Digital Security

M. Adnan Quaium

Assistant Professor
Department of Electrical and Electronic Engineering
Ahsanullah University of Science and Technology
Room – 4A07
Email – adnan.eee@aust.edu
URL- http://adnan.quaium.com/aust/cse4295

Digital Signatures

Just as with a normal signature in everyday life, there are specific requirements for a **digital signature**. It must be:

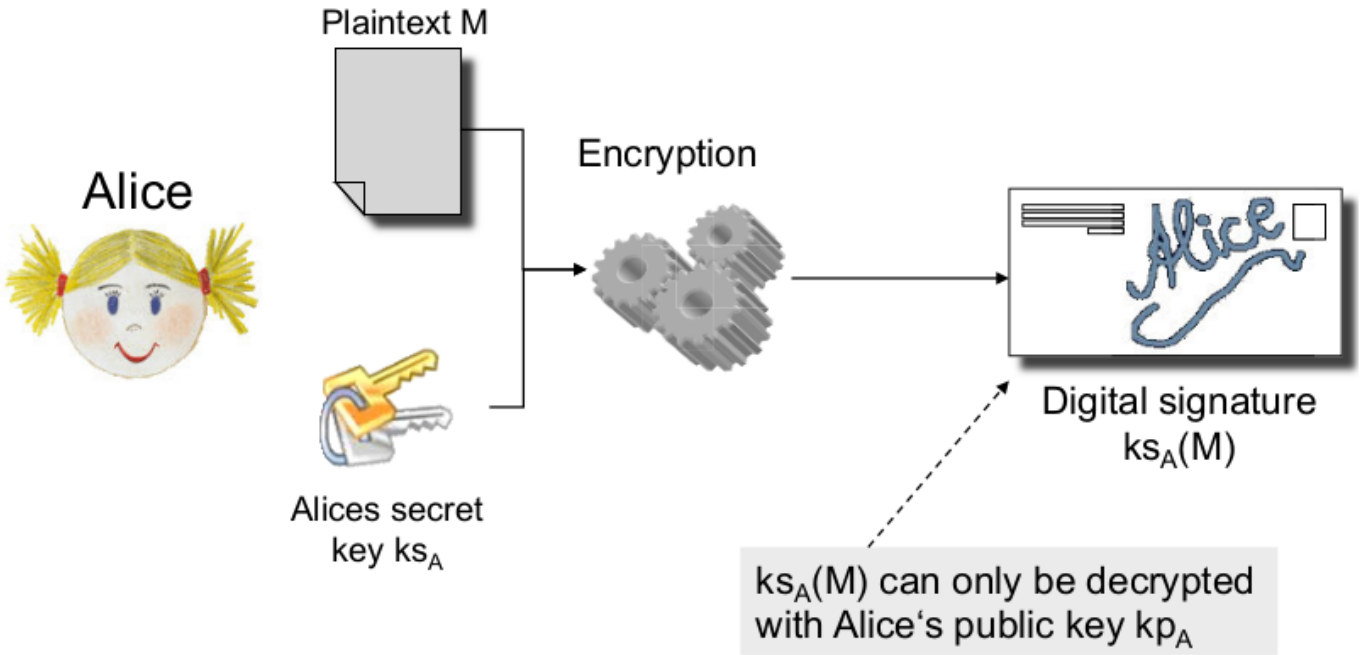
- Verifiable,
- Tamper-proof and
- Binding

There must be an assurance that the one who signs the document is indeed the person he or she claims to be.

At the same time it should not be possible for the signer of a document to later be able to dispute having signed the document in question.

