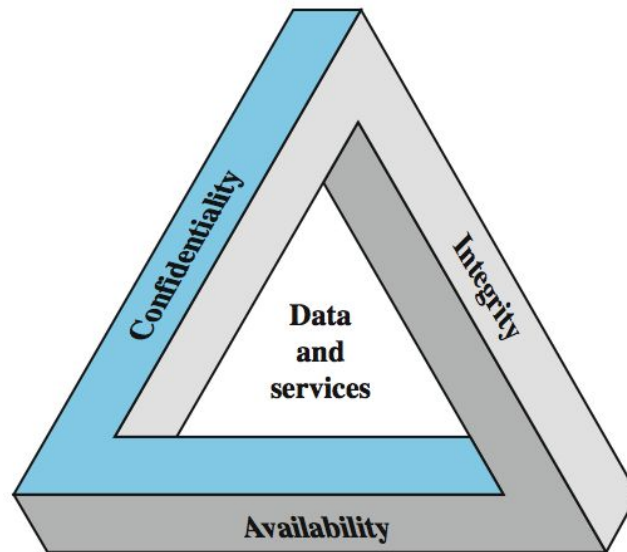


Background for Cryptocurrency Technologies

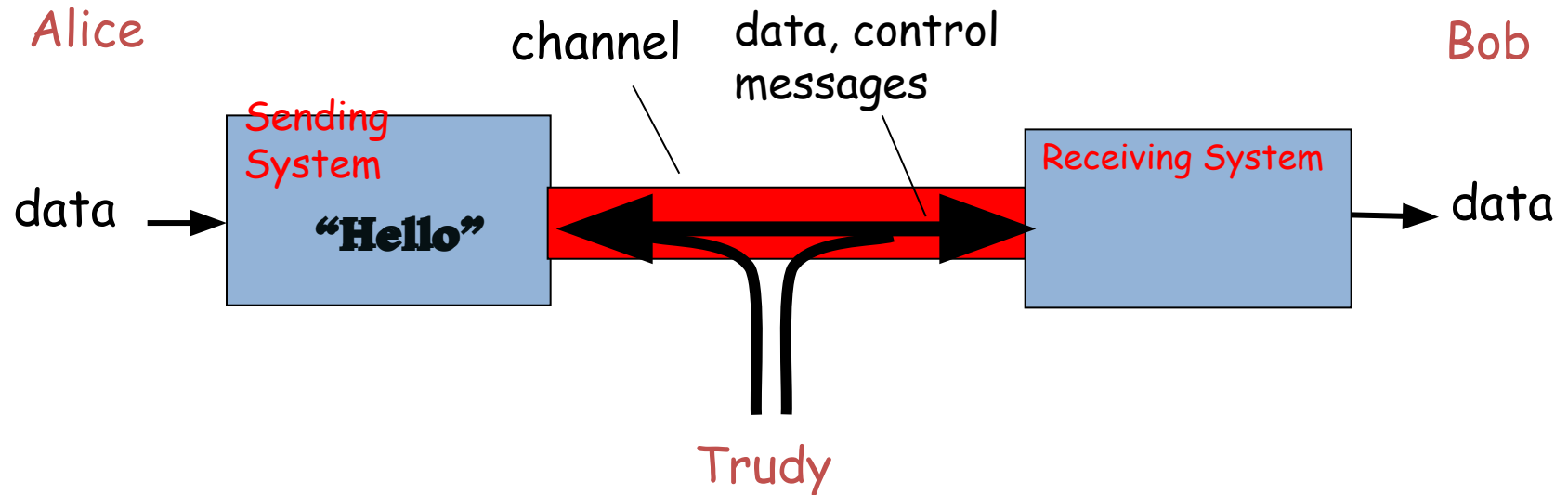
Cryptographic Tools

Information Security

Protection afforded to an automated information system in order to attain the applicable objectives of preserving the **integrity**, **availability** and **confidentiality** of information system resources (includes hardware, software, firmware, information/data, and telecommunications)



Confidentiality



Confidentiality: Preventing unauthorized disclosure of information or disclosure of information to unauthorized entities

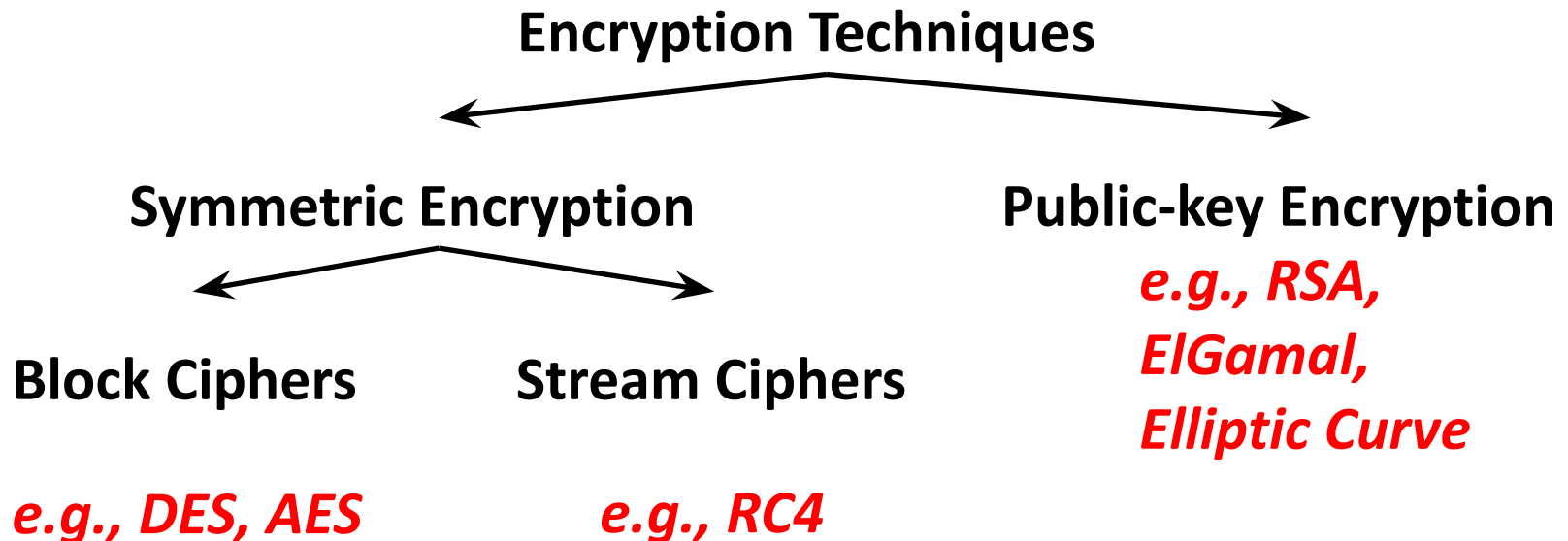
So, how to achieve confidentiality?

