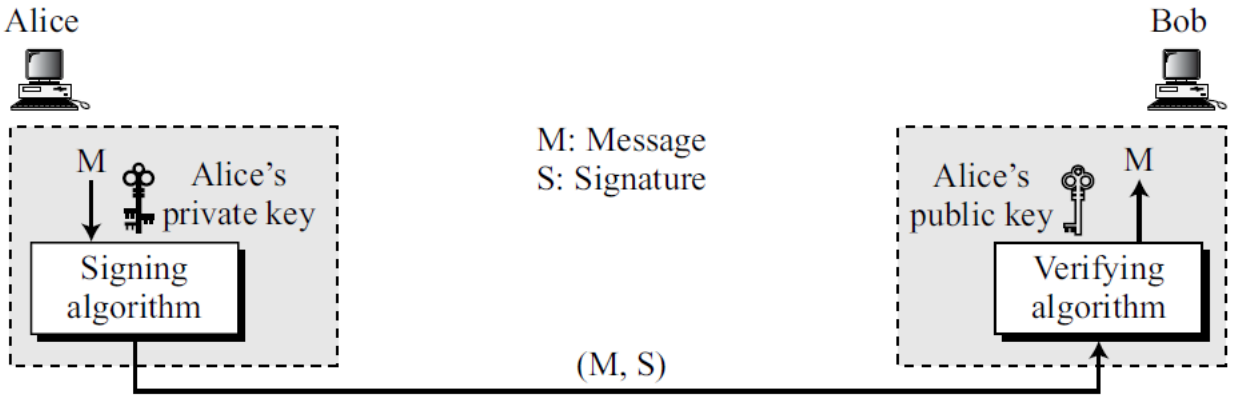
Network Security

Digital Signature

Kamalika Bhattacharjee

Digital Signature

- The most important development from the work on public-key cryptography
- provides a set of security capabilities that would be difficult to implement in any other way.



- 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

Digital Signature vs Cryptosystem

- Can we use a secret (symmetric) key to both sign and verify a signature?
- No!
- Because:
  - First, 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.
  - Second, creating a secret key for a session involves authentication, which uses a digital signature. We have a vicious cycle.
  - Third, Bob could use the secret key between himself and Alice, sign a document, send it to Ted, and pretend that it came from Alice

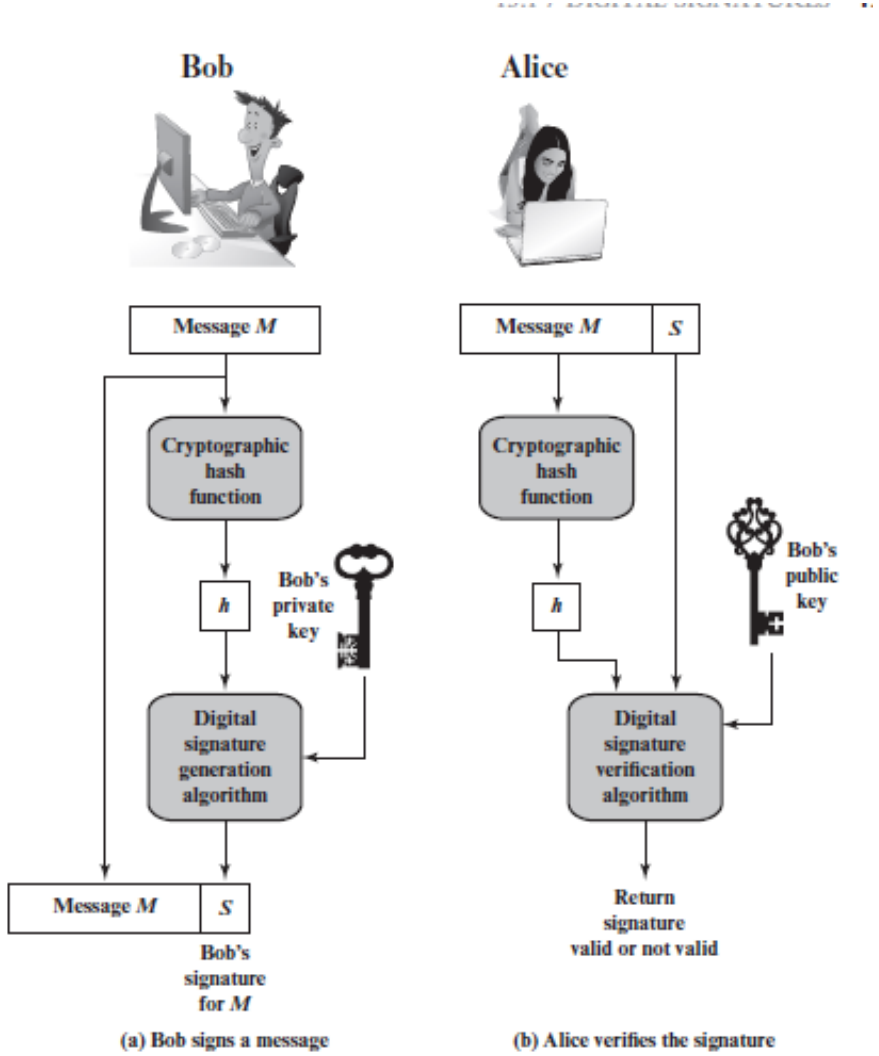
➢ A cryptosystem uses the private and public keys of the receiver: a digital signature uses the private and public keys of the sender.

