Message Digest & Digital Signature.

Checksums vs. Mess. Digests

- Checksums:
  - Used to produce a compact representation of a message
  - If the message changes the checksum will probably not match
  - Good: accidental changes to a message can be detected
  - Bad: easy to purposely alter a message without changing the checksum
- Message digests:
  - Used to produce a compact representation (called the **fingerprint** or **digest**) of a message
  - If the message changes the digest will probably not match
  - Good: accidental changes to a message can be detected
  - Good: difficult to alter a message without changing the digest

Signing the Digest

We said before that public-key encryption is efficient if the message is short.

Using a public key to sign the entire message is very inefficient if the message is very long.

The solution is to let the sender sign a digest of the document instead of the whole document. The sender creates a miniature version or digest of the document and signs it; the receiver then checks the signature on the miniature.

To create a digest of the message, we use a hash function. The hash function creates a fixed-size digest from a variable-length message, as shown in Figure

The two most common hash functions are called MD5 (Message Digest 5) and SHA-I (Secure Hash Algorithm I). The first one produces a 120-bit digest. The second produces a 160-bit digest.

Note that a hash function must have two properties to guarantee its success.

First, hashing is one-way; the digest can only be created from the message, not vice versa.

Second, hashing is a one-to-one function; there is little probability that two messages will create the same digest. We will see the reason for this condition shortly.

After the digest has been created, it is encrypted (signed) using the sender's private key. The encrypted digest is attached to the original message and sent to the receiver.

Idea of a Message Digest

The concept of message digests is based on similar principles. However, it is slightly wider in scope. For instance, suppose that we have a number 4000 and we divide it by 4 to get 1000 Thus, 4 can become a fingerprint of the number 4000. Dividing 4000 by 4 will always yield 1000. If we change either 4000 or 4, the result will not be 1000.

Another important point is, if we are simply given the number 4, but are not given any further information, we would not be able to trace back the equation 4 x 1000 = 4000. Thus, we have one more important concept here. The fingerprint of a message (in this case, the number 4) does not tell anything about the original message (in this case, the number 4000). This is because there are infinite other possible equations, which can produce the result 4.

Another simple example of message digest is shown in fig. Let us assume that we want to calculate the message digest of a number 7391753. Then, we multiply each digit in the number with the next digit (excluding it if it is 0), and disregarding the first digits of the multiplication operation, if the result is a two-digit number.

Idea of a Message Digest

Thus, we perform a hashing operation (or a message digest algorithm) over a block of data to produce its hash or message digest, which is smaller in size than the original message. This concept is shown in fig.

Actually, the message digests are not so small and straightforward to compute. Message digests usually consist of 128 or more bits. This means that the chance of any two-message digests being the same is anything between 0 and at least 2128. The message digest length is chosen to be so long with a purpose. This minimizes that the scope for two messages digests being the same.

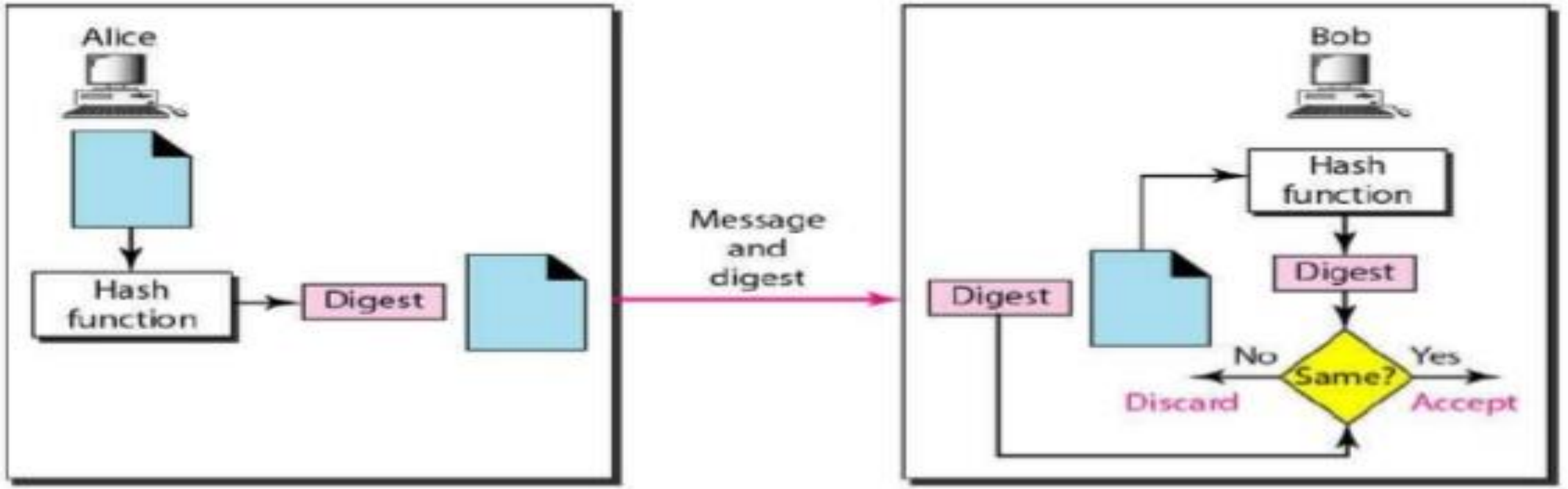
We can summarize the requirements of the message digest concept, as follows:

- Given a message, it should be very easy to find its corresponding message digest. Also for a given message, the message digest must always be the same.
- Given a message digest, it should be very difficult to find the original message for which the digest was created.
- Given any two messages, if we calculate their message digests, the two message digests must be different.

Another basis of message digest is that it should not give any clue or indication of the original message. i.e. it should not be possible to revert back to original message from the digest. Also, for a given message it's digest should be the same always.

Message Digest

Different algorithms are used to convert original message into its message digest. The popularly used ones are MD5 or Message Digest 5 (developed by Rivest) a modified version of earlier MD4, MD3 and MD2, while the first one was simply MD, and the SHA (Secure Hash Algorithm) developed by National Institute of Standards and Technology (NISI) in 1993. SHA-I is promoted & prominently used than the MD5 algorithm.



Digital Signatures

In earlier discussion of Asymmetric key cryptography, we had considered the only situation, in which if X is sender & Y receiver, then X encrypts the message with Y's public key and on receiving, Y decrypts with his own private key. This method only ensures secure communication between the two. Now consider another situation. If X is sender and Y is receiver, X encrypts the message using his own private key! On receiving, Y decrypts it using X's public key. The purpose behind this move is 'authentication'. It is clear that, only X knows his private key.

So, when Y receives this message (encrypted with X's private key), it is an indication or proof that it has originated only from X and none else! Remember that in earlier scheme, the purpose was only 'confidentiality' and the origin of message was not the concern.

Now, one may say that if someone else wants to intercept this communication it should be easy. i.e. anyone can decrypt the message who knows X's public key. This is true, but then it will not be possible for anyone to again encrypt this message as only X knows his private key. Thus receiver here will not be fooled that message came from X This scheme confirms the origin of the message. So, in this case X cannot deny that he has sent the message to Y, because it was encrypted with X's private key, known only to X

Digital Signatures

The above discussion forms the basis for the concept called ' Digital Signature" In case of our normal operations, we make use of our (handwritten) signatures. These are used to confirm the 'origin' or the 'authentication' of the individual. In the Internet world, it would be difficult to use any such method in practice. Hence the concept of 'Digital signatures' was evolved.

This technique is vitally important in the E-commerce concept used in the Internet. It proves as a valid mechanism for 'authenticity' of individual. Most of the financial transactions done over Internet make use of this method.
Techniques of Digital signatures:

Actual working of Digital signatures involves the use of a concept called 'Message digest' or 'hash'. Message digest is something like the summary of original message. (works similar to the CRC checksum concept) This is basically used to verify the 'integrity' of data i.e. to ensure that the message has not been modified after it was sent by sender and before it reaches the receiver.

Digital Signatures

The Digital Signature Standard (DSS) was developed by NIST first in 1991. It suggests using the SHA-1 algorithm for calculating the message digest. This digest is further used for performing Digital signatures, by using the algorithm called Digital Signature Algorithm (DSA). In DSA, message digest is encrypted with the sender's private key to form the Digital Signature (DS). This signature is transmitted further along with the original message. It is also possible to use the earlier RSA algorithm for performing digital signatures. RSA is prominently used over DSA as DSA turns out to be more complicated.