Encryption or Encipherment!

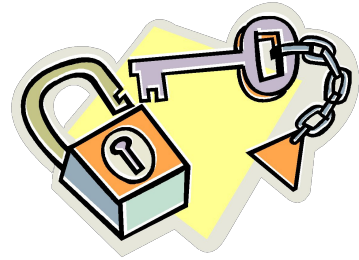
Encryption

What is Encryption (a.k.a Encipherment)?

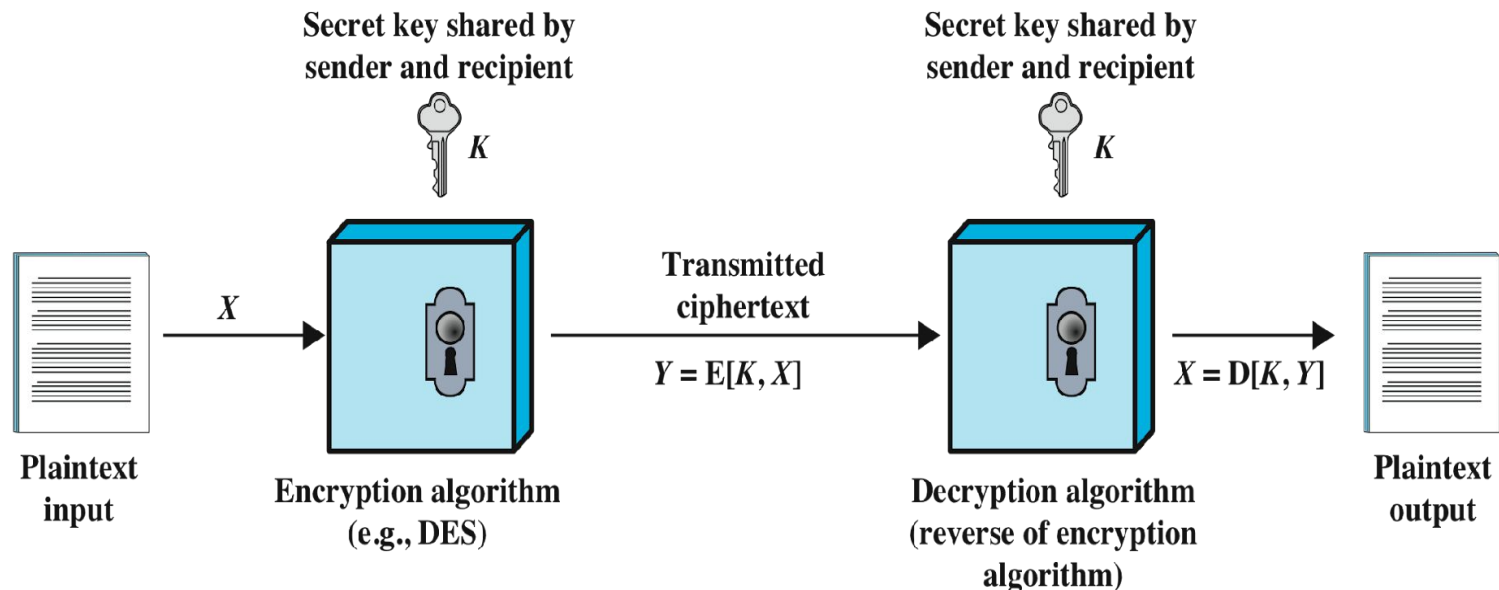
- Process of perturbing or transforming information that needs to be protected (a.k.a ***plaintext***) into something that is unintelligible (a.k.a ***ciphertext***) to everyone except authorized receivers.



Symmetric Encryption



- Encryption algorithm based on permutation and substitution operations
- Algorithm uses the same key for encryption and decryption - also referred to as **single-key** encryption
- Two requirements for secure use:
 - Needs a strong encryption algorithm
 - Sender and receiver must have obtained copies of the secret key in a secure fashion and must keep the key secure



Attacking Symmetric Encryption

1. Brute-Force Attack

- Try all possible keys on some ciphertext until an intelligible translation into plaintext is obtained
 - On average half of all possible keys must be tried to achieve

