Data Encryption System

(Security of Information on Internet)

Parimal Sudas
CSE/IT
Dronacharya College of Engg.

Security in Distributed Systems

1. Introduction and Design Issues
   a) Threats to security systems
   b) Security Mechanisms
   c) Design Issues: Focus of Data Control, Layering of security mechanisms and Simplicity

2. Cryptography
   a) Basics of Cryptography
   b) Types of encryption: Symmetric, Asymmetric and Hash Functions

3. Secure Channels
   a) Using Symmetric Keys
   b) Using Public/Private Keys
   c) Signing/Digital Signatures

4. Secure Mobile Code
   a) Sandboxing

Security Threats

4 possible threats to security in computer systems:

- Interception: unauthorized party has gained access to a service or data e.g. eavesdropping, illegal copying of data (or files)

- Interruption: when a service or data becomes unavailable, unusable, destroyed etc. e.g. when a file is lost or corrupted, denial of service by malicious attacks

- Modification: unauthorized changing the data or service.
  - intercepting and changing transmitted data
  - changing the behaviour of a service
  - altering database entries

- Fabrication: generating data or activity that would not normally exist. For example, adding a password into a password file or database or falsifying a service.

Security Mechanisms

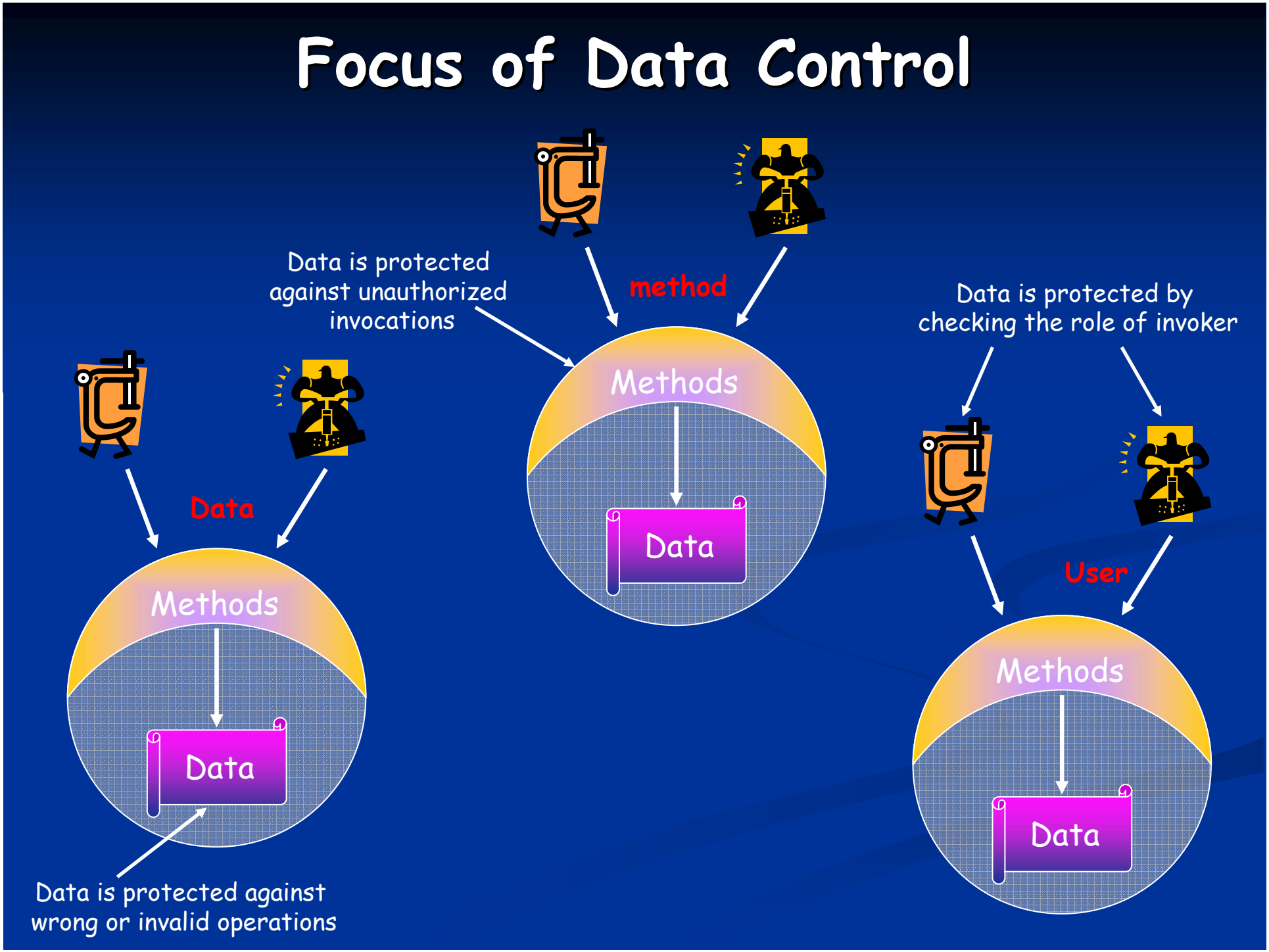
- Need a Security Policy
- Look at security mechanisms by which a policy can be enforced:

- Encryption: transforms data (encrypts) into unintelligible data i.e. implements confidentiality, integrity
- Authentication: verify the claimed identity of the user e.g. passwords, public/private-keys
- Authorization: is user authorized to access the resource e.g. unix
- Auditing: track which clients accessed what

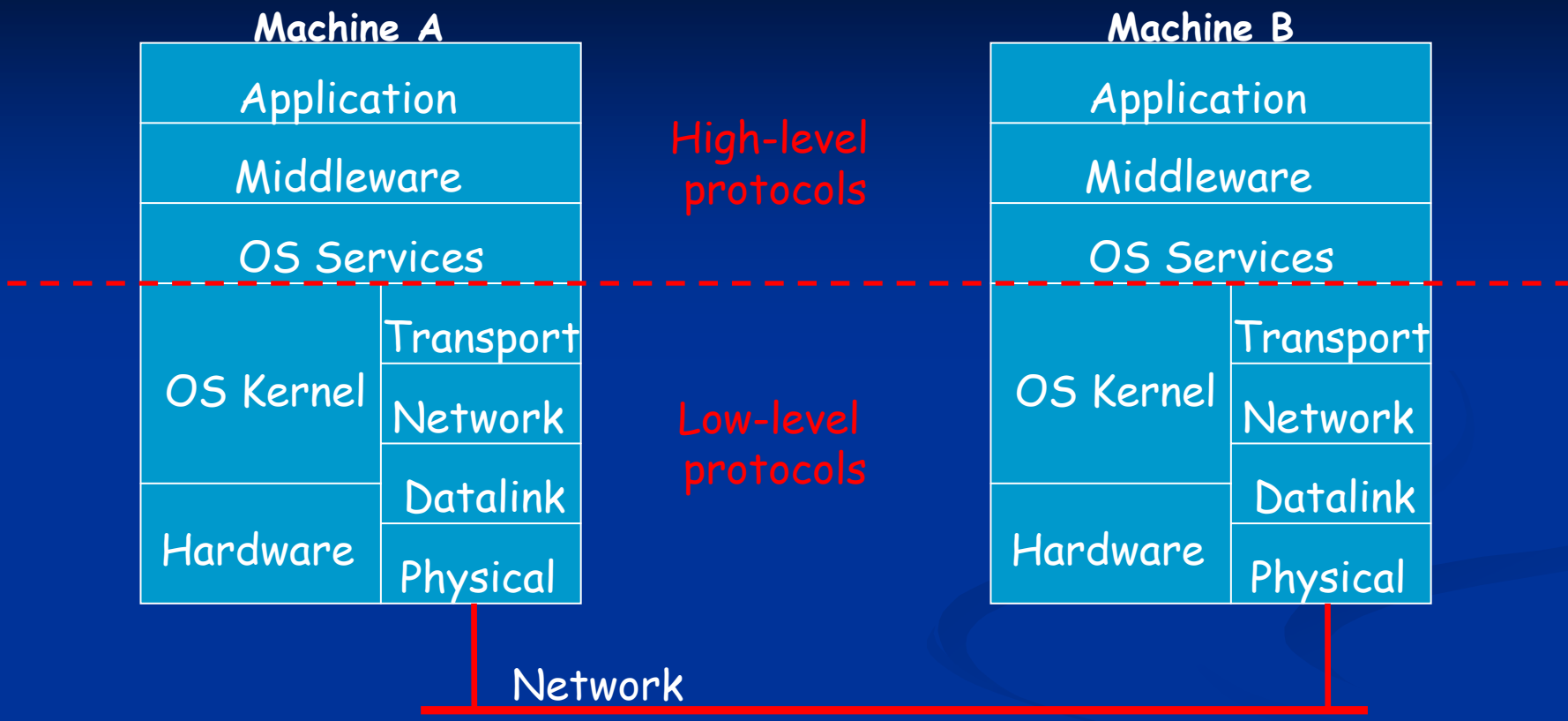
Design Issues

- Focus of Control
- Layering of Security mechanisms
- Simplicity

Focus of Data Control



