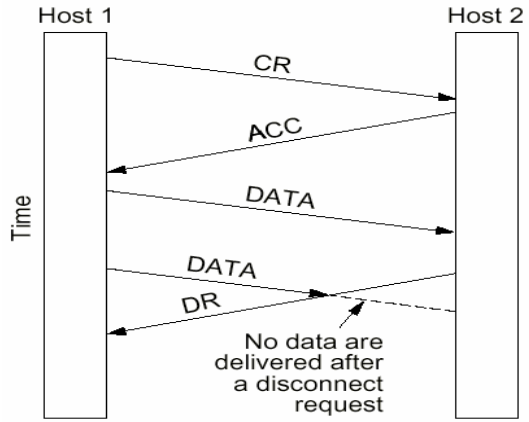
# Transport Layer(9)

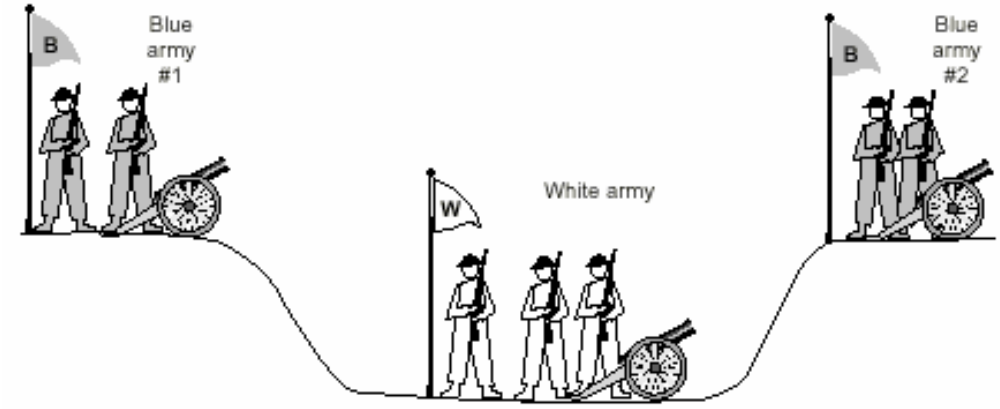
## Releasing a Connection

Easier than establishing one, but still with pitfalls

## Abrupt disconnection with loss of data



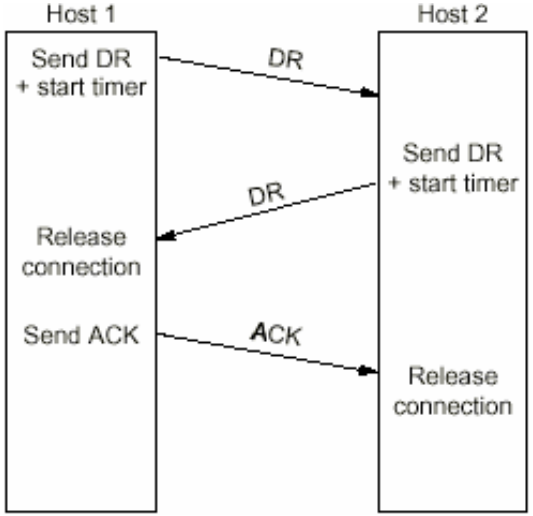

## The two-army problem



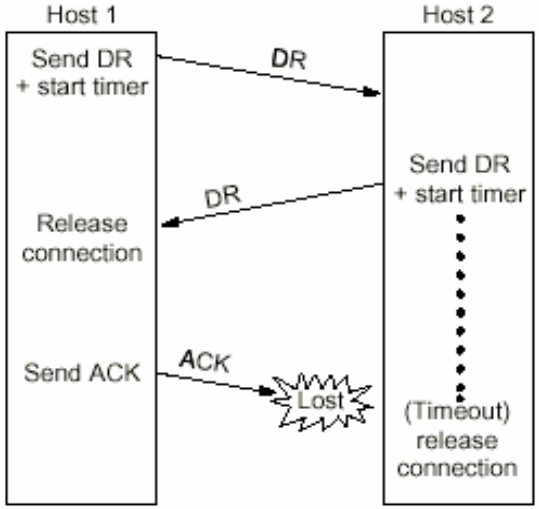
# Transport Layer(10)