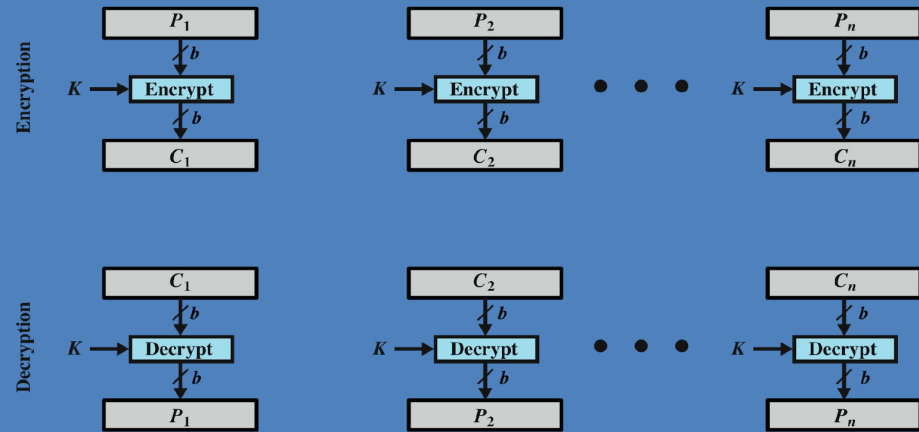
| Key Size (bits) | Number of Alternative Keys | Time Required at 1 Decryption/ μs | Time Required at 10^6 Decryptions/ μs |
|-----------------------------|--------------------------------|---|--|
| 32 | $2^{32} = 4.3 \times 10^9$ | $2^{31} \mu s = 35.8$ minutes | 2.15 milliseconds |
| 56 | $2^{56} = 7.2 \times 10^{16}$ | $2^{55} \mu s = 1142$ years | 10.01 hours |
| 128 | $2^{128} = 3.4 \times 10^{38}$ | $2^{127} \mu s = 5.4 \times 10^{24}$ years | 5.4×10^{18} years |
| 168 | $2^{168} = 3.7 \times 10^{50}$ | $2^{167} \mu s = 5.9 \times 10^{36}$ years | 5.9×10^{30} years |
| 26 characters (permutation) | $26! = 4 \times 10^{26}$ | $2 \times 10^{26} \mu s = 6.4 \times 10^{12}$ years | 6.4×10^6 years |

Attacking Symmetric Encryption

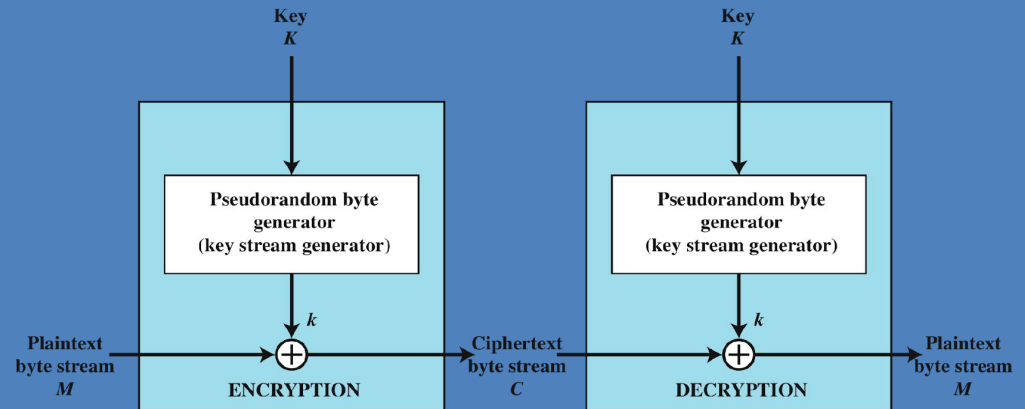
2. Cryptanalytic Attacks

- Relies on:
 - Properties of the algorithm
 - Knowledge of the plaintext
 - Sample plaintext-ciphertext pairs
- Exploits characteristics of the algorithm to deduce the specific plaintext or the key being used
 - If successful all future and past messages encrypted with that key are compromised

Block Cipher Encryption



(a) Block cipher encryption (electronic codebook mode)



