Computer Networks

Jürgen Schönwälder

August 20, 2018

Abstract

This memo contains annotated slides for the course “Computer Networks”. This is largely work in progress since annotating a large collection of slides is an effort that takes time.

Contents

Part I

Introduction

Section 1: Internet Concepts and Design Principles

1. Internet Concepts and Design Principles

2. Structure and Growth of the Internet

3. Internet Programming with Sockets

Section 2: Structure and Growth of the Internet

1. Internet Concepts and Design Principles

2. Structure and Growth of the Internet

3. Internet Programming with Sockets

Section 3: Internet Programming with Sockets

1. Internet Concepts and Design Principles

2. Structure and Growth of the Internet

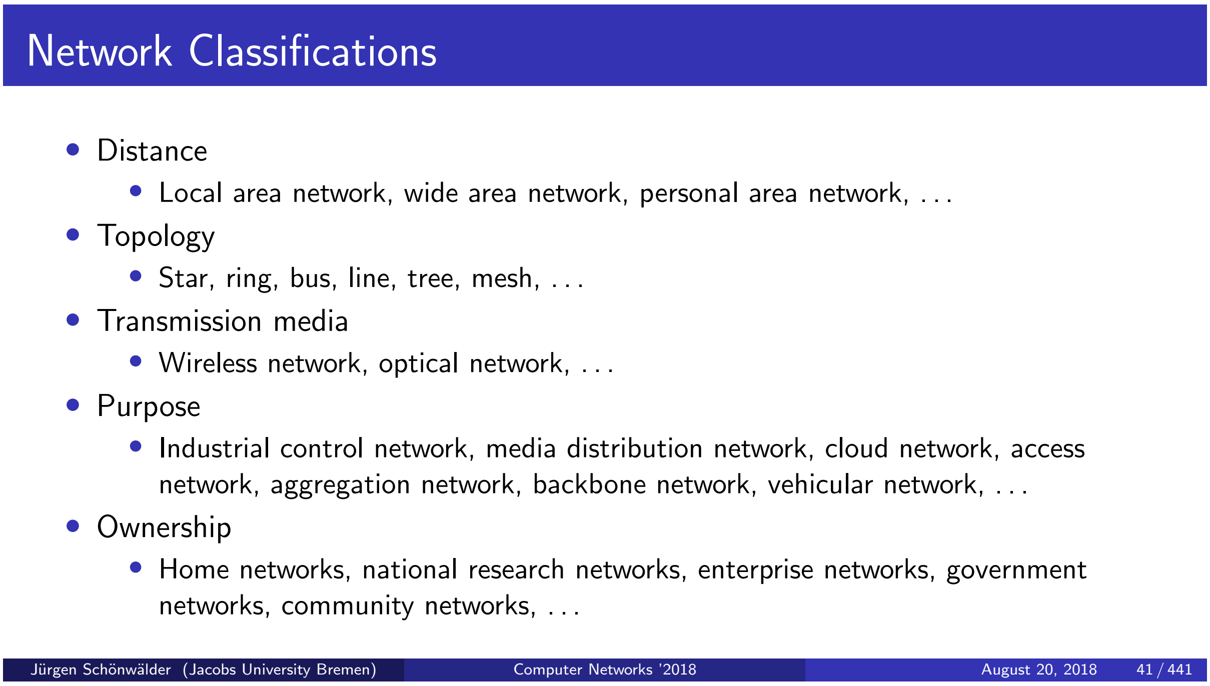
3. Internet Programming with Sockets

Part II

Fundamental Concepts

Section 4: Classification and Terminology

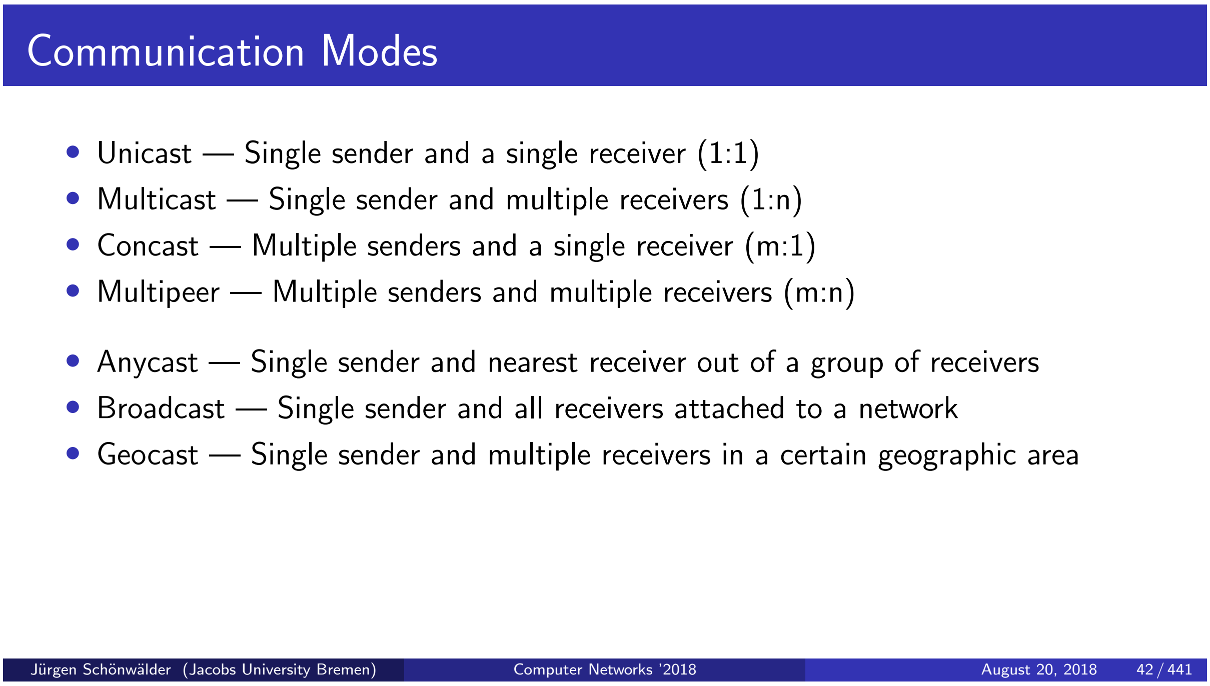
4. Classification and Terminology
5. Communication Channels and Transmission Media
6. Media Access Control
7. Transmission Error Detection
8. Sequence Numbers, Acknowledgements, Timer
9. Flow Control and Congestion Control
10. Layering and the OSI Reference Model



Network Classifications

- Distance
  - Local area network, wide area network, personal area network, . . .
- Topology
  - Star, ring, bus, line, tree, mesh, . . .
- Transmission media
  - Wireless network, optical network, . . .
- Purpose
  - Industrial control network, media distribution network, cloud network, access network, aggregation network, backbone network, vehicular network, . . .
- Ownership
  - Home networks, national research networks, enterprise networks, government networks, community networks, . . .

Computer networks can be classified according to several criteria. The list of criteria on the slides is not necessarily complete. Try to think of other possible criteria.



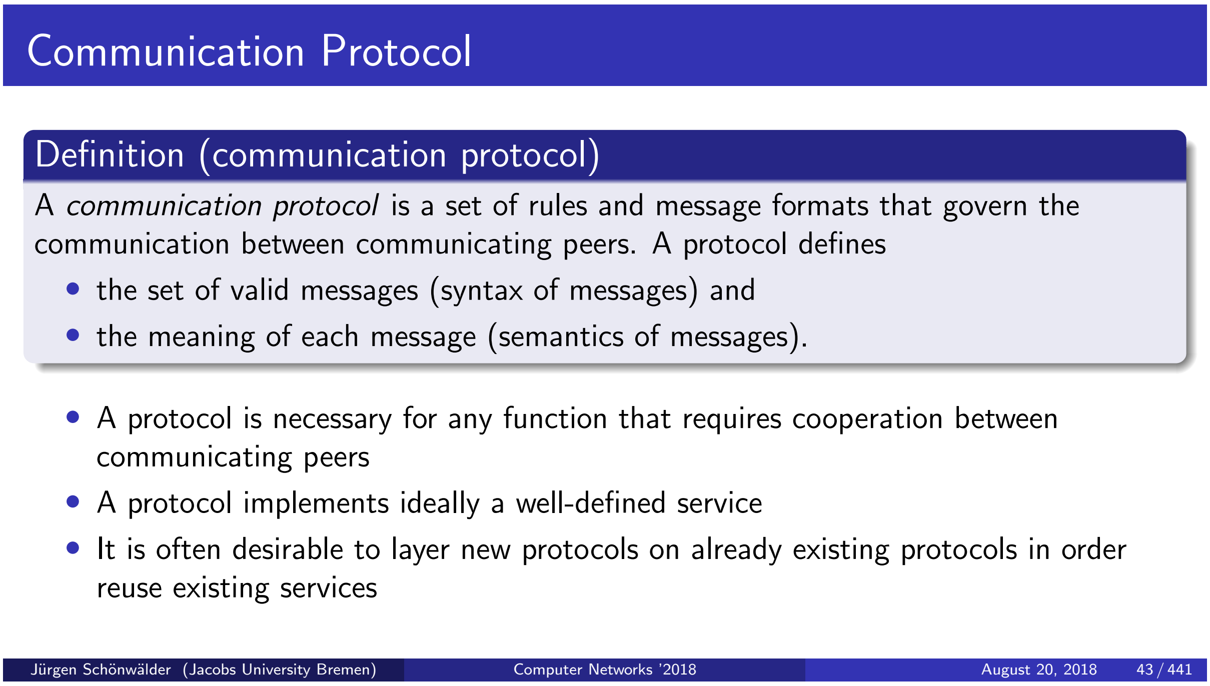
Communication Modes

- Unicast — Single sender and a single receiver (1:1)
- Multicast — Single sender and multiple receivers (1:n)
- Concast — Multiple senders and a single receiver (m:1)
- Multipeer — Multiple senders and multiple receivers (m:n)

- Anycast — Single sender and nearest receiver out of a group of receivers
- Broadcast — Single sender and all receivers attached to a network
- Geocast — Single sender and multiple receivers in a certain geographic area

We often focus on unicast communication. However, broadcast and multicast communication is a very important communication model in local area networks. Anycasting is widely deployed in today's Internet in order to prove fast access to popular content.

Concast communication is relatively rare and if it happens at large scale often associated with denial of service attacks where many senders try to overload a single receiver (the target of the attack).



Communication Protocol

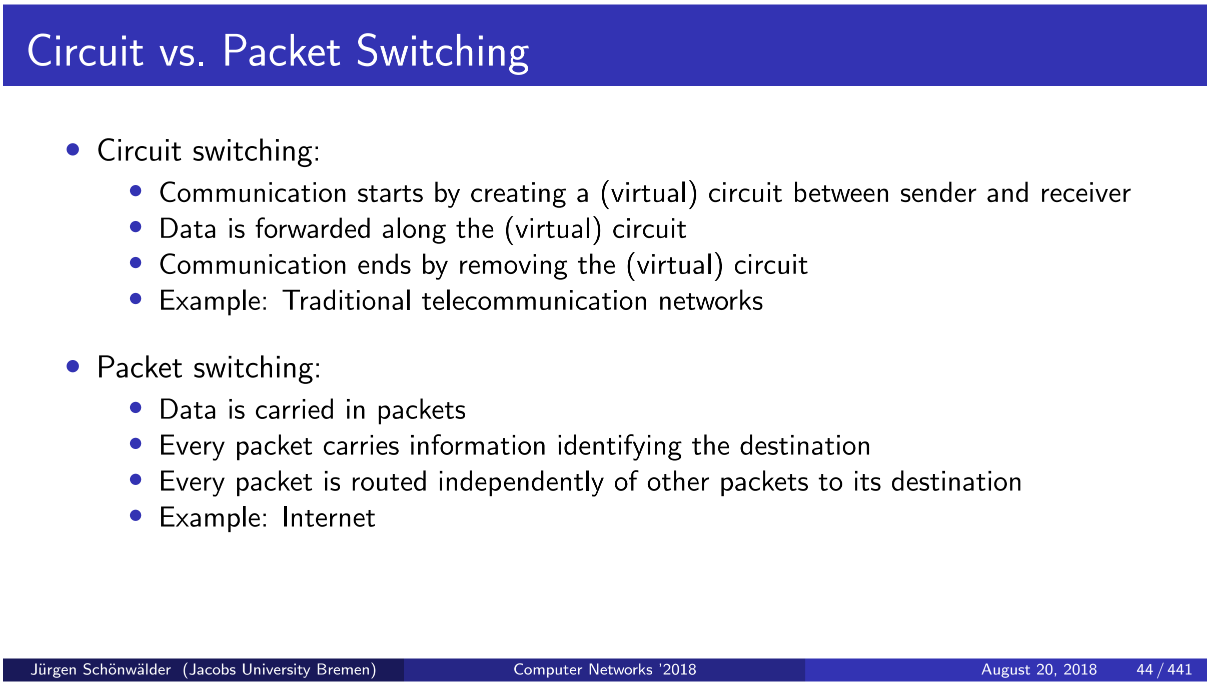
Definition (communication protocol)

A *communication protocol* is a set of rules and message formats that govern the communication between communicating peers. A protocol defines

- the set of valid messages (syntax of messages) and
- the meaning of each message (semantics of messages).

- A protocol is necessary for any function that requires cooperation between communicating peers
- A protocol implements ideally a well-defined service
- It is often desirable to layer new protocols on already existing protocols in order reuse existing services

The notion of a communication protocol is central for this course. We use an informal definition here but there are of course ways to formalize the definition of message syntax as well as message semantics.