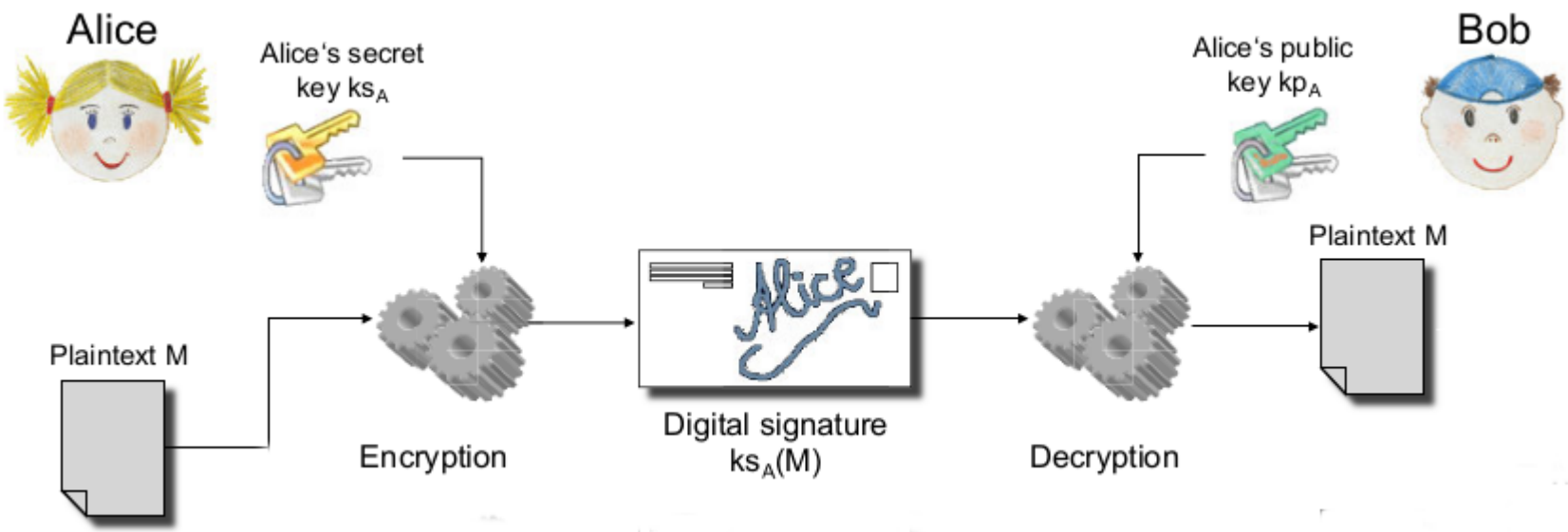
Data Integrity and Authenticity

Let us assume that Alice wants to use a digital signature for proof of the integrity of message M and its authenticity. Additionally, Bob, the receiver of message M , wants to be assured that it was in fact Alice who sent him message M



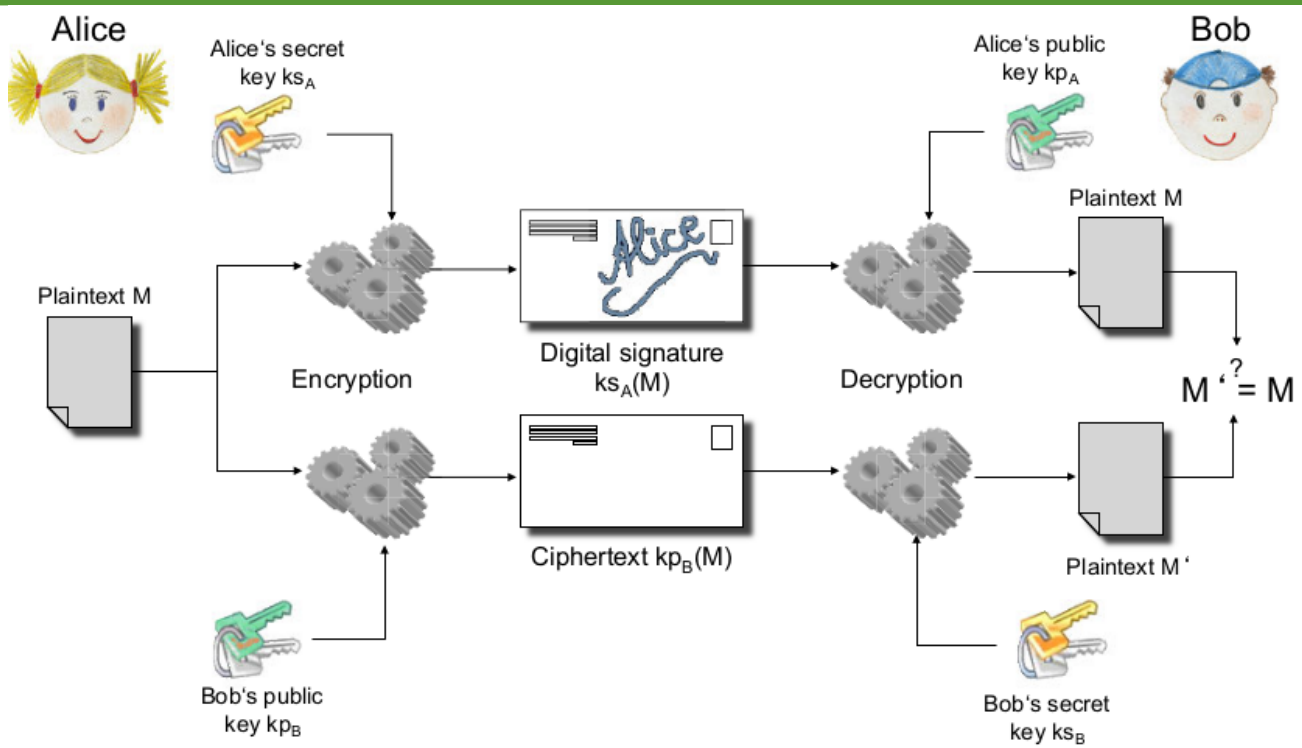
- To sign message M digitally Alice simply uses an asymmetrical encryption algorithm together with her own secret key ks_A and calculates $ks_A(M)$, the so-called digital signature of message M.
- Alice sends $ks_A(M)$ to Bob, who with Alice's public key kp_A, can decrypt the message $M = kp_A(ks_A(M))$.