## Four scenarios of releasing using a three-way handshake

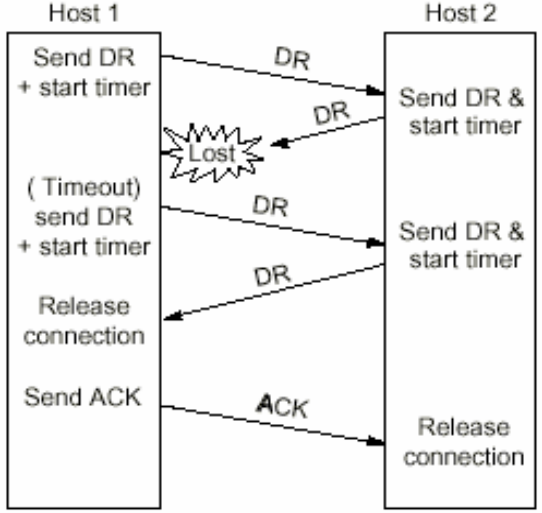
a) normal use



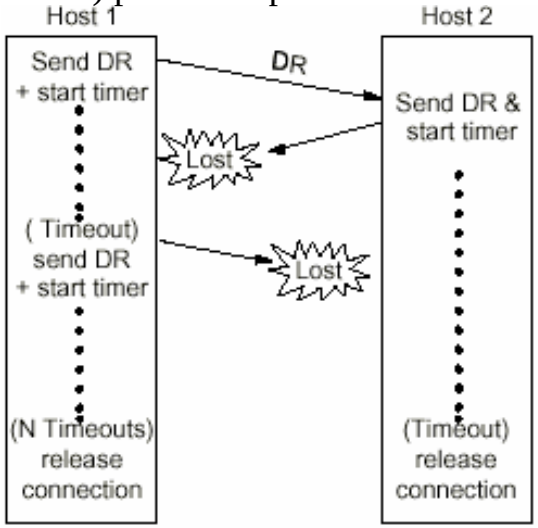

b) final *ACK* TPDU is lost




c) second DR lost




d) case c) plus attempts to retransmit also fail



# Transport Layer(11)

**Flow control and buffering**

There are similarities and differences w.r.t. to the same topic in the data link layer.

**Basic Similarity:**

sliding window or other scheme is needed on each connection to keep a fast transmitter from overrunning a slow receiver

**Main difference:**

a router usually has relatively few lines whereas a host may have numerous connections with possibly different packet sizes ---> having one buffer for all may not be adequate ---> more elaborate buffer management