Circuit vs. Packet Switching

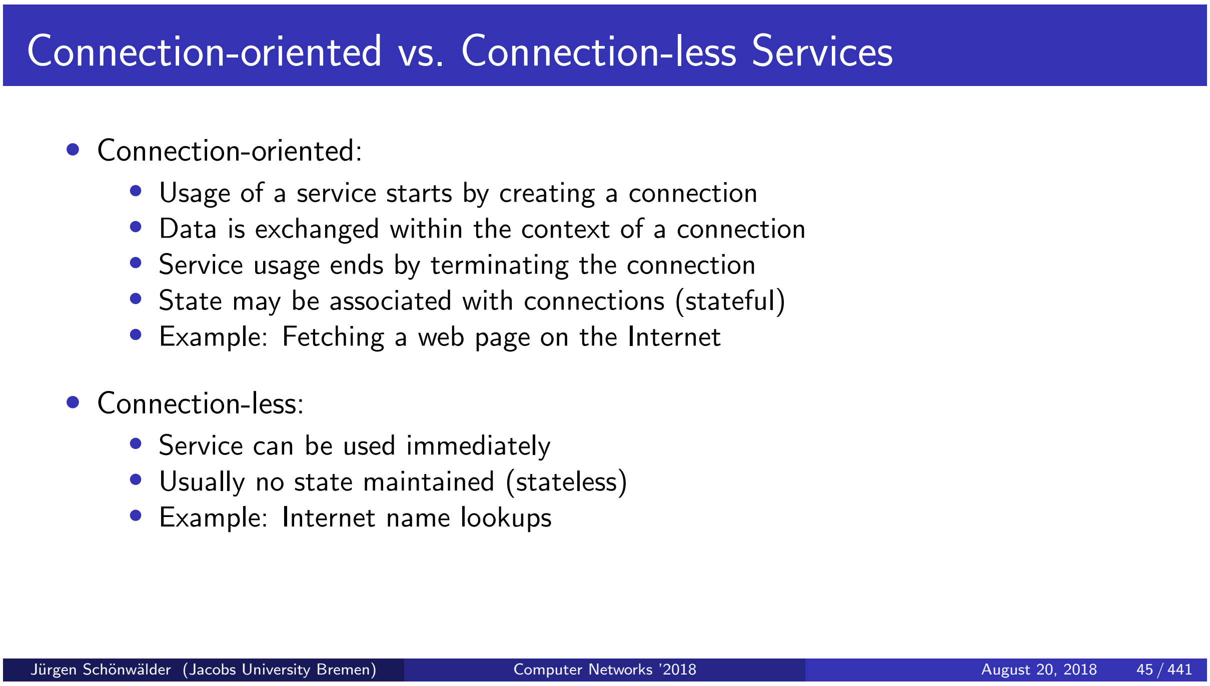
- Circuit switching:
  - Communication starts by creating a (virtual) circuit between sender and receiver
  - Data is forwarded along the (virtual) circuit
  - Communication ends by removing the (virtual) circuit
  - Example: Traditional telecommunication networks
- Packet switching:
  - Data is carried in packets
  - Every packet carries information identifying the destination
  - Every packet is routed independently of other packets to its destination
  - Example: Internet

The telephone network was designed as a circuit switched network. In the early days, human operators were plugging cables to establish a call. This was later automated but even with digital telecommunication networks, the notion of establishing a (virtual) circuit for every phone call remained. As a consequence, it was natural to charge for the length of the call (the time resources were allocated for the virtual circuit).

Packet switching ideas were developed in the 1960s, largely driven by the idea to build networks that could survive nuclear attacks. First prototype networks appeared in the late 1960s and early 1970s, leading to the development of the first Internet protocols during the 1970s.

Packet switching proved to be a far better match for data networking (compared to lets say voice call handling) since data networking usually involves the communication with many different endpoints.

During the 1990s, there was a big discussion whether packet switched networks, that rely on statistical multiplexing, could replace circuit switched networks, that allocate and reserve resources for every switched circuit. The argument was that resource reservation in circuit switched networks provides a means to guarantee a certain service quality. The counter argument was that similar quality can be achieved with statistical multiplexing by simply dimensioning the network capacity in such a way that it never operates at utilization levels where noticeable service quality degrations can appear.

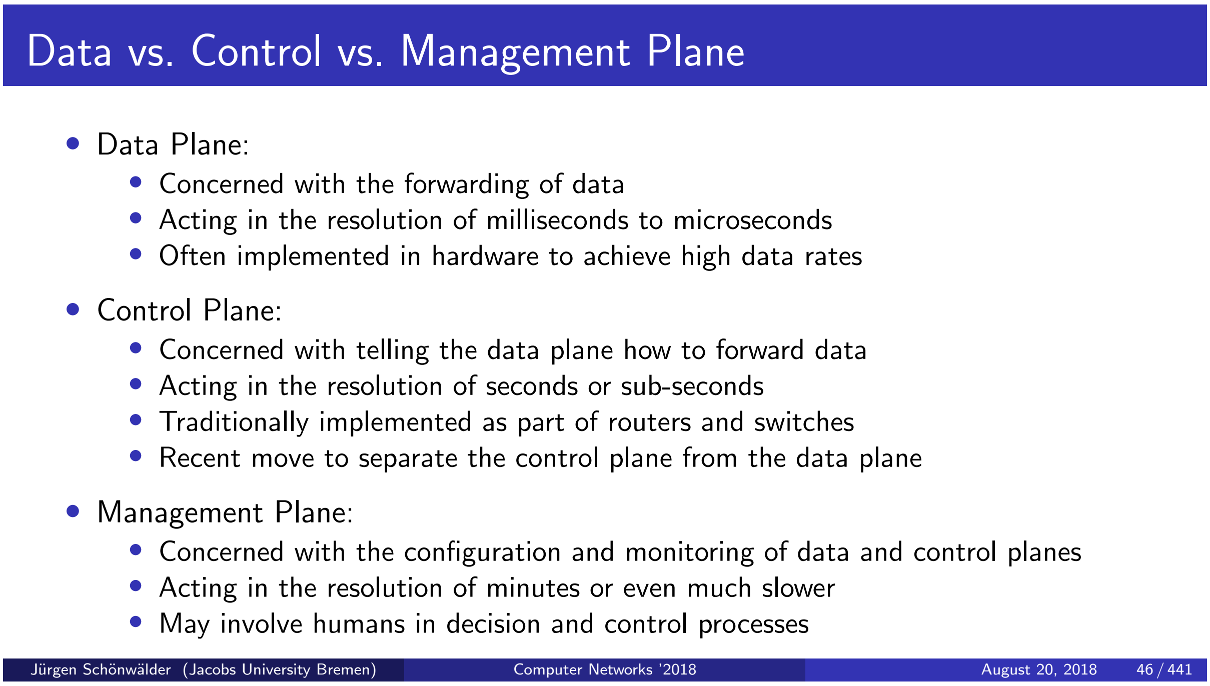


Connection-oriented vs. Connection-less Services

- Connection-oriented:
  - Usage of a service starts by creating a connection
  - Data is exchanged within the context of a connection
  - Service usage ends by terminating the connection
  - State may be associated with connections (stateful)
  - Example: Fetching a web page on the Internet
- Connection-less:
  - Service can be used immediately
  - Usually no state maintained (stateless)
  - Example: Internet name lookups

Jürgen Schönwälder (Jacobs University Bremen) Computer Networks '2018 August 20, 2018 45 / 441

It is important to not confuse circuit vs. packet switching with connection-oriented vs. connection-less service. It is very well possible to realize connection-oriented services on a packet switched network. Most of the Internet traffic today is using connection-oriented services running over a packet switched network.



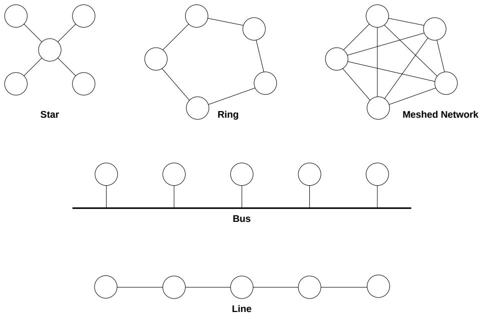
Data vs. Control vs. Management Plane

- Data Plane:
  - Concerned with the forwarding of data
  - Acting in the resolution of milliseconds to microseconds
  - Often implemented in hardware to achieve high data rates
- Control Plane:
  - Concerned with telling the data plane how to forward data
  - Acting in the resolution of seconds or sub-seconds
  - Traditionally implemented as part of routers and switches
  - Recent move to separate the control plane from the data plane
- Management Plane:
  - Concerned with the configuration and monitoring of data and control planes
  - Acting in the resolution of minutes or even much slower
  - May involve humans in decision and control processes

Networking products developed in the 1990s and the early years of this century were usually bundling the data plane and the control plane together. A networking device such as a router did include both a hardware assisted data plane and a control plane implemented in software. The management plane was usually detached and implemented according to the needs of the network operator.

Work to separate the data plane and the control plane in data networks started in the early years of this century and the Ethane project at Stanford university finally led to the OpenFlow protocol, which lead to the architectural concept of Software Defined Networks, which largely builds on the idea to separate the control plane from the forwarding plane and thereby making the control plane freely programmable.

Topologies (1/3)



- The *topology* of a network describes the way in which nodes attached to the network are interconnected

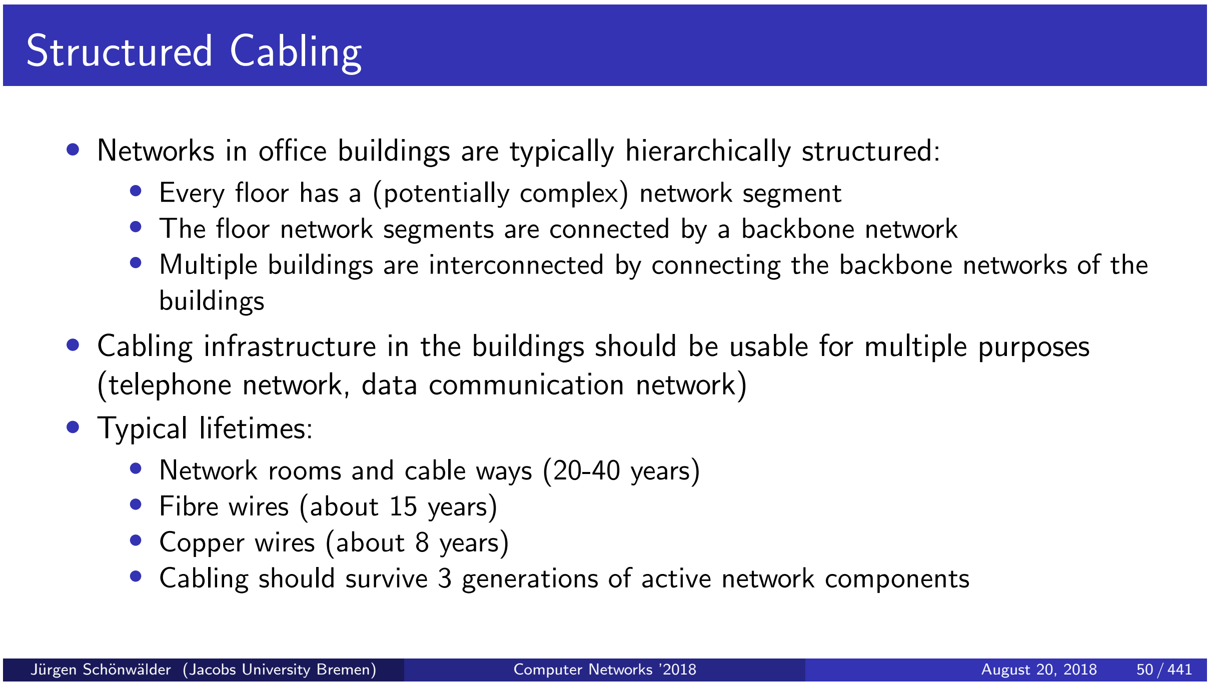
Topologies (2/3)

- Star:
  - Nodes are directly connected to a central node
  - Simple routing
  - Good availability if central node is highly reliable
- Ring:
  - Nodes are connected to form a ring
  - Simple routing
  - Limited reliability, failures require reconfiguration
- Meshed Network:
  - Nodes are directly connected to all other nodes
  - Very good reliability
  - Requires $\frac{n(n-1)}{2}$ links for n nodes

Topologies (3/3)

- Bus:
  - Nodes are attached to a shared medium
  - Simple routing
  - Good availability if the bus is highly reliable
- Line:
  - Nodes are connected to form a line
  - Average reliability
  - Network position influences transmission delays

- Line topologies are especially important in the case $n = 2$ (point to point links)



Structured Cabling

- Networks in office buildings are typically hierarchically structured:
    - Every floor has a (potentially complex) network segment
    - The floor network segments are connected by a backbone network
    - Multiple buildings are interconnected by connecting the backbone networks of the buildings
- Cabling infrastructure in the buildings should be usable for multiple purposes (telephone network, data communication network)
- Typical lifetimes:
    - Network rooms and cable ways (20-40 years)
    - Fibre wires (about 15 years)
    - Copper wires (about 8 years)
    - Cabling should survive 3 generations of active network components

The international standard ISO/IEC 11801 specifies structured cabling system suitable for a wide range of applications (telephony, data communication, building control systems, factory automation). It covers both balanced copper cabling and optical fibre cabling. The 3rd edition of ISO/IEC 11801 was released in November 2017.

Section 5: Communication Channels and Transmission Media

4 Classification and Terminology

5 Communication Channels and Transmission Media

6 Media Access Control

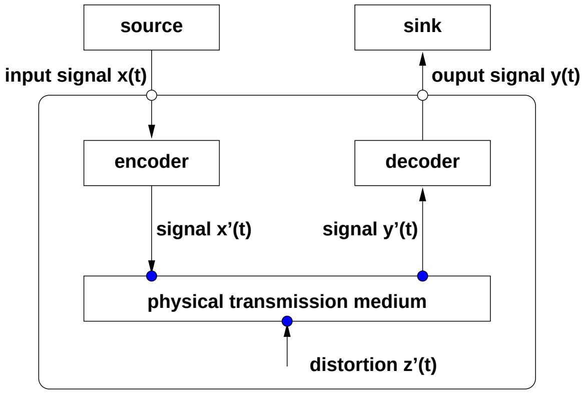
7 Transmission Error Detection

8 Sequence Numbers, Acknowledgements, Timer

9 Flow Control and Congestion Control

10 Layering and the OSI Reference Model

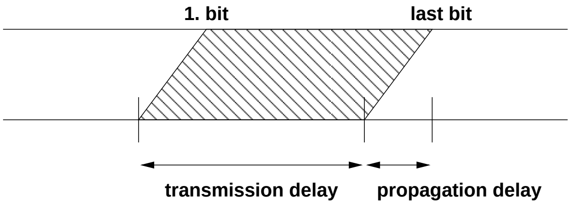
Communication Channel Model



- Signals are in general modified during transmission, leading to transmission errors.

Channel Characteristics

- The *data rate* (bit rate) describes the data volume that can be transmitted per time interval (e.g., 100 Mbit/s)
- The *bit time* is the time needed to transmit a single bit (e.g., 1 microsecond for 1 Mbit/s)



- The *delay* is the time needed to transmit a message from the source to the sink. It consists of the *propagation delay* and the *transmission delay*
- The *bit error rate* is the probability of a bit being changed during transmission

To achieve high data rates, multiple bits can be transmitted concurrently (i.e., via coding system that can encode multiple bits into a code word).

We generally assume serial data transmission where a data word (e.g., an octet) is transmitted as an ordered sequence of bits. Serial data transmission has the benefit that it requires only a single channel (i.e., a single wire). The alternative, parallel data transmission, transmits all bits of a data word (e.g., an octet) concurrently. This, however, requires multiple channels (e.g., multiple wires).

Note that some coding schemes may use multiple multiple wires for efficiency reasons but from the outside we still consider such transmission systems to be serial.

