C = EK1(P)

P = DK2(C)

K1 and K2 have strong relation but that relation should not allow you to deduce K1 from K2 or K2 from K1.

K1 (encryption key) should not lead to K2

1.Encrypt using recipient’s public key

2. Recipient’s end: he decrypts using his private key

Man-in-the-middle attack

1. Adversary can make you believe his public key as the recipient’s public key.

Trudy’s key –transmitted as—>Bob’s key

1. C = ET-P (P); P = DT-Pvt (C)

Confidentiality

Message integrity (using message digest)

Authentication

Non-repudiation

Problem: if you take 2 large prime nos. p, q and multiply them to get n, then it is difficult to obtain these prime nos. from the product n.

p \* q = n [key length should be min. 1024 bits]

If you take large prime nos., then it increase computational effort, and hence slow, and hence not practical.

That’s why, we combine asymmetric key with symmetric key [256 bits].

Asymmetric key is used as outer envelope and not so frequently as symmetric key.

RSA

1. Choose prime nos. p = 3 , q = 11
2. n = 3 \* 11 = 33
3. Compute z = (3 – 1) \* (11 – 1) = 20
4. Choose a number e such that it is coprime to z
5. Let us choose d = 7
6. Choose (d\*e)%z = 1 [Euclid’s algo]
7. Hence d = 3
8. Private key: (7, 33)
9. Public key: (3, 33)
10. C = me(mod n)
11. m = Cd(mod n)

Encrypt CAB

2 0 1

m = 2-> C = 27(mod 33) = 29

293mod 33 = 2

If a cryptanalyst could factor n and obtain p and q

Obtain z

e is already known.

d \* e = 1 (mod z)

d can be derived from e.

Symmetric key scenario

C = EK(P) :: C is unintelligible

P = DK(C) :: P back

ASymmetric key scenario

C = Ee(P) :: C is unintelligible

P = Dd(C) :: P back

Interchangeability of keys

C = Ed(P) :: C is unintelligible

P = De(C) :: P back

A(m)🡪 EB+ 🡪 EB+ (m)🡪 DB- 🡪 DB-(EB+ (m)) 🡪 m conf..

C Trudy

Digital signature

A(m)🡪 EA- 🡪 EA- (m)🡪 DA+ 🡪 D A+ (EA- (m)) 🡪 m

(Any recipient’s end)

confidentiality ?? No

Non-repudiation ?

Digital Sign: MD is signed.

MD: Message digest is a fingerprint of m.

MD is generated by a hash function (SHA1) (hash function output is MD)

Hash (m) = MD

Hash (m’) ≠MD

(collision free: two diff messages cannot give same MD)

There is no way to obtain m from MD.

Easy to compute Hash (m) = MD

But given MD, obtaining m is infeasible.

A(MD)🡪 EA- 🡪 EA- (MD)🡪 DA+ 🡪 D A+ (EA- (MD)) 🡪 MD

A B(anyone)

P, EA- (MD) P’, EA- (MD)

MD

MD(P’)

If MD(P’) == MD

Dig. Sign verified

If A wants to send a signed message to B confidentially

1. Message (m)
2. ~~B’s public key to encrypt~~
3. First sign
   1. Get MD
   2. Digital sign: m,EA-(MD)
4. B’s public key to encrypt the digital sign:

EB+(m,EA-(MD))

1. Transmit EB+(m,EA-(MD)) to B

Suppose Trudy (bad guy) access EB+(m,EA-(MD)), then can he obtain m,EA-(MD) ? no

1. B’s end: B will apply his private key to

EB+(m,EA-(MD)) as follows:

EB-( EB+(m,EA-(MD)) ) = m,EA-(MD)

DA+EA-(MD) = MD

Compute MD(m) and check if MD(m) == MD

(process of verification of Digital Sign)

This process could have been done only by B because of the need to have B- to decipher EB+(m,EA-(MD)) : Hence confidentiality

1. B’s end: Digital sign m,EA-(MD) can be verified with A+ (A’s public key) : Hence non repudiation.

Weakness

1. How to publicize the key or how to transmit the public key of recipient to sender?
2. Slow (because min key length has to be 1024 bits rather 2048 bits for better safety). Comparatively 256 bits are usually fine in symmetric key. Hence symmetric key cryptosystem is much faster and hence practical.
3. Anybody (Trudy)can compute many possible ciphertexts C’s = E Public key B (m) and checking if

C’ == C (the original C in transmission)

Trudy observes the communication channel. Trudy can see all C’s.

In contrast to symmetric key, Trudy can compute possible C’s for probable P’s (based on contextual info about different possible plain texts. Since process of encrypting is public, Trudy can compute possible cipher-texts and try to match it against the given C. If he gets success, he can know plaintext.)

A(m)🡪 EB+ 🡪 EB+ (m)🡪 DB- 🡪 DB-(EB+ (m)) 🡪 m conf..

C Trudy

Issues

How to publicize the key / transmit public key

A(m)🡪 EB+ 🡪 EB+ (m)🡪 DB- 🡪 DB-(EB+ (m)) 🡪 m conf..

C Trudy

Digital Signature recall

1. First generate MD using a hash function
2. Encrypt this MD using private key of the sender

What proof B has to prove that it is signed by A

How confidential communication happens through Asymmetric key

A –confidential commn -- > B

A -- EB+(m) –-- DB- --🡪 DB- EB+(m) = m

Digital Sign:

1. EA-(m) ------------------ DA+ --- DA+ EA-(m) = m
2. m, EA-( MD(m)) ---DA+ ---- m, DA+ EA-( MD(m)) = m, MD(m)

What proof B has to prove that it is signed by A

At B’ end: MD(m) m

If B produces this m, EA-( MD(m)) to a judge, then it can be shown that this could only be generated by A because A has access to A-. The judge can compute himself m, DA+ EA-( MD(m)) using public key of A.

The m and MD(m).