Layering of Security Mechanisms



Issues:

- which level are the security mechanisms placed ?
  - communication or services (Middleware)
- depends on the trust of the client

Simplicity

The Good news …

The fewer security mechanisms the better !

- need to be easily understood
- and trusted to work
- simplicity will contribute to the trust that end users will put into the application

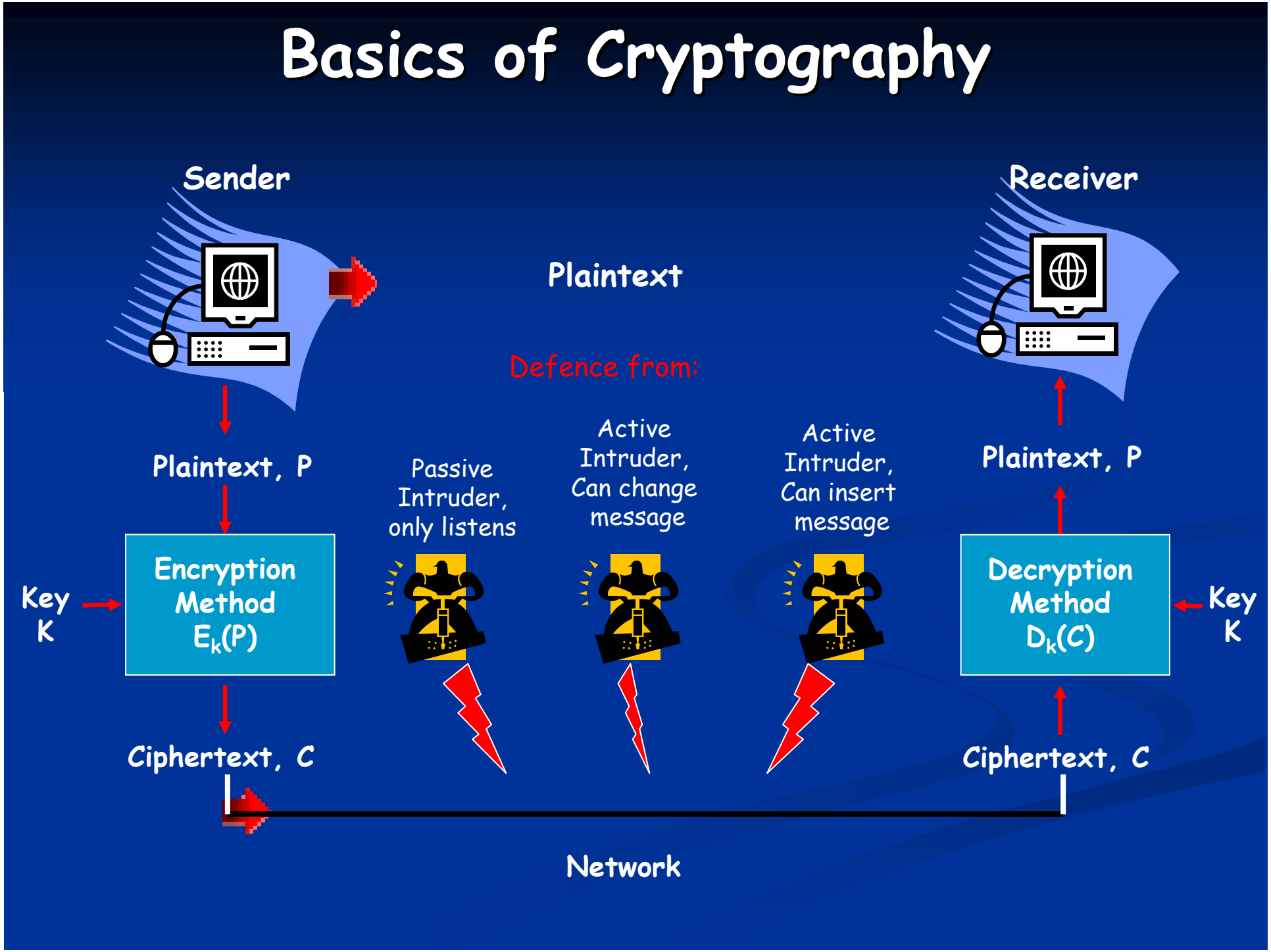
The Bad news ..