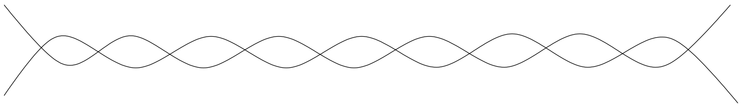
Transmission Media Overview

- Copper wires:
  - Simple wires
  - Twisted pair
  - Coaxial cables
- Optical wires:
  - Fibre (multimode and single-mode)
- Air:
  - Radio waves
  - Micro waves
  - Infrared waves
  - Light waves

Simple Electrical Wires

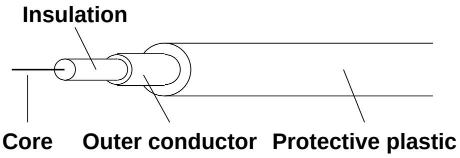
- Simple two-wire open lines are the simplest transmission medium
- Adequate for connecting equipment up to 50 m apart using moderate bit rates
- The signal is typically a voltage or current level relative to some ground level
- Simple wires can easily experience crosstalk caused by capacitive coupling
- The open structure makes wires suspectible to pick-up noise signals from other electrical signal sources

Twisted Pairs



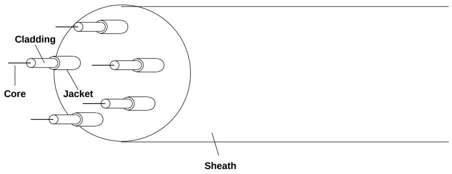
- A twisted pair consists of two insulated copper wires
- Twisting the wires in a helical form cancels out waves
- Unshielded twisted pair (UTP) of category 3 was the standard cabling up to 1988
- UTP category 5 and above are now widely used for wiring (less crosstalk, better signals over longer distances)
- Shielded twisted pair (STP) cables have an additional shield further reducing noise

Coaxial Cable



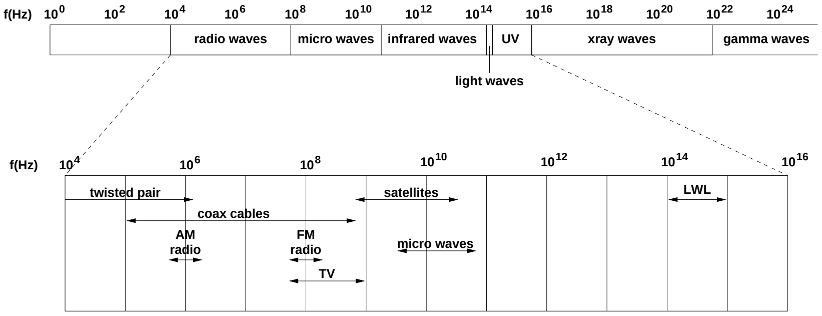
- Coax cables are shielded (less noise) and suffer less from attenuation
- Data rates of 500 mbps over several kilometers with a error probability of 10^{-7} achievable
- Widely used for cable television broadcast networks (which in some countries are heavily used for data communication today)

Fibre



- The glass core is surrounded by a glass cladding with a lower index of refraction to keep the light in the core
- Multimode fiber have a thick core (20-50 micrometer) and propagate light using continued refraction
- Single-mode fiber have a thin core (2-10 micrometer) which guides the light through the fiber
- High data rates, low error probability, thin, lightweight, immune to electromagnetic interference

Electromagnetic Spectrum



- Usage of most frequencies is controlled by legislation
- The Industrial/Scientific/Medical (ISM) band (2400-2484 MHz) can be used without special licenses

Transmission Impairments (1/2)

- *Attenuation*:
  - The strength of a signal falls off with distance over any transmission medium
  - For guided media, attenuation is generally an exponential function of the distance
  - For unguided media, attenuation is a more complex function of distance and the makeup of the atmosphere
- *Delay distortion*:
  - Delay distortion occurs because the velocity of propagation of a signal through a guided medium varies with frequency
  - Various frequency components of a signal will arrive at the receiver at different times

Transmission Impairments (2/2)

- Noise
  - Thermal noise (white noise) is due to thermal agitation of electrons and is a function of temperature
  - Intermodulation noise can occur if signals at different frequencies share the same transmission medium
  - Crosstalk is an unwanted coupling between signal paths
  - Impulse noise consists of irregular pulses or noise spikes of short duration and of relatively high amplitude

Section 6: Media Access Control

4 Classification and Terminology

5 Communication Channels and Transmission Media

6 Media Access Control

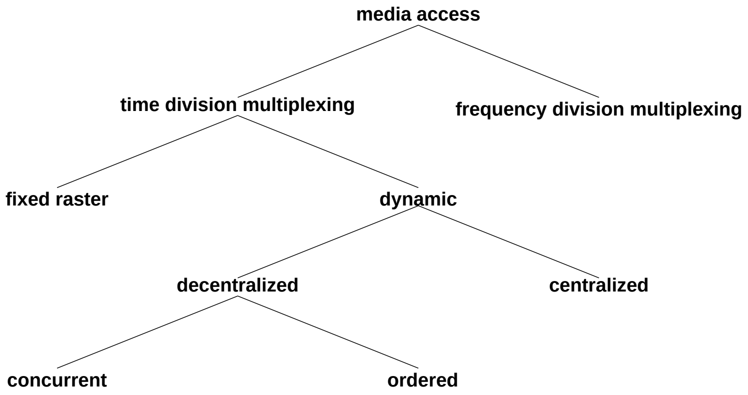
7 Transmission Error Detection

8 Sequence Numbers, Acknowledgements, Timer

9 Flow Control and Congestion Control

10 Layering and the OSI Reference Model

Media Access Control Overview



media access

time division multiplexing

frequency division multiplexing

fixed raster

dynamic

decentralized

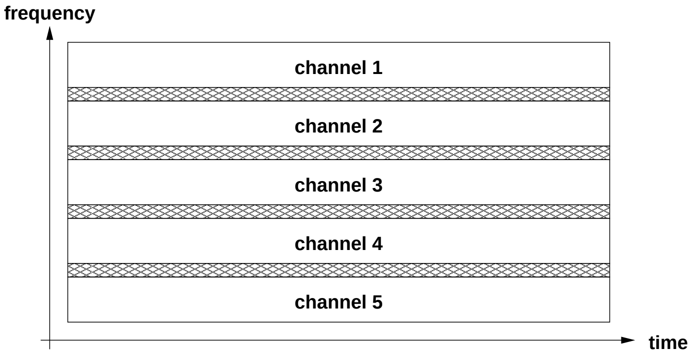
centralized

concurrent

ordered

- Shared transmission media require coordinated access to the medium (media access control)

Frequency Division Multiplexing (FDM)

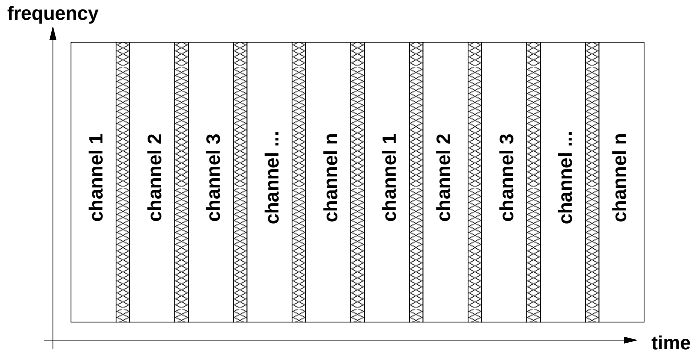


- Signals are carried simultaneously on the same medium by allocating to each signal a different frequency band

Wavelength Division Multiplexing (WDM)

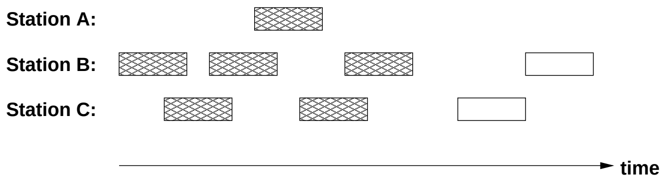
- Optical fibers carry multiple wavelength at the same time
- WDM can achieve very high data rates over a single optical fiber
- Dense WDM (DWDM) is a variation where the wavelengths are spaced close together, which results in an even larger number of channels.
- Theoretically, there is room for 1250 channels, each running at 10 Gbps, on a single fiber (= 12.5 Tbps).
- A single cable often bundles a number of fibers and for deployment or reasons, fibres are sometimes even bundled with power cables.

Time Division Multiplexing (TDM)



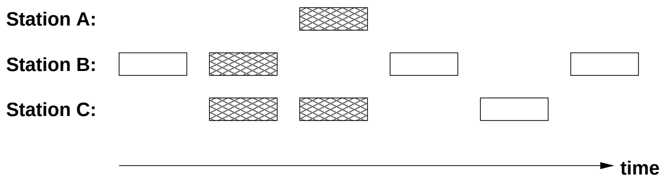
- Signals from a given sources are assigned to specific time slots
- Time slot assignment might be fixed (synchronous TDM) or dynamic (statistical TDM)

Pure Aloha



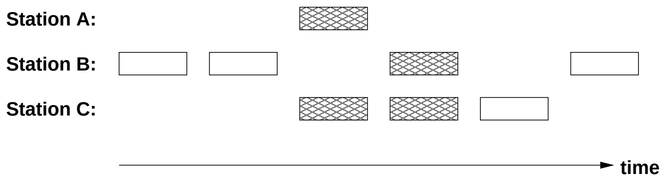
- Developed in the 1970s at the University of Hawaii
- Sender sends data as soon as data becomes available
- Collisions are detected by listening to the signal
- Retransmit after a random pause after a collision
- Not very efficient ($\approx$ 18 % of the channel capacity)

Slotted Aloha



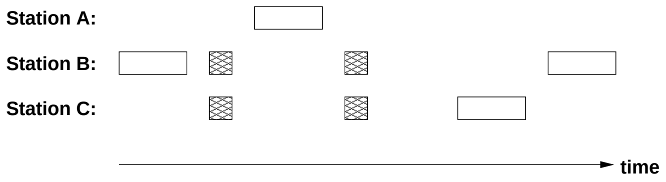
- Senders do not send immediately but wait for the beginning of a time slot
- Time slots may be advertised by short control signals
- Collisions only happen at the start of a transmission
- Avoids sequences of partially overlaying data blocks
- Slightly more efficient ($\approx$ 37 % of the channel capacity)

Carrier Sense Multiple Access (CSMA)



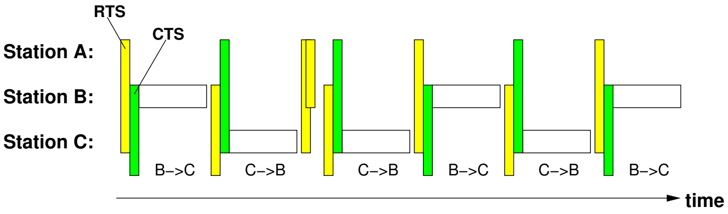
- Sense the media whether it is unused before starting a transmission
- Collisions are still possible (but less likely)
- 1-persistent CSMA: sender sends with probability 1
- p-persistent CSMA: sender sends with probability p
- non-persistent CSMA: sender waits for a random time period before it retries if the media is busy

CSMA with Collision Detection (CSMA-CD)



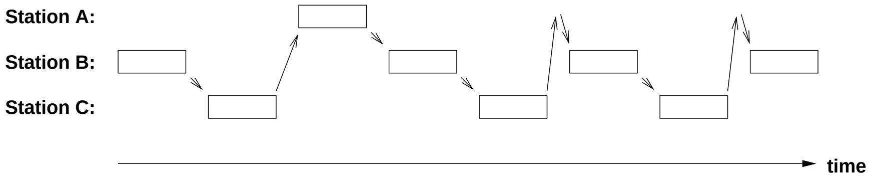
- Terminate the transmission as soon as a collision has been detected (and retry after some random delay)
- Let τ be the propagation delay between two stations with maximum distance
- Senders can be sure that they successfully acquired the medium after 2τ time units
- Used by the classic Ethernet developed at Xerox Parc

Multiple Access with Collision Avoidance (MACA)



- A station which is ready to send first sends a short RTS (ready to send) message to the receiver
- The receiver responds with a short CTS (clear to send) message
- Stations who receive RTS or CTS must stay quiet
- Solves the *hidden station* and *exposed station* problem

Token Passing



- A token is a special bit pattern circulating between stations - only the station holding the token is allowed to send data
- Token mechanisms naturally match physical ring topologies - logical rings may be created on other physical topologies
- Care must be taken to handle lost or duplicate token

Section 7: Transmission Error Detection

4 Classification and Terminology

5 Communication Channels and Transmission Media

6 Media Access Control

7 Transmission Error Detection

8 Sequence Numbers, Acknowledgements, Timer

9 Flow Control and Congestion Control

10 Layering and the OSI Reference Model

Transmission Error Detection

- Data transmission often leads to transmission errors that affect one or more bits
- Simple parity bits can be added to code words to detect bit errors
- Parity bit schemes are not very strong in detecting errors which affect multiple bits
- Computation of error check codes must be efficient (in hardware and/or software)

Cyclic Redundancy Check (CRC)

- A bit sequence (bit block) $b_n b_{n-1} \ldots b_1 b_0$ is represented as a polynomial $B(x) = b_n x^n + b_{n-1} x^{n-1} + \ldots + b_1 x + b_0$
- Arithmetic operations:

$$0 + 0 = 1 + 1 = 0 \quad 1 + 0 = 0 + 1 = 1$$

$$1 \cdot 1 = 1 \quad 0 \cdot 0 = 0 \cdot 1 = 1 \cdot 0 = 0$$

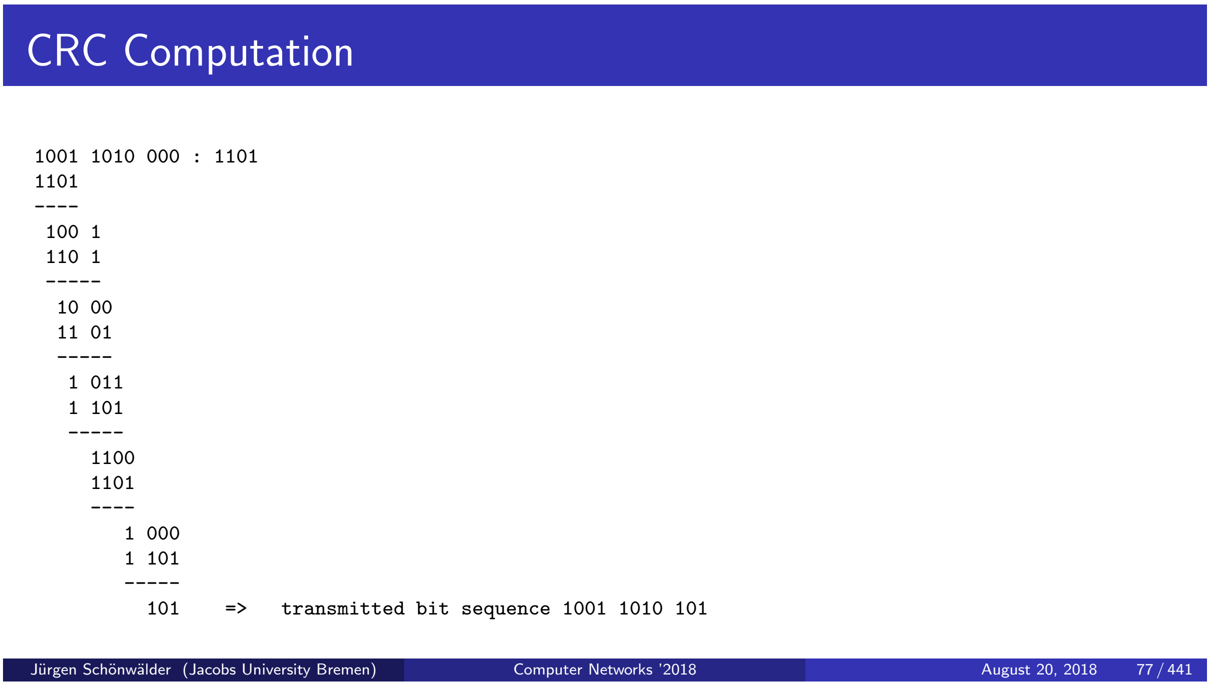
- A generator polynomial $G(x) = g_r x^r + \ldots g_1 x + g_0$ with $g_r = 1$ and $g_0 = 1$ is agreed upon between the sender and the receiver
- The sender transmits $U(x) = x^r \cdot B(x) + t(x)$ with

$$t(x) = (x^r \cdot B(x)) \bmod G(x)$$

Cyclic Redundancy Check (CRC)

