Topics to Cover

1) Principles of Public-Key Cryptosystems
2) Rivest-Shamir-Adleman (RSA)
3) Diffie Hellman Key Exchange

PRINCIPLES OF PUBLIC-KEY CRYPTOSYSTEMS

Terminologies Associated with Asymmetric Encryption

➢ Asymmetric Keys:-

- a public key that anyone can use and a private key that only one person has.
- They work together to do opposite tasks, like locking (encrypting) and unlocking (decrypting) messages, or creating (signing) and checking (verifying) signatures.

➢ Public Key Certificate:-

- A digital certificate, issued and signed by a Certification Authority, links a subscriber's name to a public key.
- The certificate confirms that the subscriber has exclusive control over the corresponding private key.

Terminologies Associated with Asymmetric Encryption (Contd..)

➢ Public Key (Asymmetric) Cryptographic Algorithm:-

- A cryptographic algorithm uses two keys: a public key that anyone can see and a private key that only you keep secret.
- It's very hard to figure out the private key just by knowing the public key.

➢ Public Key Infrastructure (PKI):-

- A system for managing certificates and keys.
- Can create, update, and cancel certificates.

Misconceptions of Public-Key Encryption

- Public-key encryption is more secure from cryptanalysis than symmetric encryption
- Public-key encryption is a general-purpose technique that has made symmetric encryption obsolete
- There is a feeling that key distribution is trivial when using public-key encryption, compared to the cumbersome handshaking involved with key distribution centers for symmetric encryption

Evolution and Principles of Public-Key Cryptosystems (PKC)

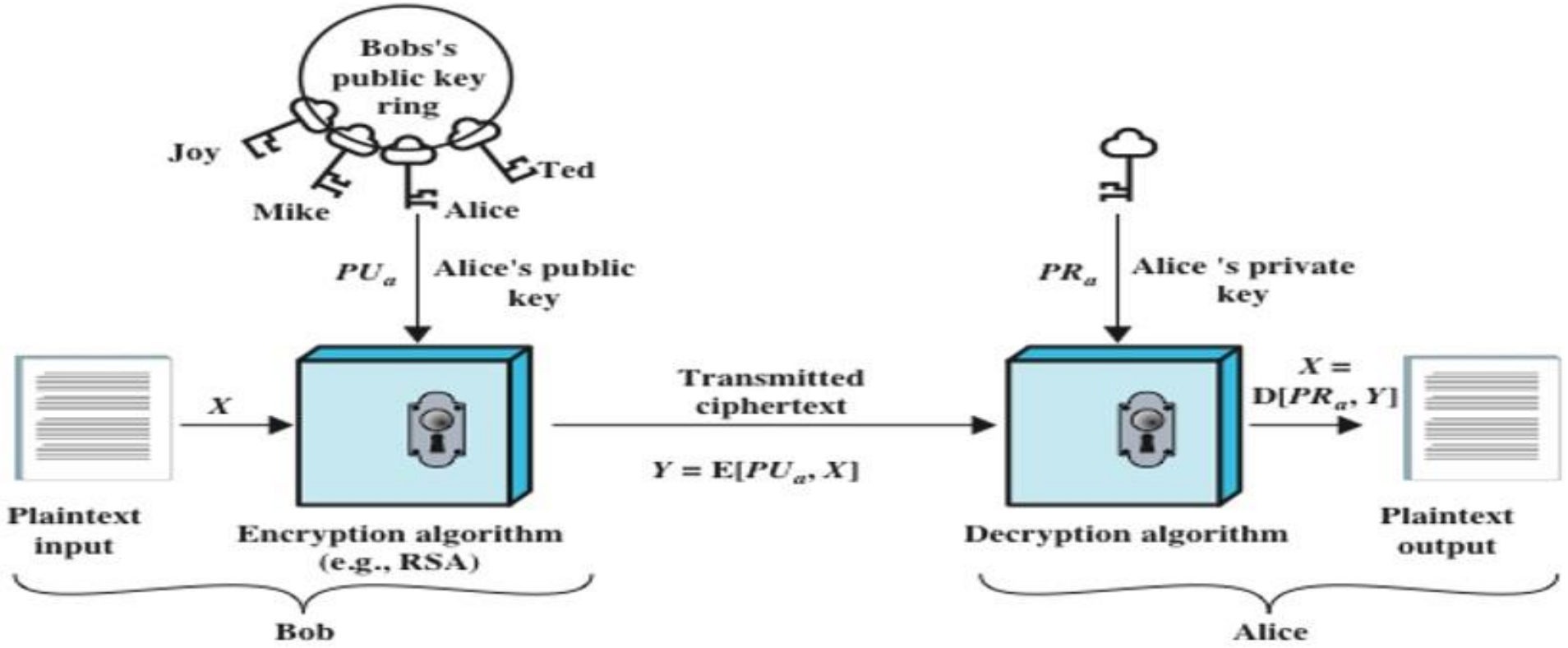
- Public-key cryptography was created to solve 2 major challenges with symmetric encryption:-
  - Key Distribution
  - Digital Signatures

- In 1976, Whitfield Diffie and Martin Hellman from Stanford University made a big discovery, which addressed both the problems, and was different from cryptographic methods in past.