Digital Signatures



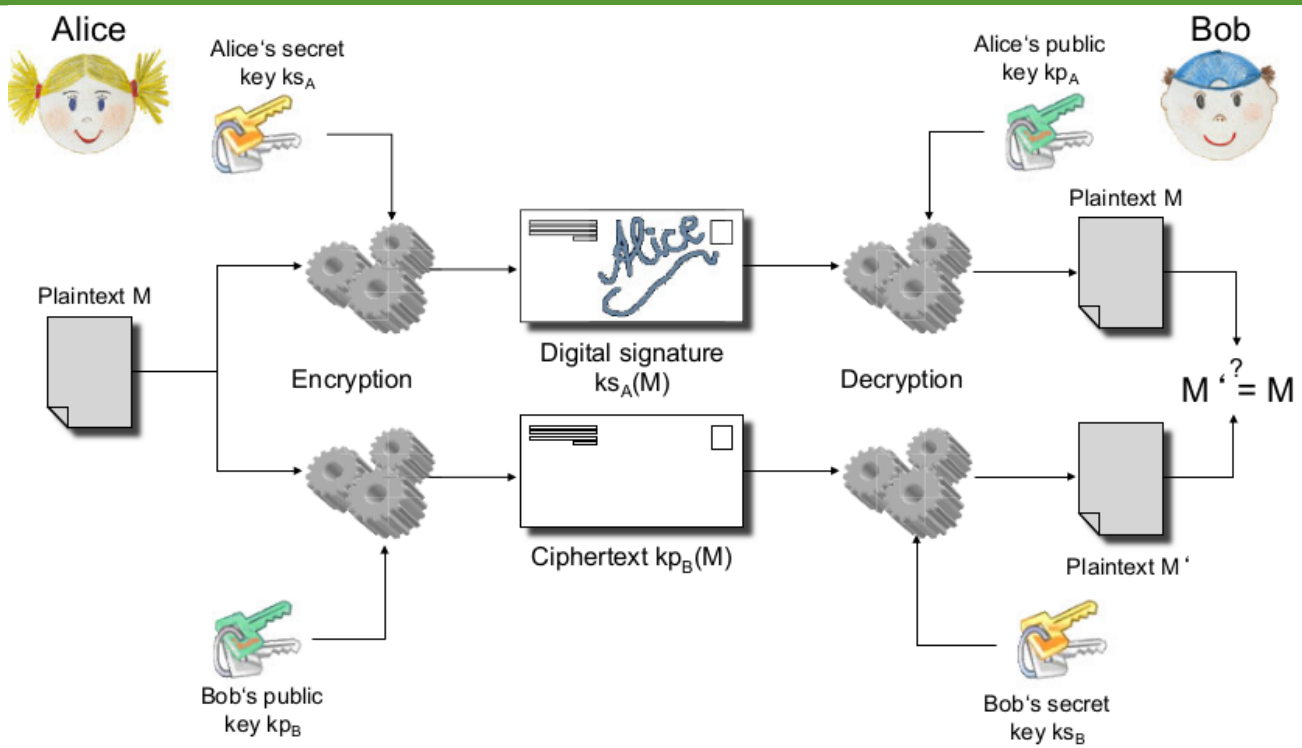
- Bob can verify whether Alice was actually the sender of the message by using the decryption algorithm with Alice's public key kp_A and calculating $kp_A(ks_A(M))$.
- If the result is then readable as message M, Bob can be sure that Alice is the originator and signer of the received message.
- $kp_A(ks_A(M)) = M$ applies *ONLY IF* the received digital signature $kp_A(M)$ is encrypted with Alice's secret key ks_A.
- However, the only person who has access to this key (according to prescribed use) is Alice. Therefore, Alice is the only one who could have generated the digital signature.

