Digital Signatures

19CSE311 Computer Security

Jevitha KP

Department of CSE

Signature

- **Signature is proof to the recipient** that the document comes from the correct entity.
- **Signature** on a document, when verified, is a sign of **authentication,** the document is authentic.
- A person signs a document **to show that it originated** from her or was **approved** by her.
- Eg:
- Signature on a bank cheque - to verify that the customer issued it
- Signature on the art - to verify whether the painting is authentic

Digital Signature

- When Alice sends a message to Bob, Bob needs to check the authenticity of the sender; he needs to be sure that the message comes from Alice and not Eve.

- Bob can ask Alice to **sign the message electronically.**

- **An electronic signature (digital signature)** can prove the authenticity of Alice as the sender of the message.

Conventional vs Digital Signature - Inclusion

- A conventional signature is included in the document;
- It is **part of the document.**
- When we write a cheque, the **signature is on the check**; it is not a separate document.

- When we sign a document digitally, **we send the signature as a separate document**.
- The sender sends two documents: **the message and the signature.**
- The recipient receives both documents and **verifies that the signature belongs to the supposed sender.**
- If this is proven, the message is kept; otherwise, it is rejected.

Conventional vs Digital Signature - Verification Method

- For a conventional signature, when the recipient receives a document, she **compares the signature on the document with the signature on file.**
- If **they are the same**, the document is authentic.
- The **recipient needs to have a copy of this signature on file** for comparison.

- For a digital signature, the recipient **receives the message and the signature**.
- A **copy of the signature is not stored anywhere.**
- The recipient needs to apply a **verification technique** to the **combination of the message and the signature** to verify the **authenticity**

Conventional vs Digital Signature - Relationship

- For a conventional signature, there is normally a **one-to-many relationship** between a **signature and documents**.

- A person uses the **same signature to sign many documents.**

- For a digital signature, there is a **one-to-one relationship between a signature and a message.**

- **Each message** has its **own signature.**

- The **signature of one message cannot be used in another message.**

- If Bob receives two messages, one after another, from Alice, he cannot use the signature of the first message to verify the second.

- Each **message** needs a **new signature**

Conventional vs Digital Signature - Duplicity

- Another difference between the two types of signatures is a quality called **duplicity**.

- In conventional signature, a **copy of the signed document can be distinguished from the original one on file**.

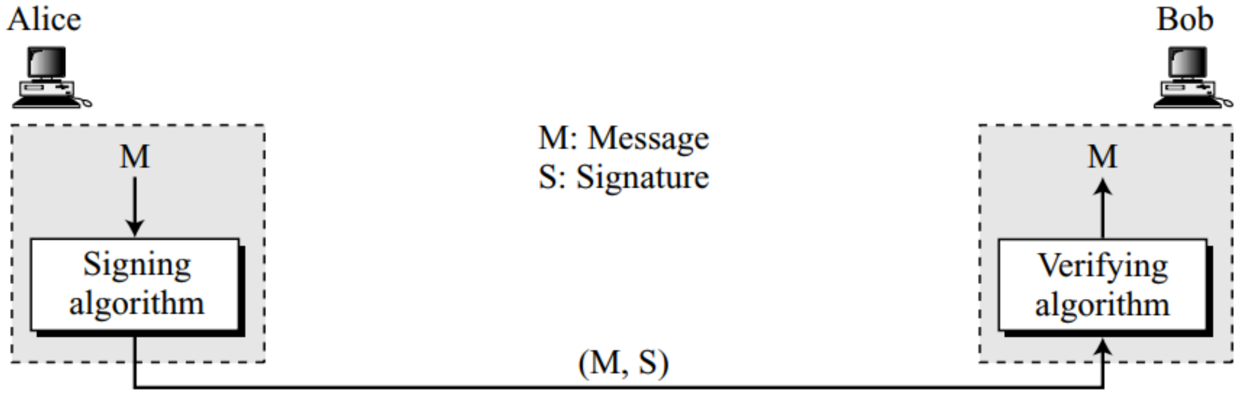
- In digital signature, **there is no such distinction** unless there is a **factor of time** (such as a timestamp) on the document.

- For example, suppose Alice sends a document instructing Bob to pay Eve.

- If Eve intercepts the document and the signature, she can replay it later to get money again from Bob.

Digital Signature Process

- The sender uses a **signing algorithm** to **sign the message.**
- The **message and the signature** are sent to the receiver.
- The **receiver receives the message and the signature** and applies the **verifying algorithm** to the combination.
- If the result is **true**, the **message** is **accepted**; otherwise, it is **rejected**.



Need for Keys

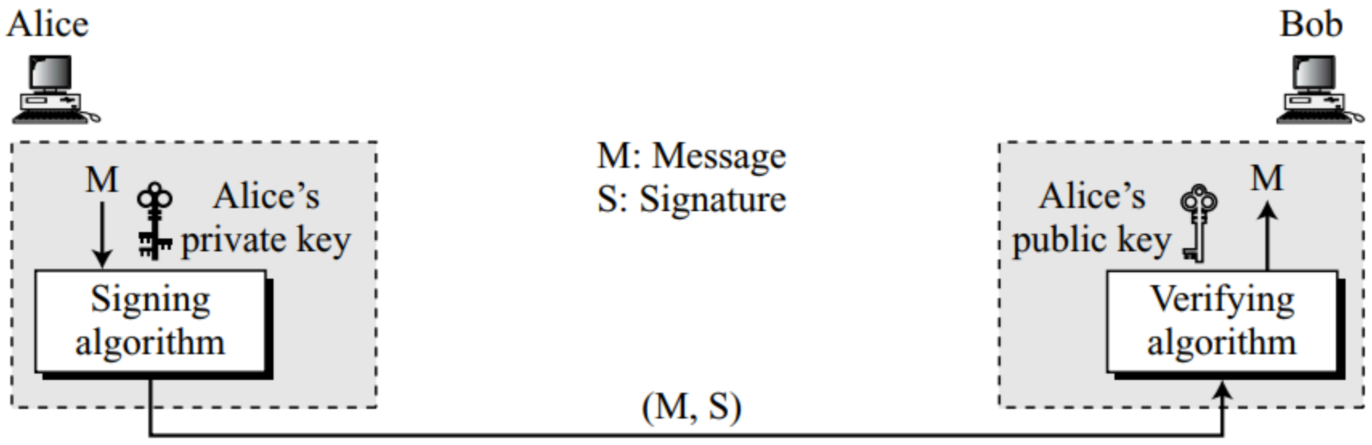
- A conventional signature is like a **private "key"** belonging to the **signer of the document**.

- The signer uses it to sign documents; no one else has this signature.

- The **copy of the signature is on file** like a **public key**; anyone can use it to verify a document, to compare it to the original signature.