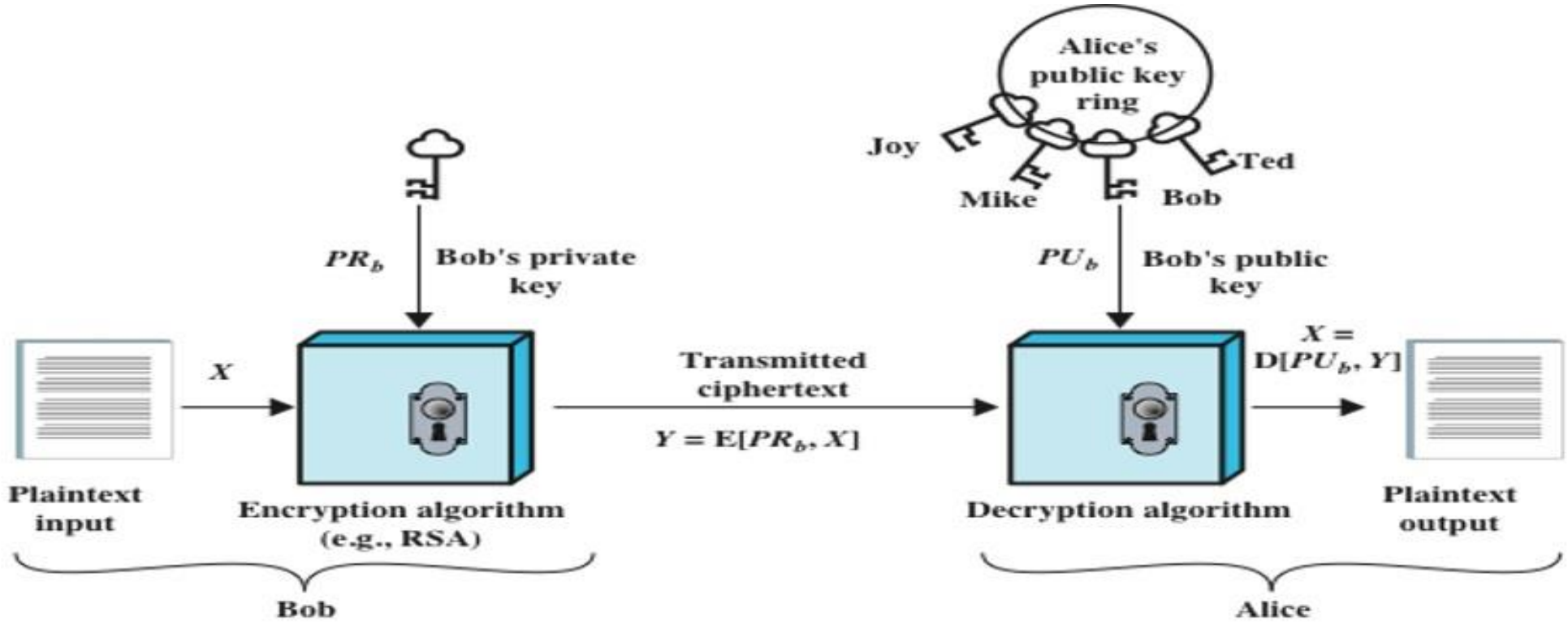
6 Ingredients of PKC

- Private key
- Public Key
- Plaintext
- Encryption Algorithm
- Ciphertext
- Decryption Algorithm

PKC (Encryption with Public Key)



PKC (Encryption with Private Key)