**Another bottleneck: the carrying capacity of the subnet**

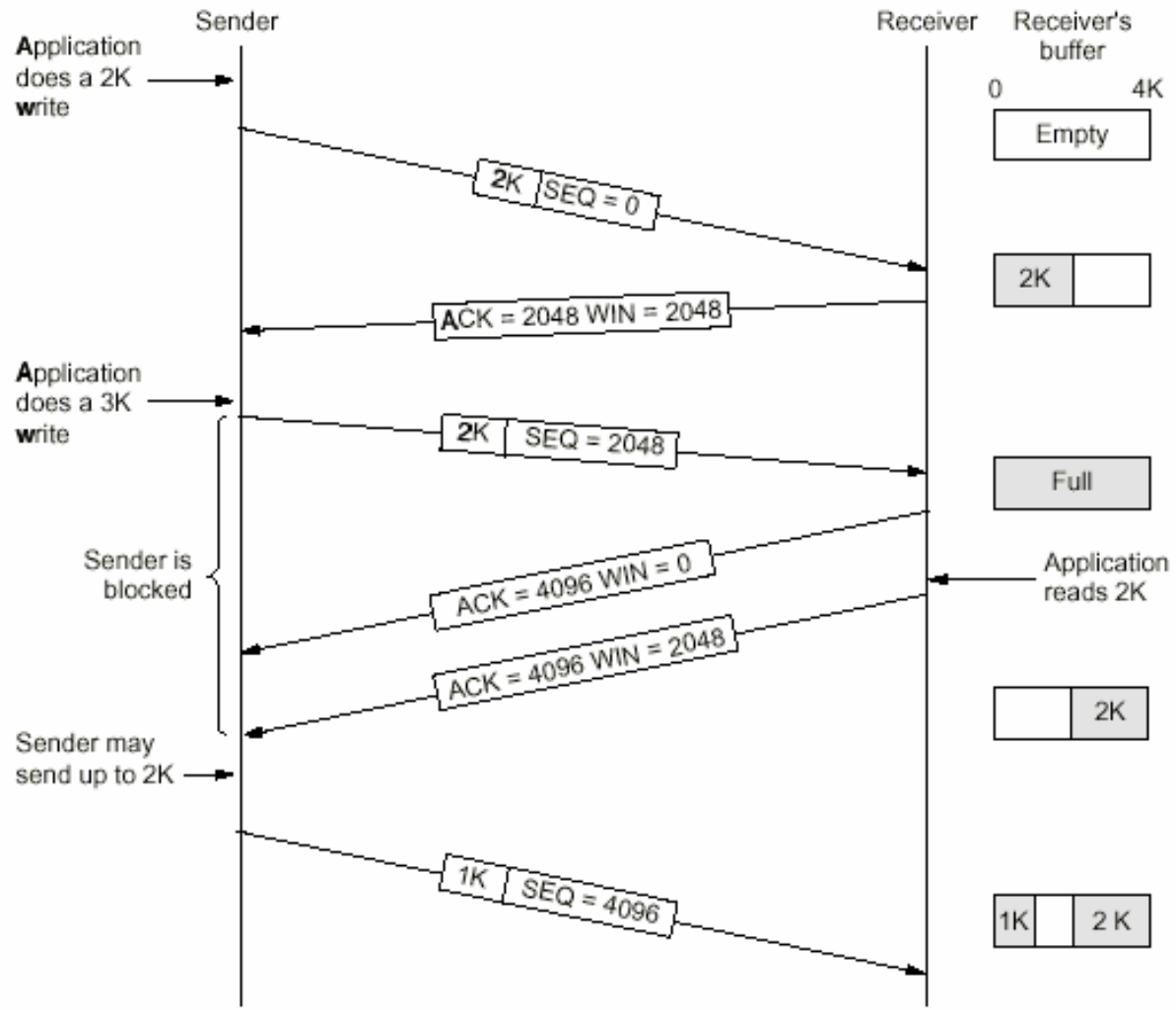
---> congestion problem and no longer a flow control problem

**Solution approach** (also adopted in TCP):

A sliding window flow control scheme in which the sender adjusts dynamically the window size to match the network's carrying capacity and not the buffer size of the receiver as it is done on the data link layer.