Steps for the process

Sender's Side:

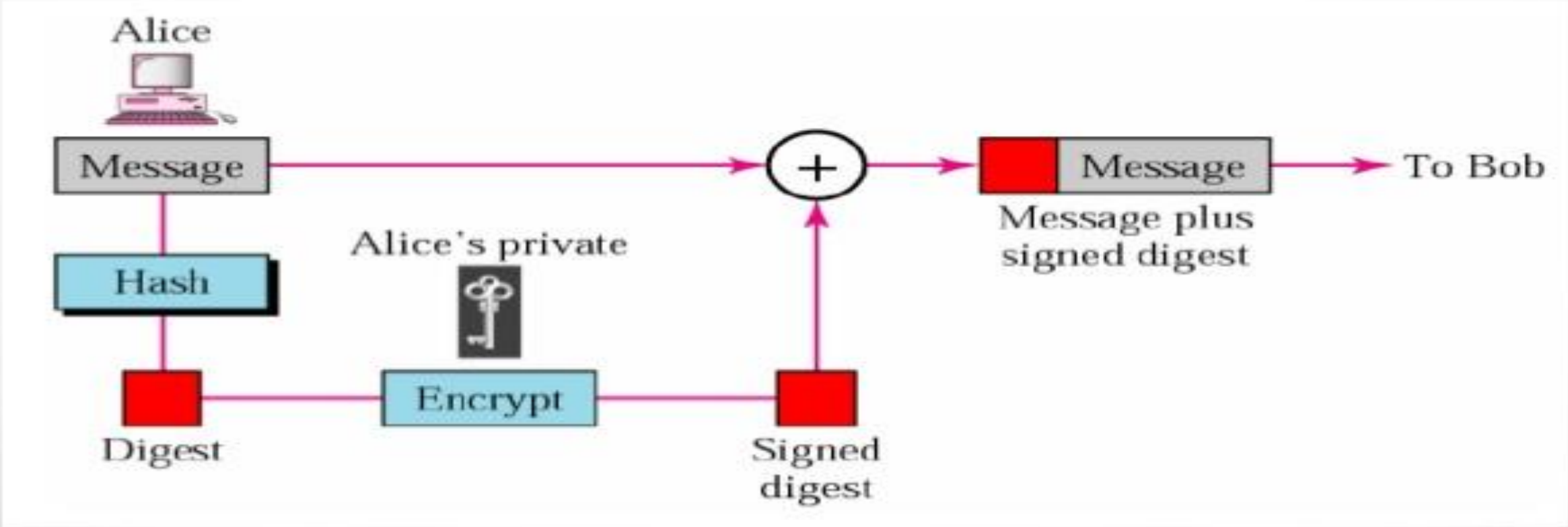
1. If X is the sender, the SHA-1 algorithm is used to first calculate the message digest (MD 1) of original message.
2. This MD1 is further encrypted using RSA with X's private key. This output is called the Digital Signature (DS) of X.
3. Further, the original message (M) along with the Digital signature (DS) is sent to receiver.

Receiver's Side:

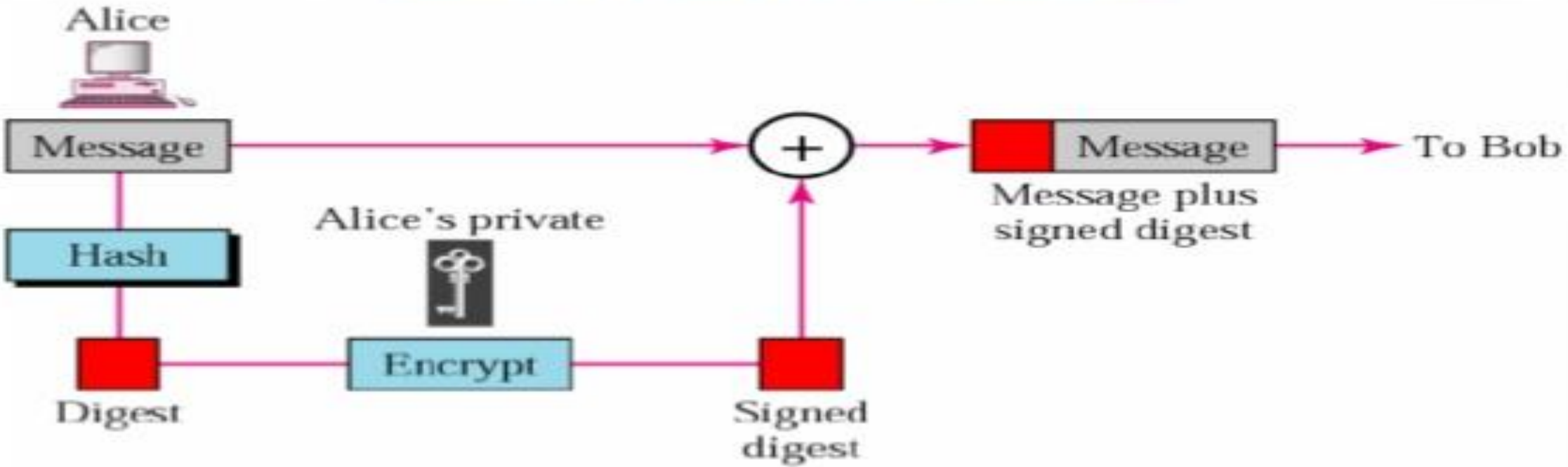
4. Y thus receives the original message (M) and X's digital signature. Y uses the same message digest algorithm used by X to calculate the message digest (MD2) of received message (M).
5. Also, Y uses X's public key to decrypt the digital signature. The outcome of this decryption is nothing but original message digest (MD1) calculated by X.
6. Y, then compares this digest MD1 with the digest MD2 he has just calculated in step 4. If both of them are matching, i.e. MDI = MD2, Y can accept the original message (M) as correctly authenticated and assured to have originated from X. whereas, if they are different, the message shall be rejected.