Signing the Digest

- 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.
- Solution:
  - 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.
  - The digest is first created out of the received message using same public hash function

Digital Signature Process

- When Alice receives the message plus signature, she (1) calculates a hash value for the message; (2) provides the hash value and Bob's public key as inputs to a digital signature verification algorithm.
- If the algorithm returns the result that the signature is valid, Alice is assured that the message must have been signed by Bob.
  - No one else has Bob's private key and therefore no one else could have created a signature that could be verified for this message with Bob's public key.
  - In addition, it is impossible to alter the message without access to Bob's private key, so the message is authenticated both in terms of source and in terms of data integrity



Properties

- Message authentication protects two parties who exchange messages from any third party. However, it does not protect the two parties against each other.
- Disputed situations between J & M while using Message Encryption:
  - M may forge a different message and claim that it came from J. M would simply have to create a message and append an authentication code using the key that J and M share.
  - J can deny sending the message. Because it is possible for M to forge a message, there is no way to prove that J did in fact send the message.
- In situations where there is not complete trust between sender and receiver, something more than authentication is needed. Solution: digital signature.
- The digital signature must have the following properties:
  - It must verify the author and the date and time of the signature.
  - It must authenticate the contents at the time of the signature.
  - It must be verifiable by third parties, to resolve disputes.

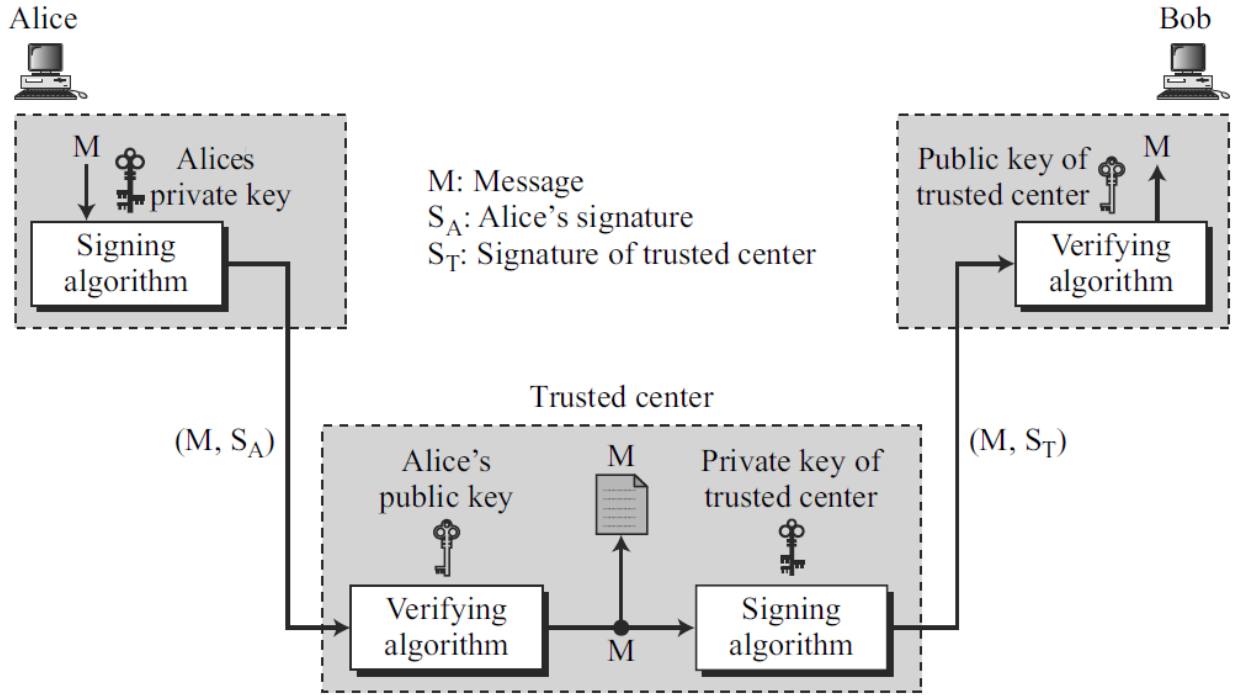
Services

- Message Authentication
  - A secure digital signature scheme, like a secure conventional signature (one that cannot be easily copied) can provide message authentication (also referred to as data-origin authentication).