Digital Signatures



- To additionally ensure that the sent message was not manipulated during transmission, besides sending the digital signature Alice can also send Bob the message ***M*** .
- Alice can protect message ***M*** from being seen by third parties by encrypting ***M*** with Bob's public key kp_B and transmitting it as $kp_B(M)$.
- Upon receipt, Bob decrypts $kp_B(M)$ with his secret key ks_B and obtains the original message $M = ks_B(kp_B(M))$.
- By comparing message ***M*** with the decrypted digital signature $M=kp_A(ks_A(M))$, Bob can tell whether the content of the original message ***M*** was manipulated in any way during the course of transmission.

Digital Signatures



If an unauthorized third party has forged the original message M as a new message M' before Bob has received it, Bob recognizes in encryption that $kp_A(ks_A(M)) \neq M'$ now applies.

Digital Signatures

However, there are ***two serious problems*** with this variation of the digital signature.

- ***First,*** the calculation complexity of asymmetrical encryption procedures increases when compared to symmetrical encryption procedures by a factor of 1,000. The calculation of a digital signature, which is based on encryption of the whole document, is extremely complex in the case of a large document.

- ***Second,*** it cannot be guaranteed that the message content remains secret. It is namely always possible to decipher Alice's digital signature with Alice's public key. Because this key is accessible to everyone there is an ever-present danger that the contents of the original message might be disclosed.

Message Digest

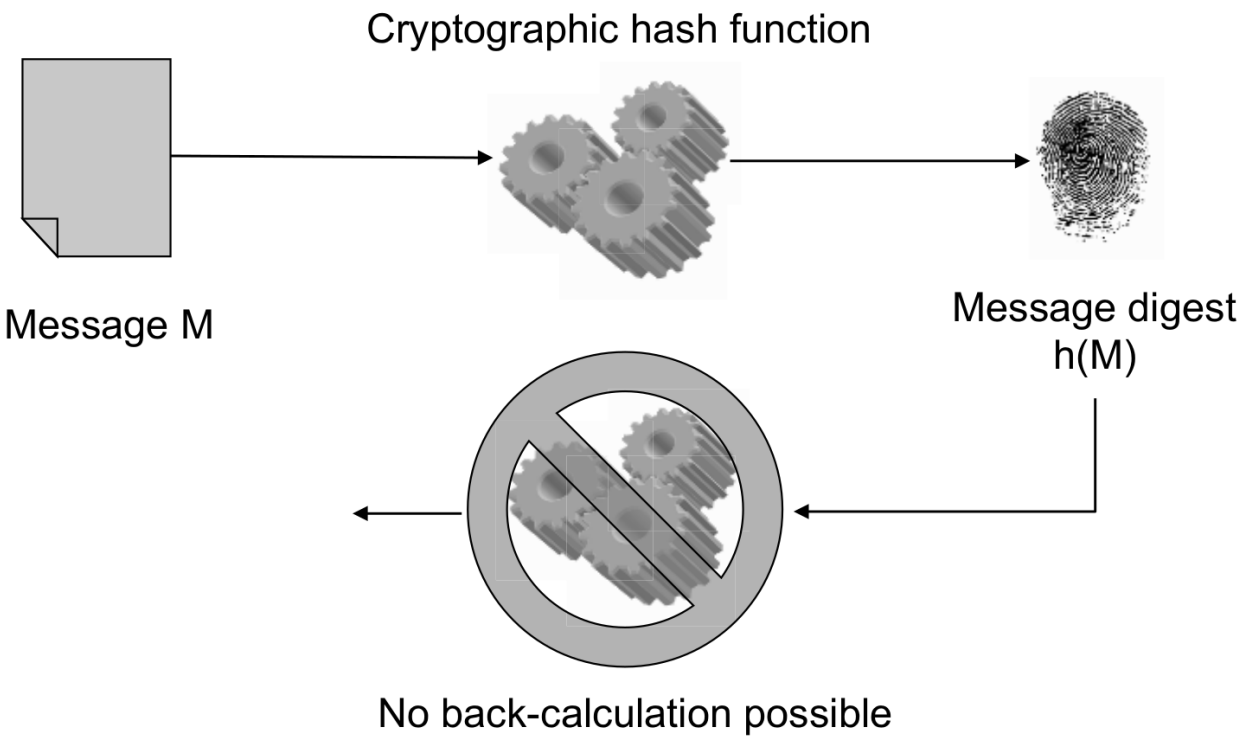
The encryption of an entire message in order to generate a digital signature is in many cases an extremely cumbersome process.