- The receiver tests whether the polynomial corresponding to the received bit sequence can be divided by $G(x)$ without a remainder
- Efficient hardware implementation possible using XOR gates and shift registers
- Only errors divisible by $G(x)$ will go undetected

- Example:
  - Generator polynomial $G(x) = x^3 + x^2 + 1$ (corresponds to the bit sequence 1101)
  - Message $M = 1001\ 1010$ (corresponds to the polynomial $B(x) = x^7 + x^4 + x^3 + x$)



CRC Computation

```
1001 1010 000 : 1101
1101
----
 100 1
 110 1
 -----
  10 00
  11 01
  -----
   1 011
   1 101
   -----
     1100
     1101
     ----
        1 000
        1 101
        -----
          101    =>   transmitted bit sequence 1001 1010 101
```

CRC Verification

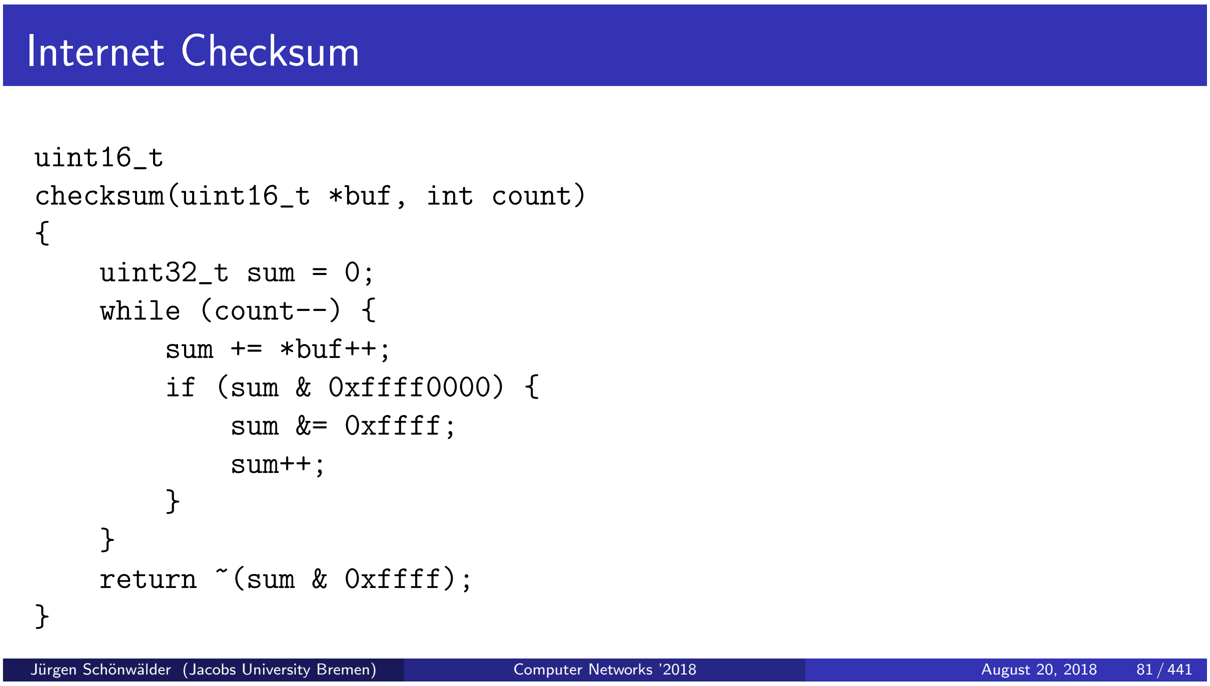
```
1001 1010 101 : 1101
1101
----
 100 1
 110 1
 -----
  10 00
  11 01
  -----
   1 011
   1 101
   -----
     1100
     1101
     ----
        1 101
        1 101
        -----
            0    =>   remainder 0, assume no transmission error
```

Choosing Generator Polynomials

- $G(x)$ detects all single-bit errors if $G(x)$ has more than one non-zero term
- $G(x)$ detects all double-bit errors, as long as $G(x)$ has a factor with three terms
- $G(x)$ detects any odd number of errors, as long as $G(x)$ contains the factor $(x + 1)$
- $G(x)$ detects any burst errors for which the length of the burst is less than or equal to r
- $G(x)$ detects a fraction of error bursts of length $r + 1$; the fraction equals to $1 - 2^{-(r-1)}$
- $G(x)$ detects a fraction of error bursts of length greater than $r + 1$; the fraction equals to $1 - 2^{-r}$

Well-known Generator Polynomials

- The HEC polynomial $G(x) = x^8 + x^2 + x + 1$ is used by the ATM cell header
- The CRC-16 polynomial $G(x) = x^{16} + x^{15} + x^2 + 1$ detects all single and double bit errors, all errors with an odd number of bits, all burst errors with 16 or less bits and more than 99% of all burst errors with 17 or more bits
- The CRC-CCITT polynomial $G(x) = x^{16} + x^{15} + x^5 + 1$ is used by the HDLC protocol
- The CRC-32 polynomial $G(x) = x^{32} + x^{26} + x^{23} + x^{22} + x^{16} + x^{12} + x^{11} + x^{10} + x^8 + x^7 + x^5 + x^4 + x^2 + x + 1$ is used by the IEEE 802 standards



Internet Checksum

```
uint16_t
checksum(uint16_t *buf, int count)
{
    uint32_t sum = 0;
    while (count--) {
        sum += *buf++;
        if (sum & 0xffff0000) {
            sum &= 0xffff;
            sum++;
        }
    }
    return ~(sum & 0xffff);
}
```

Internet Checksum Computation

```
data[] = dead cafe face (hexadecimal)

    0000                               verification:     0000
+   dead (data[0])                                   +'  dead (data[0])
   -----                                                -----
    dead                                                 dead
+   cafe (data[1])                                   +'  cafe (data[1])
   -----                                                -----
   1a9ab                                                 a9ac
+  '-->1                                             +'  face (data[2])
   -----                                                -----
    a9ac                                                 a47b
+   face (data[2])                                   +'  5b84 (checksum)
   -----                                                -----
   1a47a                                                 ffff (test passed)
+  '-->1
   -----     complement
    a47b   -------------> 5b84 (checksum)
```

Internet Checksum Properties

- Summation is commutative and associative
- Computation independent of the byte order
- Computation can be parallelized on processors with word sizes larger than 16 bit
- Individual data fields can be modified without having to recompute the whole checksum
- Can be integrated into copy loop
- Often implemented in assembler or special hardware
- For details, see RFC 1071, RFC 1141, and RFC 1624

Section 8: Sequence Numbers, Acknowledgements, Timer

4 Classification and Terminology

5 Communication Channels and Transmission Media

6 Media Access Control

7 Transmission Error Detection

8 Sequence Numbers, Acknowledgements, Timer

9 Flow Control and Congestion Control

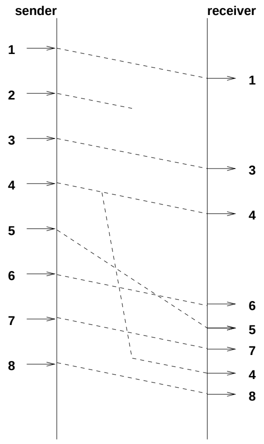
10 Layering and the OSI Reference Model

Errors Affecting Complete Data Frames

- Despite bit errors, the following transmission errors can occur
  - Loss of complete data frames
  - Duplication of complete data frames
  - Receipt of data frames that were never sent
  - Reordering of data frames during transmission
- In addition, the sender must adapt its speed to the speed of the receiver (*end-to-end flow control*)
- Finally, the sender must react to congestion situations in the network (*congestion control*)

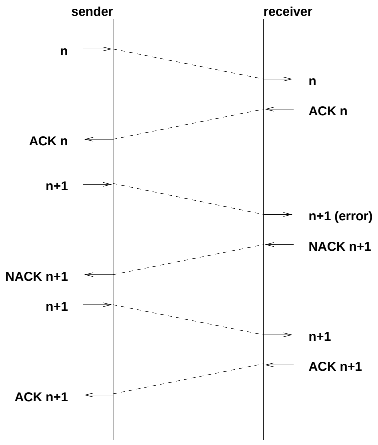
Sequence Numbers

- The sender assigns growing sequence numbers to all data frames
- A receiver can detect reordered or duplicated frames
- Loss of a frame can be determined if a missing frame cannot travel in the network anymore
- Sequence numbers can grow quickly on fast networks



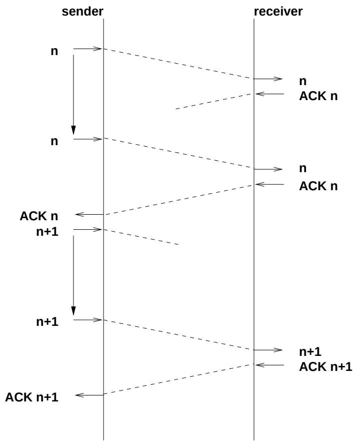
Acknowledgements

- Retransmit to handle errors
- A positive acknowledgement (ACK) is sent to inform the sender that the transmission of a frame was successful
- A negative acknowledgement (NACK) is sent to inform the sender that the transmission of a frame was unsuccessful
- Stop-and-wait protocol: a frame is only transmitted if the previous frame was been acknowledged



Timers

- Timer can be used to detect the loss of frames or acknowledgments
- A sender can use a timer to retransmit a frame if no acknowledgment has been received in time
- A receiver can use a timer to retransmit acknowledgments
- Problem: Timers must adapt to the current delay in the network



Section 9: Flow Control and Congestion Control

4 Classification and Terminology

5 Communication Channels and Transmission Media

6 Media Access Control

7 Transmission Error Detection

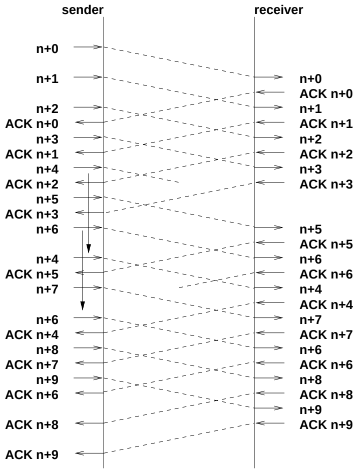
8 Sequence Numbers, Acknowledgements, Timer

9 Flow Control and Congestion Control

10 Layering and the OSI Reference Model

Flow Control

- Allow the sender to send multiple frames before waiting for acknowledgments
- Improves efficiency and overall delay
- Sender must not overflow the receiver
- The stream of frames should be smooth and not bursty
- Speed of the receiver can vary over time



Sliding Window Flow Control

- Sender and receiver agree on a window of the sequence number space
- The sender may only transmit frames whose sequence number falls into the sender's window
- Upon receipt of an acknowledgement, the sender's window is moved
- The receiver only accepts frames whose sequence numbers fall into the receiver's window
- Frames with increasing sequence number are delivered and the receiver window is moved.
- The size of the window controls the speed of the sender and must match the buffer capacity of the receiver

Sliding Window Implementation

- Implementation on the sender side:
  - `SWS` (send window size)
  - `LAR` (last ack received)
  - `LFS` (last frame send)
  - Invariant: $\texttt{LFS} - \texttt{LAR} + 1 \leq \texttt{SWS}$
- Implementation on the receiver side:
  - `RWS` (receiver window size)
  - `LFA` (last frame acceptable)
  - `NFE` (next frame expected)
  - Invariant: $\texttt{LFA} - \texttt{NFE} + 1 \leq \texttt{RWS}$

Congestion Control

- Flow control is used to adapt the speed of the sender to the speed of the receiver
- Congestion control is used to adapt the speed of the sender to the speed of the network
- Principles:
  - Sender and receiver reserve bandwidth and puffer capacity in the network
  - Intermediate systems drop frames under congestion and signal the event to the senders involved
  - Intermediate systems send control messages (choke packets) when congestion builds up to slow down senders

Section 10: Layering and the OSI Reference Model

4 Classification and Terminology

5 Communication Channels and Transmission Media

6 Media Access Control

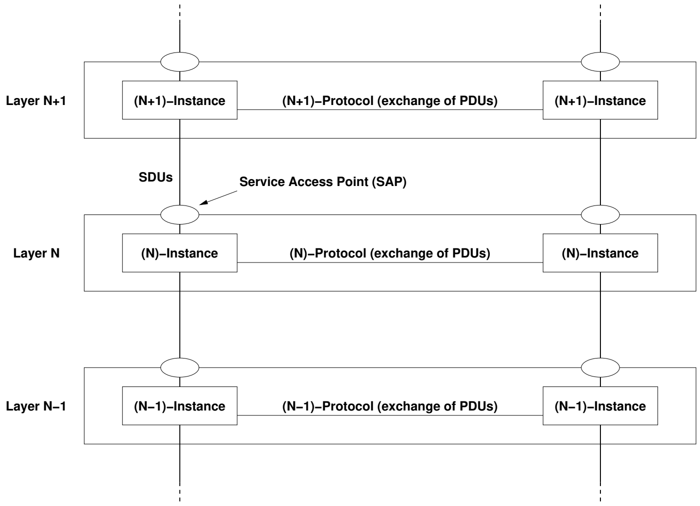
7 Transmission Error Detection

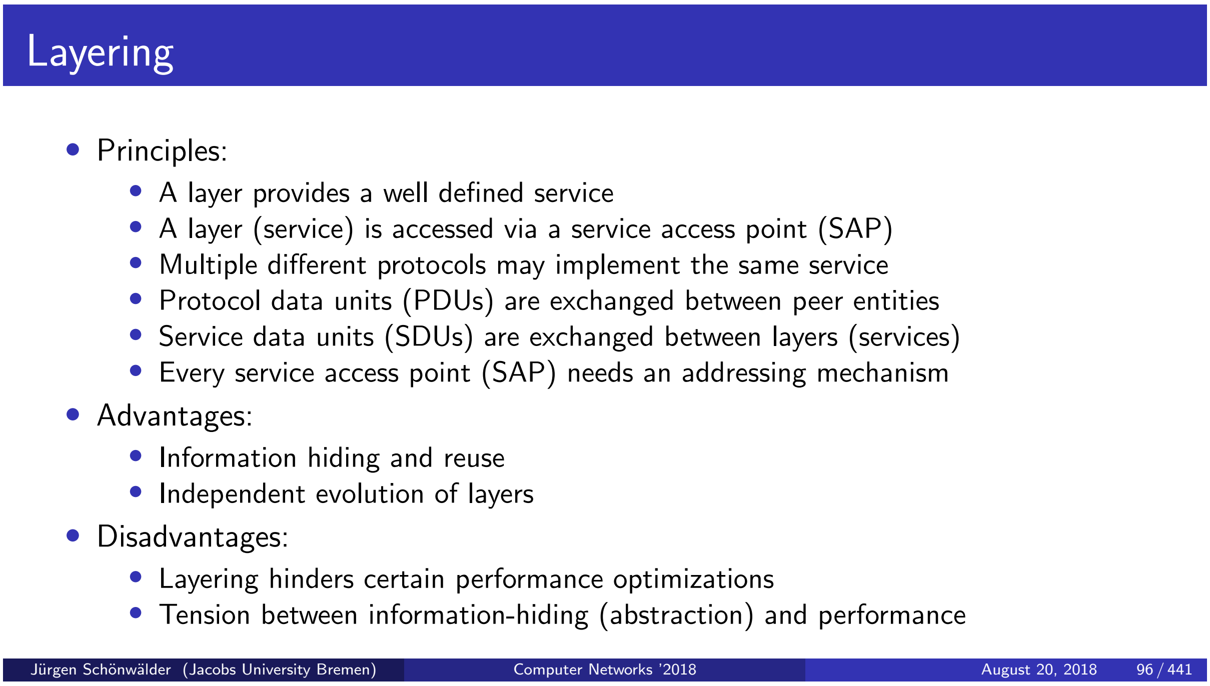
8 Sequence Numbers, Acknowledgements, Timer