- Message Integrity
  - The integrity of the message is preserved even if we sign the whole message because we cannot get the same signature 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

Services

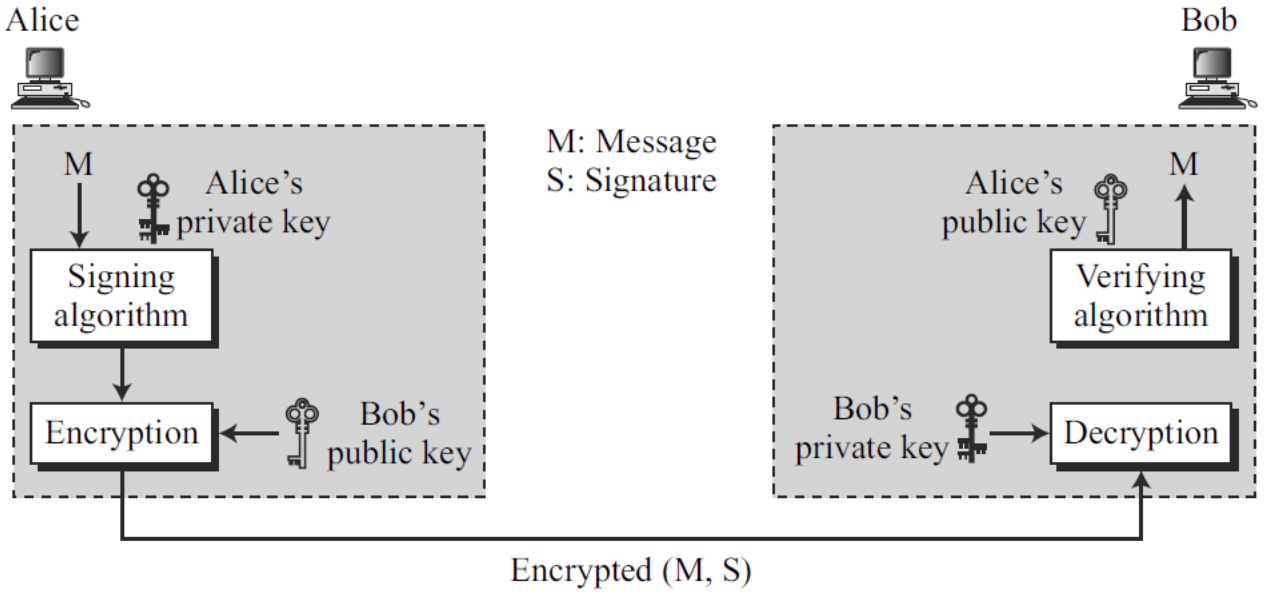
- Nonrepudiation: Need a trusted Third Party



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

Services

- Confidentiality: 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

Attacks and Forgeries

- Let A denotes the user whose signature method is being attacked, and C denotes the attacker.
- **Key-only attack:** C only knows A's public key -- same as the *ciphertext-only attack*
- **Known message attack:** C is given access to a set of messages and their signatures. Similar to *known-plaintext attack*
- **Chosen Message Attack:** Similar to chosen-plaintext attack
  - **Generic chosen message attack:** C chooses a list of messages before attempting to breaks A's signature scheme, independent of A's public key. C then obtains from A's valid signatures for the chosen messages. The attack is generic, because it does not depend on A's public key; the same attack is used against everyone.

Attacks and Forgeries

- Let A denotes the user whose signature method is being attacked, and C denotes the attacker.
- **Chosen Message Attack**
  - **Directed chosen message attack:** Similar to the generic attack, except that the list of messages to be signed is chosen after C knows A's public key but before any signatures are seen.
  - **Adaptive chosen message attack:** C is allowed to use A as an "oracle." This means that C may request from A signatures of messages that depend on previously obtained message-signature pairs.

Attacks and Forgeries

- If the attack is successful, the result is a forgery
- Success at breaking a signature scheme as an outcome in which C can do any of the following with a non-negligible probability:
  - **Total break:** C determines A's private key.
  - **Universal forgery:** C finds an efficient signing algorithm that provides an equivalent way of constructing signatures on arbitrary messages.
  - **Selective forgery:** C forges a signature for a particular message chosen by C.
  - **Existential forgery:** C forges a signature for at least one message. C has no control over the message. Consequently, this forgery may only be a minor nuisance to A. Here, a document has been forged, but the content is randomly calculated.