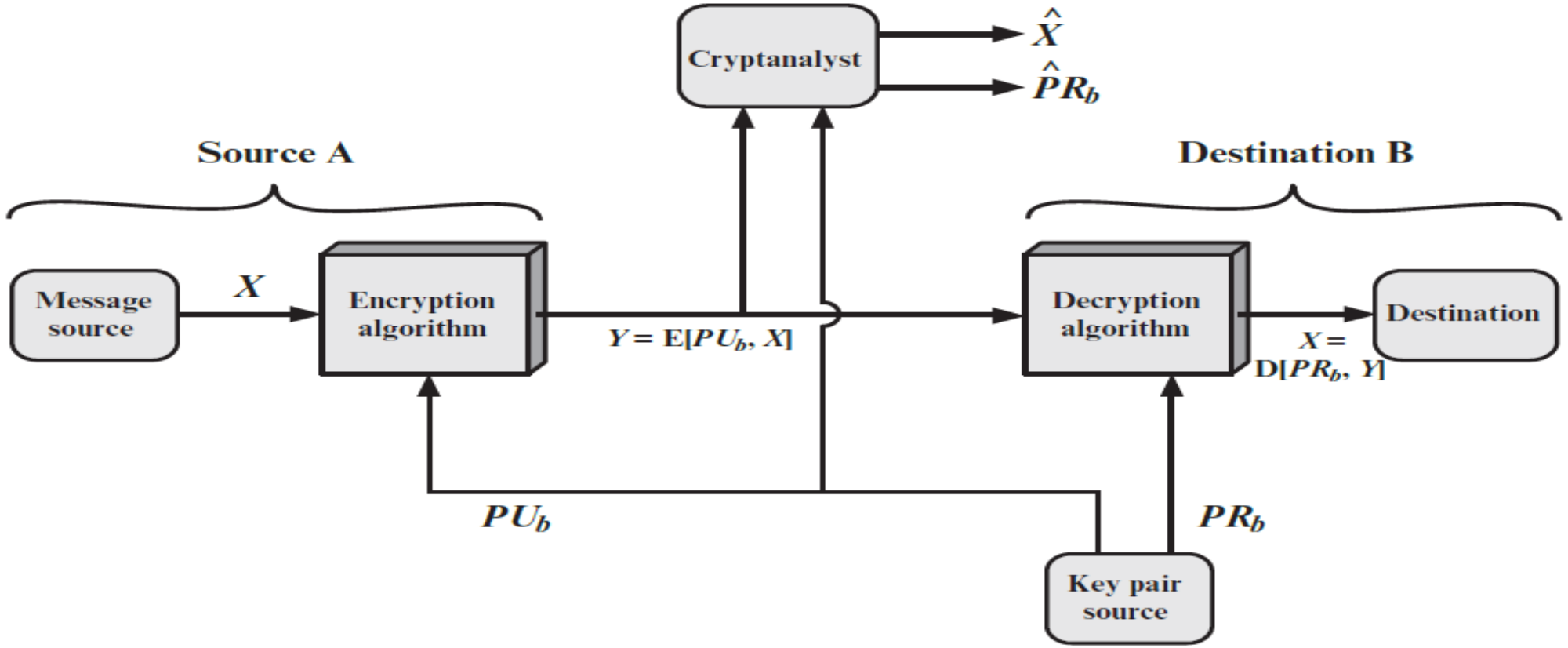
Conventional and Public Key Encryption

Conventional Encryption	Public-Key Encryption
Needed to Work: 1. The same algorithm with the same key is used for encryption and decryption. 2. The sender and receiver must share the algorithm and the key. *Needed for Security:* 1. The key must be kept secret. 2. It must be impossible or at least impractical to decipher a message if the key is kept secret. 3. Knowledge of the algorithm plus samples of ciphertext must be insufficient to determine the key.	*Needed to Work:* 1. One algorithm is used for encryption and a related algorithm for decryption with a pair of keys, one for encryption and one for decryption. 2. The sender and receiver must each have one of the matched pair of keys (not the same one). *Needed for Security:* 1. One of the two keys must be kept secret. 2. It must be impossible or at least impractical to decipher a message if one of the keys is kept secret. 3. Knowledge of the algorithm plus one of the keys plus samples of ciphertext must be insufficient to determine the other key.

Security Goals provided by PKC

- Confidentiality
- Authentication
- Integrity
- Accountability

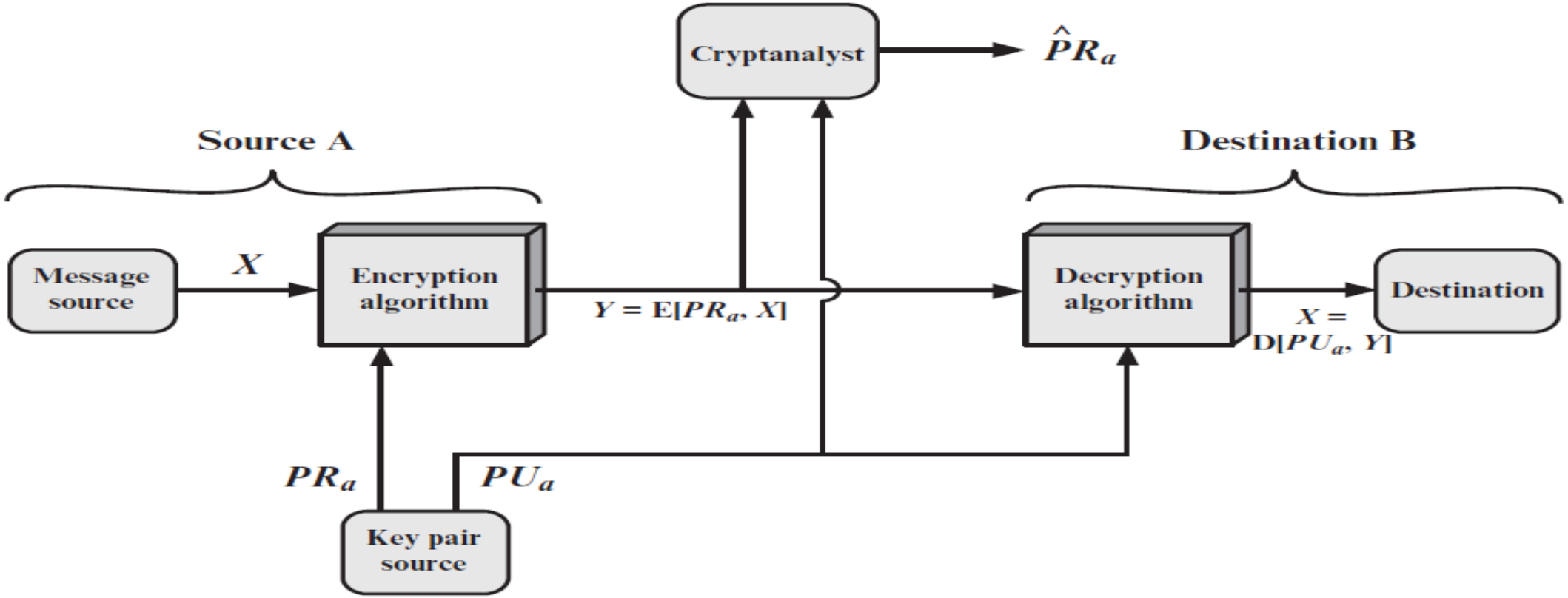
PKC for Confidentiality



PKC for Confidentiality (Contd..)

- $Y = E(PU_b, X)$
- $X = D(PR_b, Y)$
- An attacker can capture Y and has the public key (PU_b), but not the private key (PR_b) or X.
- The attacker wants to find X and/or the private key (PR_b).
- The attacker has knowledge of E() and D().
- If the attacker cares about only one message, he/she tries to guess X by generating a PT estimate $\hat{X}$.
- If the enemy wants to read future messages too, he/she will also try to guess the private key (PR_b) by generating an estimate $\widehat{(PR_b)}$.