Need for Keys

- In a digital signature, the **signer uses her private key**, applied to a **signing algorithm**, to **sign the document.**
- The verifier, uses the **public key of the signer,** applied to the **verifying algorithm**, to **verify the document.**
- When a document is signed, anyone can verify it because everyone has access to Alice's public key.
- Alice **must not use her public key to sign the document** because, **anyone could forge her signature.**



Symmetric keys for DS?

- Can we use a secret (symmetric) key to both sign and verify a signature?

Symmetric keys for DS?

- Can we use a secret (symmetric) key to both sign and verify a signature?

- No, because

    - A secret key is known by only two entities (Alice and Bob, for example). So if Alice needs to sign another document and send it to Ted, she needs to **use another secret key.**

    - Creating a secret key for a session involves **authentication**, which uses a **digital signature.** We have a vicious cycle

    - Bob could use the **secret key between himself and Alice**, sign a document, send it to Ted, and **pretend that it came from Alice**

Keys in Asym Crypto. vs DS

- We should make a distinction between private and public keys as used in **digital signatures** and public and private keys as used in a **cryptosystem** for **confidentiality**.

- In the cyptosystems, the **private and public keys of the receiver are used** in the process.

    - The sender uses the public key of the receiver to encrypt;

    - the receiver uses his own private key to decrypt.

- In a digital signature, the **private and public keys of the sender are used**.

    - The sender uses her private key to sign the document

    - the receiver uses the sender's public key to verify the signature

Signing the Digest

- The asymmetric-key cryptosystems are very **inefficient** when dealing with long messages.

- In a digital signature system, **the messages are normally long**, but we have to use asymmetric-key schemes.

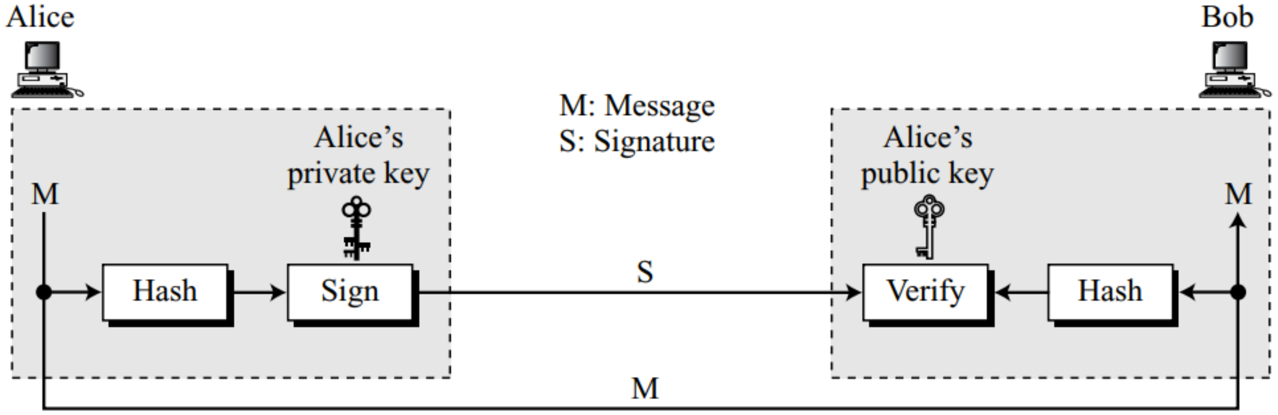
- The solution is to **sign a digest of the message**, which is much **shorter than the message.**

- A carefully selected **message digest** has a **one-to-one relationship** with the **message**.

- **The sender can sign the message digest and the receiver can verify the message digest.** The effect is the same.

Signing the Digest

- **Sender:**
  - A digest is made out of the message at Alice's site.
  - The digest then goes through the signing process using Alice's private key.
  - Alice then sends the message and the signature to Bob.
- **Receiver:**
  - At Bob's site, **using the same public hash function**, a digest is first created out of the received message.
  - Calculations are done on the signature and the digest.
  - The verifying process also **applies criteria on the result of the calculation t**o determine the **authenticity of the signature.**
  - If **authentic**, the message is **accepted**; otherwise, it is **rejected**



Services Provided by Digital Signatures

- Security Services provided by Digital Signature are **message authentication, message integrity, and nonrepudiation.**

- For **Message confidentiality** we still need **encryption/ decryption.**

Message Authentication

- This is also referred to as **data-origin authentication**

- A **secure digital signature scheme**, like a secure conventional signature (one that cannot be easily copied) can provide message authentication.

- Bob can verify that the message is sent by Alice because **Alice’s public key is used in verification.**

- Alice’s public key **cannot verify the signature signed by Eve’s private key**

Message Integrity

- The **integrity of the message is preserved even** if we sign the message digest, **because we cannot get the same digest if the message is changed.**
- The digital signature schemes today use a **hash function in the signing and verifying algorithms** that preserve the integrity of the message

Nonrepudiation

- If Alice signs a message and then denies it, can Bob later prove that Alice actually signed it?

- For example, if Alice sends a message to a bank (Bob) and asks to transfer $10,000 from her account to Ted's account, can Alice later deny that she sent this message?

- With the scheme we have presented so far, Bob might have a problem.

- Bob **must keep the signature on file** and later use Alice's public key to create the original message to prove the message in the file and the newly created message are the same.

- This is not feasible because **Alice may have changed her private or public key during this time**; she may also claim that the **file containing the signature is not authentic**