Digital Signature Requirements

- The signature must be a bit pattern that depends on the message being signed.
- The signature must use some information only known to the sender to prevent both forgery and denial.
- It must be relatively easy to produce the digital signature.
- It must be relatively easy to recognize and verify the digital signature.
- It must be computationally infeasible to forge a digital signature, either by constructing a new message for an existing digital signature or by constructing a fraudulent digital signature for a given message.
- It must be practical to retain a copy of the digital signature in storage

Digital Signature Requirements

- The signature must be a bit pattern that depends on the message being signed.
- The signature must use some information only known to the sender to prevent both forgery and denial.
- It must be relatively easy to produce the digital signature.
- It must be relatively easy to recognize and verify the digital signature.
- It must be computationally infeasible to forge a digital signature, either by constructing a new message for an existing digital signature or by constructing a fraudulent digital signature for a given message.
- It must be practical to retain a copy of the digital signature in storage

  - A secure hash function provides a basis for satisfying these requirements. However, care must be taken in the design of the details of the scheme.

Direct Digital Signature

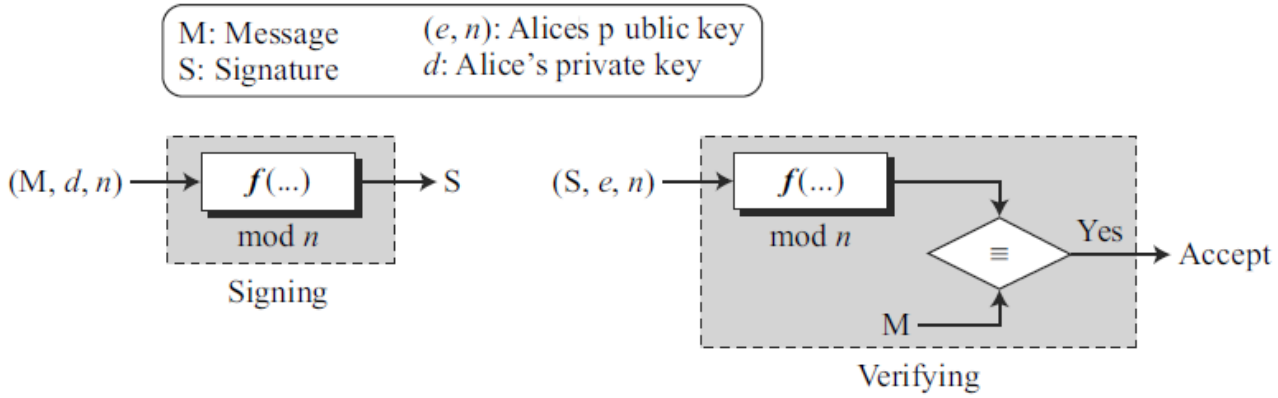
- Refers to a digital signature scheme that involves only the communicating parties (source, destination).
- It is assumed that the destination knows the public key of the source.
- Confidentiality can be provided by encrypting the entire message plus signature with a shared secret key (symmetric encryption).
  - it is important to perform the signature function first and then an outer confidentiality function.
  - In case of dispute, some third party must view the message and its signature. If the signature is the inner operation, then the recipient can store the plaintext message and its signature for later use in dispute resolution.
- The validity of the scheme depends on the security of the sender's private key.

Direct Digital Signature

- If a sender later wishes to deny sending a particular message, the sender can claim that the private key was lost or stolen and the signature is forged.
  - Administrative controls relating to the security of private keys can be employed to thwart or at least weaken this ploy, but the threat is still there,
  - Example is to require every signed message to include a **timestamp** (date and time) and to require prompt reporting of compromised keys to a central authority.
- Another threat: a private key might actually be stolen from X at time T.
  - The opponent can then send a message signed with X's signature and stamped with a time before or equal to T.
  - The universally accepted technique for dealing with these threats is the use of a digital certificate and certificate authorities.

RSA Digital Signature Scheme

- The digital signature scheme changes the roles of the private and public keys.
  - First, the private and public keys of the sender, not the receiver, are used.
  - Second, the sender uses her own private key to sign the document; the receiver uses the sender's public key to verify it.