9 Flow Control and Congestion Control

10 Layering and the OSI Reference Model

Layering Overview

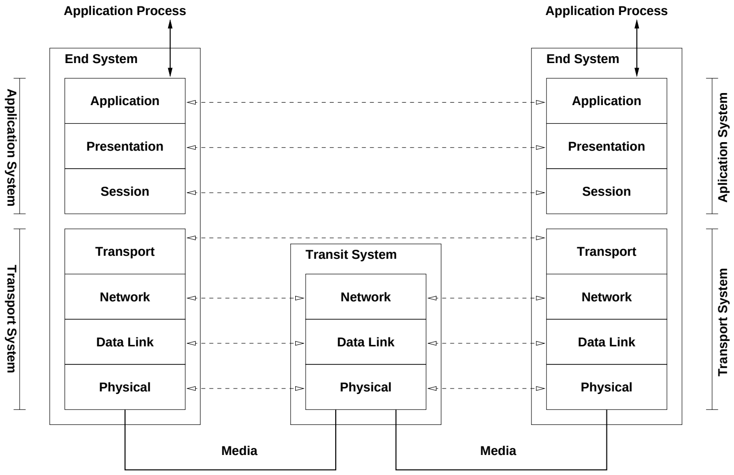

Layer N+1
(N+1)–Instance
(N+1)–Protocol (exchange of PDUs)
(N+1)–Instance
SDUs
Service Access Point (SAP)
Layer N
(N)–Instance
(N)–Protocol (exchange of PDUs)
(N)–Instance
Layer N–1
(N–1)–Instance
(N–1)–Protocol (exchange of PDUs)
(N–1)–Instance

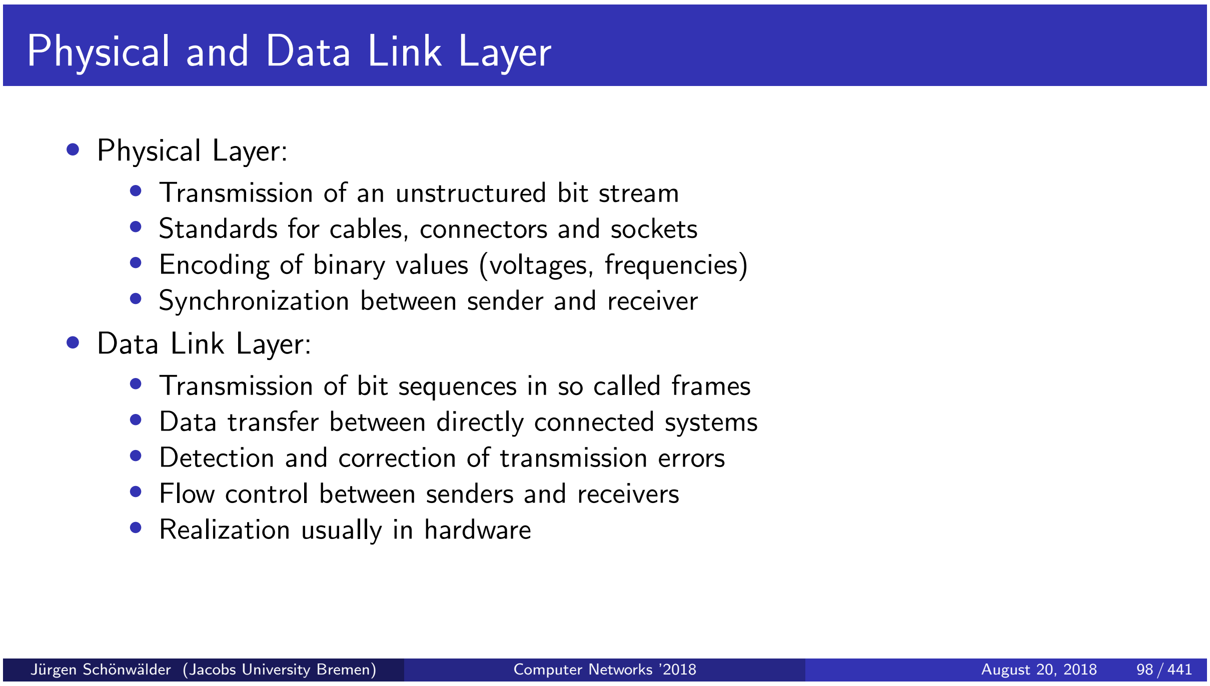


Layering

- Principles:
  - A layer provides a well defined service
  - A layer (service) is accessed via a service access point (SAP)
  - Multiple different protocols may implement the same service
  - Protocol data units (PDUs) are exchanged between peer entities
  - Service data units (SDUs) are exchanged between layers (services)
  - Every service access point (SAP) needs an addressing mechanism
- Advantages:
  - Information hiding and reuse
  - Independent evolution of layers
- Disadvantages:
  - Layering hinders certain performance optimizations
  - Tension between information-hiding (abstraction) and performance

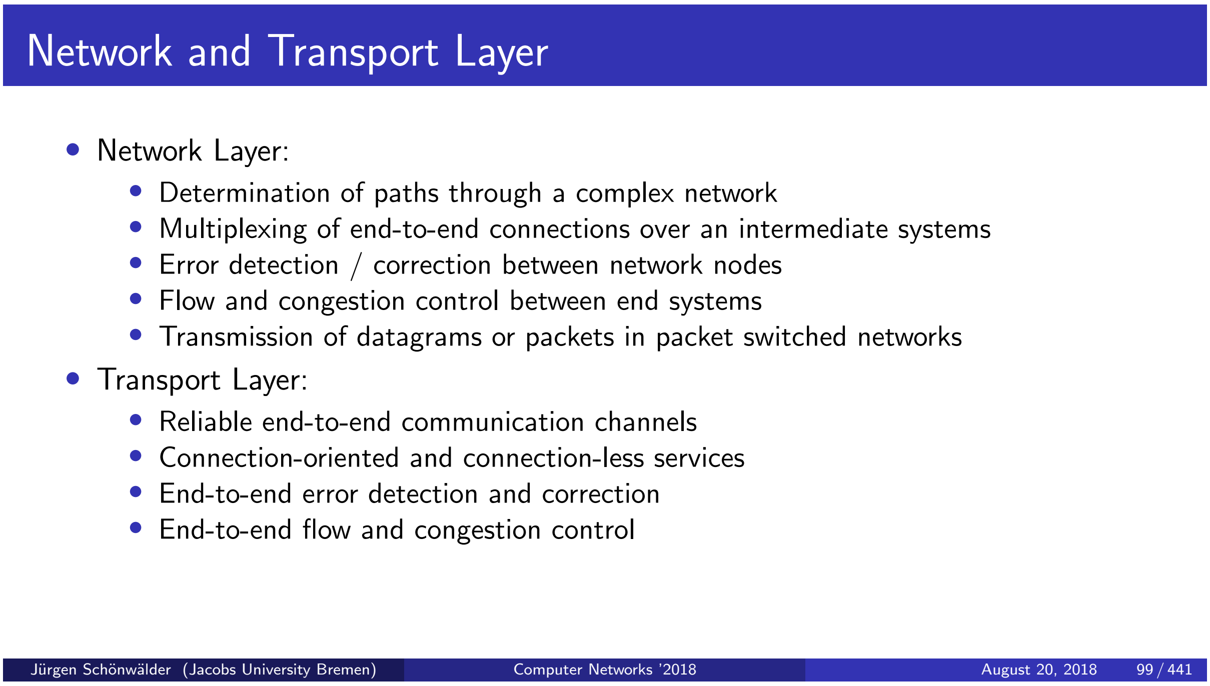
OSI Reference Model Overview





Physical and Data Link Layer

- Physical Layer:
  - Transmission of an unstructured bit stream
  - Standards for cables, connectors and sockets
  - Encoding of binary values (voltages, frequencies)
  - Synchronization between sender and receiver
- Data Link Layer:
  - Transmission of bit sequences in so called frames
  - Data transfer between directly connected systems
  - Detection and correction of transmission errors
  - Flow control between senders and receivers
  - Realization usually in hardware



Network and Transport Layer

- Network Layer:
  - Determination of paths through a complex network
  - Multiplexing of end-to-end connections over an intermediate systems
  - Error detection / correction between network nodes
  - Flow and congestion control between end systems
  - Transmission of datagrams or packets in packet switched networks
- Transport Layer:
  - Reliable end-to-end communication channels
  - Connection-oriented and connection-less services
  - End-to-end error detection and correction
  - End-to-end flow and congestion control

Session, Presentation and Application Layer

- Session Layer:
  - Synchronization and coordination of communicating processes
  - Interaction control (check points)
- Presentation Layer:
  - Harmonization of different data representations
  - Serialization of complex data structures
  - Data compression
- Application Layer:
  - Fundamental application oriented services
  - Terminal emulationen, name and directory services, data base access, network management, electronic messaging systems, process and machine control, . . .

Part III

Local Area Networks (IEEE 802)

Local area networks connects computers within a limited area such as a university campus, an office building or a data center. Local area networking technology appeared in the 1970s and became commercially viable in the 1980s. The IEEE standardizes local area networking technology since 1980 under the project 802 (hence the name IEEE 802). IEEE organizes work in working groups, named after the project they belong to. The 802.11 working group, for example, defines standards for wireless networks while the 802.3 working group is responsible for the Ethernet standards.

Section 11: Local Area Networks Overview

11 Local Area Networks Overview

12 Ethernet (IEEE 802.3)

13 Bridges (IEEE 802.1)

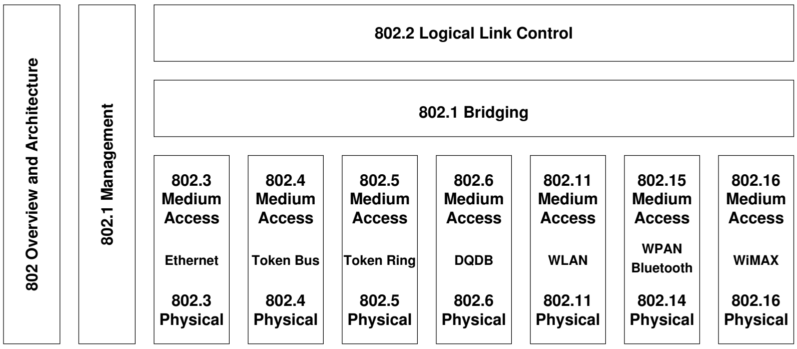
14 Virtual Local Area Networks (IEEE 802.1Q)

15 Port Access Control (IEEE 802.1X)

16 Wireless LAN (IEEE 802.11)

17 Logical Link Control (IEEE 802.2)

IEEE 802 Overview

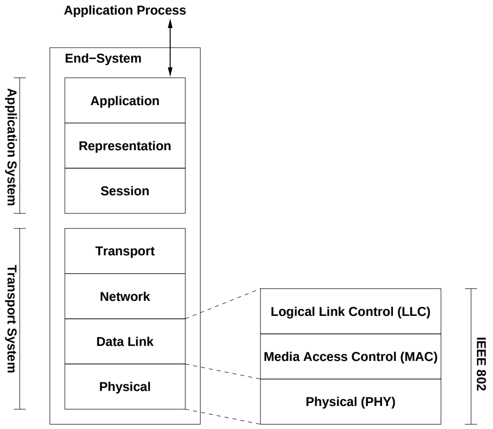


- IEEE 802 standards are developed since the early 1980s
- Dominating technology in local area networks (LANs)

Jürgen Schönwälder (Jacobs University Bremen) Computer Networks '2018 August 20, 2018 104 / 441

There are many more working groups than shown on the slide. IEEE 802 has been successful by standardizing technology developed originally outside the IEEE even if the standards were competing against each other in the market. One benefit of this approach is that IEEE 802 did not have to select a technology and it “owns” the winners that the market selected. Ethernet technology (802.3) has been a tremendous success as has been wireless LAN technology (802.11). Bridging technology (802.1) has been extremely successful as well, even though bridging technology is challenged these days by disruptive new approaches replacing distributed control protocols with logically centralized control software, leading to so called software-defined networks (SDN).

IEEE 802 Layers in the OSI Model



- The Logical Link Control layer provides a common service interface for all IEEE 802 protocols
- The Medium Access Control layer defines the method used to access the transmission media used
- The Physical layer defines the physical properties for the various transmission media that can be used with a certain IEEE 802.x protocol

The IEEE 802 standards cover the physical and data link layers of the OSI reference model. The data link layer is split into two parts, the media access control layer that is tied to the physical layer and the logical link control layer that aims at providing a common interface to higher layers in the protocol stack.

IEEE 802 Addresses

- IEEE 802 addresses (sometimes called MAC addresses or meanwhile also EUI-48 addresses) are 6 octets (48 bit) long
- The common notation is a sequence of hexadecimal numbers with the bytes separated from each other using colons or hyphens (`00:D0:59:5C:03:8A` or `00-D0-59-5C-03-8A`)
- The highest bit indicates whether it is a *unicast* address (0) or a *multicast* address (1). The second highest bit indicates whether it is a *local* (1) or a *global* (0) address
- The *broadcast* address, which represents all stations within a broadcast domain, is `FF-FF-FF-FF-FF-FF`
- Globally unique addresses are created by vendors who apply for a number space delegation by the IEEE

Since number spaces are delegated to vendors, it is possible to associate a MAC address to the vendor. For example, a MAC address can reveal that a system on a network was built by Apple or that a printer was built by Konica. The address assignment happens usually relatively late in the production process. While MAC addresses were often stored in erasable programmable read-only memory (EPROMs) in the 1990s and hence not easy to modify, this has changed over the years and meanwhile it is often possible to modify MAC addresses. But note that networking technology generally assumes MAC addresses to be unique and hence you have to be careful with changing your MAC address.