- The necessary data transfer alone is doubled thus hindering an effective encryption of the content.
- In addition, there are a lot of messages regularly exchanged by a large number of network components and processes that do not need encryption, e.g., routers or email agents.
- Solely the identity of the sender must be secured and the receiver assured that the message content has not been tampered with by an unauthorized third party.

In these cases the application of a hash function is recommended to generate a message digest.

Message Digest

Such hash functions calculated from the content of the message to be sent M as short as possible "fingerprint" h(M) of the message of fixed-length, known as a message digest.



Message Digest

- It is sent together with the message to the receiver.
- With the help of the message digest it is possible to check whether the contents of the received message ***M'*** have arrived intact and therefore if ***M = M'*** is valid.
- To do this, using the same hash function the receiver calculates the fingerprint ***h(M')*** of the received message ***M'*** and verifies whether it matches the received fingerprint ***h(M)***.
- If ***h(M') = h(M)***, then due to the requirements placed on the hash function, ***M = M'*** is also valid.
- The received message is thus identical with the message originally transmitted

Message Digest

From this point of view it makes sense to use the message digest for the digital signature. Namely, if Alice carries out encryption with her secret key ks_A on the message digest $h(M)$, instead of encrypting the whole message it is sufficient to transfer $ks_A(h(M))$ with the original message instead of $ks_A(M)$. This fulfills all the requirements of a digital signature. At the same time the message digest function must fulfill the following requirements to qualify as forgery-proof:

- With a given message digest value d it is not possible to reconstruct the original message M with a reasonable effort so that $h(M) = d$ applies.
- It is impossible to find two different messages M and N with a reasonable effort so that $h(M) = h(N)$ applies.