(b) Stream encryption

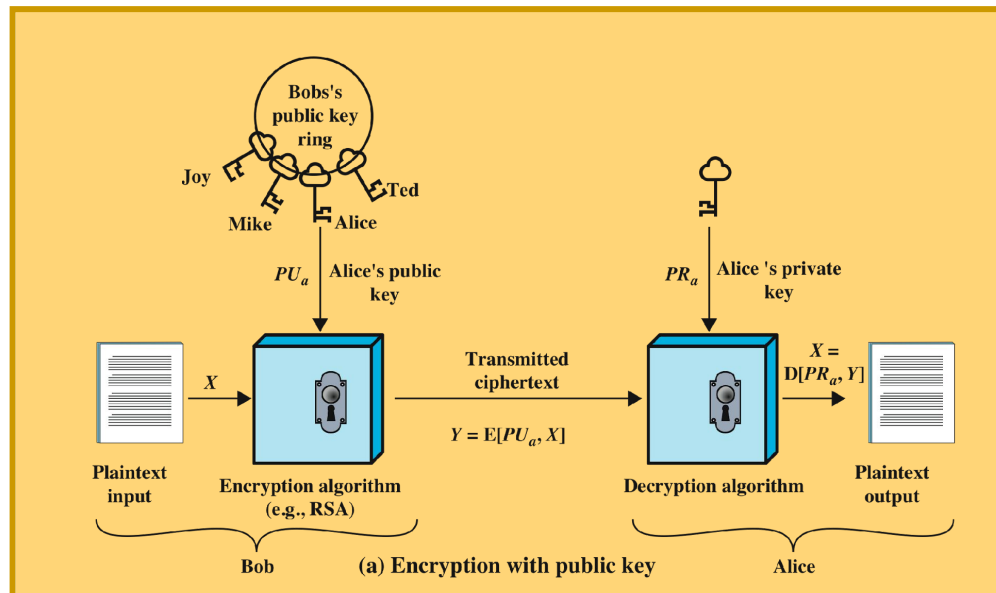
Stream Encryption

Figure 2.3 Types of Symmetric Encryption

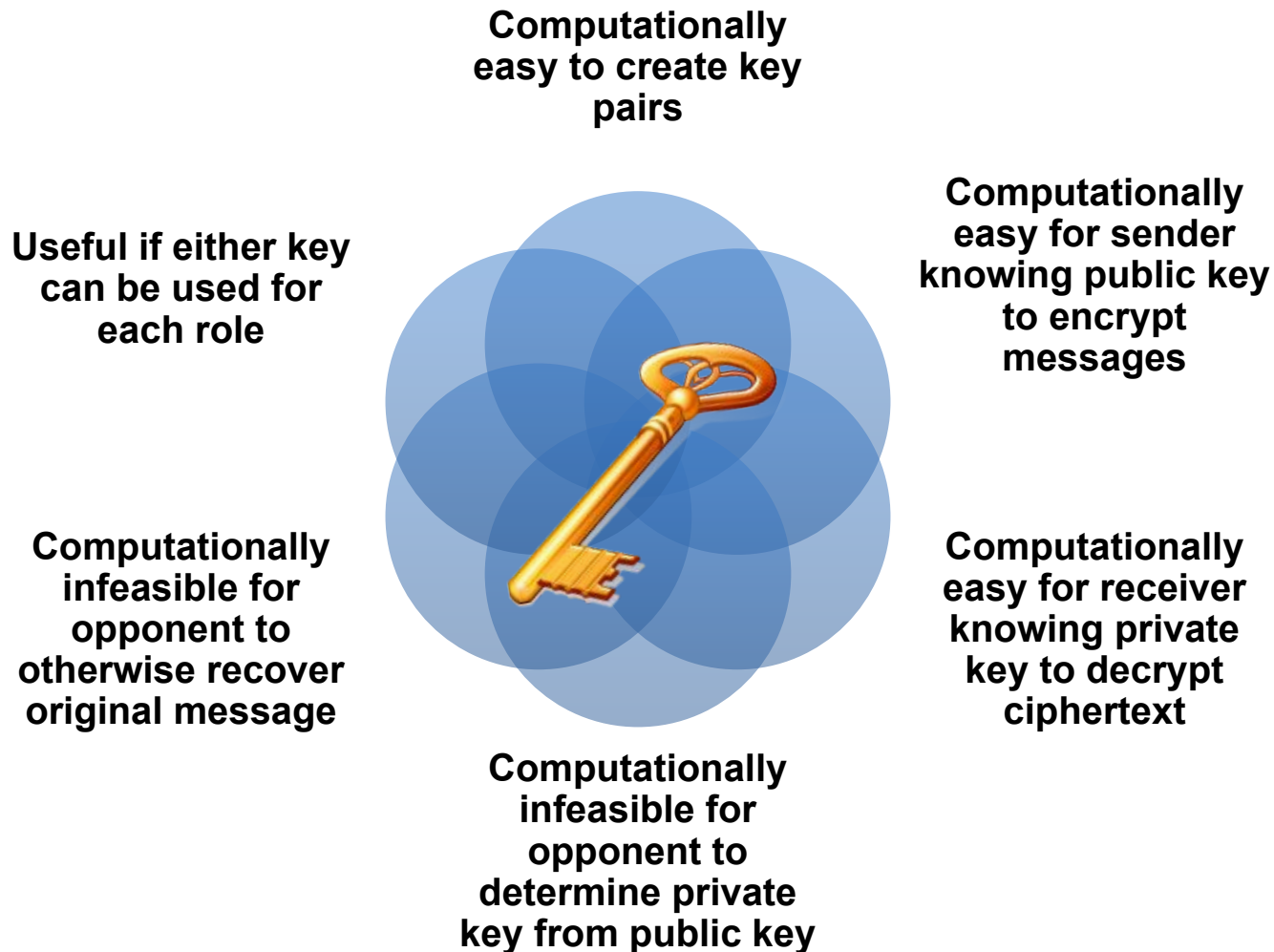
Public-Key Encryption



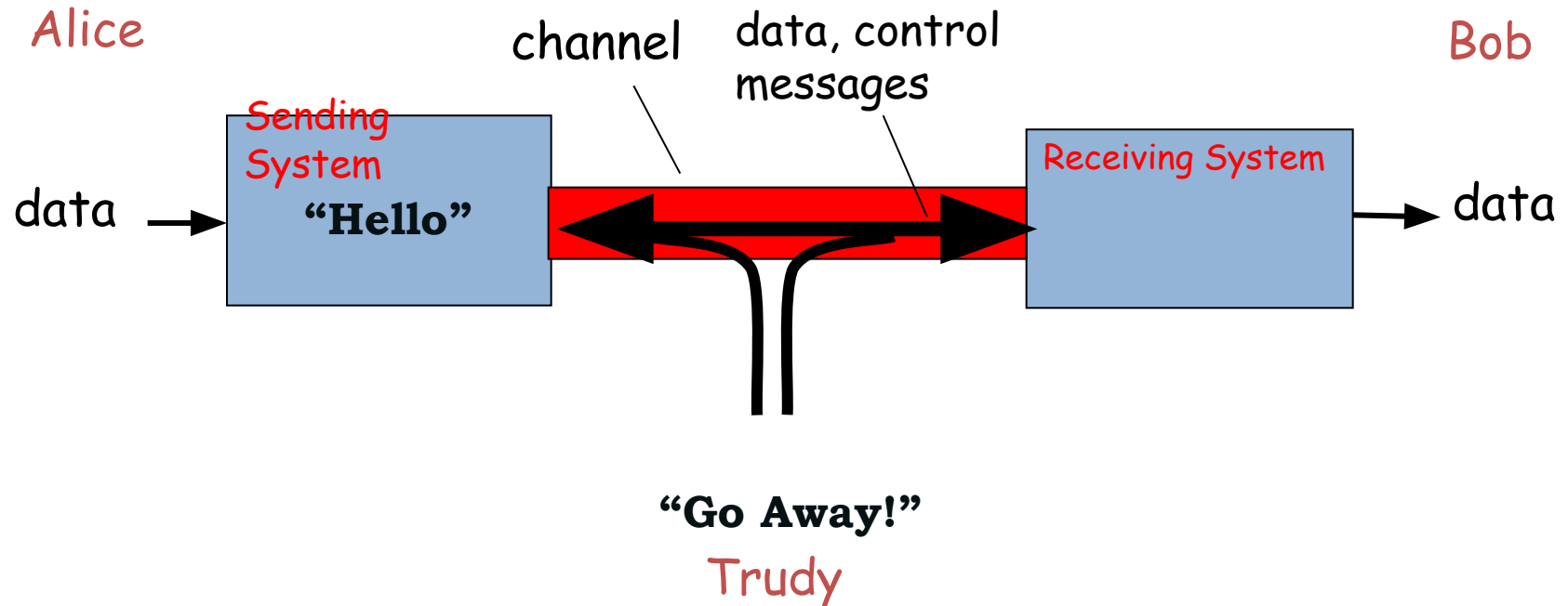
- Encryption algorithms rely on mathematical functions that are easy to compute but difficult to invert!
- **Asymmetric** - uses two separate keys
 - **Public key** is made public for others to use
 - **Private key** is secret and is never released



