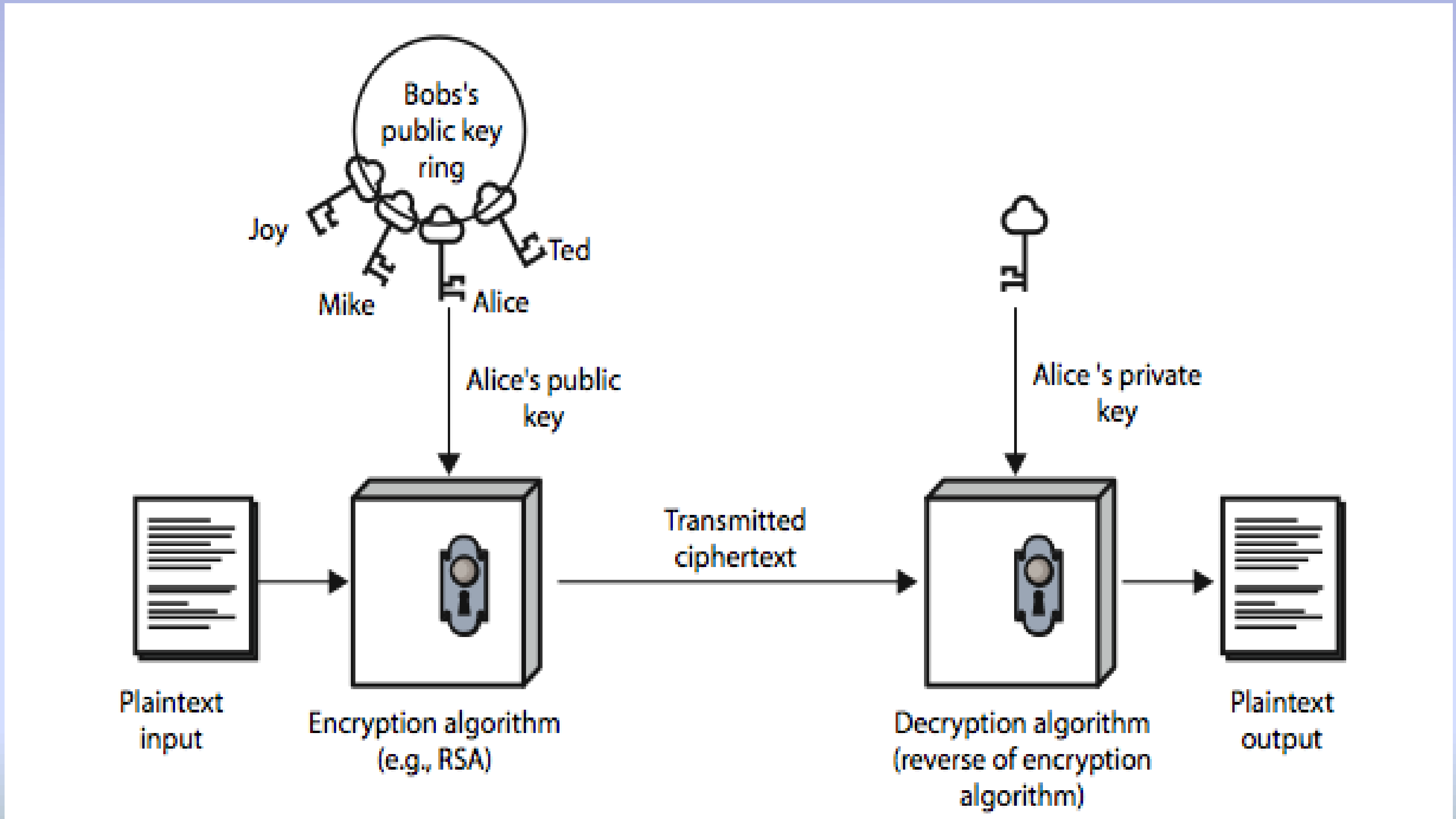
Public Key Cryptography

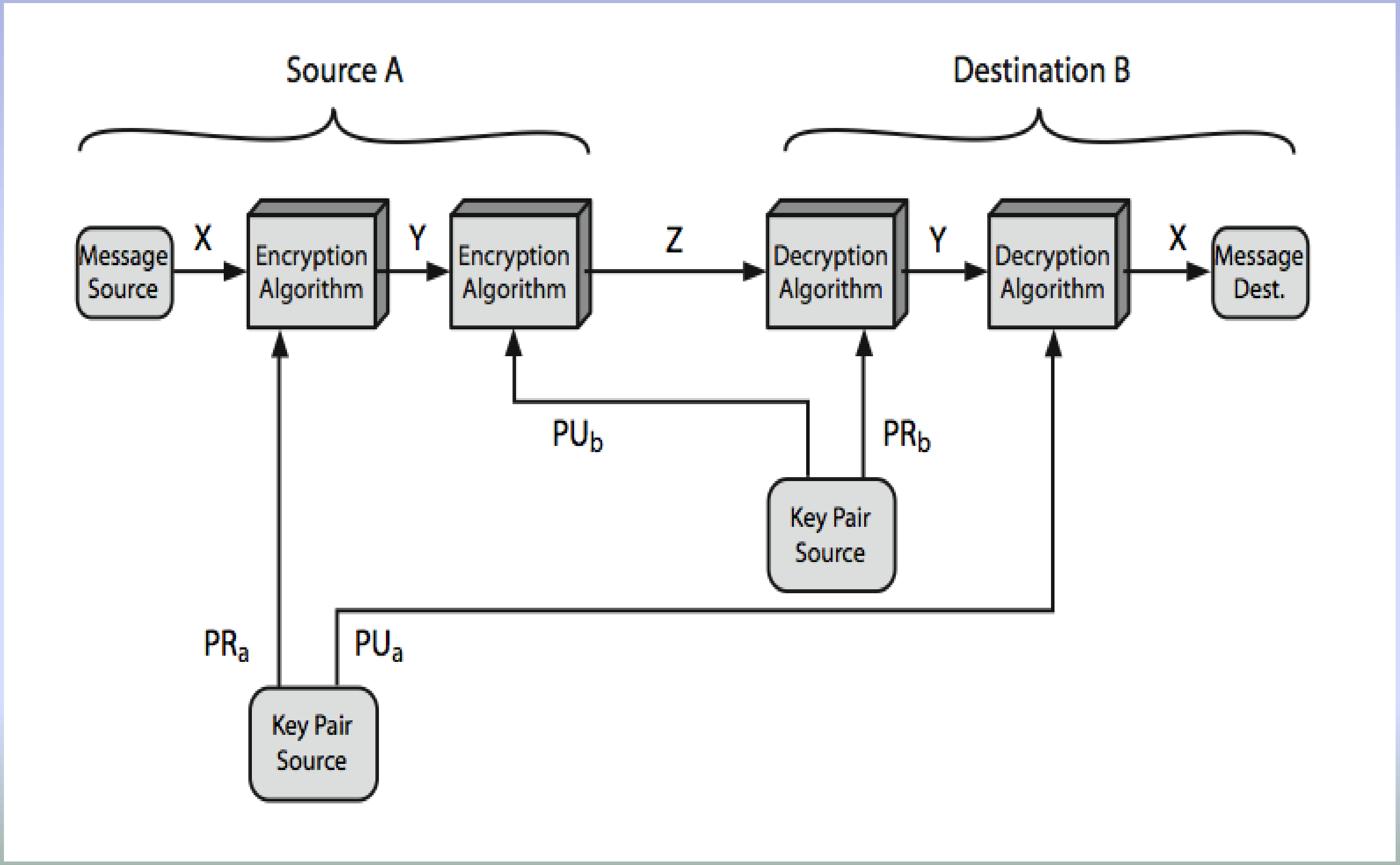
- **Operations**: Addition, Multiplication, and exponentiation
- Security is based on complexity of mathematical functions (intractable problems)
- **Examples:** RSA, Diffie Hellman, Rabin, ElGamal, Elliptic curve cryptography etc.

Confidentiality without authentication



(a) Encryption

Confidentiality with Authentication



Intractable problems

- **Integer factorization**: n = p * q. Finding primes p and q, given n. Eg: RSA (Soln: NFS with subexponetial running time)
- **Discrete Logarithmic Problem**: $x = y^z \% p$. Finding z, given x, y and p. Eg: Diffie Hellman alg, ElGamal Alg, DSA. (Soln:Pollard rho with Fully Exp, NFS with sub exponential time)
- **ECDLP:** P= nQ, where P and Q are points on elliptic curve. Finding n, given P and Q. Eg: ECC (Soln: Pollard rho having Fully exponential time)

Asymmetric Key Algorithms

- RSA Algorithm
- Diffie Hellman Algorithm
- DSA-Digital Signature Algorithm
- Elliptic Curve Cryptography

RSA Algorithm

- Key Generation:
  - Select any two distinct prime numbers p & q.
  - Compute $n=p*q$ and $\varphi(n) = (p-1)*(q-1)$
  - Select a random number 'e' such that $1<e<\varphi(n)$ and $\gcd(e, \varphi(n)) =1$.
  - Compute e's inverse $d=e^{-1} \bmod \varphi(n)$
  - Public Key = (n,e); Private Key = (p,q,d)

RSA…

- Encryption: Let $x<n$ be a plain text (number)
  - $y = E_{Pub}(x) = x^e \bmod n$
- Decryption:
  - $x = D_{pvt}(y) = y^d \bmod n$
- Note: Message is represented as a seqn of 0's and 1's. Find a no k such that $2^k < n$ (i.e., $k=\log_2 n$). Then k bits of message (whose integer value is x) is encrypted. For better speed, take k s.t, $2^k < n <= 2^{k+1}$
- Above encr/decr provides confidentiality. Swap(Pub, Pvt) keys in the above for providing authentication

RSA Example.