Section 12: Ethernet (IEEE 802.3)

11 Local Area Networks Overview

12 Ethernet (IEEE 802.3)

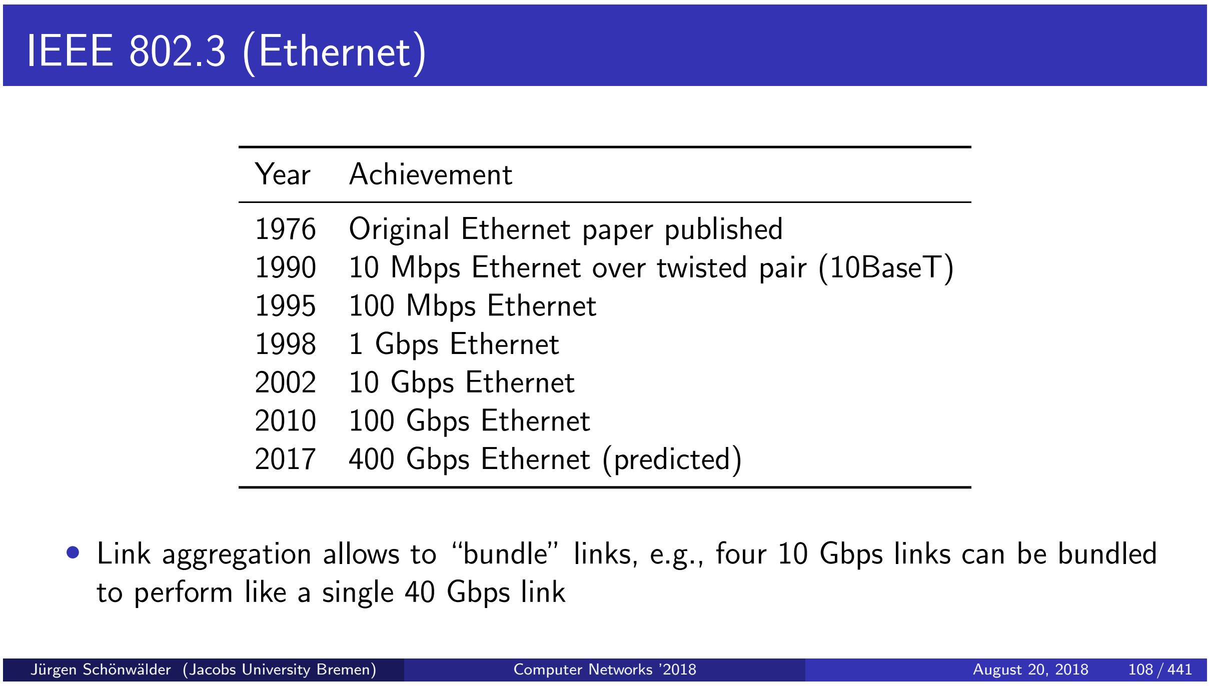
13 Bridges (IEEE 802.1)

14 Virtual Local Area Networks (IEEE 802.1Q)

15 Port Access Control (IEEE 802.1X)

16 Wireless LAN (IEEE 802.11)

17 Logical Link Control (IEEE 802.2)



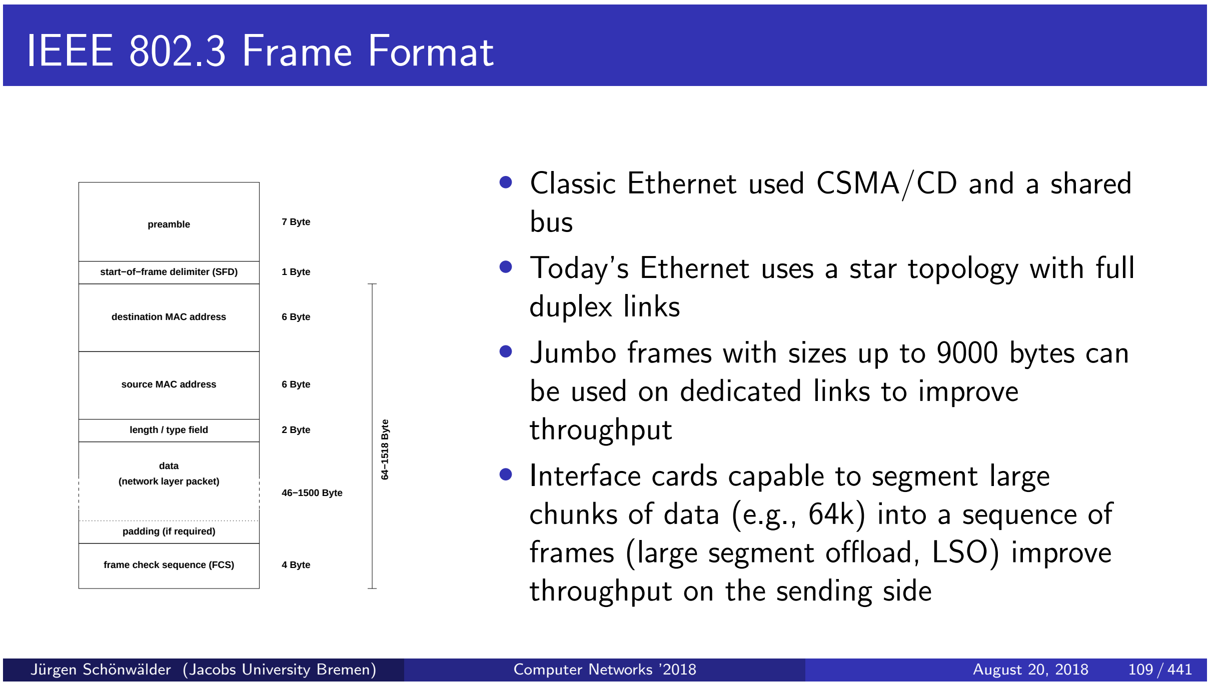
IEEE 802.3 (Ethernet)

Year	Achievement
1976	Original Ethernet paper published
1990	10 Mbps Ethernet over twisted pair (10BaseT)
1995	100 Mbps Ethernet
1998	1 Gbps Ethernet
2002	10 Gbps Ethernet
2010	100 Gbps Ethernet
2017	400 Gbps Ethernet (predicted)

- Link aggregation allows to “bundle” links, e.g., four 10 Gbps links can be bundled to perform like a single 40 Gbps link

Jürgen Schönwälder (Jacobs University Bremen) Computer Networks '2018 August 20, 2018 108 / 441

The increase of the data rate is slowing down. This has technical reasons but also economic reasons. The number of systems that need a port speed of 1 Tbps or more is relatively small. Breaking the relatively high development costs down on a relatively small market makes the cost per port very high. It is thus often cheaper to bundle links running at lower speeds to obtain the desired data rates. 1 Tbps Ethernet may still happen at some point in time but the market first has to grow to make this a viable development effort.



IEEE 802.3 Frame Format

- Classic Ethernet used CSMA/CD and a shared bus
- Today's Ethernet uses a star topology with full duplex links
- Jumbo frames with sizes up to 9000 bytes can be used on dedicated links to improve throughput
- Interface cards capable to segment large chunks of data (e.g., 64k) into a sequence of frames (large segment offload, LSO) improve throughput on the sending side

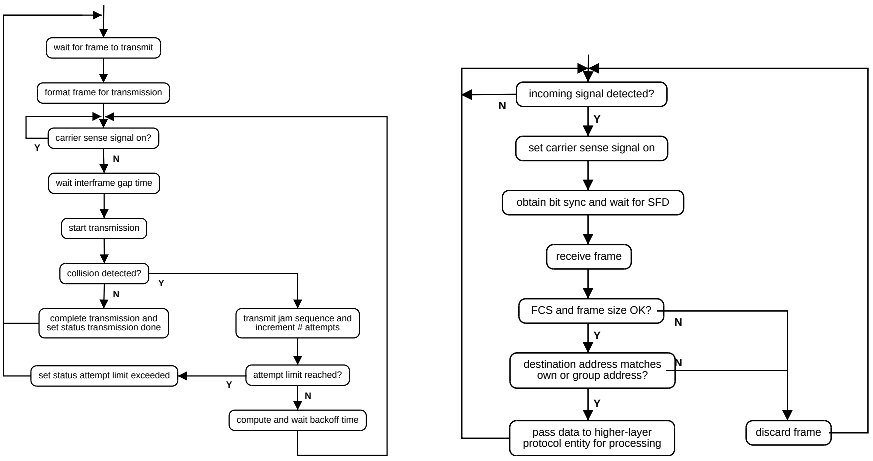
The 802.3 frame format starts with a preamble and a start-of-frame delimiter so that the receiving station can synchronize with the sender and identity the start of the data frame. The fields of the data frame have the following meaning:

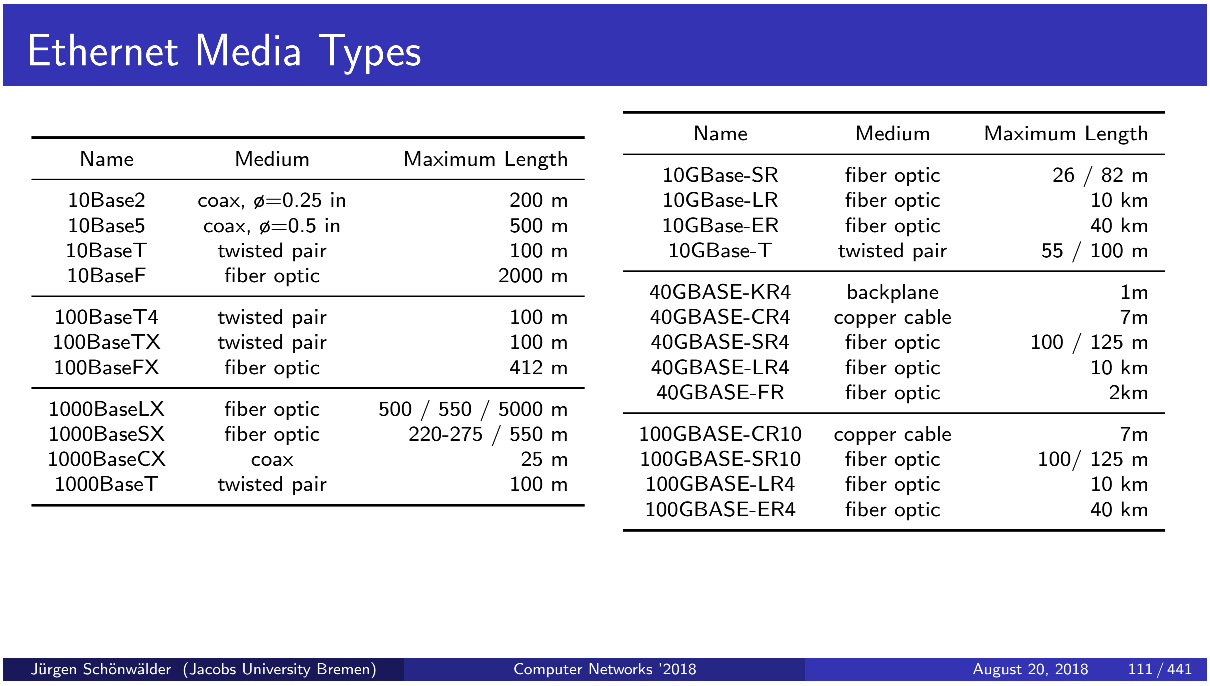
- Destination MAC Address: The MAC address of the intended recipient of the data frame.
- Source MAC Address: The MAC address of the originator of the data frame.
- Length / Type: The length of the frame (IEEE 802.3) or the type (Ethertype) of the data carried in the data frame. Type codes are larger than 1535 so that they do not collide with length values.
- Data: The payload carried in the data frame. If there is no type code, then the first bytes in the data field determine the type of the payload.
- Padding: Data may be padded in order to achieve a minimum frame size.
- Frame Check Sequence: A CRC-32 frame check sequence. The FCS is calculated over all data frame fields except the FCS itself.

The IEEE 802.3 frame format in a different notation:

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                           Preambel                            :
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
:                                               |     SFD       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                   Destination MAC Address                     :
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
:                               |                               :
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
:                      Source MAC Address                       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|      Type / Length            |           Data                :
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
:                                                               :
:            (46-1500 octets, padding if required)              :
:                                                               :
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                     Frame Check Sequence                      :
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
:                           (CRC-32)                            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Transmitting and Receiving IEEE 802.3 Frames (CSMA/CD)


wait for frame to transmit
format frame for transmission
carrier sense signal on?
Y
N
wait interframe gap time
start transmission
collision detected?
Y
N
complete transmission and set status transmission done
transmit jam sequence and increment # attempts
set status attempt limit exceeded
attempt limit reached?
Y
N
compute and wait backoff time
incoming signal detected?
N
Y
set carrier sense signal on
obtain bit sync and wait for SFD
receive frame
FCS and frame size OK?
N
Y
destination address matches own or group address?
N
Y
pass data to higher-layer protocol entity for processing
discard frame



Ethernet Media Types

Name	Medium	Maximum Length
10Base2	coax, ø=0.25 in	200 m
10Base5	coax, ø=0.5 in	500 m
10BaseT	twisted pair	100 m
10BaseF	fiber optic	2000 m
100BaseT4	twisted pair	100 m
100BaseTX	twisted pair	100 m
100BaseFX	fiber optic	412 m
1000BaseLX	fiber optic	500 / 550 / 5000 m
1000BaseSX	fiber optic	220-275 / 550 m
1000BaseCX	coax	25 m
1000BaseT	twisted pair	100 m

Name	Medium	Maximum Length
10GBase-SR	fiber optic	26 / 82 m
10GBase-LR	fiber optic	10 km
10GBase-ER	fiber optic	40 km
10GBase-T	twisted pair	55 / 100 m
40GBASE-KR4	backplane	1m
40GBASE-CR4	copper cable	7m
40GBASE-SR4	fiber optic	100 / 125 m
40GBASE-LR4	fiber optic	10 km
40GBASE-FR	fiber optic	2km
100GBASE-CR10	copper cable	7m
100GBASE-SR10	fiber optic	100/ 125 m
100GBASE-LR4	fiber optic	10 km
100GBASE-ER4	fiber optic	40 km

Twisted pair cables of the category CAT5e and CAT6 are meanwhile pretty much standard. These cables support 1000BaseT and hence they can be used for 1 Gbps Ethernet.

10 Gbps Ethernet requires CAT7 twisted pair cables. Higher speeds almost always require optical fibers, since copper cables only work on very short distances (e.g., within a rack in a data center).