Requirements for Public-Key Cryptosystems



Integrity

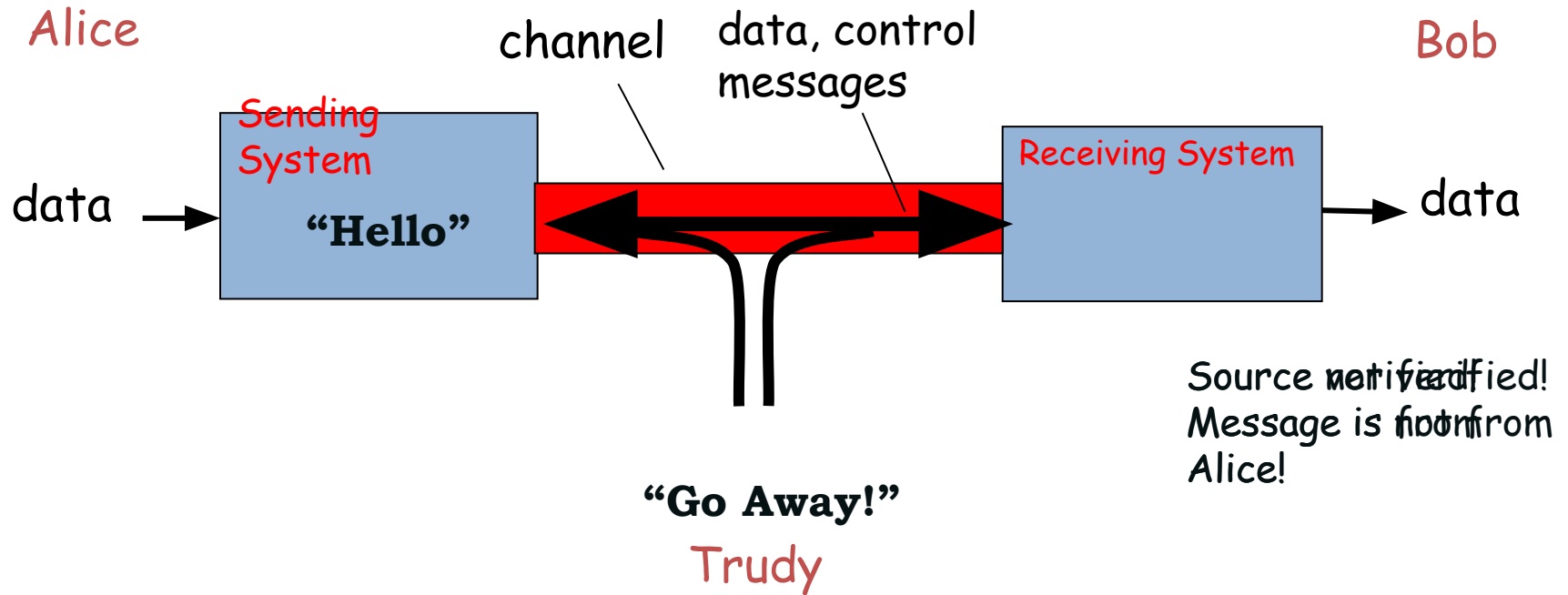


Integrity: Preventing unauthorized modification of information or modification of information by unauthorized entities

So, how to achieve Integrity?

Message Authentication Codes (MAC)!

Authenticity



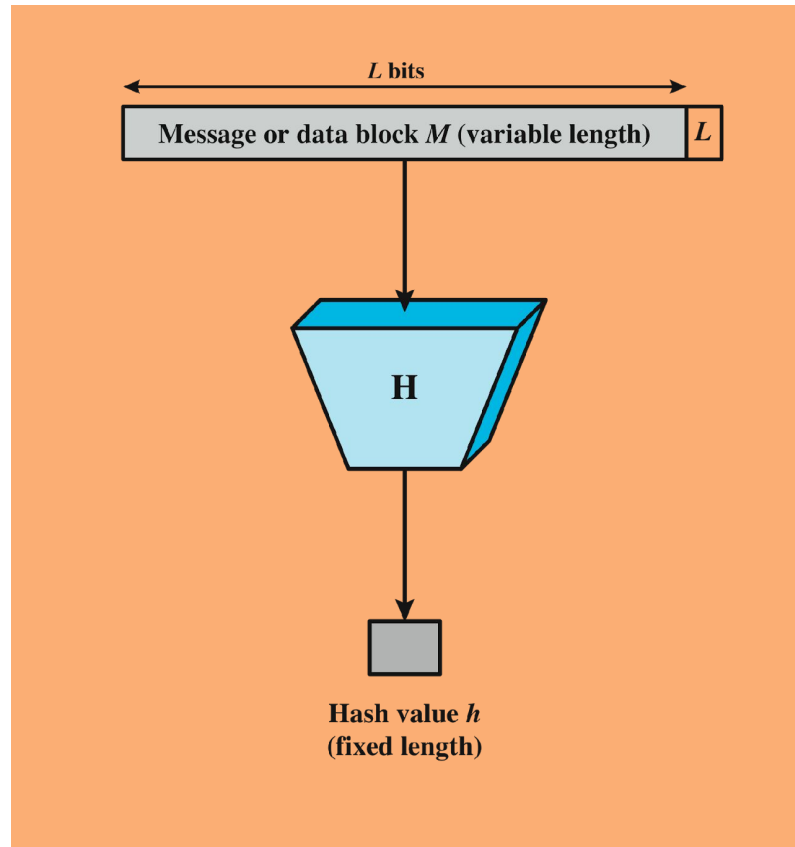
Authenticity: Integrity protection + enabling verification of the authenticity of the information (and its origin)

So, how to achieve Authenticity?

Digital Signatures!