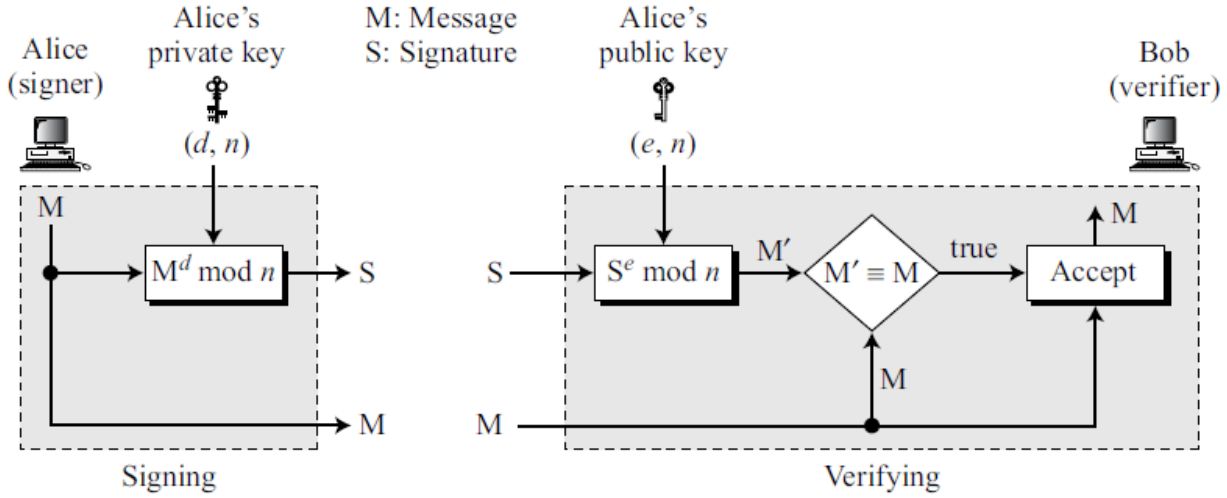
General idea behind the RSA digital signature scheme

  - 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 Digital Signature Scheme

- **Key Generation**
  - Key generation in the RSA digital signature scheme is exactly the same as key generation in the RSA cryptosystem
  - Choose two primes p and q and calculate $n = p \times q$.
  - Calculates $\phi(n) = (p - 1)(q - 1)$.
  - Chooses e, the public exponent, and calculate d, the private exponent such that $e \times d = 1 \bmod \phi(n)$.
  - Keep d; publicly announce n and e.

RSA digital signature scheme



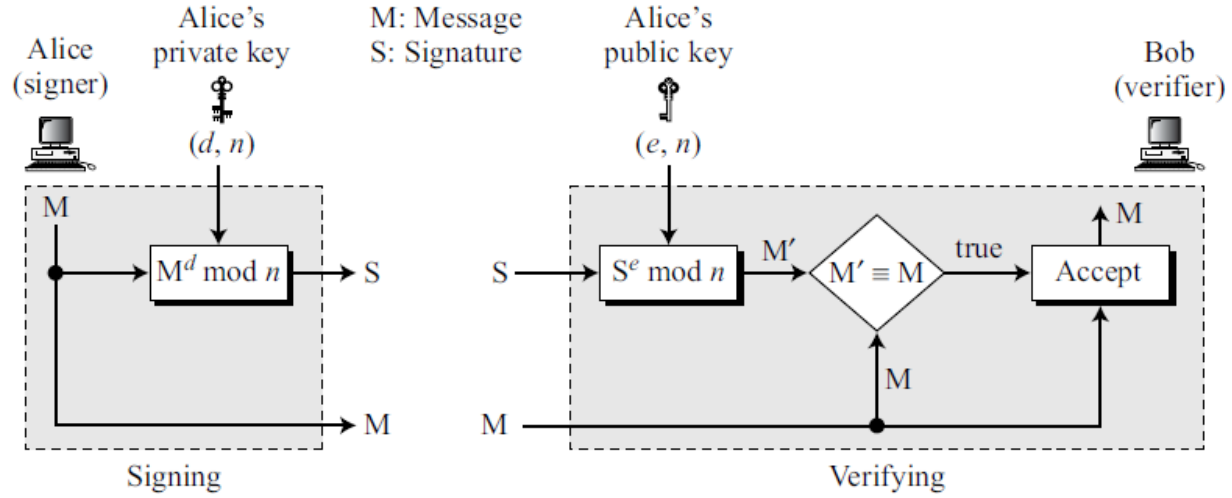
RSA Digital Signature Scheme

- **Key Generation**
  - Key generation in the RSA digital signature scheme is exactly the same as key generation in the RSA cryptosystem
  - Choose two primes p and q and calculate n = p × q. Calculates ϕ(n) = (p − 1) (q − 1).
  - Chooses e, the public exponent, and calculate d, the private exponent such that e × d = 1 mod ϕ(n).
  - Keep d; publicly announce n and e.
- **Signing and Verifying**
  - Verification Criteria

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

  - The last congruent holds because

    d × e = 1 mod ϕ(n)



RSA digital signature scheme

RSA Digital Signature Scheme

- For the security of the signature, the value of p and q must be very large.
- A trivial example:
  - suppose that Alice chooses p = 823 and q = 953, and calculates n = 784319. The value of $\phi(n)$ is 782544.
  - Now she chooses e = 313 and calculates d = 160009. At this point key generation is complete.
  - Now imagine that 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. Bob receives the message and the signature. He calculates

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

  - Signature Verified and message accepted!

Attacks on RSA Digital Signature Scheme

- **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' ≡ Se(mod n).
  - This problem is as difficult to solve as the discrete logarithm problem and is an *existential forgery* and normally is useless to Eve.
- **Known-Message Attack**
  - Here 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.