Digital Signatures

This method turns out to be foolproof. Even if an attacker intercepts anywhere in between, it is not likely for him to again sign the modified/read message, as only X in this case will know the private key! Hence, even if intercepted, this method remains very much secure and reliable!

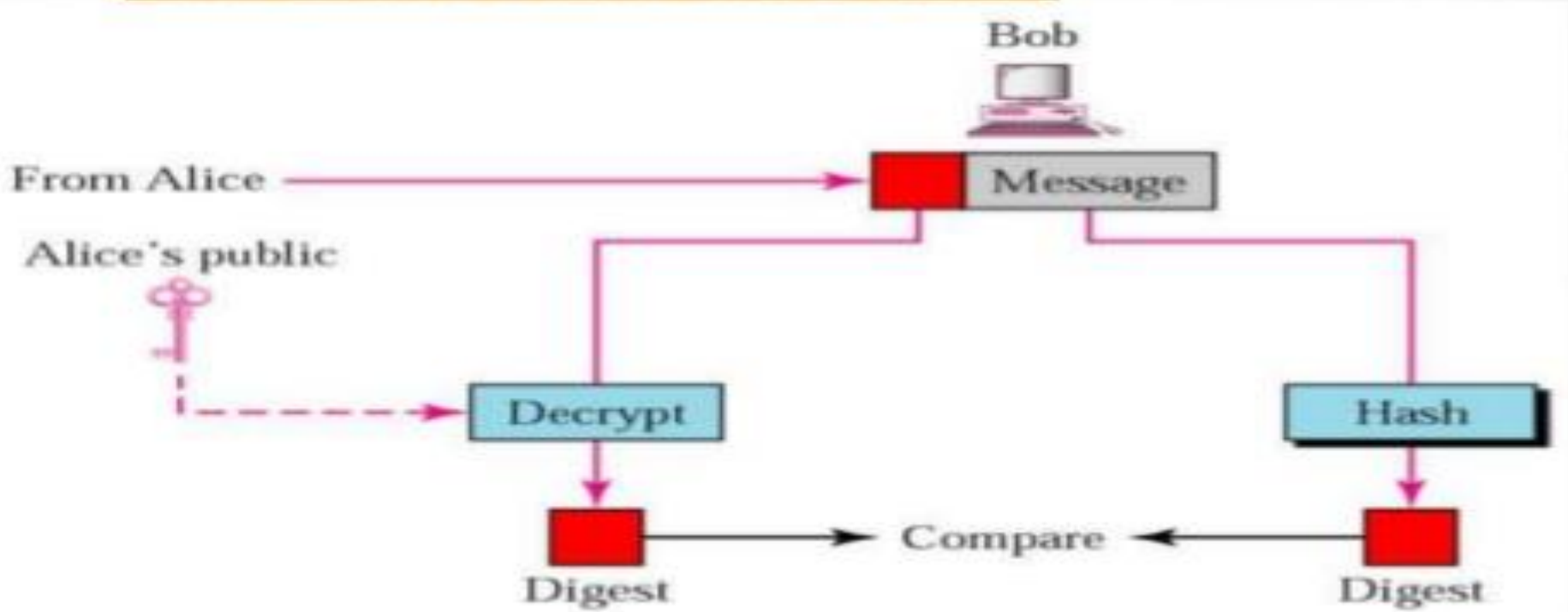


The Sender's Side



After the digest has been created, it is encrypted (signed) using the sender's private key. The encrypted digest is attached to the original message and sent to the receiver.