- Assume the following:
  - Sender: A   Receiver: B
  - B's (pub, pvt) pair generation:
    - B selects primes p=17, q=11 and computes n=p*q=187 and φ(n) = (p-1)*(q-1)=160
    - B selects a random no e=7 satisfying the reqd condition and calculates its inverse d as 23 using extended Euclid's algorithm.
    - B's public Key=(n,e)=(187,7) and private key = (p,q,d) = (17,11, 23). B announces its public key to all

- A wants to send encrypted message to B
  - ‘A’ generates a message and say its integer value is x=88 (<n)
  - ‘A’ encrypts the plain text x with B’s public key as: $y=88^{7} \bmod 187 = 11$ and transmits it to B
  - ‘B’ decrypts the cipher text 11 as $x=11^{23} \bmod 187 = 88$.

Diffie-Hellman Key Exchange Alg

- **Global Public Elements:**
  - Select prime: q
  - Find primitive root of q: α (<q)
- **Sender A:**
  - Select a random no X_A (Private of A)
  - Find $Y_A = \alpha$ ^ X_A% q (Public of A)

- **Receiver B:**
  - Select a random no X_B (Private of B)
  - Find $Y_B = \alpha \wedge X_B \% q$ (Public of B)
- Secret Key Generation at A:
  - $K = (Y_B \wedge X_A) \% q$
- Secret Key Generation at B:
  - $K = (Y_A \wedge X_B) \% q$

Drawback of Public Key Crypt

- Time Consumption for Encryption and decryption.
- A Secret key algorithm is at least 1000 times faster than a public key algorithm.
- **Solution:** Hybrid cryptosystem.
- Use public key algorithm for key sharing and secret key algorithm for encryption & decryption. Used in IBM**.**

Key sizes for equivalent security levels

Symmetric	ECC	DH/DSA/RSA
80	160	1024
128	256	3072
192	384	7680
256	512	15360

Key sizes for equivalent security levels

Symmetric	ECC	DH/DSA/RSA
80	160	1024
128	256	3072
192	384	7680
256	512	15360

Computational Efforts for Cryptanalysis

- Computing power needed to compute ECDLP using Pollard rho method.

Key size	**MIPS – years**
160	$8.5 \text{ X } 10^{11}$
186	$7.0 \text{ X } 10^{15}$
234	$1.2 \text{ X } 10^{23}$
354	$1.4 \text{ X } 10^{11}$
426	$9.2 \text{ X } 10^{51}$

- Computing power needed to compute integer factorization using NFS method.

Key size	**MIPS – years**
512	$3 \text{ X } 10^4$
768	$2 \text{ X } 10^8$
1024	$3 \text{ X } 10^{11}$
1280	$1 \text{ X } 10^{14}$
1536	$3 \text{ X } 10^{16}$
2048	$3 \text{ X } 10^{20}$

ECC Challenges

- Ref:http://www.certicom.com/index.php/the-certicom-ecc-challenge
- Announced in 1997.
- Solved 109 bits key in 2002 and 2004
- Unsolved for keys131, 163, 191, 239, 359 bits
- Prizes $ 20K, 30K, 40K, 50K, 100K

Top International Cryptography Conferences

- ASIACRYPT
- CRYPTO
- EUROCRYPT
- INDOCRYPT
- CHES- Cryptographic h/w & embd systems
- PKC – Workshop on practice & theory in Public Key Crypt
- ANTS- Algorithmic Number theory symposium
- Proceedings of above conferences are published by springer – LNCS

Some Useful S/W

- **LiDIA/LC 2.0.x.** It is a C++ library for computational number theory which provides a collection of highly optimized implementations of various multi-precision data types and time-intensive algorithms. The entire LiDIA functionality can be used interactively through LiDIA's LC interpreter. LC implements a subset of C++ and provides in addition to standard programming facilities, function overloading and automatic coercions. Functions and statements are treated as ordinary objects and may be manipulated at run-time. Because of the interpreted language, LC functions can be easily transformed to C++ programs which can then be compiled.

- **Pari-GP 2.1.x.** It is a Calculator for number theory. The PARI system is a package which is capable of doing formal computations on recursive types at high speed; it is primarily aimed at number-theorists, but can be used by people whose primary need is speed. It is possible to use PARI in two different ways: (1) as a library, which can be called from any upper-level language application, (2) as a sophisticated programmable calculator, named GP, which contains most of the standard control instructions of a standard language like C.

- GAP
- KANT/KASH
- Magma
- Maple
- Mathematica
- MuPAD
- Cryptlib

- Crypto++
- GNU MP
- Libgcrypt
- MIRACL
- NTL
- OpenSSL
- SAGE