Section 13: Bridges (IEEE 802.1)

11 Local Area Networks Overview

12 Ethernet (IEEE 802.3)

13 Bridges (IEEE 802.1)

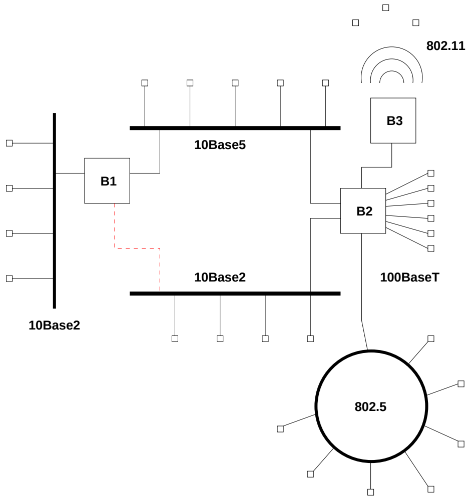
14 Virtual Local Area Networks (IEEE 802.1Q)

15 Port Access Control (IEEE 802.1X)

16 Wireless LAN (IEEE 802.11)

17 Logical Link Control (IEEE 802.2)

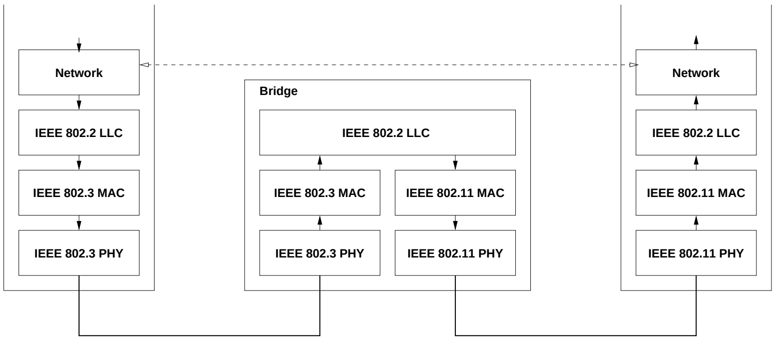
Bridged IEEE 802 Networks



As networks grew, it became necessary to interconnect network segments. The different segments often use different IEEE 802 technologies. The solution was the introduction of bridges that "bridge network segments". In the market, bridges are often called switches. Bridges that bridge a wireless segment to an Ethernet segment are often called access points (and they may contain additional functionality not found in plain bridges).

A key features of Ethernet and bridging technology was that it was very easy to deploy (plug and play). Mass production made the technology very cheap as well. However, whenever a technology is plug and play, it opens a number of attack vectors that can be exploited to capture, redirect, or even rewrite traffic. And it a plug and play technology is usually also open to relatively simple denial of service attacks.

IEEE 802 Bridges

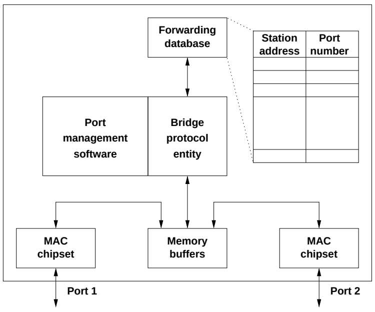


- *Source Routing Bridges*: Sender routes the frame through the bridged network
- *Transparent Bridges*: Bridges are transparent to senders and receivers

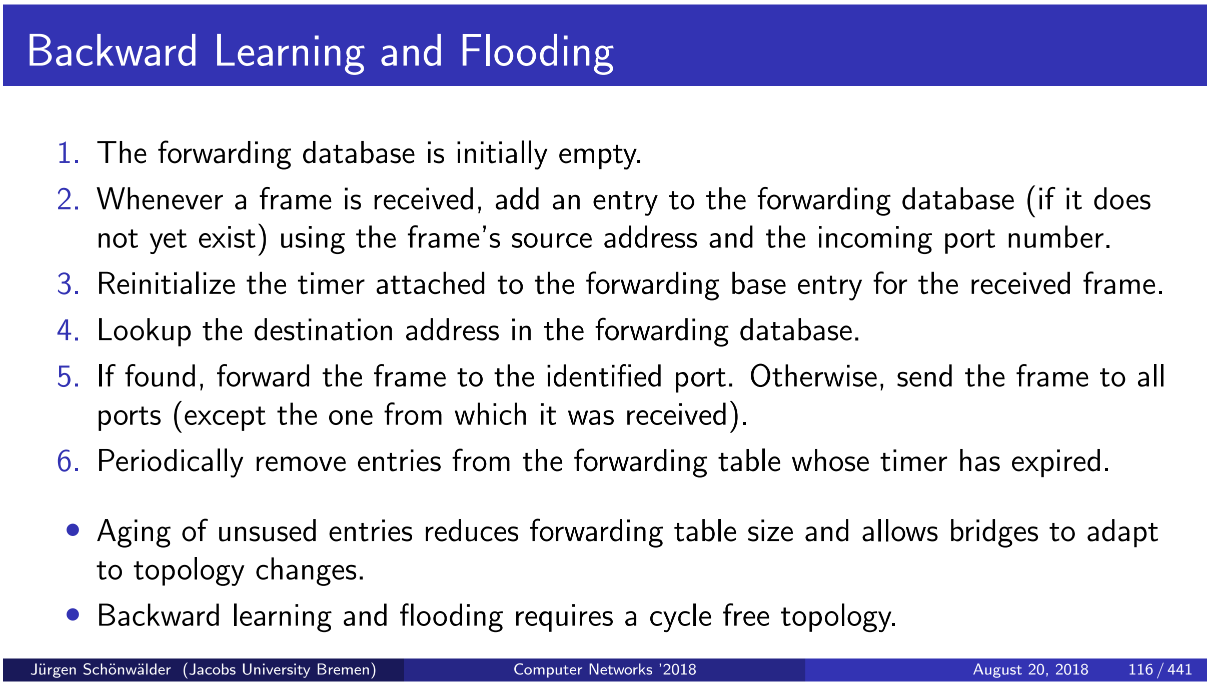
There was a large debate about source routing bridges versus transparent bridges when bridging technology was developed. Transparent bridges did win in the market since they reduce complexity at the connected stations and they were easier to deploy. In the following, we will only look at transparent bridges.

Nowadays, there is a movement to replace bridges with switches where an external controller guides the traffic flows through the network.

Transparent Bridges (IEEE 802.1D)



- Lookup an entry with a matching destination address in the forwarding database and forward the frame to the associated port.
- If no matching entry exists, forward the frame to all outgoing ports except the port from which the frame was received (flooding).



Backward Learning and Flooding

1. The forwarding database is initially empty.
2. Whenever a frame is received, add an entry to the forwarding database (if it does not yet exist) using the frame's source address and the incoming port number.
3. Reinitialize the timer attached to the forwarding base entry for the received frame.
4. Lookup the destination address in the forwarding database.
5. If found, forward the frame to the identified port. Otherwise, send the frame to all ports (except the one from which it was received).
6. Periodically remove entries from the forwarding table whose timer has expired.

- Aging of unused entries reduces forwarding table size and allows bridges to adapt to topology changes.
- Backward learning and flooding requires a cycle free topology.

Backward learning is simple to implement and plug and play (as long as there is a cycle free topology). Flooding is, however, not scalable on very large topologies. The backward learning algorithm also opens the floor to a number of attacks.

- A malicious station may generate frames with a source address of another station in order to redirect traffic to the malicious station.
- A malicious station may generate frames with random source addresses causing the bridges to age out valid entries.

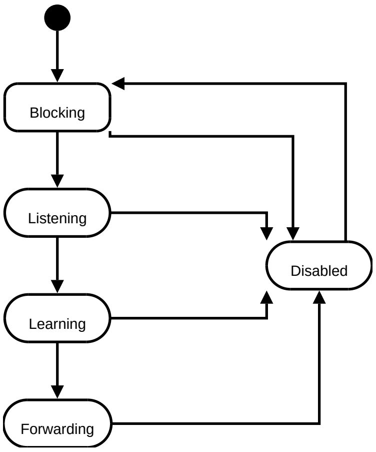
Spanning Tree

1. The root of the spanning tree is selected (root bridge). The root bridge is the bridge with the highest priority and the smallest bridge address.
2. The costs for all possible paths from the root bridge to the various ports on the bridges is computed (root path cost). Every bridge determines which root port is used to reach the root bridge at the lowest costs.
3. The designated bridge is determined for each segment. The designated bridge of a segment is the bridge which connects the segment to the root bridge with the lowest costs on its root port. The ports used to reach designated bridges are called designated ports.
4. All ports are blocked which are not designated ports or root ports. The resulting active topology is a spanning tree.

Port States

- Blocking: A port in the blocking state does not participate in frame forwarding.
- Listening: A port in the transitional listening state has been selected by the spanning tree protocol to participate in fame forwarding.
- Learning: A port in the transitional learning state is preparing to participate in frame forwarding.
- Forwarding: A port in the forwarding state forwards frames.
- Disabled: A port in the disabled state does not participate in frame forwarding or the operation of spanning tree protocol.

Port State Transitions



Bridge Protocol PDUs

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|      Protocol Identifier      |  Version ID   |  BPDU Type    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     Flags     |                                               :
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
:                     Root Bridge Identifier                    :
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
:               |                Root Path Cost                 :
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
:               |                                               :
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
:                        Bridge Identifier                      :
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
:               |        Port Identifier        |    Message    :
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
:     Age       |          Hello Time           |    Forward    :
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
:    Delay      |           Length              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

- BPDUs are sent periodically over all ports (including blocked ports) to a special multicast address
- BPDUs are usually encapsulated in an LLC header (see below)
- Failure to transmit or deliver BPDUs may result in briding errors

Broadcast Domains

- A bridged LAN defines a single *broadcast domain*:
  - All frames send to the broadcast address are forwarded on all links in the bridged networks.
  - Broadcast traffic can take a significant portion of the available bandwidth.
  - Devices running not well-behaving applications can cause *broadcast storms*.
  - Bridges may flood frames if the MAC address cannot be found in the forwarding table.
- It is desirable to reduce the size of broadcast domains in order to separate traffic in a large bridged LAN.
- Do not confuse a broadcast domain with a collision domain, i.e., segments on which media access collisions can occur.

Section 14: Virtual Local Area Networks (IEEE 802.1Q)

11 Local Area Networks Overview

12 Ethernet (IEEE 802.3)

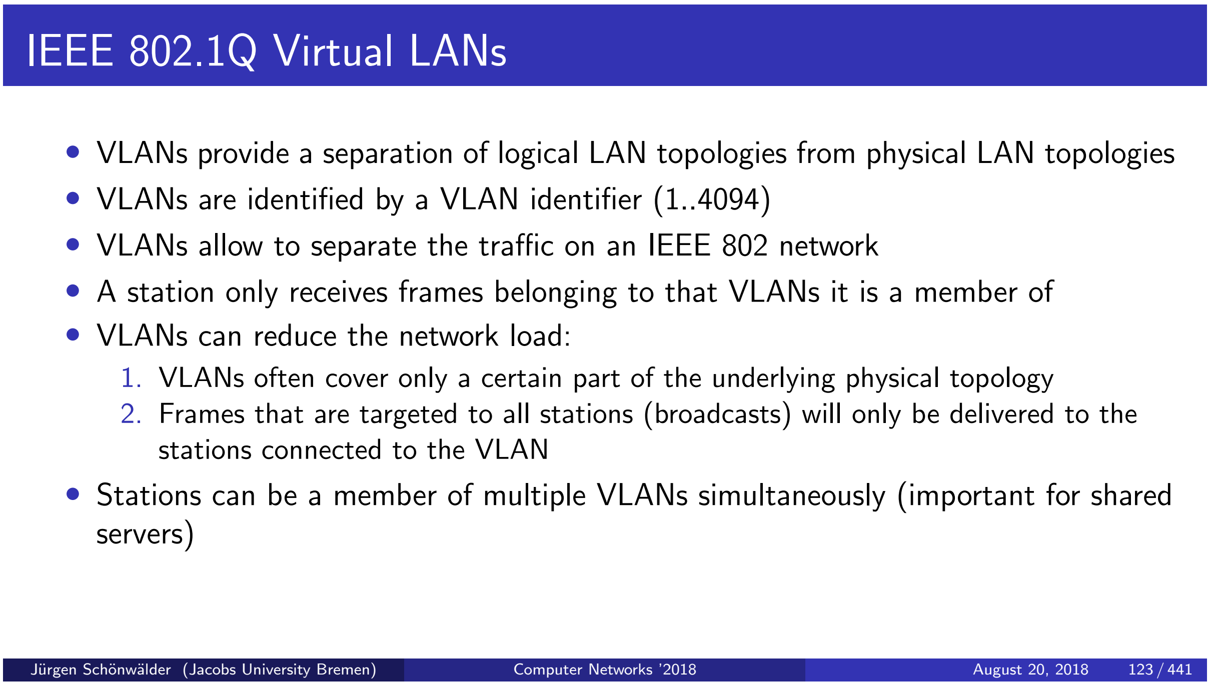
13 Bridges (IEEE 802.1)

14 Virtual Local Area Networks (IEEE 802.1Q)

15 Port Access Control (IEEE 802.1X)

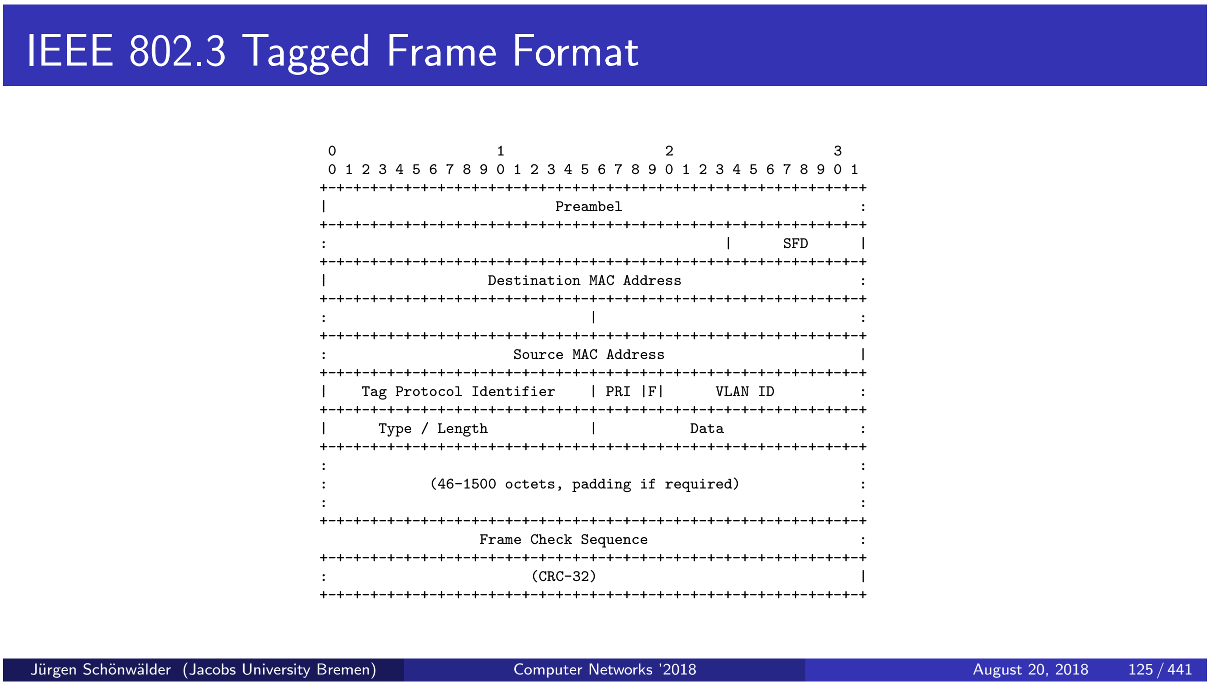
16 Wireless LAN (IEEE 802.11)

17 Logical Link Control (IEEE 802.2)



IEEE 802.1Q Virtual LANs

- VLANs provide a separation of logical LAN topologies from physical LAN topologies
- VLANs are identified by a VLAN identifier (1..4094)
- VLANs allow to separate the traffic on an IEEE 802 network
- A station only receives frames belonging to that VLANs it is a member of
- VLANs can reduce the network load:
  1. VLANs often cover only a certain part of the underlying physical topology
  2. Frames that are targeted to all stations (broadcasts) will only be delivered to the stations connected to the VLAN
- Stations can be a member of multiple VLANs simultaneously (important for shared servers)