PKC for Authentication

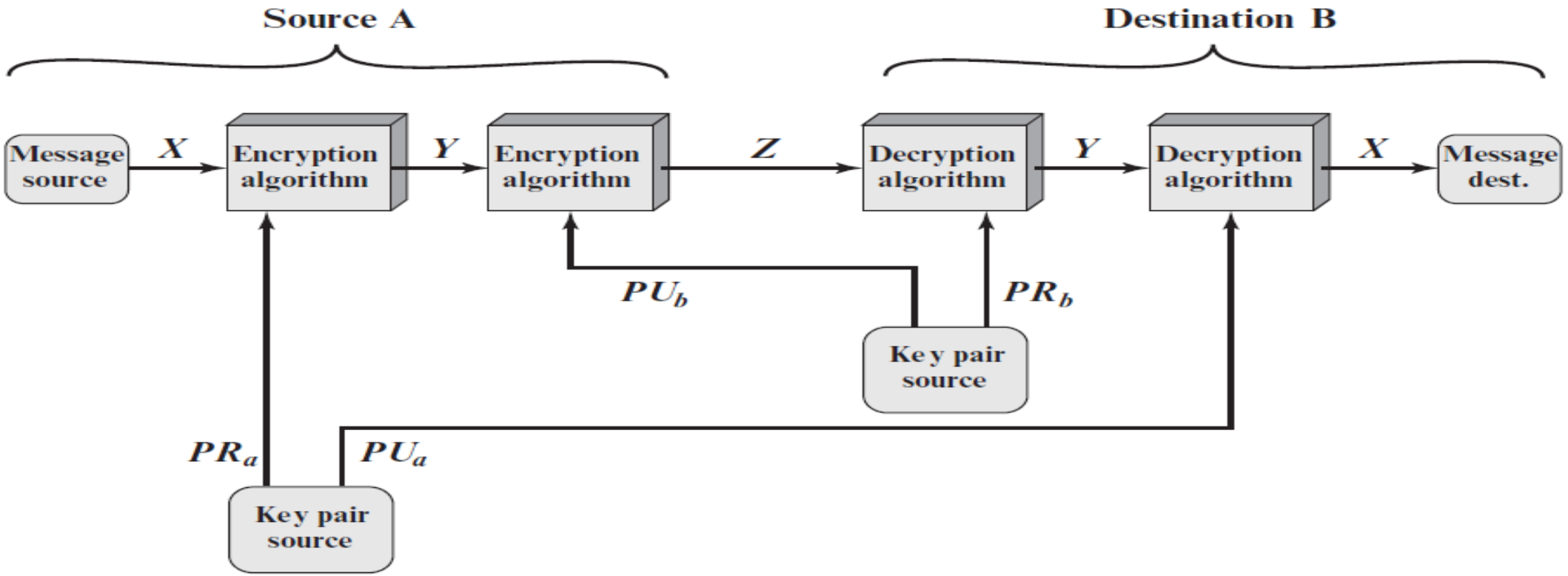


PKC for Authentication (Contd..)

- $Y = E(PR_a, X)$
- $X = D(PU_a, Y)$

- Only A could have created the message since it was encrypted with A’s private key.
- The encrypted message acts as a digital signature.
- The message can't be changed without A’s private key.
- This confirms both who sent it (source) and that it hasn’t been altered (data integrity).

PKC for Confidentiality and Authentication



PKC for Confidentiality and Authentication (Contd..)

- $Z = E(PU_b, E(PR_a,X))$
- $X = D(PU_a, D(PR_b,Z))$

- 'A' encrypts a message with the sender's private key for a digital signature (Result = Y).
- 'A' encrypts Y using receiver's public key (PU_b). (Result = X)
- Only the intended receiver can decrypt the final message with their private key.
- This method ensures that the message remains confidential.
- However, the complete process requires the complex public-key process to be done four times instead of just two.

Applications for PKC

- The sender uses their private key, the receiver's public key, or both to encrypt or sign messages, depending on the application.

- Public-key cryptosystems can be classified into three categories:

• Encryption/Decryption
• Digital Signature
• Key Exchange

- Some algorithms work for all three applications.

- Others can only be used for one or two applications.

Applications for PKC (Contd..)

Algorithm	Encryption/Decryption	Digital Signature	Key Exchange
RSA	Yes	Yes	Yes
Elliptic Curve	Yes	Yes	Yes
Diffie-Hellman	No	No	Yes
DSS	No	Yes	No

Requirements for PKC

- B (PU_b, PR_b):- Computationally easy
- C = E(PU_b, M):- Computationally easy
- M = D(PR_b, C) = D[PR_b, E(PU_b, M)]:- Computationally easy
- Deriving PR_b from PU_b should be computationally infeasible for an attacker.
- Recovering M from C and PU_b should be computationally infeasible for an attacker.
- M = D[PU_b, E(PR_b, M)] = D[PR_b, E(PU_b, M)]

Requirements for PKC (Contd..)

- Trap Door One-Way Function

- $Y = f(X)$:- Computationally feasible
- $X = f^{-1}(Y)$:- Computationally infeasible
- $Y = f_k(X)$ easy, if k and X are known
- $X = f_k^{-1}(Y)$ easy, if k and Y are known
- $X = f_k^{-1}(Y)$ infeasible, if Y known but k not known

- A practical public-key scheme depends on a suitable trap-door one-way function

Cryptanalysis on PKC

- Public-key encryption can be attacked using brute-force, so using large keys is important for security.
- However, bigger keys can slow down the encryption and decryption process.
- Because of this, public-key encryption is mainly used for key management and signatures, rather than regular data encryption.
- It’s possible to calculate the private key from the public key, but it's not yet been proven Mathematically that this can't be done for any public-key system.
- This means that all algorithms, even popular ones like RSA, could potentially be vulnerable because new ways to solve problems can be discovered over time.