Often applications are too complex and security just makes it worse!

Cryptography

- Basics of Cryptography
- Types of Encryption
- Symmetric Encryption
- Asymmetric Encryption
- Hash Functions

Basics of Cryptography



Types Of Encryption

Text is converted to ciphertext by use of an algorithm and key

- Algorithm is publicly known
- Key is held private

Three Main Categories

- Secret Key (symmetric cryptosystem)
  - single key is used to encrypt and decrypt information
- Public/Private Key (asymmetric cryptosystem)
  - two keys are used: one for encryption (public key) and one for decryption (private key)
- One-way Function (hash functions)
  - information is encrypted to produce a "digest" of the original information that can be used later to prove its authenticity

Symmetric Encryption

Sender and receiver have same secret key that will encrypt and decrypt plain text

- Strength of encryption technique depends on key length
- Known symmetrical algorithms
  - Data Encryption Standard (DES) - 56 bit key
  - Triple DES, DESX, GDES, RDES - 168 bit key
  - RC2, RC4, RC5 - variable length up to 2048 bits
  - IDEA - basis of PGP - 128 bit key
  - Blowfish - variable length up to 448 bits

Example: Data Encryption Standard (DES)

- widely-used

- private (secret) key - judged so difficult to break it was restricted for export by US Gov.

- 72,000,000,000,000,000 (72 quadrillion) or more possible encryption keys
- key chosen at random - both sender and receiver must know and use the same private key

- can run in several modes and involves 16 rounds or operations

- many companies use "triple DES" - applies three keys in succession

- in 1997, Rivest-Shamir-Adleman, owners of another encryption approach, offered a $10,000 reward for breaking a DES message

- cooperative effort on the Internet of over 14,000 computer users trying out various keys finally deciphered the message, discovering the key after running through only 18 quadrillion of the 72 quadrillion possible keys!

Asymmetric Encryption

- Better Known as Public/Private Key
  - user X has a pair of keys one public and one private
  - To encrypt a message to X use X's public key
  - X will decrypt encrypted message using X's private key that "matches" X's public key

• Most common algorithm is the RSA (Rivest Shamir Adelman) algorithm with key lengths from 512 to 1024 bits

• Uses modular arithmetic & elementary number theory
• based on the fact that it is extremely difficult to find the prime factors of large numbers.

o Pretty Good Privacy (PGP),
o the Secure Sockets Layer (SSL),
o S/MIME, Secure Electronic Transactions (SET),
o Secure Shell (SSH).
o X. 509 V.3 certificates as used in JXTA, Globus/OGSA
o included in WWW browsers e.g. Netscape and Microsoft Internet Explorer

Hash Functions

- One-Way Functions
  - non-reversible "quick" encryption
  - produces a fixed length value called a *hash* or *message digest*
  - used to authenticate contents of a message
  - Common message digest functions
    - MD4 and MD5
      - produces 128 bit hashes
    - SHA
      - produces 160 bit hashes

Secure Channels

- Protecting communication between two users e.g. peer-to-peer or client server
- Two Types:
  - Symmetric
    - Shared Secret Keys
    - Session Keys
  - Asymmetric i.e. Public/Private Key

Uses of Secure Channels

- Secure Socket Layer (SSL):
  - helps improve the safety of Internet communications
  - standard for encrypted client/server communication
  - protocol that runs on top of TCP/IP
  - utilizes several security techniques e.g. *public keys*, *symmetric keys*, and *certificates*.
  - web sites commonly use SSL to guard private information such as credit card numbers.