Secure Hash Functions

Important cryptographic primitive required for both Message Authentication Codes and Digital Signatures



Hash functions are key in the design of cryptocurrencies – will be clear later in the class

Hash Function Requirements

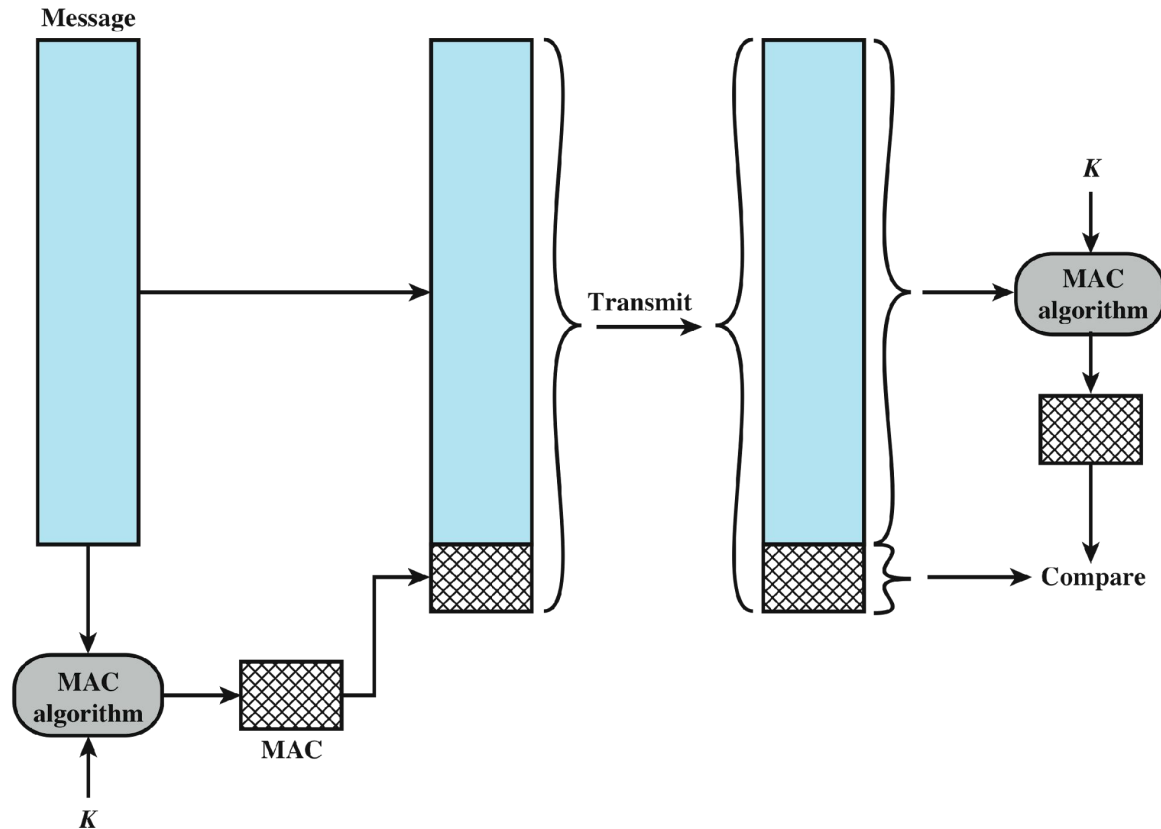
- Can be applied to a block of data of any size
- Produces a fixed-length output
- $H(x)$ is relatively easy to compute for any given x
- **One-way or pre-image resistant**
 - Computationally infeasible to find x such that $H(x) = h$
- Second pre-image resistant or **weak collision resistant**
 - Computationally infeasible to find $y \neq x$ such that $H(y) = H(x)$
- Collision resistant or **strong collision resistance**
 - Computationally infeasible to find any pair (x, y) such that $H(x) = H(y)$

We will come back to these properties again later in the class and see how they are relevant to cryptocurrencies

Security of Hash Functions

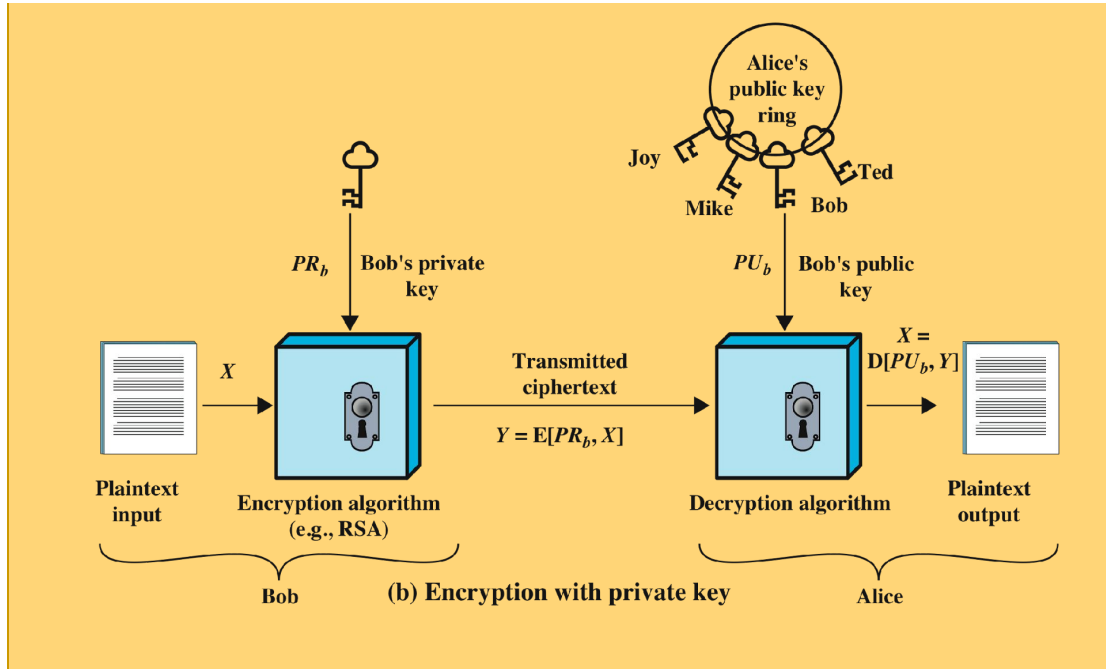
- Two approaches to attacking a secure hash function:
 - Cryptanalysis
 - Exploit logical weaknesses in the algorithm
 - Brute-force attack
 - Strength of hash function depends solely on the length of the hash code produced by the algorithm
- SHA family of algorithms are the most widely used hash functions
- Applications:
 - MACs and Digital signatures
 - Cryptocurrencies
 - Storage of passwords
 - Intrusion detection