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

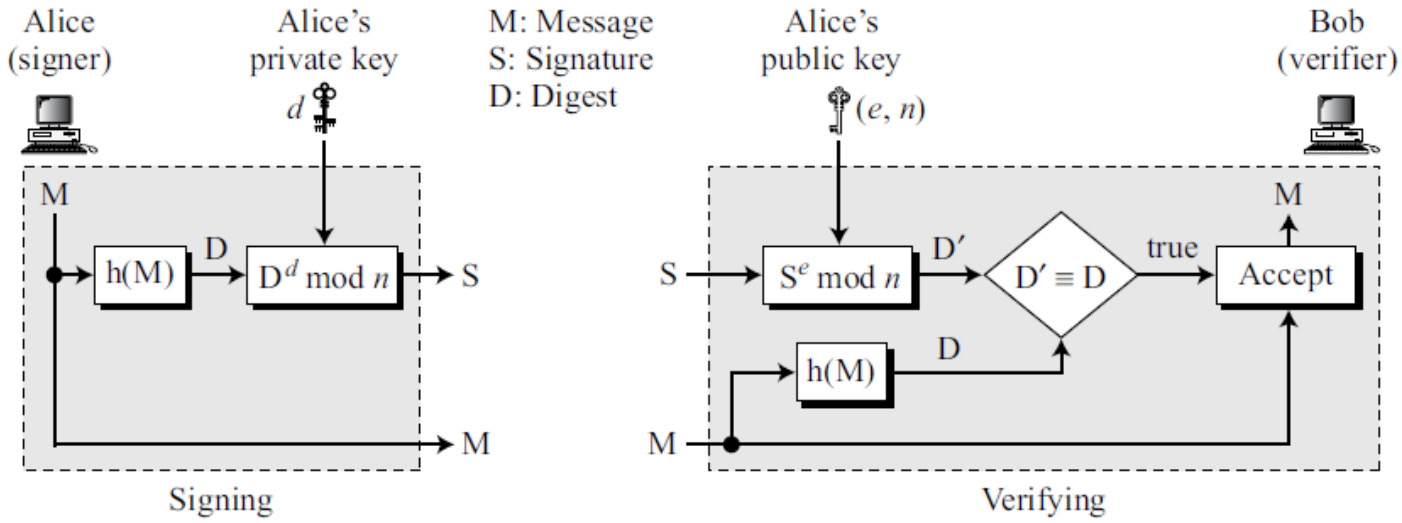
  - Eve can create M = (M1 × M2) mod n, and she can create S = (S1 × S2) mod n, and fool Bob into believing that S is Alice's signature on the message M.
  - This attack is sometimes referred to as multiplicative attack, and is *existential forgery*

Attacks on RSA Digital Signature Scheme

- **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 $M = M1 \times M2$. 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).

RSA Signature on the Message Digest

- In the case of RSA, it can make the signing and verifying processes much faster because the RSA digital signature scheme is 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



Attacks on RSA Signed Digests

- **Key-Only 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

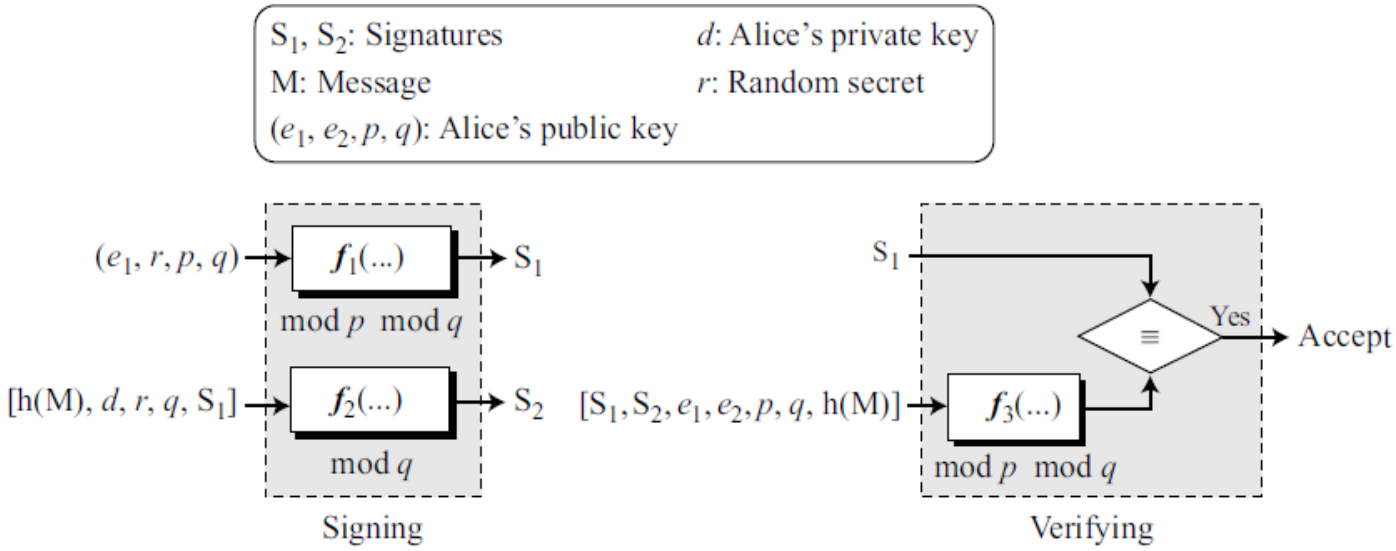
- **Known-Message Attack**
  - Let us assume Eve has two message-signature pairs (M1, S1) and (M2, S2) which have been created using the same private key. Eve calculates $S \equiv S1 \times S2$. If she can find a message M such that $h(M) \equiv h(M1) \times h(M2)$, she has forged a new message. However,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