Solution

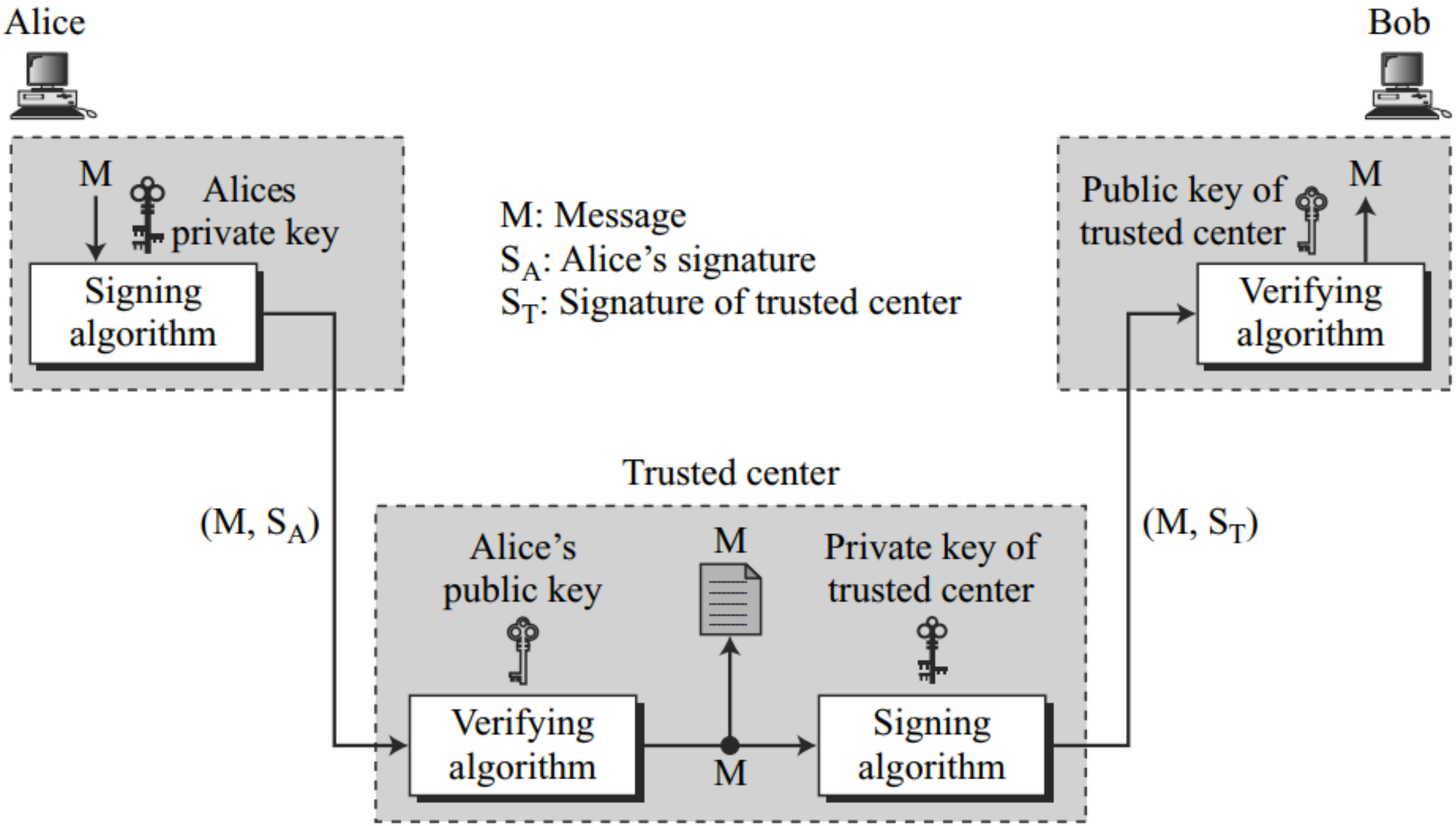
- One solution is a **trusted third party.**

- People can create an established trusted party among themselves.

- Trusted party can solve many other problems concerning security services and key exchange.

- A trusted party can prevent Alice from denying that she sent the message.

Nonrepudiation - TTP



Nonrepudiation - TTP

- Alice creates a signature from her message (S_A) and sends the message, her identity, Bob's identity, and the signature to the center.
- The center, after checking that Alice's public key is valid, verifies through Alice's public key that the message came from Alice.
- The center then saves a copy of the message with the sender identity, recipient identity, and a timestamp in its archive.
- The center uses its private key to create another signature (S_T) from the message.
- The center then sends the message, the new signature, Alice's identity, and Bob's identity to Bob.
- Bob verifies the message using the public key of the trusted center

Nonrepudiation - TTP

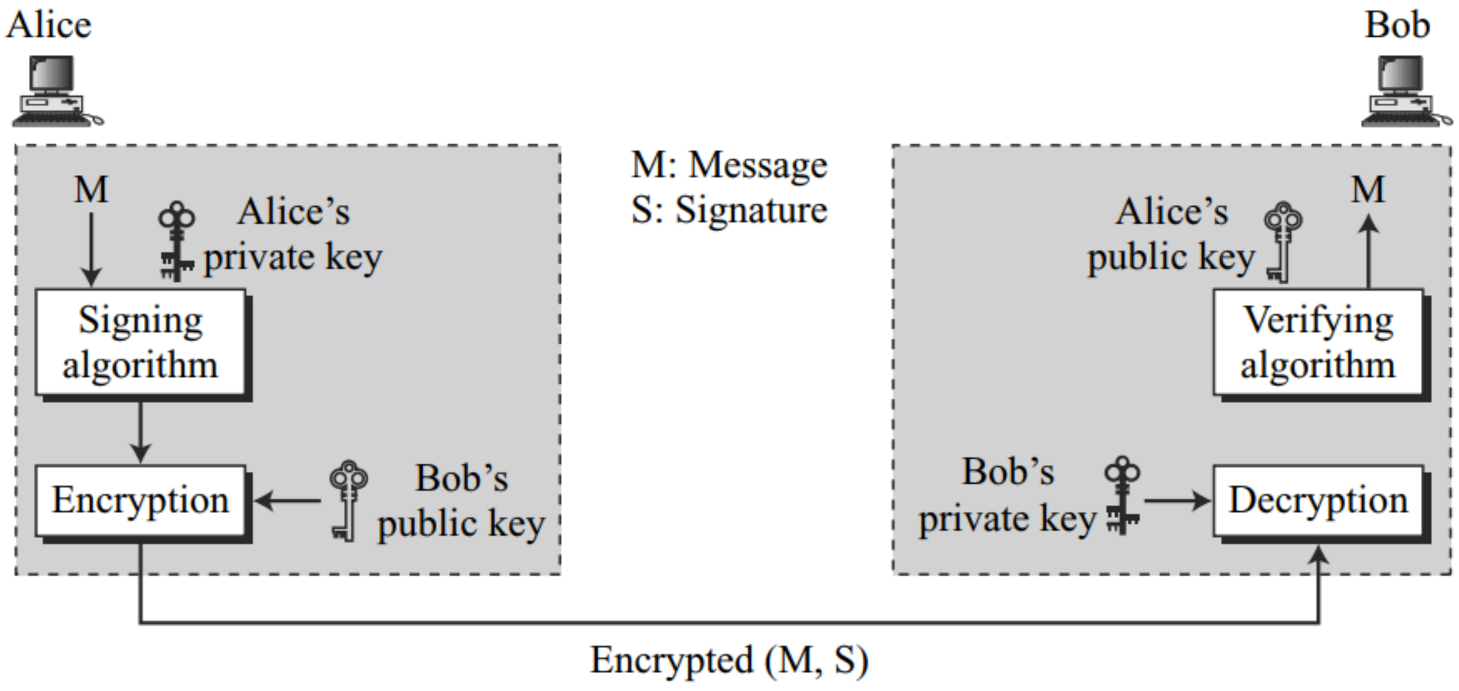
- If in the future Alice denies that she sent the message, the center can show a **copy of the saved message.**

- If Bob's message is a duplicate of the message saved at the center, Alice will lose the dispute.

