g. Financial Transactions (you want to be sure that the receipt of a huge transaction is duly signed by the bank entity).
- A digital signature is the digital equivalent of a manuscript signature, but applied to electronic documents. It is a **sequence of bytes** attached to the electronic document that has **no meaning if considered alone and out of the context of the signing / verification algorithm**.
- The implementation of digital signatures however is not an easy task to accomplish, mostly due to the following reasons:
 - It is extremely easy to copy computer files;
 - If a simple digitalized form of an manuscript signature was used, it would be easily tampered or reused;
 - Computer files are easy to modify without leaving any trace.

Digital Signatures

A Little of History

- The notion of a digital signature scheme was first described in **1976** by Whitfield Diffie and Martin Hellman, although they only conjectured that such schemes existed.
- Soon afterwards, Ronald Rivest, Adi Shamir, and Len Adleman invented the RSA algorithm that could be used for primitive digital signatures, though the *plain* RSA signatures are not secure (see below for the description on how to use RSA to produce secure digital signatures).
- The first widely marketed software package to offer digital signature was Lotus Notes 1.0, released in **1989**, which used the RSA algorithm.

Digital Signatures

Security Notions

In the seminal paper of Goldwasser, Micali, and Rivest, the hierarchy of attack models against digital signatures is as follows:

- In a key-only attack, the attacker is only given the public verification key.
- In a known message attack, the attacker is given valid signatures for a variety of messages known by the attacker but not chosen by the attacker.
- In an adaptive chosen message attack, the attacker first learns signatures on arbitrary messages of his choice.

They also describe a hierarchy of attack results:

- A total break results in the recovery of the signing key.
- A universal forgery attack results in the ability to forge signatures for any message.
- A selective forgery attack results in the ability to forge signature on a message of the choice of the adversary.
- An existential forgery results in valid message/signature pair not already known to the adversary.

The strongest notion of security is security against existential forgery under an adaptive chosen message attack.

Digital Signatures

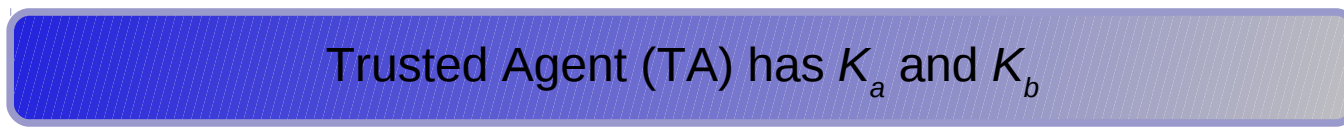
Definition and Implementation (I)

- The construct a digital signature scheme, 2 main algorithms are needed:
 - An algorithm that accepts (I) the message to be signed and (II) a private key (or a secret used with the same purpose) as inputs, and generates the ***digital signature***.
 - An algorithm that accepts (I) the signed message, (II) the public key (or a secret used with the same purpose) and (III) the signature, and verifies that all the previously mentioned security properties are met.
- Notice that the previously mentioned algorithms are the ones responsible for assuring that all the required security properties are met.

Digital Signatures

Definition and Implementation (II) – Using Symmetric Cryptography

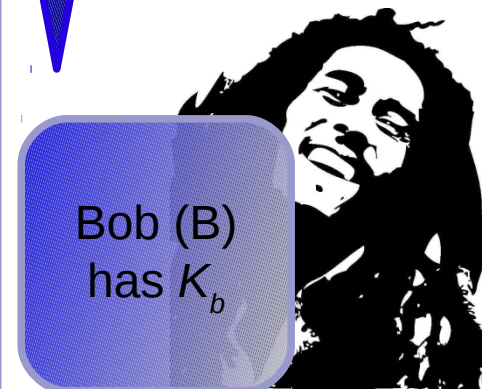
(I)



Alice sends the message and its encryption to the TA. TA verifies that the message was indeed sent by Alice, generates a certificate and sends it to Bob.