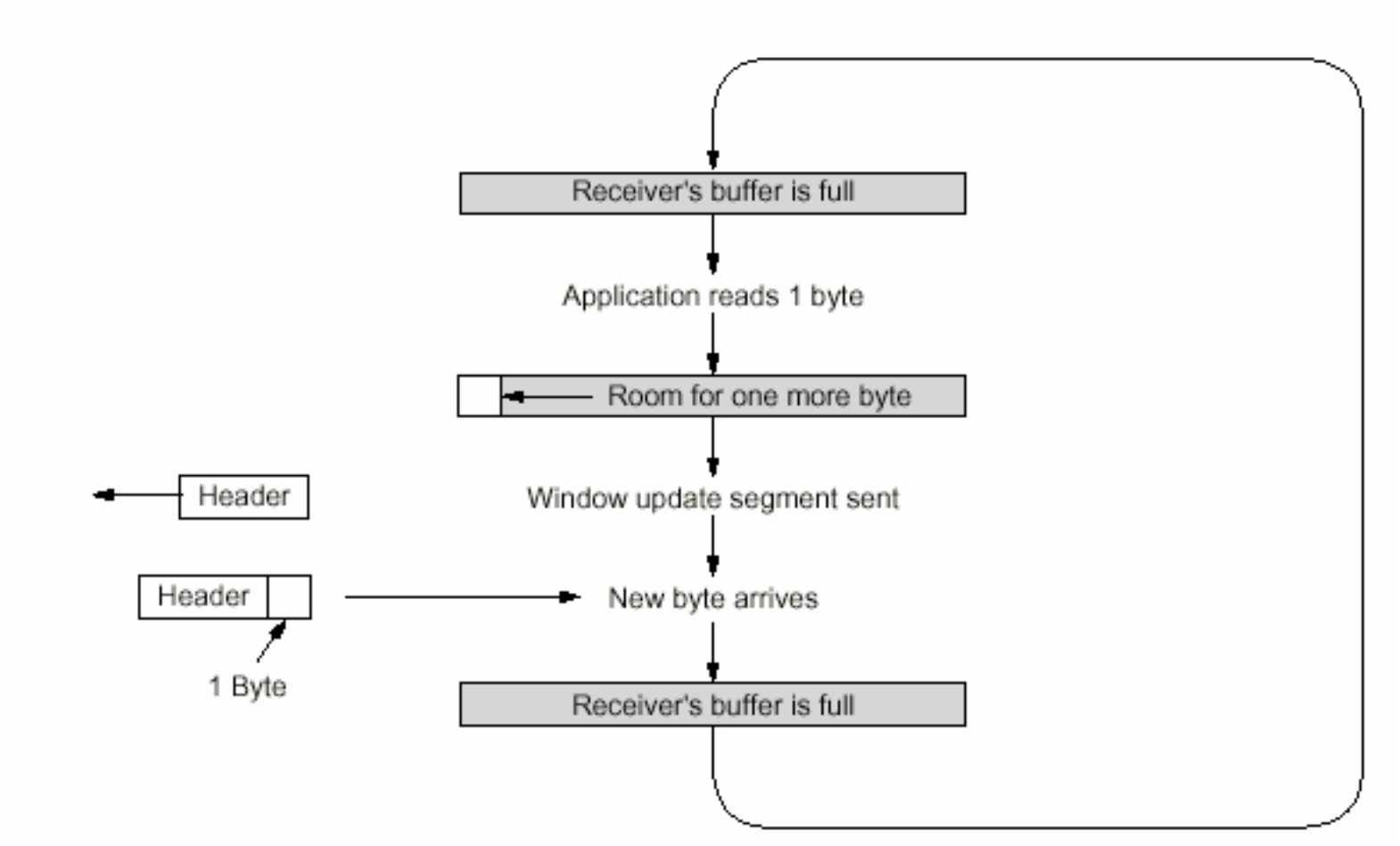
# Transport Layer(12)

## The window management (transmission policy) in TCP



# Transport Layer(13)

## The silly window syndrome



# Transport Layer(14)

## The Internet TCP (Transmission Control Protocol)

**Goal:**

Provide a reliable (connection-oriented) end-to-end byte stream over an unreliable internetwork