- To make everything confidential, a level of encryption/ decryption can be added to the scheme

Confidentiality

- A digital signature does not provide confidential communication.
- If confidentiality is required, the message and the signature must be encrypted using either a **secret-key or public-key cryptosystem.**
- Asymmetric-key encryption/decryption just to emphasize the type of keys used at each end.
- Encryption/decryption can also be done with a symmetric key.



ATTACKS ON DIGITAL SIGNATURE

- Attack Types
- Forgery Types

Types of Attack

- Three kinds of attacks on digital signatures:
  - Key-Only Attack
  - Known-message Attack and
  - Chosen-message Attack

Types of Attack

- **Key-Only Attack**
- Eve has access **only to the public information** released by **Alice**.
- To forge a message, Eve needs to **create Alice's signature** to convince Bob that the message is coming from Alice.
- This is the same as the **ciphertext-only attack** in encipherment.

Types of Attack

- **Known-Message Attack**
- Eve has access to **one or more message-signature pairs**.
- In other words, she has **access to some documents previously signed by Alice**.
- Eve tries to **create another message and forge Alice's signature on it.**
- This is similar to the **known-plaintext attack** in encipherment.

Types of Attack

- **Chosen-Message Attack**

- Eve somehow makes **Alice sign one or more messages for her.**

- Eve now has a **chosen-message/signature pair.**

- Eve later creates another message, with the content she wants, and forges Alice's signature on it.

- This is similar to the **chosen-plaintext attack** in encipherment.

Types of Forgery

- If the attack is successful, the result is a forgery.
- We can have two types of forgery:
  - Existential Forgery
  - Selective Forgery

Types of Forgery

- **Existential Forgery**
- In an existential forgery, Eve may be able to create a **valid message-signature pair**, but not one that she can really use.
- In other words, **a document has been forged, but the content is randomly calculated.**
- This type of forgery is **probable**, but fortunately Eve cannot benefit from it very much.
- Her **message** could be **syntactically or semantically unintelligible.**

Types of Forgery

- **Selective Forgery**

- In selective forgery, Eve may be able to forge Alice’s signature on a message with the content selectively chosen by Eve.

- Although this is beneficial to Eve, and may be very detrimental to Alice, the **probability of such forgery is low,** but not negligible.

DIGITAL SIGNATURE SCHEMES

- There are several digital signature schemes have evolved during the last few decades:
  - **RSA Digital Signature Scheme**
  - ElGamal Digital Signature Scheme
  - Schnorr Digital Signature Scheme
  - Digital Signature Standard (DSS)
  - Elliptic Curve Digital Signature Scheme
- We will focus only on RSA Digital Signature scheme

RSA Digital Signature Scheme

- The RSA scheme can be used for signing and verifying a message, called the RSA digital signature scheme.

- The digital signature scheme **changes the roles of the private and public keys**.

  - The **private and public keys of the sender**, not the receiver, **are used**.

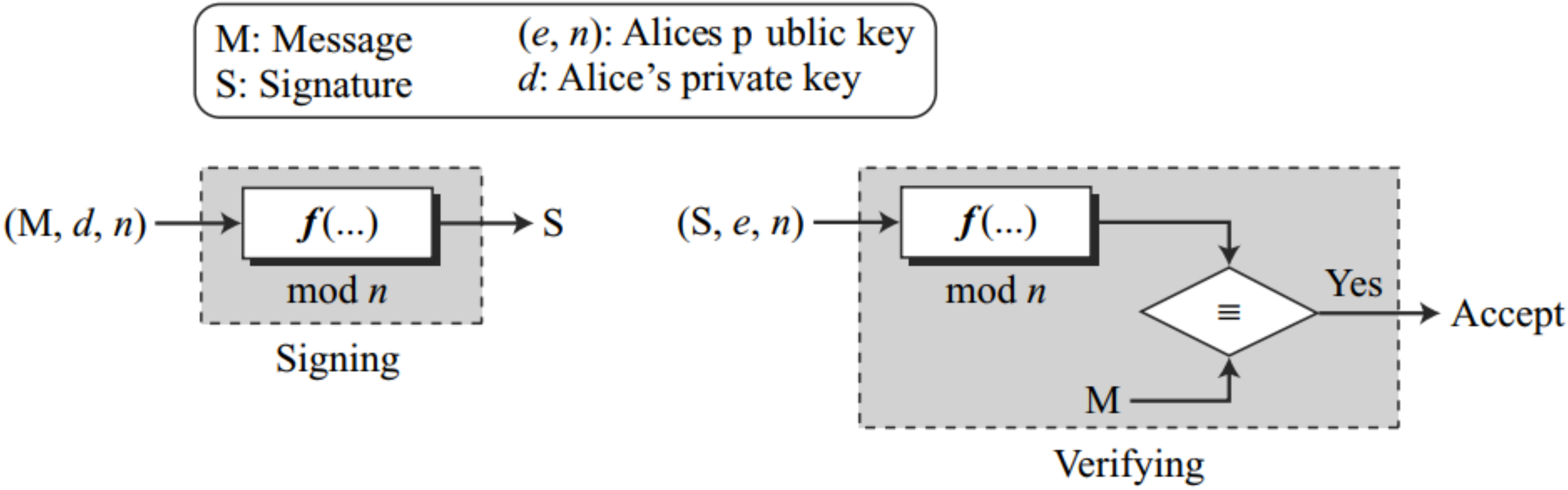
  - The sender uses **her own private key to sign the document**; the receiver uses the **sender's public key to verify it.**

- If we compare the scheme with the conventional way of signing, we see that the **private key plays the role of the sender's own signature**, the **sender's public key plays the role of the copy of the signature that is available to the public.**

- Obviously, **Alice cannot use Bob's public key to sign the message** because then any other person could do the same.