Message Authentication Codes (MAC)



- MACs verify that information has not been altered and is from an authentic source.
- **But, can it really provide non-repudiation?**

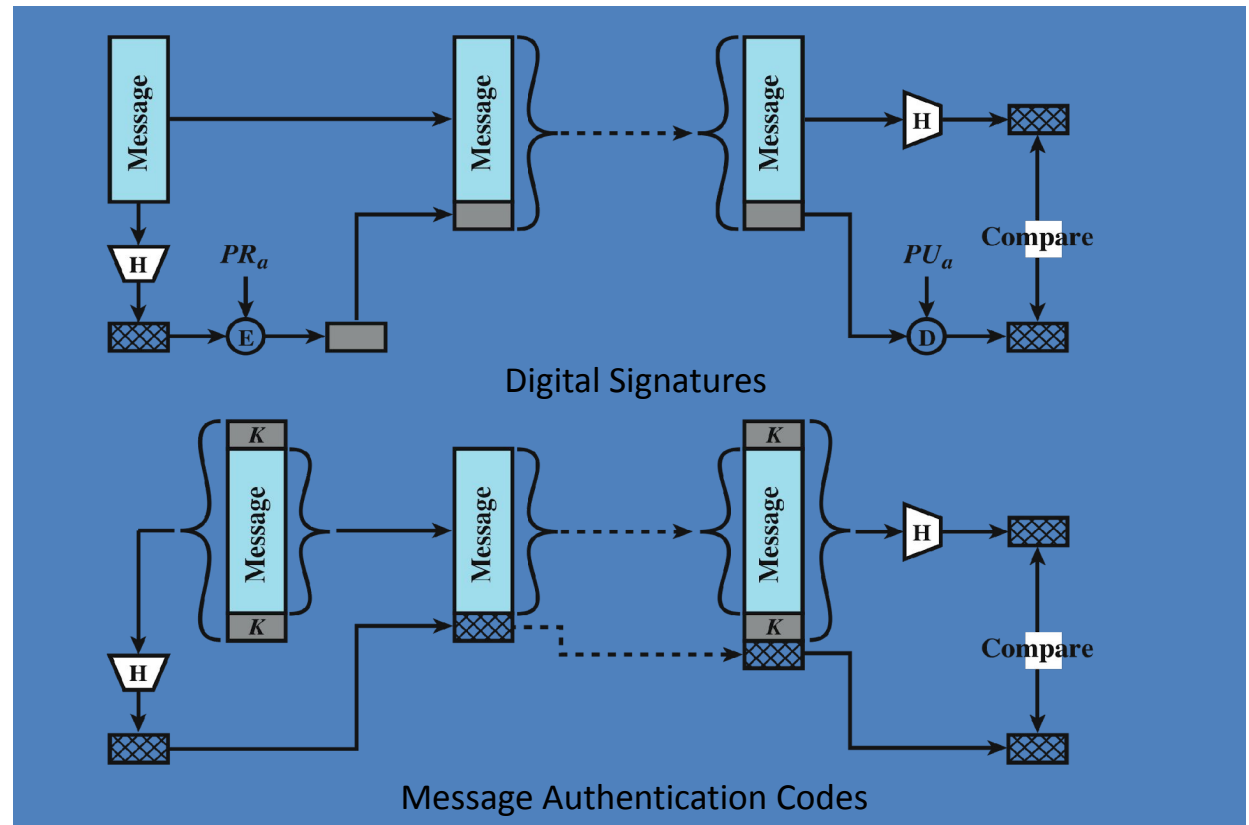
Digital Signatures



- Also known as **private-key encryption**.
- User **encrypts** data using his or her own **private key**
- Anyone who knows the corresponding **public key** will be able to **decrypt** the message
- Directed towards providing **authentication** with **non-repudiation**

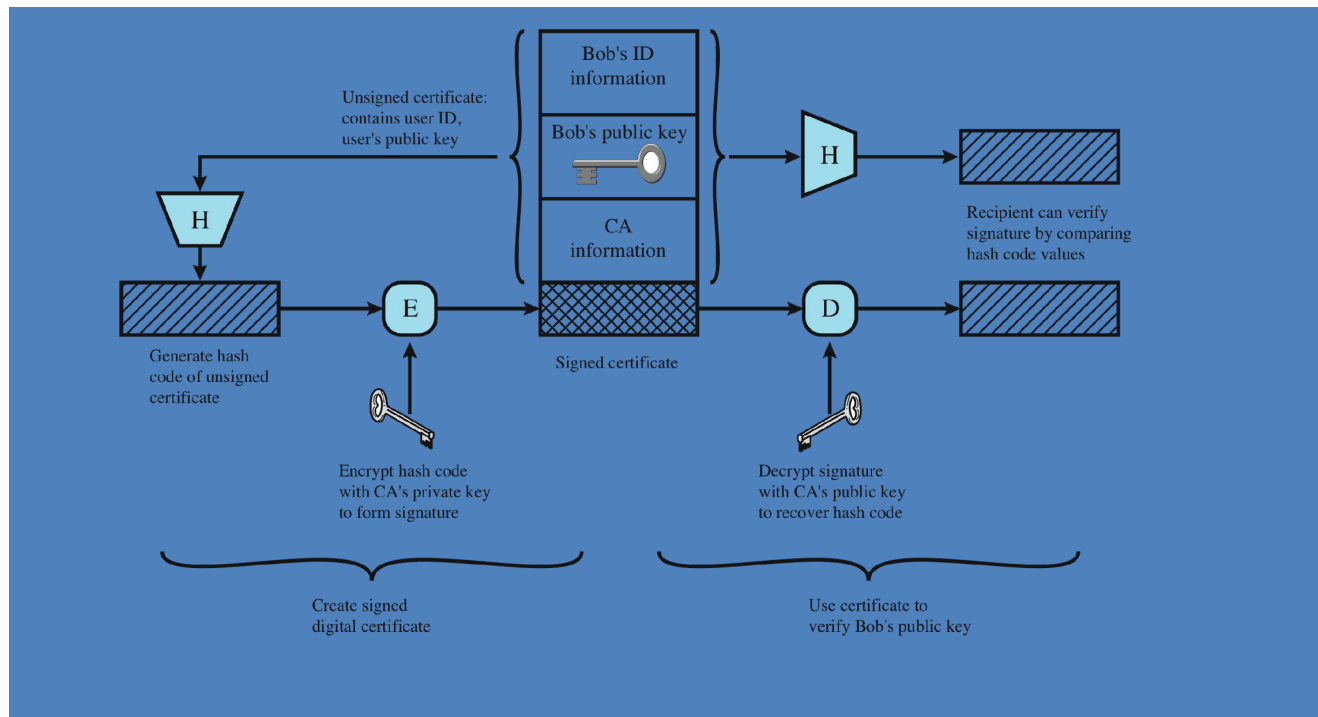
So, how do Secure hash functions enable MACs and Digital Signatures?