Rivest-Shamir-Adleman (RSA)

RSA (Overview)

- Developed in 1977 at MIT, Cambridge (USA) by Ron Rivest, Adi Shamir & Len Adleman.
- It was one of the first public-key cryptosystems.
- RSA is one of the most adopted algorithms in practical applications.
- RSA is versatile and can be applied to a wide range of applications, from securing emails to enabling secure connections for websites (via SSL/TLS).
- RSA is also used for applications like digital signatures, VPNs, Blockchain, Cryptocurrencies, etc.

RSA (Overview)

- The PT and CT falls in the range [0,n-1], where n has a typical size of atleast 1024 bits.
- RSA makes use of modular exponentiation.
- PT is divided into blocks.
- Number of bits in each block $\leq (\log_2(n)+1)$.

RSA (Key Generation)

- Select large primes ‘p’ and ‘q’, such that $p \neq q$.
- $n = p*q$
- $\Phi(n) = (p-1)*(q-1)$
- Select ‘e’, such that $1<e<\Phi(n)$ and $GCD(e, \Phi(n))=1$
- $d = MI(e) \bmod \Phi(n)$
- Public Key = (e, n)
- Private Key = d

RSA (Encryption and Decryption)

- Encryption:- $C = P^e \pmod n$
- Decryption:- $P = C^d \pmod n$
- $P,C \in Z_n$

- Inputs to Encryption:- (P, e, n)
- Inputs to Decryption:- (C, d, n)

Requirements for RSA algorithm

- Large ‘p’ and ‘q’ (at least 512 bits each).
- $p \neq q$
- Significant difference between ‘p’ and ‘q’.
- It is possible to find values of ‘e’, ‘d’, and ‘n’ such that $(M^e)^d \bmod n = M$ for all $M<n$.
- It is relatively easy to calculate $M^e \bmod n$ and $C^d \bmod n$ for all values of $M<n$.
- It is infeasible to determine ‘d’ given ‘e’ and ‘n’.
- $e*d \bmod \Phi(n) = 1$

Proof for $C^d \bmod n = M$

- $C^d \bmod n = (M^e)^d \bmod n$
- $e*d \equiv 1 \bmod \Phi(n)$
- $e*d = k*\Phi(n) + 1$
- $M^{\Phi(n)} \bmod n = 1$
- $M^{k*\Phi(n)} \bmod n = 1$
- $M^{k*\Phi(n)+1} \bmod n = M$
- $M^{e*d} \bmod n = M$
- $C^d \bmod n = M$

Computational Aspects of RSA Encryption and Decryption

- Both encryption and decryption in RSA involve raising an integer to an integer power, mod n
- Can make use of a property of modular arithmetic: [(a mod n) * (b mod n)] mod n = (a * b) mod n
- Exponentiations Efficiency is also an important aspect to consider.
- CT or DT can be calculated using Fast modular exponentiation.

Efficient RSA Public Key Operation

- To make the RSA algorithm faster, a specific number for ‘e’ is chosen.
- The most common choice for ‘e’ is 65537 (i.e. $2^{16}+1$).
- Other popular choices for e are 3 and 17.
- These choices have only two 1 bits, which reduces the number of multiplications needed.
- RSA can be weak with a small public key (e = 3).

Efficient RSA Public Key generation (e=3)

- Assume that there are 3 different recipients who use e=3, but unique values of n (i.e. n_1, n_2, n_3)

- In that case, the sender would send 3 different CTs to the 3 different recipients:-
  - $C_1 = M^3 \bmod n_1$
  - $C_2 = M^3 \bmod n_2$
  - $C_3 = M^3 \bmod n_3$

- If n_1, n_2, n_3, are pairwise relatively prime, then the attacker can calculate M^3 using CRT.

Efficient RSA Private Key Operation

- Small value of ‘d’ will make RSA vulnerable to BFA and other cryptanalysis methods.
- We can speed up the calculation $M=C^d \bmod n$ using CRT.
- $V_p = C^d \bmod p$; $V_q = C^d \bmod q$
- Let $X_p = q*MI(q) \bmod p$; $X_q = p*MI(p) \bmod q$
- According to CRT, $M = (V_p*X_p + V_q*X_q) \bmod n$

Efficient RSA Private Key Operation (Contd..)

- if $d>p$, $V_p = C^d \bmod p = C^{d \bmod (p-1)} \bmod p$
- if $d>q$, $V_q = C^d \bmod q = C^{d \bmod (q-1)} \bmod q$

- if $d=p$, $V_p = C^d \bmod p = C$
- if $d=q$, $V_q = C^d \bmod q = C$

- If $d>p$, and $d>q$, then the CRT method is 4 times the speed of calculating ($M = C^d \bmod n$) directly.

Computational Aspects for Key Generation

- Finding large prime numbers efficiently is important.

- There's no perfect way to find very large primes, so a different method is used:
  - Randomly pick an odd number and check if it's prime.
  - If it's not prime, keep picking new numbers until a prime is found.

- Primality can be tested using various tests (Common one is Miller Rabin algorithm)

Computational Aspects for Key Generation

➢Steps to find a prime number:-

- Randomly choose an odd number ‘n’.
- Randomly choose another number ‘a’ (a<n).
- As long as ‘n’ fails the test, start over from Step 1.
- If ‘n’ passes enough number of tests, accept ‘n’ as a prime, else go to step 2.

➢Approximate number of trials required to identify a prime near a number N = ln(N)/2.

Numerical 1

- A Plaintext = 123 is encrypted using RSA with key generation primes 61 and 59, and public exponent = 17. Calculate the private key, and CT.