RSA DSS

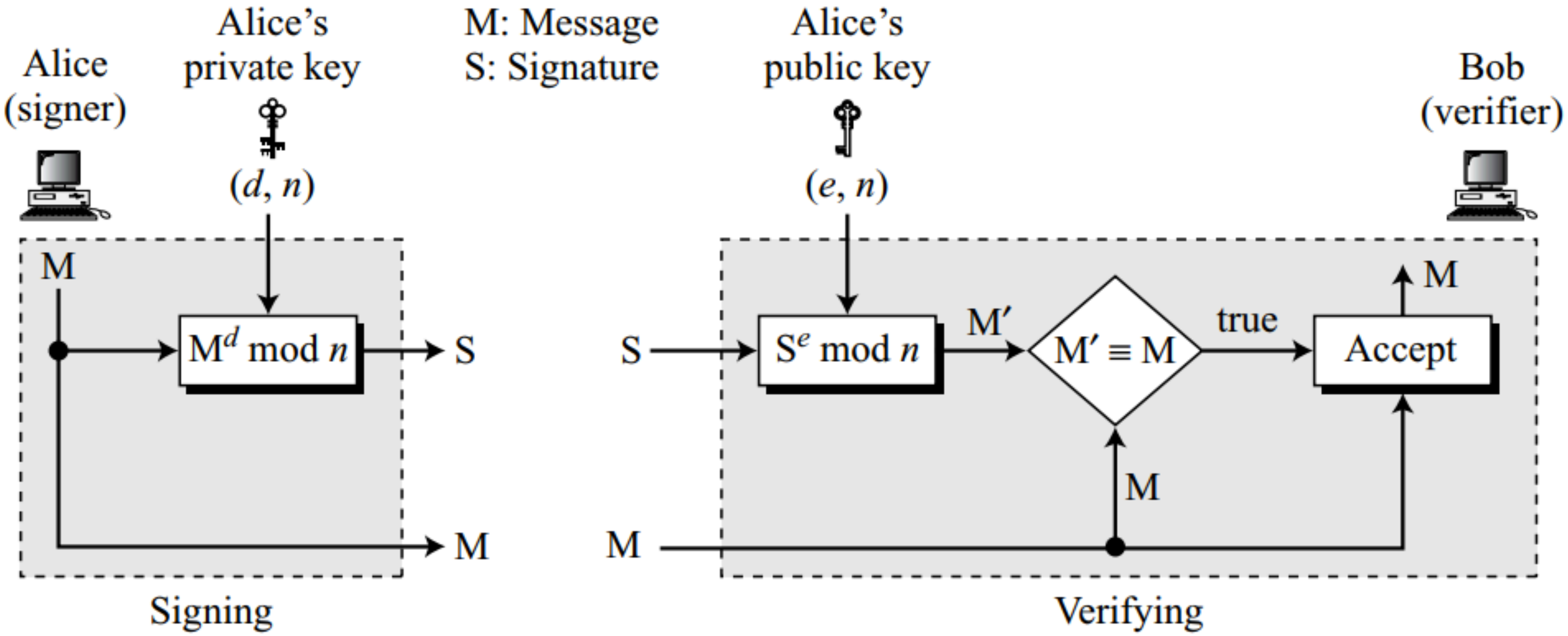
- The signing and verifying sites **use the same function,** but with **different parameters.**
- The **verifier compares the message and the output of the function** for congruence.
- If the **result is true, the message is accepted.**



RSA DSS - Key Generation

- Key generation in the RSA digital signature scheme is exactly the same as key generation in the RSA cryptosystem.
- Alice chooses two primes p and q and calculates **n = p × q.**
- Alice calculates **ϕ(n) = (p − 1) (q − 1).**
- She then chooses e, the public exponent, and calculates d, the private exponent such that **e × d = 1 mod ϕ(n).**
- Alice keeps d; **she publicly announces n and e.**
- **d is private key ; e and n are public key.**

RSA DSS - Signing and Verifying



RSA DSS - Signing and Verifying

- **Signing :**
  - Alice creates a signature out of the message using her private exponent, **$S = M^d \bmod n$**
  - Sends the message(M) and the signature(S) to Bob.
- **Verifying :**
  - Bob receives M and S.
  - Bob applies Alice’s public exponent to the signature to create a copy of the message **$M' = S^e \bmod n$**.
  - Bob compares the value of M′ with the value of M.
  - If the two values are congruent, Bob accepts the message.
- **Proof :** As per Euler's theorem, $d \times e = 1 \bmod \phi(n)$:

$$M' \equiv M \pmod{n} \quad \rightarrow \quad S^e \equiv M \pmod{n} \quad \rightarrow \quad M^{d \times e} \equiv M \pmod{n}$$

Example - Key Generation

- If p = 823 and q = 953,
- n = 784319.
- The value of ϕ(n) is 782544.
- Let e = 313 => d = 160009
- Public key = (313, 784319)
- Private Key = (160009, 784319)

Example - Signing and Verification

- **Signing**
  - If Alice wants to send a message with the value of M = 19070 to Bob.
  - She uses her private exponent, 160009, to sign the message:

$$\text{M: } 19070 \quad \rightarrow \quad \text{S} = (19070^{160009}) \bmod 784319 = 210625 \bmod 784319$$

  - Alice sends the message and the signature to Bob.
- **Verification**
  - Bob verifies the signature, and accepts the message since the signature is valid.

$$\text{M}' = 210625^{313} \bmod 784319 = 19070 \bmod 784319 \quad \rightarrow \quad \text{M} \equiv \text{M}' \bmod n$$

Attacks on RSA Signature

- There are some attacks that Eve can apply to the RSA digital signature scheme to forge Alice’s signature.

- **Key-Only Attack**

  - Eve has access only to Alice’s public key.

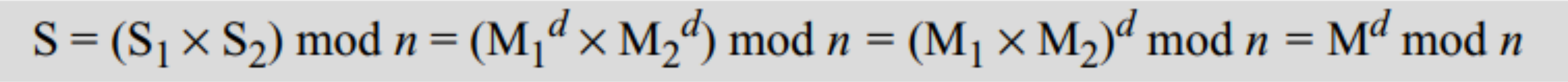
  - Eve intercepts the pair (M, S) and tries to create another message M′ such that **$M' \equiv S^e (\mathrm{mod}\ n)$**.

  - This problem is as difficult to solve as the discrete logarithm problem we

  - This is an existential forgery and normally is useless to Eve.

Attacks on RSA Signature

- **Known-Message Attack**
- Eve uses the multiplicative property of RSA.
- Assume that Eve has intercepted **two message-signature pairs (M1, S1) and (M2, S2)** that have been created using the **same private key.**
- If M = (M1 × M2) mod n, then S = (S1 × S2) mod n.
- This is simple to prove because we have



$$S = (S_1 \times S_2) \bmod n = ({M_1}^d \times {M_2}^d) \bmod n = (M_1 \times M_2)^d \bmod n = M^d \bmod n$$

Attacks on RSA Signature

- Eve can create M = (M1 × M2) mod n, and she can create S = (S1 × S2) mod n, and fool Bob into believing that S is Alice’s signature on the message M.

- This attack, which is sometimes referred to as multiplicative attack, is easy to launch.