Secure Hash Functions in the realization of MACs and Digital Signatures

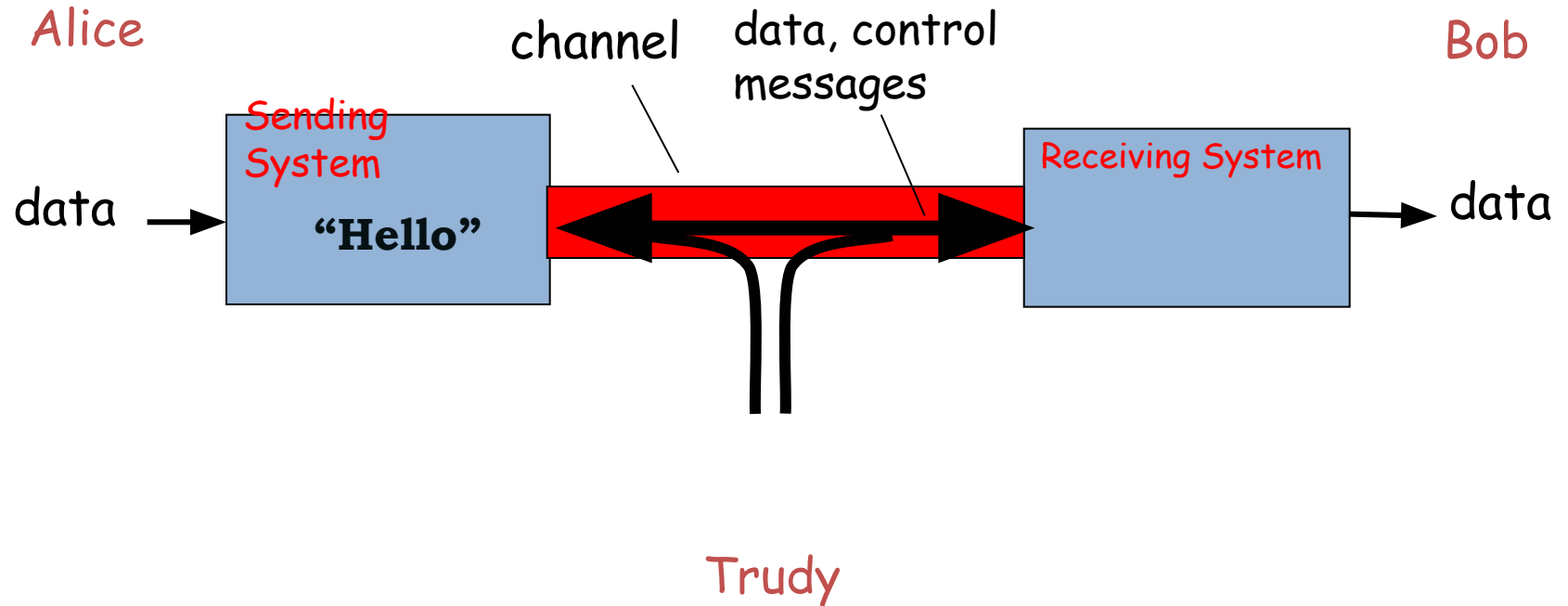


Public-Key Certificates

- How do receivers verify that the public key of the sender actually belongs to the sender?
- Senders employ public-key certificates to distribute their public-keys -> certificates contains the sender's public-key signed using some trusted third-parties private key.



Availability



Random Numbers

- Primarily used for:
 - Generating keys for public-key and symmetric algorithms
 - Generating key stream for symmetric stream cipher
 - Sequence numbers in networking protocols
 - Many other uses in cryptographic algorithms and networking protocols
 - Including in cryptocurrencies (we will see more later)

Random Number Requirements

Randomness

- Uniform distribution
 - Frequency of occurrence of each of the numbers should be approximately the same
- Independence
 - Each number is statistically independent of other numbers in the sequence
 - Opponent should not be able to predict future elements of the sequence on the basis of earlier elements

Random versus Pseudorandom

- Cryptographic applications typically use fixed algorithms for random number generation
 - Algorithms are deterministic and therefore produce sequences of numbers that are not statistically random
 - Called pseudorandom numbers are:
 - Almost random or statistically close to random numbers
 - Sequences produced that satisfy statistical randomness tests
 - Likely to be predictable
- True random number generator (TRNG):
 - Uses a nondeterministic source to produce randomness
 - Most operate by measuring unpredictable natural processes
 - e.g. radiation, gas discharge, leaky capacitors
 - Increasingly provided on modern processors