The Receiver's Side



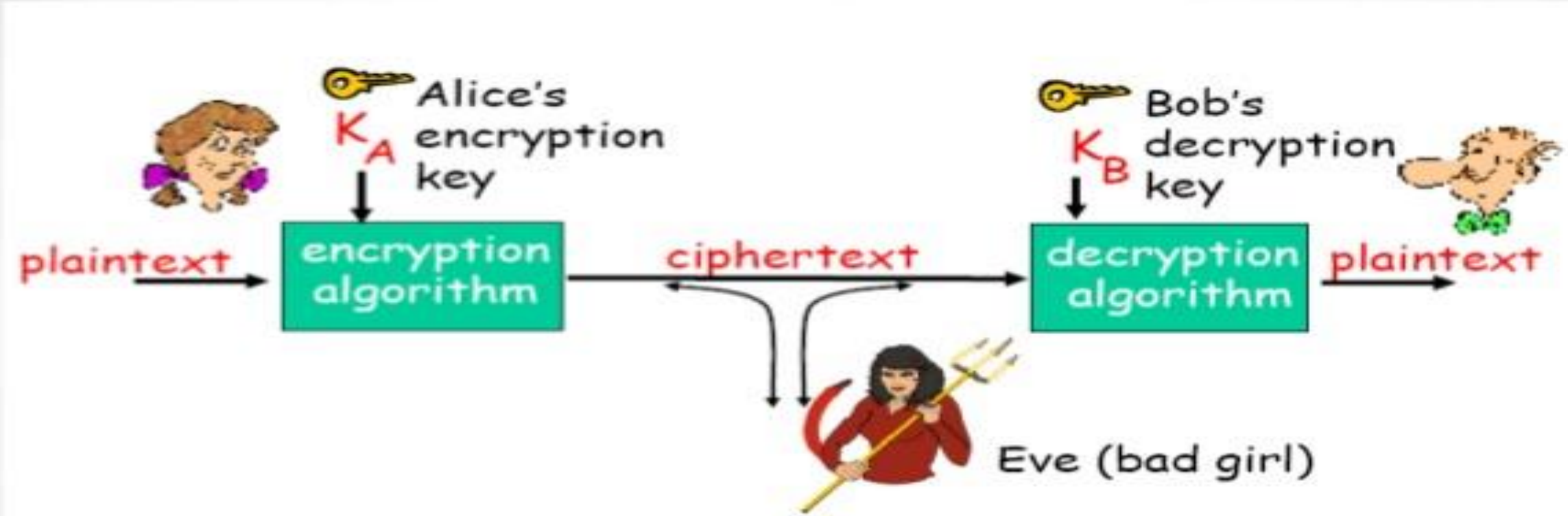
The receiver receives the original message and the encrypted digest. He separates the two. He applies the same hash function to the message to create a second digest. He also decrypts the received digest, using the public key of the sender. If the two digests are the same, all three security measures are preserved.

Properties of Digital Signatures

- Digital signature does not provide privacy. If there is a need for privacy, another layer of encryption/decryption must be applied.
- **Digital signatures can provide**

1. Integrity,
2. Authentication, and
3. Nonrepudiation.

Properties of Digital Signatures



K_A Alice's key

K_B Bob's key

Alice, Bob are two entities (person, process, client, server) that like to communicate. Eve is another entity which for eg. intercepts the communication.

Properties of Digital Signatures

1. Integrity The integrity of a message is preserved because if Eve intercepted the message and partially or totally changed it, the decrypted message would be unreadable.
2. Authentication We can use the following reasoning to show how a message can be authenticated. If Eve sends a message while pretending that it is coming from Alice, she must use her own private key for encryption. The message is then decrypted with the public key of Alice and will therefore be nonreadable. Encryption with Eve's private key and decryption with Alice's public key result in garbage.

Properties of Digital Signatures

3. Nonrepudiation Digital signature also provides for nonrepudiation. Bob saves the message received from Alice. If Alice later denies sending the message, Bob can show that encrypting and decrypting the saved message with Alice's private and public key can create a duplicate of the saved message. Since only Alice knows her private key, she cannot deny sending the message.