- This is an **existential forgery** as the message M is a multiplication of two previous messages created by Alice, not Eve; M is normally useless

Attacks on RSA Signature

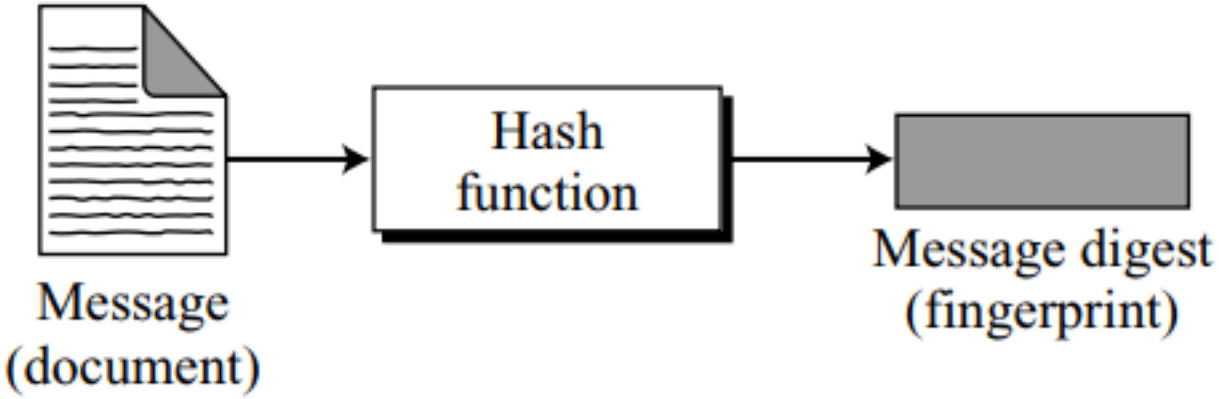
- **Chosen-Message Attack**
  - This attack also uses the **multiplicative property of RSA.**
  - Eve can somehow ask Alice to sign two legitimate messages, M1 and M2, for her and later creates a new message **M = M1 × M2.**
  - Eve can later claim that Alice has signed M.
  - The attack is also referred to as multiplicative attack.
  - This is a **very serious attack on the RSA digital signature scheme because it is a selective forgery** (Eve can manipulate M1 and M2 to get a useful M).

Cryptographic Hash Functions

- Cryptographic hash functions that are used to create a message digest from a message.

- The function creates a compressed image of the message that can be used like a fingerprint

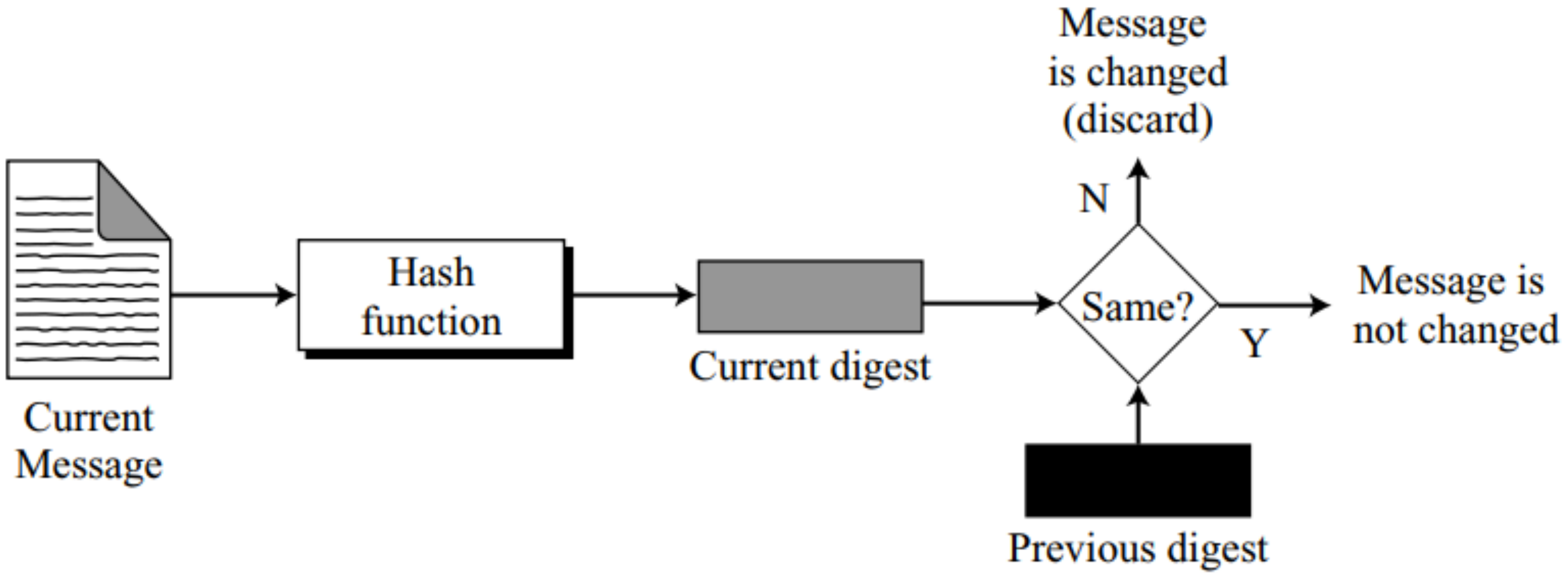
- Message digests guarantee the integrity of the message.



Cryptographic Hash Functions

- To check the integrity of a message, or document, we run the cryptographic hash function again and compare the new message digest with the previous one.

- If both are the same, we are sure that the original message has not been changed.

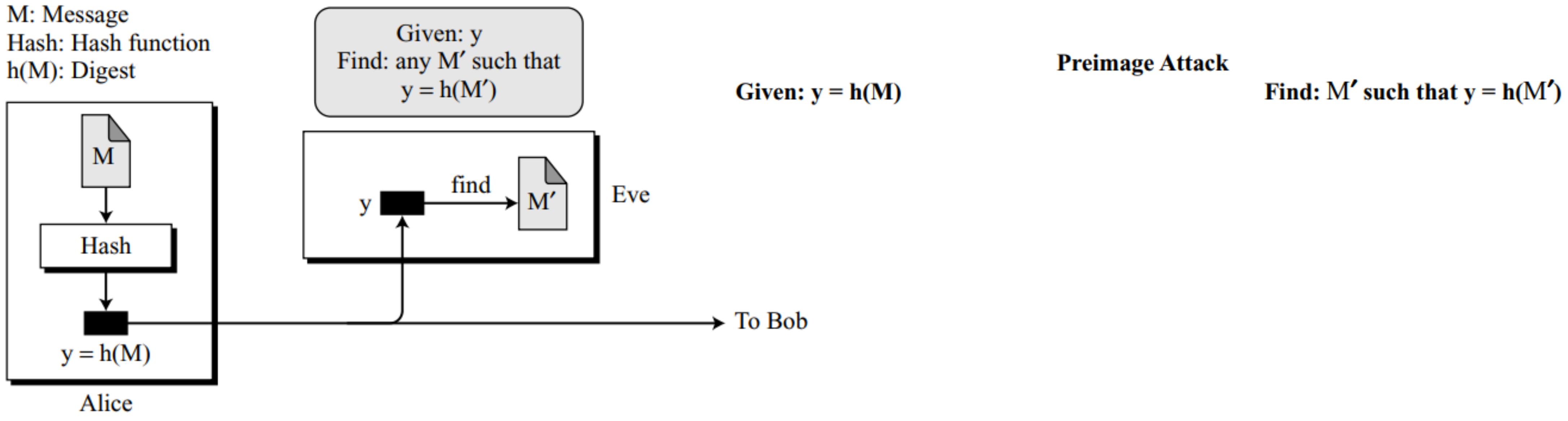


Cryptographic Hash Function Criteria

- A cryptographic hash function must satisfy three criteria:
  - Preimage resistance,
  - Second preimage resistance, and
  - Collision resistance