- **Chosen-Message Attack**
  - Eve can ask Alice to sign two legitimate messages M1 and M2 for her. Eve then creates a new signature $S \equiv S1 \times S2$. Since Eve can calculate $h(M) \equiv h(M1) \times h(M2)$, 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.

    - 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.

Digital Signature Standard (DSS)

- Adopted by the National Institute of Standards and Technology (NIST) in 1994. NIST published DSS as FIPS 186.
- DSS uses a digital signature algorithm (DSA) based on the ElGamal scheme with some ideas from the Schnorr scheme.
- Criticized from the time it was published regarding the secrecy of DSS design and the size of the prime, 512 bits. Later NIST made the size variable to respond to this complaint.



Digital Signature Standard (DSS)

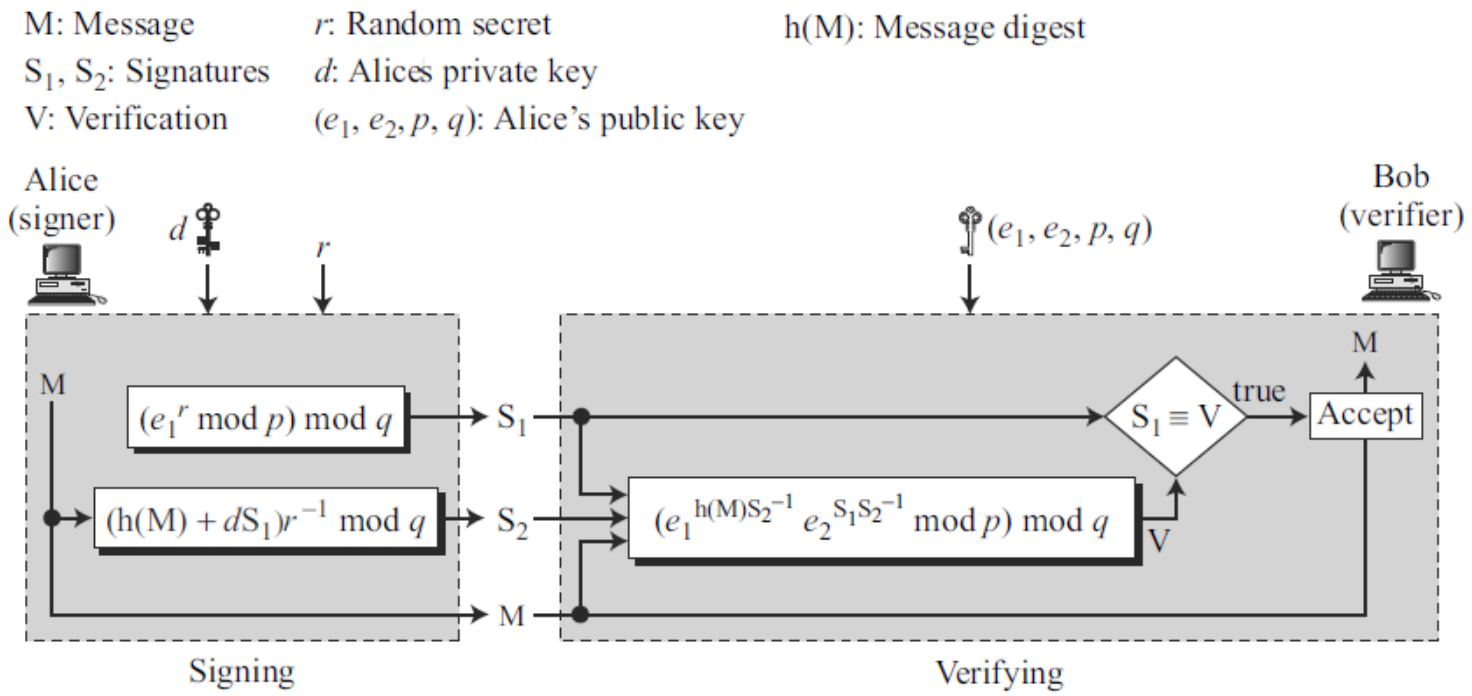
Key Generation

Before signing a message to any entity, Alice needs to generate keys and announce the public ones to the public.