Solution:-

- p = 61, q = 59, M = 123
- n = p*q = 3599
- $\Phi(n) = (p-1)*(q-1) = 3480$
- e = 17
- d = MI(e) mod $\Phi(n)$ = MI(17) mod 3480

Numerical 1 (Contd..)

q	a	b	r	v1	v2	v
204	3480	17	12	0	1	-204
1	17	12	5	1	-204	205
2	12	5	2	-204	205	-614
2	5	2	1	205	-614	1433
2	2	1	0	-614	1433	-3480
	1	0		**1433**		

- Private Key = d = 1433

Numerical 1 (Contd..)

- $C = M^e \bmod n = 123^{17} \bmod 3599$

➢ $123^{17} \bmod 3599$:-

- $123^2 \bmod 3599 = 733$
- $123^4 \bmod 3599 = 1038$
- $123^{16} \bmod 3599 = 550$
- $123^{17} \bmod 3599 = (123^{16} \bmod 3599 * 123) \bmod 3599 = (550*123) \bmod 3599 = 2868$

➢ $CT = C = 2868$

Numerical 2

- In reference to the previous numerical, calculate the PT when CT=2868.

Solution:-

- $M = C^d \bmod n = 2868^{1433} \bmod 3599$
- $1433 = 1024 + 256 + 128 + 16 + 8 + 1$

➢ $2868^{1433} \bmod 3599$:-

- $2868^2 \bmod 3599 = 1709$
- $2868^4 \bmod 3599 = 1892$
- $2868^8 \bmod 3599 = 2258$

Numerical 2 (Contd..)

- $2868^{16} \bmod 3599 = 2380$
- $2868^{32} \bmod 3599 = 3173$
- $2868^{64} \bmod 3599 = 1526$
- $2868^{128} \bmod 3599 = 123$
- $2868^{256} \bmod 3599 = 733$
- $2868^{512} \bmod 3599 = 1038$
- $2868^{1024} \bmod 3599 = 1343$

Numerical 2 (Contd..)

- $2868^{1433} \bmod 3599 = (1343 * 733 * 123 * 2380 * 2258 * 2868) \bmod 3599$
- $2868^{1433} \bmod 3599 = [(1343 * 733 * 123) \bmod 3599 * (2380 * 2258 * 2868) \bmod 3599] \bmod 3599$
- $2868^{1433} \bmod 3599 = (2380 * 428) \bmod 3599 = 123$

➢ PT = M = 123

Numerical 3

- A PT (consists of 5 characters) is encrypted by using its corresponding ASCII equivalents, and each character is encrypted separately. The key generation primes are 11 and 19, and the public exponent is 47. The CT has a sequence (41, 141, 76, 76, 29). Calculate the private key and PT.

Solution:-

- $p = 11, q = 19, e = 47$
- $n = p*q = 209$
- $\Phi(n) = (p-1)*(q-1) = 180$
- $d = MI(e) \bmod \Phi(n) = MI(47) \bmod 180$

Numerical 3 (Contd..)

q	a	b	r	v1	v2	v
3	180	47	39	0	1	-3
1	47	39	8	1	-3	4
4	39	8	7	-3	4	-19
1	8	7	1	4	-19	23
7	7	1	0	-19	23	-180
	1	0		**23**	-180	

- Private Key = d = 23 = 16 + 4 + 2 + 1

Numerical 3 (Contd..)

➢ $M_1 = C_1^{d} \bmod n = 41^{23} \bmod 209$:-

- $41^{2} \bmod 209 = 9$
- $41^{4} \bmod 209 = 81$
- $41^{16} \bmod 209 = 36$
- $41^{23} \bmod 209 = (41^{16} \bmod 209 * 41^{4} \bmod 209 * 41^{2} \bmod 209 * 41) \bmod 209$
- $41^{23} \bmod 209 = (36 * 81 * 9 * 41) \bmod 209 = 72$
- M_1 = ‘H’

Numerical 3 (Contd..)

➢ $M_2 = C_2^{d} \bmod n = 141^{23} \bmod 209$:-

- $141^{2} \bmod 209 = 26$
- $141^{4} \bmod 209 = 49$
- $141^{16} \bmod 209 = 163$
- $141^{23} \bmod 209 = (141^{16} \bmod 209 * 141^{4} \bmod 209 * 141^{2} \bmod 209 * 141) \bmod 209$
- $141^{23} \bmod 209 = (163 * 49 * 26 * 141) \bmod 209 = 69$
- M_2 = ‘E’

Numerical 3 (Contd..)

➢ $M_3 = C_3^{\,d} \bmod n = 76^{23} \bmod 209$:-

- $76^2 \bmod 209 = 133$
- $76^4 \bmod 209 = 133$
- $76^{16} \bmod 209 = 133$
- $76^{23} \bmod 209 = (76^{16} \bmod 209 * 76^4 \bmod 209 * 76^2 \bmod 209 * 76) \bmod 209$
- $76^{23} \bmod 209 = (133 * 133 * 133 * 76) \bmod 209 = 76$
- M_3 = ‘L’

➢ $M_4 = C_4^{\,d} \bmod n = 76^{23} \bmod 209 = 76$ = ‘L’

Numerical 3 (Contd..)