Cryptographic Hash Function Criteria

- **Preimage Resistance**
  - A cryptographic hash function must be **preimage resistant**.
  - Given a **hash function h and the hash of a message y = h(M)**, it must be **extremely difficult for Eve to find any message, M′, such that y = h(M′).**
  - If the hash function is not preimage resistant, Eve can intercept the digest h(M) and create a message M′.
  - Eve can then send M′ to Bob pretending it is M.



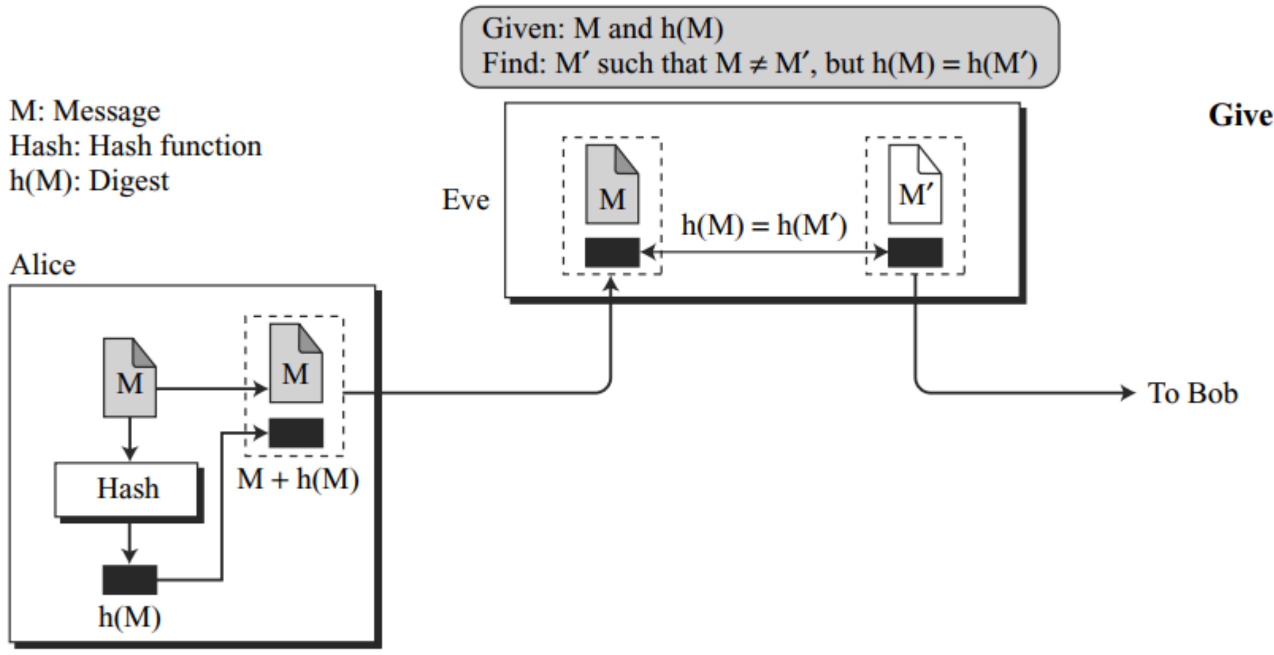
Cryptographic Hash Function Criteria

- **Second Preimage Resistance**

- This ensures that a message cannot easily be forged.

- If Alice creates a message(M) and a digest(H) and sends both to Bob, this criterion ensures that Eve cannot easily create another message(M') that hashes to the exact same digest(H).

- In other words, given a specific message and its digest, it is impossible (or at least very difficult) to create another message with the same digest.

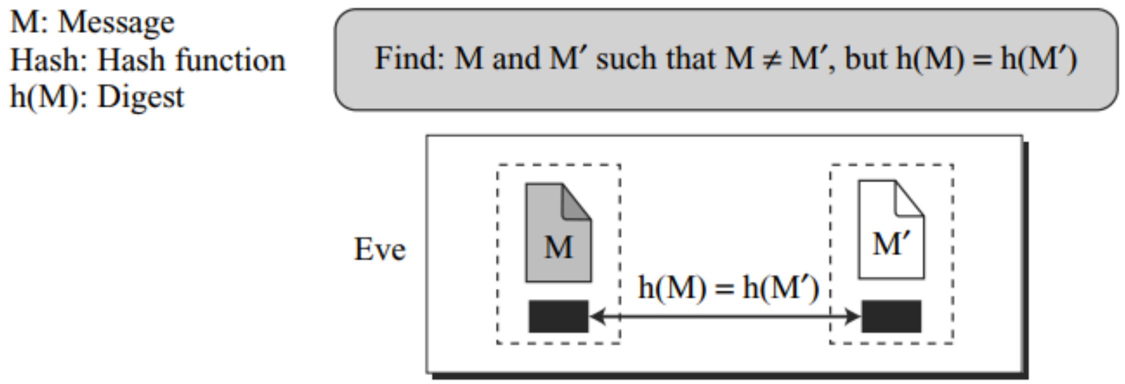


Second Preimage Attack

Given: M and h(M) **Find: M′ ≠ M such that h(M) = h(M′)**

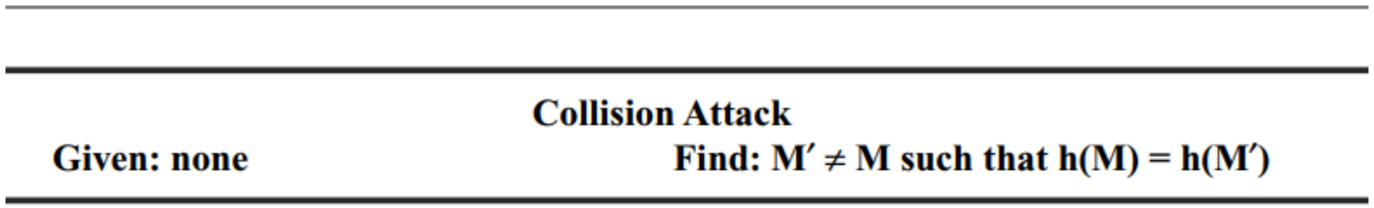
Cryptographic Hash Function Criteria

- **Collision Resistance**
- This ensures that Eve cannot find two messages that hash to the same digest.
- Here the adversary can create two messages (out of scratch) and hashed to the same digest.
- Eg: If two different wills can be created that hash to the same digest. When the time comes for the execution of the will, the second (forged) will is presented to the heirs.
- Because the digest matches both wills, the substitution is undetected.
- This type of attack is much easier to launch than the two previous kinds.
- In other words, we need particularly be sure that a hash function is collision resistant.