- Transport Layer Security (TLS):
  - protocol - ensures privacy between users
  - successor to the SSL.

Symmetric: Shared Secret Keys

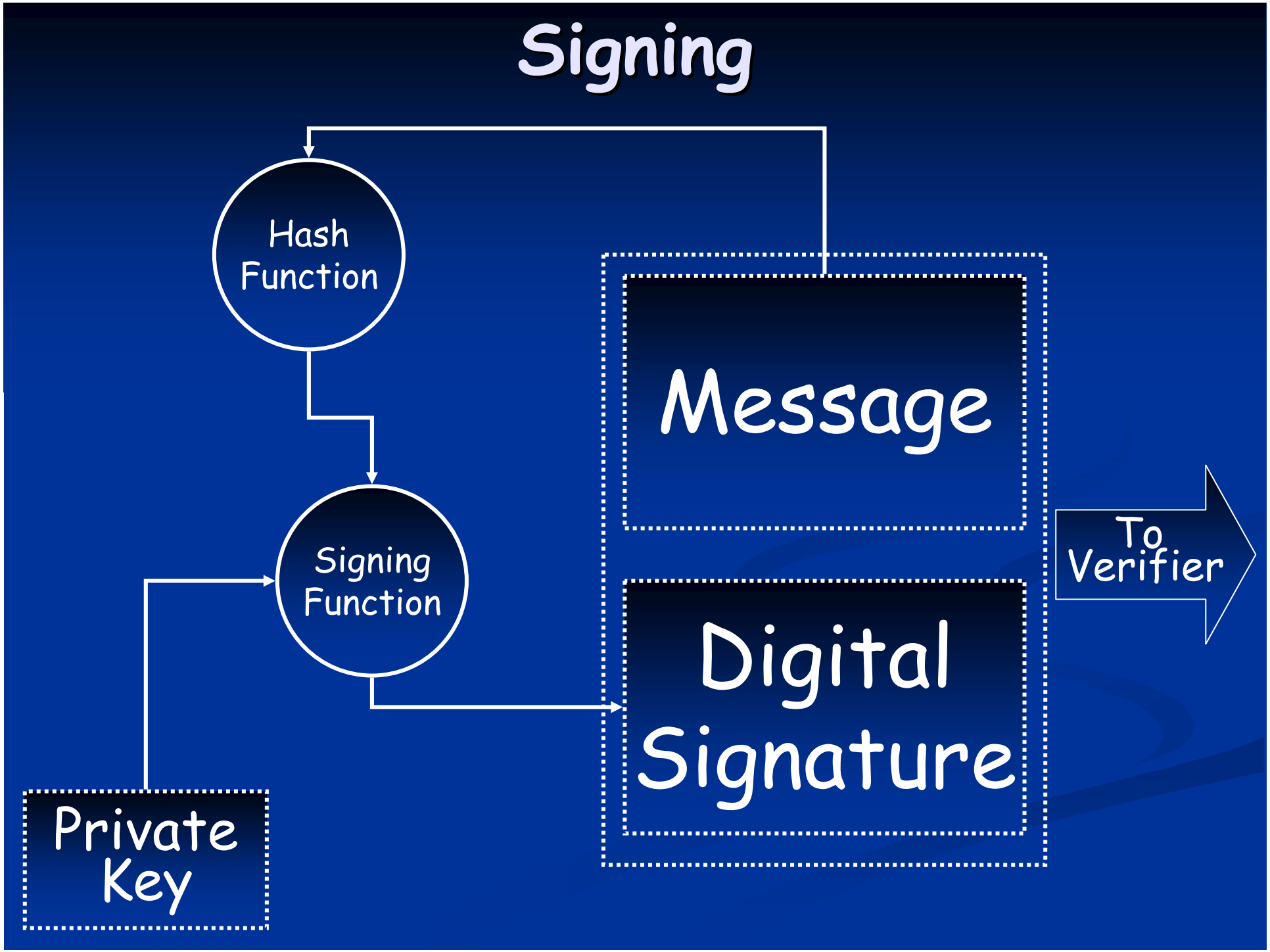
- generated once and secretly passed to the individuals
- This can be done in a number of ways:
  - other methods e.g. by using public-keys
  - telephone each other
  - post it to each other.
- Example system that uses this is Kerberos.