- $M_5 = C_5^{d} \bmod n = 29^{23} \bmod 209$:-
- $29^2 \bmod 209 = 5$
- $29^4 \bmod 209 = 25$
- $29^{16} \bmod 209 = 4$
- $29^{23} \bmod 209 = (29^{16} \bmod 209 * 29^4 \bmod 209 * 29^2 \bmod 209 * 29) \bmod 209$
- $29^{23} \bmod 209 = (4 * 25 * 5 * 29) \bmod 209 = 79$
- M_5 = ‘O’

- PT = ‘HELLO’

Diffie-Hellmann Key Exchange (DHKE)

DHKE

Alice

Bob

Alice	Bob
Alice and Bob share a prime q and α, such that $\alpha < q$ and α is a primitive root of q	Alice and Bob share a prime q and α, such that $\alpha < q$ and α is a primitive root of q
Alice generates a private key X_A such that $X_A < q$	Bob generates a private key X_B such that $X_B < q$
Alice calculates a public key $Y_A = \alpha^{X_A} \bmod q$	Bob calculates a public key $Y_B = \alpha^{X_B} \bmod q$
Alice receives Bob's public key Y_B in plaintext	Bob receives Alice's public key Y_A in plaintext
Alice calculates shared secret key $K = (Y_B)^{X_A} \bmod q$	Bob calculates shared secret key $K = (Y_A)^{X_B} \bmod q$

Y_A

Y_B

DHKE (Contd..)

- Used to establish secret key between only 2 communicating entities, at a time.
- The key's value depends on the participants' private and public keys.
- Uses exponentiation in a finite field (modulo a prime or polynomial):- Easy to calculate
- Security relies on the difficulty of solving discrete logarithms.
- Discrete Logarithm for calculating private key of Alice:- $X_A = \text{dlog}_{\alpha,q}(Y_A)$
- Discrete Logarithm for calculating private key of Bob:- $X_B = \text{dlog}_{\alpha,q}(Y_B)$

Proof of same secret key generation by Alice and Bob

$$
\begin{aligned}
K &= (Y_B)^{X_A} \bmod q \\
&= (\alpha^{X_B} \bmod q)^{X_A} \bmod q \\
&= (\alpha^{X_B})^{X_A} \bmod q \\
&= \alpha^{X_B X_A} \bmod q \\
&= (\alpha^{X_A})^{X_B} \bmod q \\
&= (\alpha^{X_A} \bmod q)^{X_B} \bmod q \\
&= (Y_A)^{X_B} \bmod q
\end{aligned}
$$

Numerical 1

- Alice and Bob agree upon the DHKE global parameters (59 and 6), where 6 is the primitive root of 59. The private key of Alice is 4 and that of Bob is 8. Calculate the public keys generated by the 2 communicating entities, and the secret key established by both the entities (show secret key calculations for both the entities).

Solution:-

- $q = 59, \alpha = 6, X_A = 4, X_B = 8$
- $Y_A = \alpha^{X_A} \bmod q = 6^4 \bmod 59 = 57$
- $Y_B = \alpha^{X_B} \bmod q = 6^8 \bmod 59 = 4$

Numerical 1 (Contd..)

- $K_A = (Y_B)^{X_A} \bmod q = 4^4 \bmod 59 = 20$

- $K_B = (Y_A)^{X_B} \bmod q = 57^8 \bmod 59$
  - $57^4 \bmod 59 = 16$
  - $57^8 \bmod 59 = 20$
  - $K_B = 20$

Numerical 2

- Alice and Bob agree upon the DHKE global parameters (1009 and 22), where 22 is the primitive root of 1009. The private key of Alice is 25 and that of Bob is 75. Calculate the public keys generated by the 2 communicating entities, and the secret key established by both the entities.

Solution:-

➢ $q = 1009, \alpha = 22, X_A = 25, X_B = 75$

➢ $Y_A = \alpha^{X_A} \bmod q = 22^{25} \bmod 1009$

Numerical 2 (Contd..)

- $25 = 16 + 8 + 1$
- $22^4 \bmod 1009 = 168$
- $22^8 \bmod 1009 = 981$
- $22^{16} \bmod 1009 = 784$
- $22^{25} \bmod 1009 = (784 * 981 * 22) \bmod 1009 = 367$
- $Y_A = 367$

➢ $Y_B = \alpha^{X_B} \bmod q = 22^{75} \bmod 1009$

- $75 = 64 + 8 + 2 + 1$

Numerical 2 (Contd..)

- $22^2 \bmod 1009 = 484$
- $22^4 \bmod 1009 = 168$
- $22^8 \bmod 1009 = 981$
- $22^{16} \bmod 1009 = 784$
- $22^{32} \bmod 1009 = 175$
- $22^{64} \bmod 1009 = 355$
- $22^{75} \bmod 1009 = (355 * 981 * 484 * 22) \bmod 1009 =$
- $22^{75} \bmod 1009 = [(355 * 981) \bmod 1009 * (484 * 22) \bmod 1009] \bmod 1009$
- $22^{75} \bmod 1009 = (150 * 558) \bmod 1009 = 962 = Y_B$