M: Message
Hash: Hash function
h(M): Digest

Find: M and M′ such that $M \neq M'$, but $h(M) = h(M')$



Collision Attack	
Given: none	Find: $M' \neq M$ such that $h(M) = h(M')$

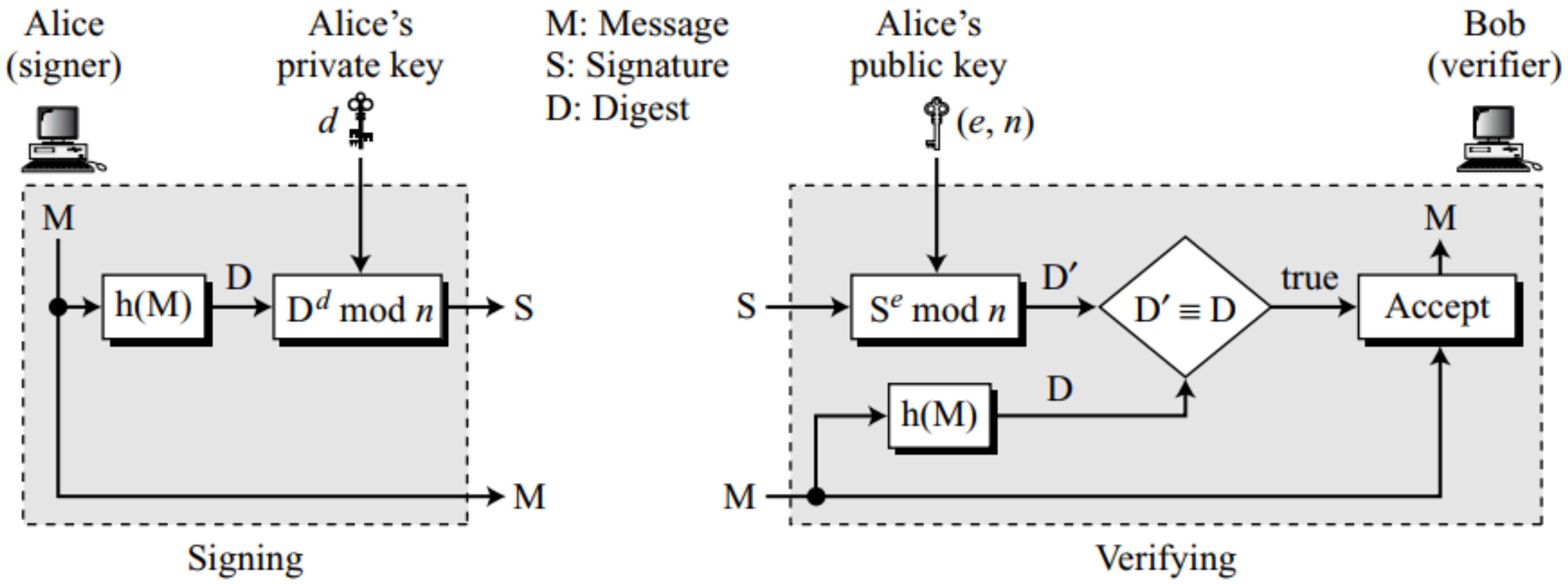
RSA Signature on the Message Digest

- Signing a message digest using a **strong hash algorithm** has several advantages.

- In the case of RSA, it can make the signing and verifying processes much faster because the RSA digital signature scheme is just **encryption with the private key and decryption with the public key.**

- The use of a strong cryptographic hashing function also makes the attack on the signature much more difficult.

RSA Signature on the Message Digest



RSA Signature on the Message Digest

- Alice, the signer, first uses an agreed-upon hash function to create a digest from the message, **H = hash(M).**
- She then signs the digest, **$S = H^d \bmod n$.**
- The message and the signature are sent to Bob.
- Bob, the verifier, receives the message and the signature.
- He first uses Alice's public exponent to retrieve the digest, **$H' = S^e \bmod n$.**
- He then applies the hash algorithm to the message received to obtain **H = hash(M).**
- Bob now compares the two digests, H and H′.
- If they are **congruent to modulo n,** he accepts the message.

Attacks on RSA Signed Digests

- **Key-Only Attack** : We can have three cases of this attack:
  - Eve intercepts the pair (S, M) and tries to find another message M′ that creates the same digest, h(M) = h(M′). If the hash algorithm is second preimage resistant, this attack is very difficult.
  - Eve finds two messages M and M′ such that h(M) = h(M′). She lures Alice to sign h(M) to find S. Now Eve has a pair (M′, S) which passes the verifying test, but it is the forgery. If the hash algorithm is collision resistant, this attack is very difficult.
  - Eve may randomly find message digest D, which may match with a random signature S. She then finds a message M such that D = h(M). If the hash function is preimage resistant, this attack is very difficult to launch

Attacks on RSA Signed Digests

- **Known-Message Attack**
  - Let us assume Eve has two message-signature pairs (M1, S1) and (M2, S2) which have been created using the same private key.
  - Eve calculates $S \equiv S1 \times S2$. If she can find a message M such that $h(M) \equiv h(M1) \times h(M2)$, she has forged a new message.
  - However, finding M given h(M) is very difficult if the hash algorithm is **preimage resistant.**

Attacks on RSA Signed Digests

- **Chosen-Message Attack**

- Eve can ask Alice to sign two legitimate messages M1 and M2 for her.

- Eve then creates a new signature S ≡ S1 × S2.

- Since Eve can calculate h(M) ≡ h(M1) × h(M2), if she can find a message M given h(M), the new message is a forgery.

- However, finding M given h(M) is very difficult if the hash algorithm is **preimage resistant.**

Attacks on RSA Signed Digests

- When the digest is signed instead of the message itself, the susceptibility of the RSA digital signature scheme depends on the **strength of the hash algorithm**