Message Digest

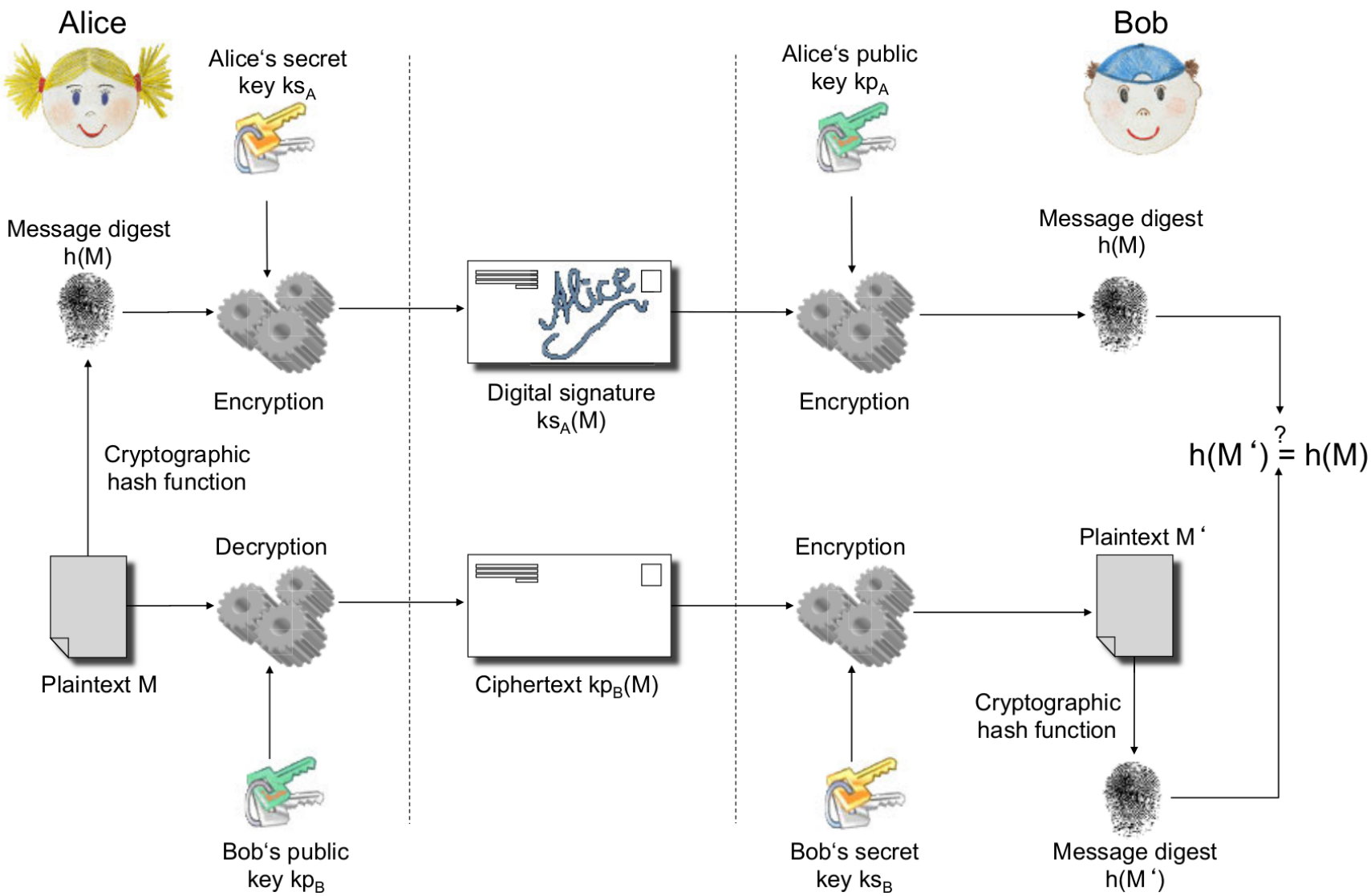


Fig. 5.16 Use of digital signature and message digest.

Message Digest

- For generating the message digest, Alice uses the hash function h on the message to be sent M and encrypts $h(M)$ with the encryption function and its secret key ks_A.
- Alice sends M together with $ks_A(h(M))$.
- Bob uses the decryption function together with Alice's public key kp_A on the received digital signature $ks_A(h(M))$ of Alice's message.
- Bob applies the hash function h on the received message M' and compares whether $h(M') = kp_A(ks_A(h(M)))$ applies.
- If both values are identical, Bob can assume that the received message is really from Alice and has not been altered during transmission.

Public Key Infrastructures and Certificates

Secure key exchange or secure key assignment can be ensured by using a trusted intermediary (trusted third party).

- In a ***symmetric encryption procedure*** this trusted third party is also called the ***Key Distribution Center (KDC)***.
- The KDC manages the private, secret keys necessary for secure communication via a symmetric encryption procedure.
- It creates secure and reliable distribution thereby preventing unauthorized third parties from gaining access to the private keys.

Public Key Infrastructures and Certificates

- In an ***asymmetric encryption procedure*** it is necessary for a communication partner to provide assurance that the public key indeed comes from him. This is done by way of ***certificates***.
- These are issued and digitally signed by a trusted third party who provides the guarantee of a ***certificate authority (CA)*** or a ***trust center or trust authority (TA)***.
- Generally, such certification authorities form a hierarchical structure with a root authority at the top, and various subordinate bodies and users below.
- Such a hierarchy of certification authorities and all the associated data technology (e.g., certificate formats) and organizational requirements (security policy) is called a ***public key infrastructure (PKI)***. This infrastructure is necessary to perform an asymmetrical key procedure securely.

Key Distribution Center (KDC)