Bob gets the message and the certificate encrypted with K_b



Digital Signatures

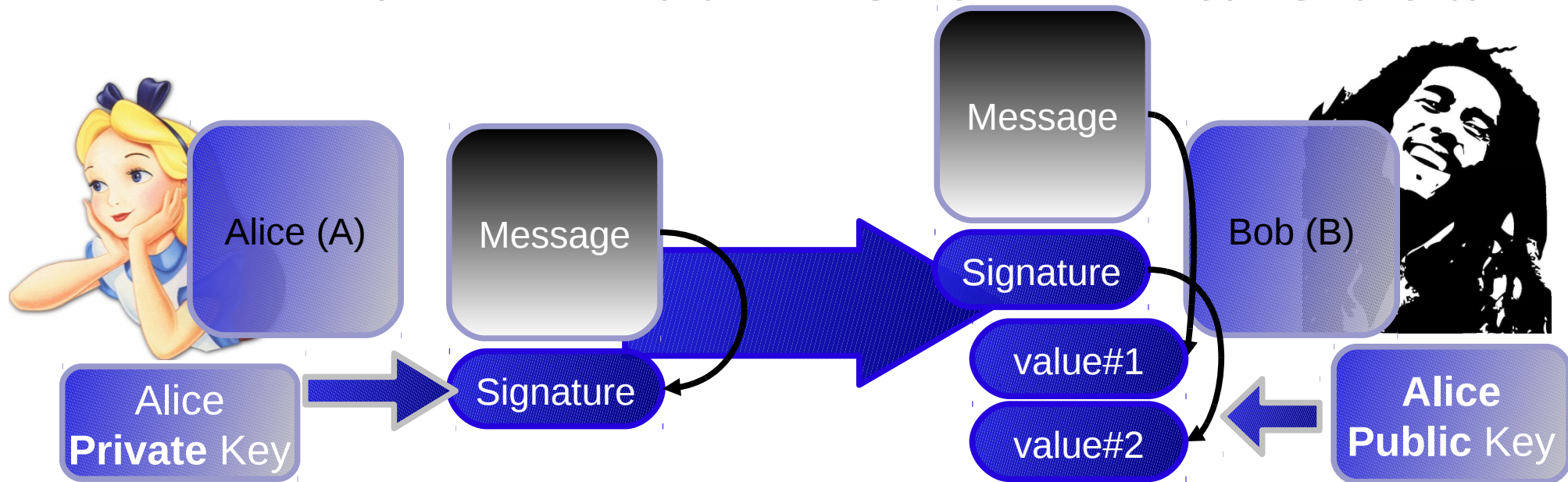
Definition and Implementation (II) – Using Symmetric Cryptography (II)

The verification is inherently embedded in the communication system:

- Bob trusts the TA and accepts the certificate the latter send to him.
 - The signature is authentic, because the TA assures Bob that Alice signed the message.
- The signature cannot be forged because only Alice and the TA know K_a (Bob never gets to need it).
- The signature cannot be used later on, because Bob would not be able to generate another message encrypted with K_a .
- The message cannot be altered because of the same reason.
- Alice cannot say that she did not issued the message, because Bob has the message she sent to the TA, encrypted with her private key.
- A Data Base could be used to prevent the TA to send messages encrypted with K_a to Bob.

Digital Signatures

Definition and Implementation (III) – Using Asymmetric Cryptography (I)



- Creates / alters the message;
- The signing algorithm generates the signature, receiving the message and the Private Key as inputs.
- Sends the message **and** the signature.

- Receives the message **and** the signature;
- A hash algorithm is typically used to digest the message.
- Check if the calculated hash is equal to value obtained after decrypting the signature with the Public Key of Alice.

Digital Signatures

Definition and Implementation (III) – Using Asymmetric Cryptography (II)

- The verification is performed by decrypting the signature with the public key of the sender of the message, and by checking if the decryption matches the plain text of the message or of its hash.
- The signature is authentic, since only Alice knows its private key.
- The signature cannot be forged nor reused, because Bob would not be able to generate another message encrypted with the private key of Alice.
- The message cannot be altered because of the same reason.
- Alice cannot say that she did not issued the message, because Bob has the message she encrypted with her private key.