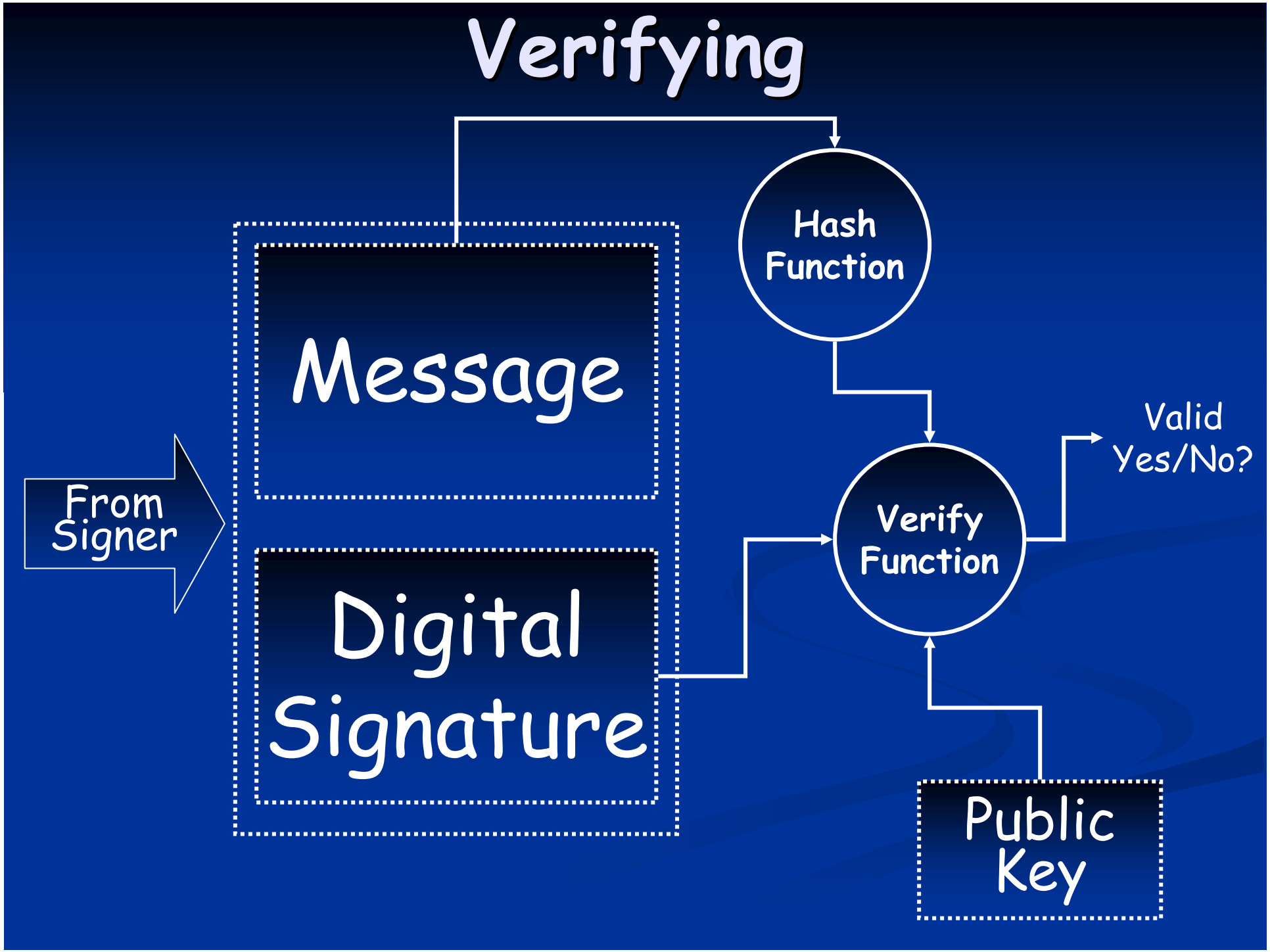
An Example

1. Digital Signatures
   a) Signing a document/email
   b) Verifying an email etc
2. Globus
   a) Globus Security
   b) Mutual Authentication

Digital Signatures

- Asymmetric cryptosystems allow users to digitally sign messages
  - allows a user to establish their authenticity.
- A hash function is used in the process of creating and verify a digital signature
  - Converts the document into a hash
  - Concise and efficient for calculation


Signing
Hash Function
Message
Signing Function
To Verifier
Digital Signature
Private Key


Verifying
Hash Function
Message
From Signer
Digital Signature
Verify Function
Valid Yes/No?
Public Key

Digital Signature Verification

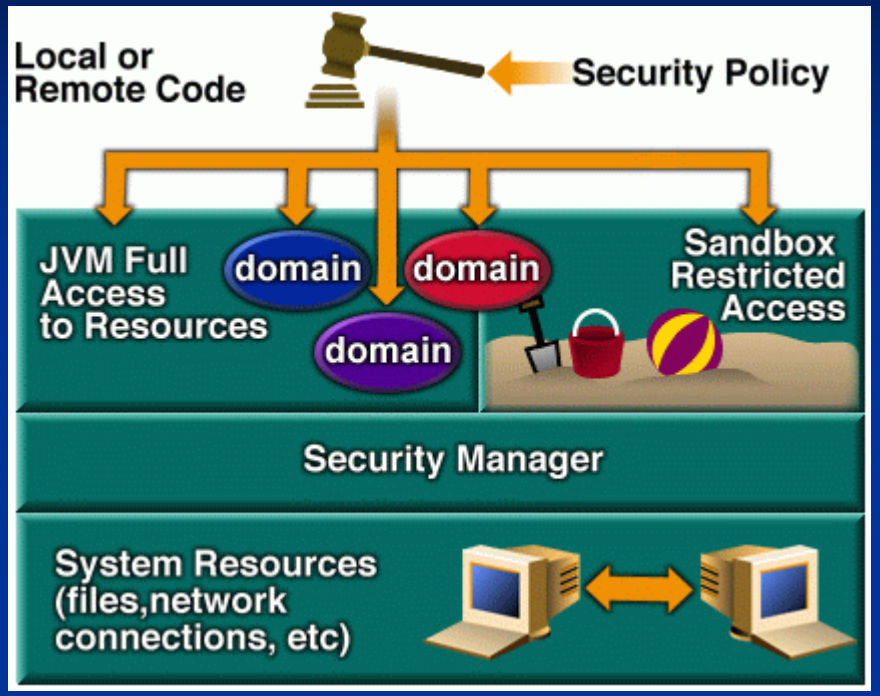
- Verification indicates that:
  - the digital signature was created by the signer (i.e. s/he is the only person with access to the private key)
  - that the message was not altered since it was signed (because has collisions are considered mathematically improbably).
- There exists a number of different mathematical formulas and procedures, but all share this overall operational pattern
- Note: Signing does not encrypt a message – it is merely a method of verifying identity
  - But encrypting a message with a private key also verifies a message – but much less efficient if this is its only purpose

Secure Mobile Code

- How do you trust remote code to run locally?
- traditionally, you prevented malicious code running on your system

• Use the **Sandbox** security model
• Sandbox is a technique by which a downloadable program is executed in such a way that each of its instructions can be fully controlled

Java Sandbox



A Java Sandbox prohibits:

- Reading or writing to the local disk
- Making network connections to any host, except the host that hosted the applet
- Creating a new process
- Loading a new dynamic library and therefore directly calling a native method

- Typically for applets
- …but for P2P also ? CPU sharing etc
- "Signed Applets" are trusted and treat like local code