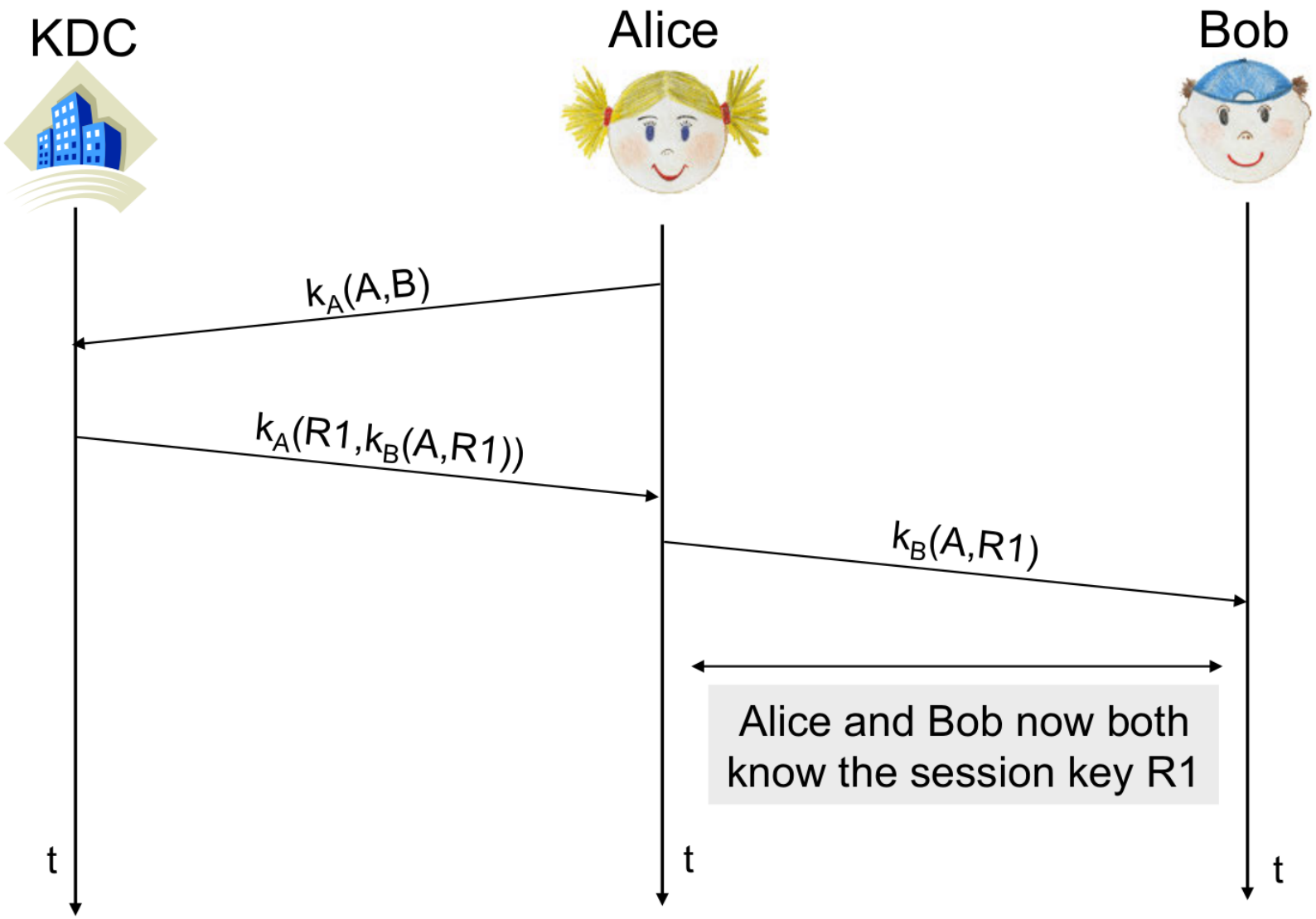


Fig. 5.17 Generation and secure distribution of a shared, secret session key for Alice and Bob via a key distribution center (KDC).

Key Distribution Center (KDC)

- Alice takes the initiative and sends a message $k_A(A, B)$ encrypted with k_A to the KDC, that she (A) wishes to communicate with Bob (B).
- The KDC has Alice's secret key k_A and can therefore decrypt Alice's message $k_A(A, B)$. The KDC then randomly generates key $R1$, which can be used for Alice and Bob's subsequent communication as a **unique session key**. The KDC sends a message to Alice encrypted with t k_A containing the following:

  - the unique session key $R1$ and
  - a pair of values consisting of Alice's name A and the session key $R1$, which is encrypted with Bob's secret key k_B: $k_B(A, R1)$.

  The KDC sends the encrypted message $k_A(R1, k_B(A, R1))$ to Alice.
- Alice receives and decrypts the KDC's message. She extracts the unique session key $R1$ and saves it for the subsequent communication with Bob and then directs the second part of the message $k_B(A, R1)$ (not understood) to Bob.
- Bob receives $k_B(A, R1)$, decrypts the message with his own secret key k_B and thereby learns of Alice/A's communication wish and the shared unique session key $R1$. The encrypted communication between Alice and Bob can now begin. If necessary, Alice and Bob can agree upon their own shared key in the first communication round, which is now securely encrypted by R1. This key is not known to the KDC.