1. Alice chooses a prime p, between 512 and 1024 bits in length. The number of bits in p must be a multiple of 64.
2. Alice chooses a 160-bit prime q in such a way that q divides $(p - 1)$.
3. Alice uses two multiplication groups $<\mathbf{Z}_p^*, \times>$ and $<\mathbf{Z}_q^*, \times>$; the second is a sub-group of the first.
4. Alice creates e_1 to be the qth root of 1 modulo p ($e_1^p = 1 \bmod p$). To do so, Alice chooses a primitive element in $\mathbf{Z}_p$, e_0, and calculates $e_1 = e_0^{(p-1)/q} \bmod p$.
5. Alice chooses d as the private key and calculates $e_2 = e_1^d \bmod p$.
6. Alice's public key is (e_1, e_2, p, q); her private key is (d).

Digital Signature Standard (DSS)

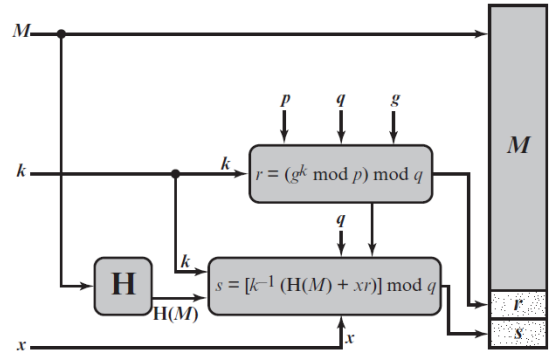
- **Verifying and Signing**



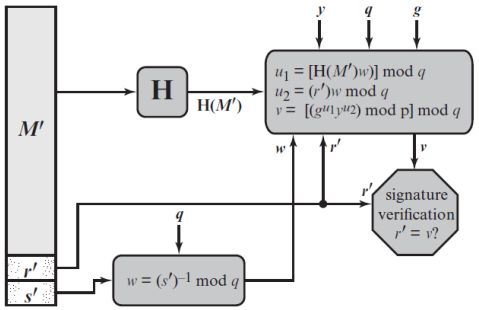
Digital Signature Standard (DSS)

Signing The following shows the steps to sign the message:

1. Alice chooses a random number r ($1 \le r \le q$). Note that although public and private keys can be chosen once and used to sign many messages, Alice needs to select a new r each time she needs to sign a new message.
2. Alice calculates the first signature $S_1 = (e_1{}^r \bmod p) \bmod q$. Note that the value of the first signature does not depend on M, the message.
3. Alice creates a digest of message h(M).
4. Alice calculates the second signature $S_2 = (h(M) + d\ S_1)r^{-1} \bmod q$. Note that the calculation of S_2 is done in modulo q arithmetic.
5. Alice sends M, S_1, and S_2 to Bob.



(a) Signing



(b) Verifying

Verifying Following are the steps used to verify the message when M, S_1, and S_2 are received:

1. Bob checks to see if $0 < S_1 < q$.
2. Bob checks to see if $0 < S_2 < q$.
3. Bob calculates a digest of M using the same hash algorithm used by Alice.
4. Bob calculates $V = [(e_1{}^{h(M)S_2{}^{-1}}\ e_2{}^{S_1S_2{}^{-1}}) \bmod p] \bmod q$.
5. If S_1 is congruent to V, the message is accepted; otherwise, it is rejected.

Digital Signature Standard (DSS)

Alice chooses $q = 101$ and $p = 8081$. Alice selects $e_0 = 3$ and calculates $e_1 = e_0^{(p-1)/q} \bmod p = 6968$. Alice chooses $d = 61$ as the private key and calculates $e_2 = e_1^{d} \bmod p = 2038$. Now Alice can send a message to Bob. Assume that h(M) = 5000 and Alice chooses $r = 61$:

$$h(M) = 5000 \qquad r = 61$$
$$S_1 = (e_1^{r} \bmod p) \bmod q = 54$$
$$S_2 = ((h(M) + d\,S_1)\, r^{-1}) \bmod q = 40$$

Alice sends M, S_1, and S_2 to Bob. Bob uses the public keys to calculate V.

$$S_2^{-1} = 48 \bmod 101$$
$$V = [(6968^{5000 \times 48} \times 2038^{54 \times 48}) \bmod 8081] \bmod 101 = 54$$

Because S_1 and V are congruent, Bob accepts the message.