**Service to provide:**

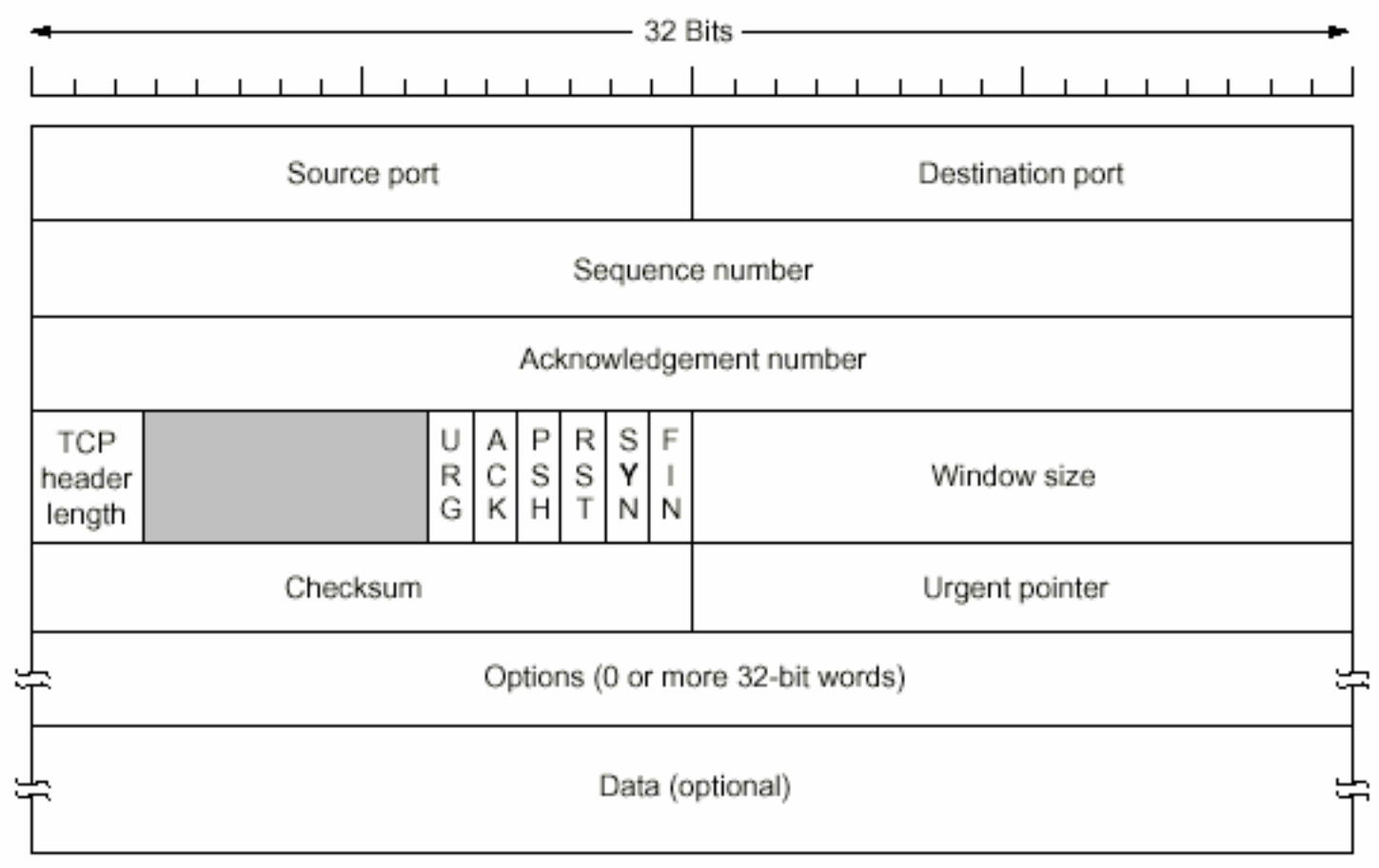
- Accepted user data streams from local processes are broken into pieces called segments not exceeding 64K bytes (in order to fit in the IP payload field)
- Reconstruction of the original byte stream by reassembling the pieces in the proper sequence
- Time out and retransmission in order to guarantee proper delivery by executing a sliding window protocol

**The TCP service model**

- both the sender and receiver create end points called sockets
- Each socket has an identifier (address):= (IP address, 16-bit number local to that host called port)
- To obtain TCP service, a connection must be established between sockets of the sender and receiver
- A socket may be used for multiple connections at the same time
- Connections are identified by the socket identifiers at both ends: (socket1, socket2)
- TCP does **not** support multicasting or broadcasting
- data is exchanged in the form of segments beginning with a 20 byte header (+ optional part)

# Transport Layer(16)

## The TCP Segment header



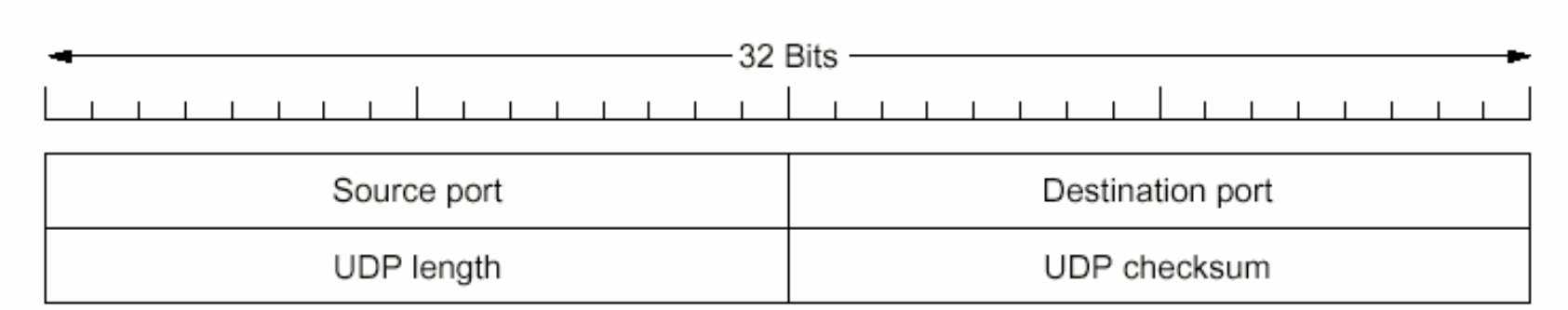
# Transport Layer(19)