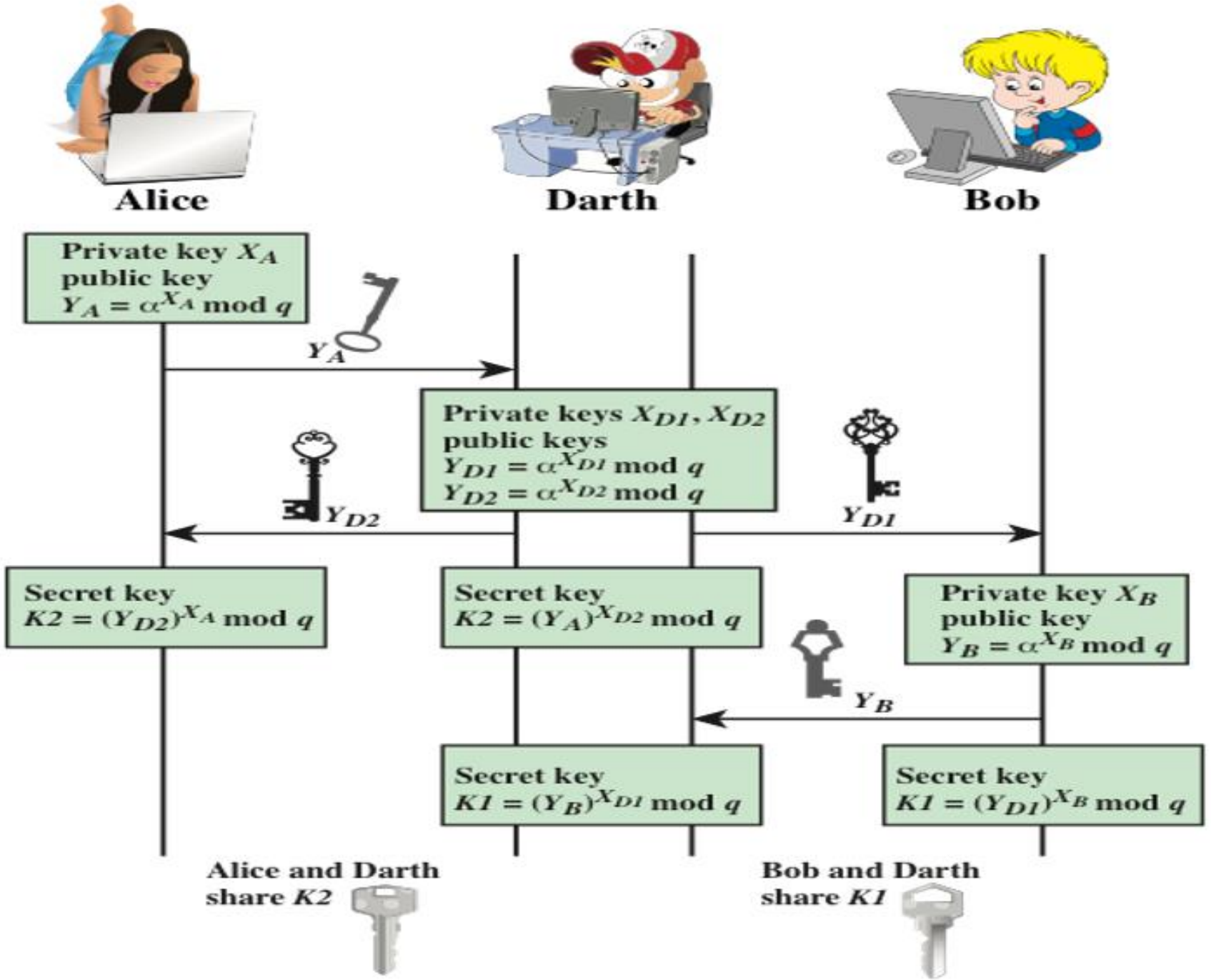
Numerical 2 (Contd..)

- $K_A = (Y_B)^{X_A} \bmod q = 962^{25} \bmod 1009$
- $25 = 16 + 8 + 1$
- $962^2 \bmod 1009 = 191$
- $962^4 \bmod 1009 = 157$
- $962^8 \bmod 1009 = 433$
- $962^{16} \bmod 1009 = 824$
- $962^{25} \bmod 1009 = (824 * 433 * 962) \bmod 1009 = 356$
- $K_A = 356$

Use case of DHKE for a group of users

- Assume that a group of users (for example on LAN) agree upon the global public parameters (‘α’ and ‘q’)
- Assume that a group of users (for example on LAN) each creates a long-lasting private key (X_i).
- Each user calculates a public number (Y_i) from their private.
- All public keys are stored in a trusted central location.
- Any user (j) can access another user’s (i) public value and compute a secret key.
- This key can be used to send encrypted messages between users.
- Only users ‘i’ and ‘j’ can figure out the key, so no one else can read the messages.
- User ‘i’ knows that only user ‘j’ could have sent the message, confirming their identity. (Authentication, and Eventually Resistance to Replay attack).

MITM on DHKE


Alice
Darth
Bob
Private key X_A
public key
$Y_A = \alpha^{X_A} \bmod q$
Y_A
Private keys X_{D1}, X_{D2}
public keys
$Y_{D1} = \alpha^{X_{D1}} \bmod q$
$Y_{D2} = \alpha^{X_{D2}} \bmod q$
Y_{D2}
Y_{D1}
Secret key
$K2 = (Y_{D2})^{X_A} \bmod q$
Secret key
$K2 = (Y_A)^{X_{D2}} \bmod q$
Private key X_B
public key
$Y_B = \alpha^{X_B} \bmod q$
Y_B
Secret key
$K1 = (Y_B)^{X_{D1}} \bmod q$
Secret key
$K1 = (Y_{D1})^{X_B} \bmod q$
Alice and Darth
share $K2$
Bob and Darth
share $K1$

MITM on DHKE (Contd..)

- Alice and Bob think they have a secret key together.
- But Bob has a secret key (K_1) with Darth, and Alice has a secret key (K_2) with Darth.
- This setup allows Darth to intercept and manipulate messages between Alice and Bob.

- Alice sends a PT to Bob, encrypted with K_2: $E(K_2, PT)$.
- Darth intercepts this message and decrypts it to get PT.
- Darth can send the original message PT to Bob, encrypted with K1: $E(K_1, PT)$
- Or Darth can even send a modified message to Bob, also encrypted with K_1: $E(K_1, PT_m)$
- Hence, there is severe threat of Authentication + Confidentiality.