Certification Authority (CA)

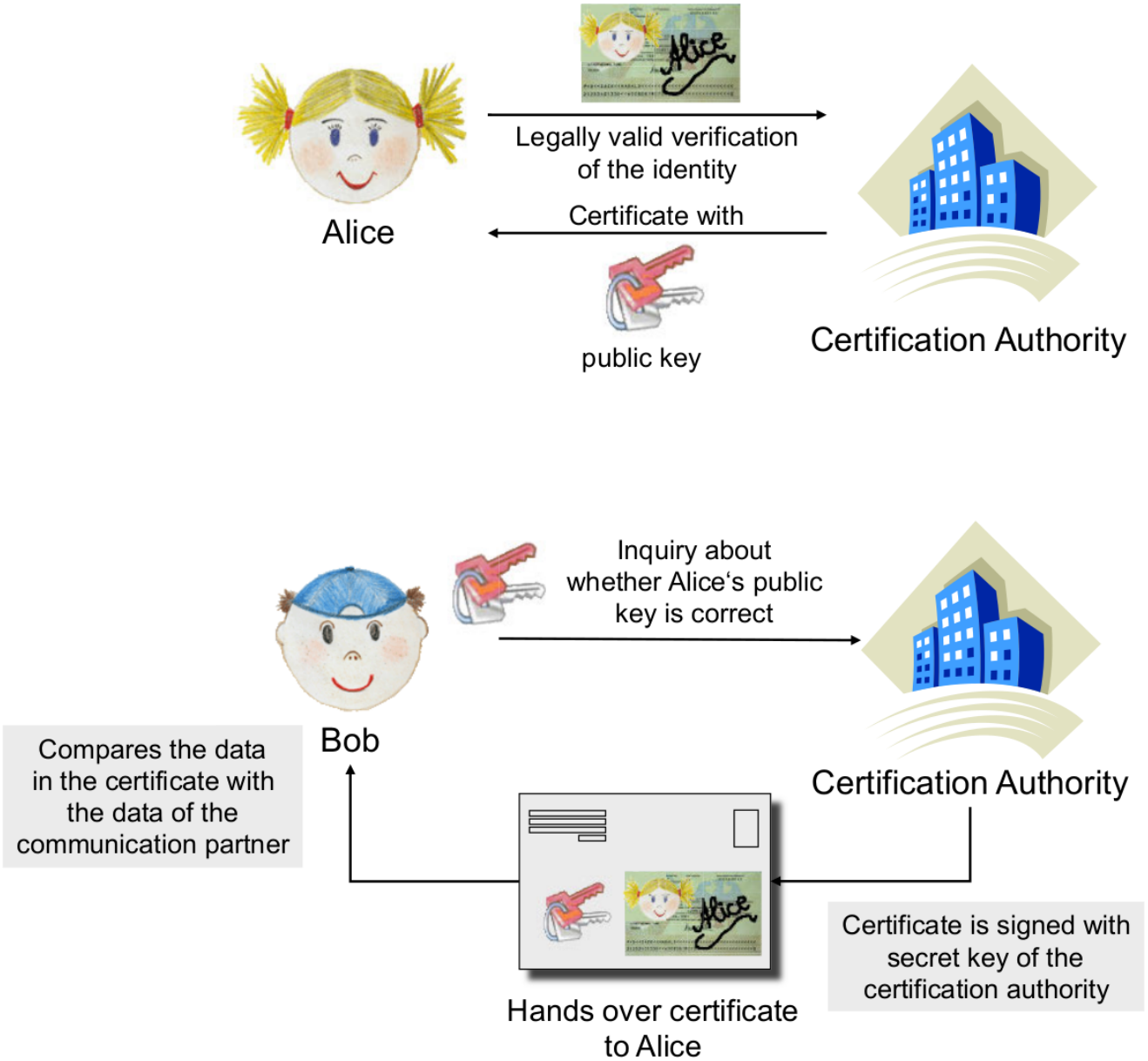


Fig. 5.18 Alice and Bob using a CA to verify the public key.

Certification Authority (CA)

To ensure that a public key passed by a user in fact corresponds to its specified identity, the following steps are taken:

- If Alice would like to communicate with Bob via an asymmetrical encryption procedure, she sends him her message together with her certificate (the certificate can also be requested by the CA).
- The respective CA has made its own public key accessible to all users in a secure manner (e.g., published in a prominent place in a reputable newspaper). Using this public key from the CA, Bob decrypts the certificate from Alice.
- If Bob is able to decrypt the certificate and if the information it contains corresponds to the information from Alice, Bob can be sure that he is really in contact with Alice and then uses her public key for further communication.