**UDP (User Data Protocol)**

**Goal:**

Provide a connectionless (unreliable) way for applications to send encapsulated raw IP datagrams --->
UDP is nothing else than IP + transport layer header

**The UDP header**




**Typical use:**

Client-server applications that have one request and one response

# Transport Layer(20)

## Summary

The transport layer is the key to understanding layered protocols. Its most important service is to provide an end-to-end, reliable, connection-oriented byte stream from sender to receiver. To do so, it must

- establish connections over unreliable networks
  ---> it must cope with delayed duplicate packets
  ---> it is done by means of a three-way-handshake
- release connections which is easier but still has to face the two-army problem
- handle the service primitives that permit establishing, using, and releasing of connections
- manage connections(transmission policy) and timer

The main Internet transport protocol is TCP

- data is exchanged in the form of segments
- it uses a fixed 20-byte header + optional part + zero or more data bytes
- the basic transmission protocol used is the sliding window protocol
- a great deal of work has gone into optimizing TCP performance using various algorithms

# Application Layer (1)

**Functionality:**

- providing applications (e-mail, www, USENET etc)
- providing support protocols to allow the real applications to function properly
  - *security* comprising a large number of concepts and protocols
  - *DNS* (Distributed Name Service) handling naming within the Internet
  - *network management* (SNMP)

Network security nowadays is regarded as one of the biggest problems for upcoming Internet applications

**Objective:**

providing secure Internet applications like

- secure e-mail system
- secure e-commerce - transactions

# Network Security (1)

Secure communication in general addresses four different topics:

- *secrecy*: keeping information secret, i.e. out of the hands of unauthorized users
- *authentication*: determining whom you are talking to before revealing information
- *nonrepudiation*: ensuring trustworthiness by means of digital signatures
- *integrity*: ensuring the originality of received messages (not being modified)

Solutions to all problem areas are mainly based on **cryptographic** (Greek: secret writing) principles

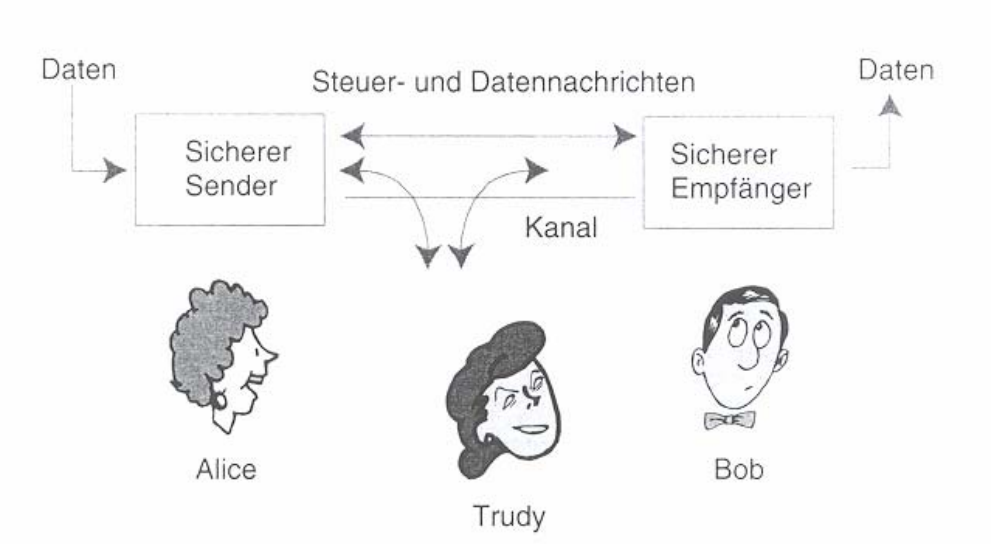
Historically, mainly 4 human areas have used and contributed to the art of cryptography:

- military
- diplomacy
- diary writing
- private affairs

Today, economy is becoming the most important area where cryptography is used.

# Network Security (2)

**Representatives for sender, recipient, and intruder**




**The encryption model:**