IEEE 802.3 Tagged Frame Format

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                           Preambel                            :
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
:                                               |      SFD      |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                   Destination MAC Address                     :
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
:                               |                               :
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
:                      Source MAC Address                       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|    Tag Protocol Identifier    | PRI |F|      VLAN ID          :
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|      Type / Length            |           Data                :
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
:                                                               :
:            (46-1500 octets, padding if required)              :
:                                                               :
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
                   Frame Check Sequence                         :
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
:                        (CRC-32)                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

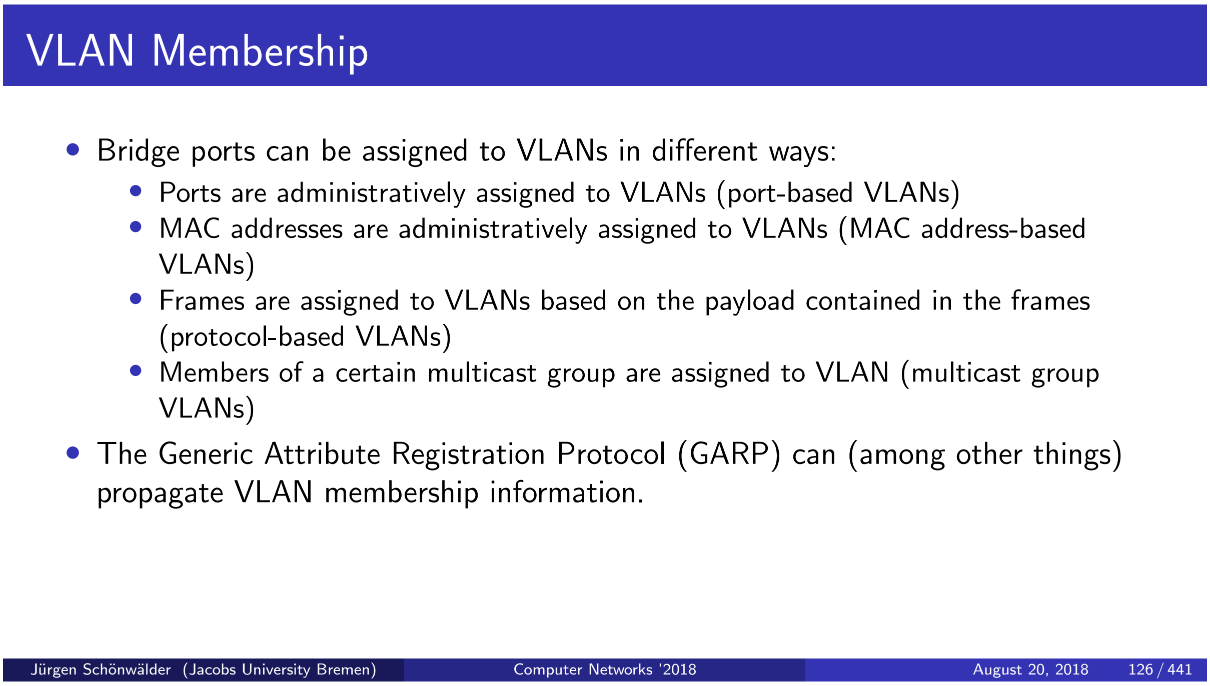
Jürgen Schönwälder (Jacobs University Bremen) Computer Networks '2018 August 20, 2018 125 / 441

In order to carry a VLAN identifier, also called a VLAN tag, it was necessary to extend the IEEE 802.3 frame format. The new fields are:

- Tag Protocol Identifier: 16-bit value set to of 0x8100 used to identify a frame as a tagged frame.
- PRI: 3-bit priority code point used by IEEE 802.1p to differentiate different classes of service.
- F: bit to indicate that data frames are eligible to be dropped in the presence of congestion.
- VLAN ID: 12-bit unsigned number identifying the VLAN the data frame belongs to. The VLAN ID value 0x000 indicates that the tagged frame does not belong to a VLAN. (This may be used in situations where only priority tagging is needed.)

The extension required to change the maximum size of IEEE 802.3 frames in order to make space available for the VLAN header. Since oversized frames are rejected by hardware that is not VLAN aware, the deployment of VLANs had to start with making the bridges VLAN capable. Once the bridges were updated to support tagged frames, VLAN tagged frames were exchanged between VLAN-aware switches while stations were largely unaware of the existence of VLANs. Meanwhile, networking hardware is generally able to deal with VLAN tagged frames.

Network segments carrying traffic belonging to multiple VLANs are often called trunks.



VLAN Membership

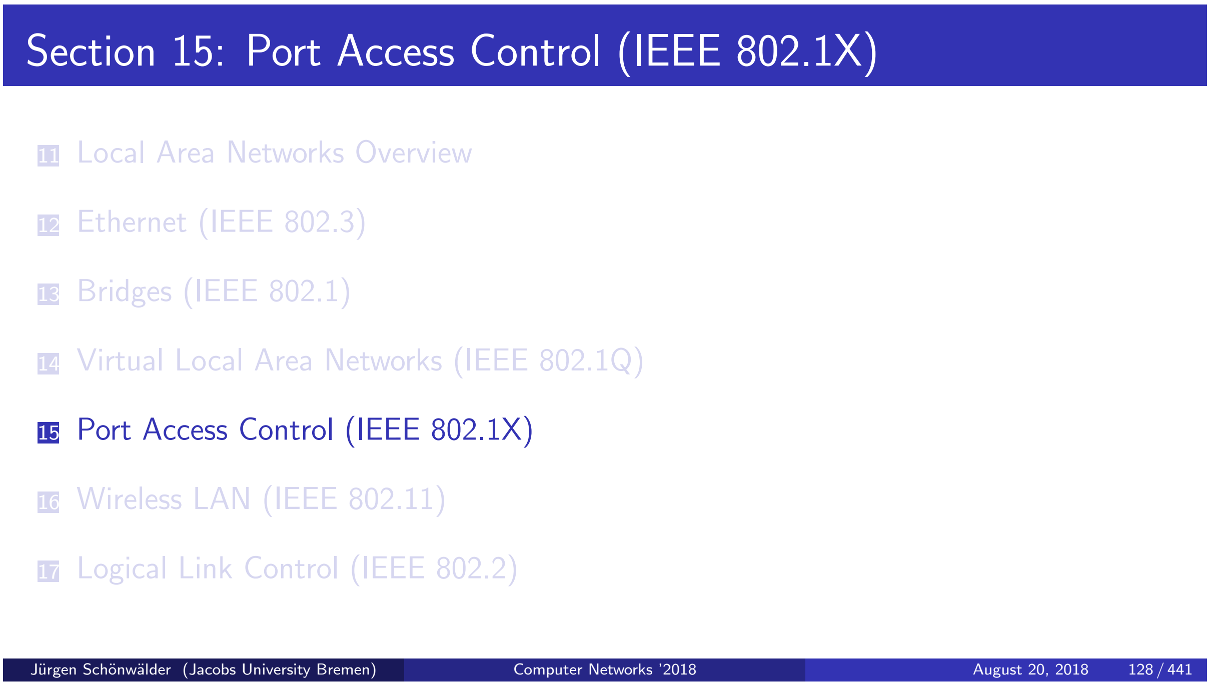
- Bridge ports can be assigned to VLANs in different ways:
  - Ports are administratively assigned to VLANs (port-based VLANs)
  - MAC addresses are administratively assigned to VLANs (MAC address-based VLANs)
  - Frames are assigned to VLANs based on the payload contained in the frames (protocol-based VLANs)
  - Members of a certain multicast group are assigned to VLAN (multicast group VLANs)
- The Generic Attribute Registration Protocol (GARP) can (among other things) propagate VLAN membership information.

In practice, VLAN membership is usually port-based.

IEEE 802.1 Q-in-Q Tagged Frames (IEEE 802.1ad)

- With two tags, a theoretical limit of $4096 \cdot 4096 = 16777216$ different tags can be achieved (larger tag space)
- A tag stack allows bridges to easily modify the tags since bridges can easily “push” or “pop” tags
- A tag stack creates a mechanism for ISPs to encapsulate customer single-tagged 802.1Q traffic with a single outer tag; the outer tag is used to identify traffic from different customers
- QinQ frames are convenient means of constructing Layer 2 tunnels, or applying Quality of service (QoS) policies, etc.
- 802.1ad is upward compatible with 802.1Q and although 802.1ad is limited to two tags, there is no ceiling on the standard allowing for future growth
- Double tagging is relatively easy to add to existing products

VLANs are quite common in corporate networks and they are getting increasingly used in home networks. Furthermore, access networks operated by Internet service providers are moving towards IEEE 802 technology. In order to connect for example a home office network using VLANs to a corporate network using VLANs via an Internet service provider network using IEEE 802 technology, it is necessary to carry VLAN tagged frames inside of VLAN tagged frames.



Section 15: Port Access Control (IEEE 802.1X)

11 Local Area Networks Overview

12 Ethernet (IEEE 802.3)

13 Bridges (IEEE 802.1)

14 Virtual Local Area Networks (IEEE 802.1Q)

15 Port Access Control (IEEE 802.1X)

16 Wireless LAN (IEEE 802.11)

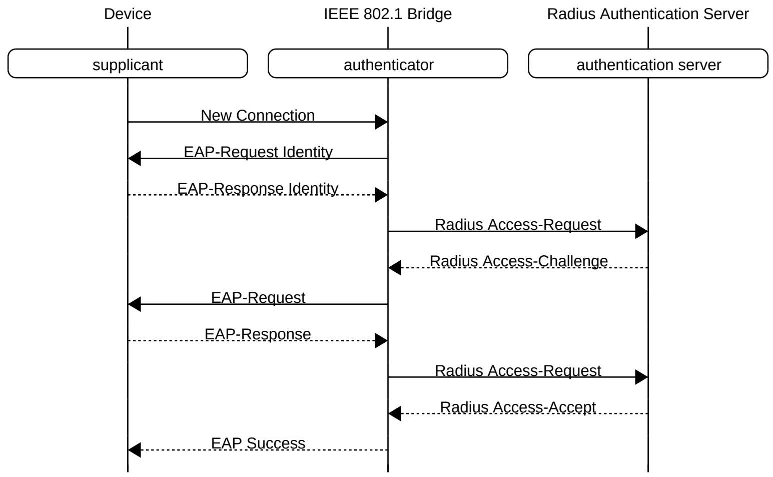
17 Logical Link Control (IEEE 802.2)

IEEE 802.1X Port Access Control

- Port-based network access control grants access to a switch port based on the identity of the connected machine.
- The components involved in 802.1X:
  - The *supplicant* runs on a machine connecting to a bridge and provides authentication information.
  - The *authenticator* runs on a bridge and enforces authentication decisions.
  - The *authentication server* is a (logically) centralized component which provides authentication decisions (usually via RADIUS).
- The authentication exchange uses the Extensible Authentication Protocol (EAP).
- IEEE 802.1X is becoming increasingly popular as a roaming solution for IEEE 802.11 wireless networks.

IEEE 802.1X provides a mechanism to securely and automatically join a network. Using IEEE 802.1X is much better for regular network access than other network access systems that attempt to redirect web traffic to web-based login pages (so called captive portals).

IEEE 802.1X Sequence Diagram



The Education Roaming (eduroam) network is an international roaming service for users in research, higher education and further education. It provides researchers, teachers, and students easy and secure network access when visiting an institution other than their own. The eduroam architecture is largely based on IEEE 802.1X, the Extensible Authentication Protocol (EAP), and RADIUS as the authentication protocol. For details, see the discussion in RFC 7593 [35].

Section 16: Wireless LAN (IEEE 802.11)

11 Local Area Networks Overview

12 Ethernet (IEEE 802.3)

13 Bridges (IEEE 802.1)

14 Virtual Local Area Networks (IEEE 802.1Q)

15 Port Access Control (IEEE 802.1X)

16 Wireless LAN (IEEE 802.11)

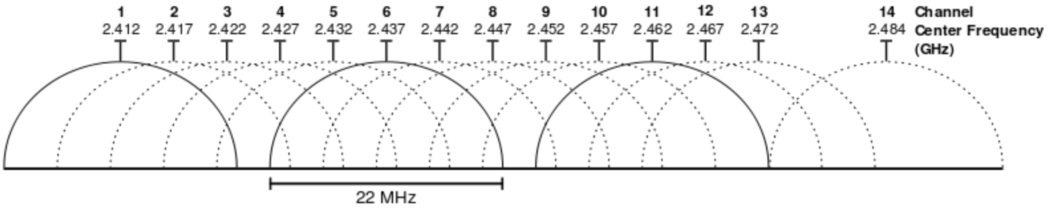
17 Logical Link Control (IEEE 802.2)

IEEE 802.11 Wireless LAN

Protocol	Released	Frequency	Data Rate	Indoor	Outdoor
802.11a	1999	5 GHz	54 Mbps	35m	120m
802.11b	1999	2.4 GHz	11 Mbps	38m	140m
802.11g	2003	2.4 GHz	54 Mbps	38m	140m
802.11n	2009	2.4/5 GHz	248 Mbps	70m	250m
802.11ac	2014	5 GHz	600 Mbps	70m	250m

- Very widely used wireless local area network (WLAN).
- As a consequence, very cheap equipment (base stations, interface cards).
- Wired equivalent privacy (WEP) was a disaster (at least for those who believe a wire is secure).
- Recommended is WPA-2 (Wifi Protected Access), in particular in combination with 802.1X and EAP-TLS.

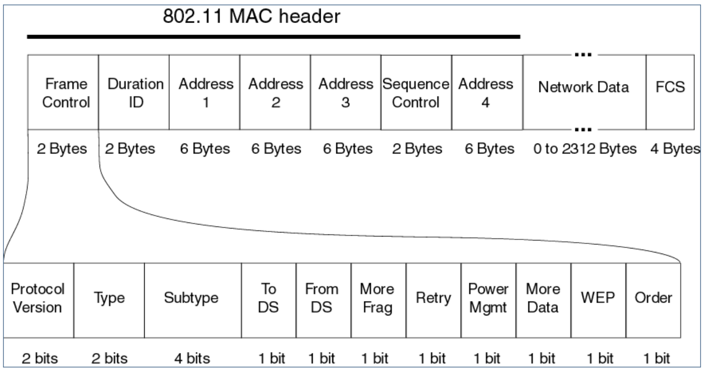
IEEE 802.11 2.4 GHz Channels



IEEE 802.11 Frame Types

- Data Frames: Carrying “useful” payloads
- Control Frames: Facilitate the exchange of data frames
  - Ready-to-send (RTS) and Clear-to-send (CTS) frames
  - Acknowledgement (ACK) frames
- Management Frames: Maintenance of the network
  - Beacon frames
  - Authentication / deauthentication frames
  - Association / deassociation frames
  - Probe request / response frames
  - Reassociation request / response frames

IEEE 802.11 Frame Format



The IEEE 802.11 frame format in a different notation:

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|         Frame Control         |         Duration ID           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                         Address #1                            :
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
:                               |                               :
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
:                         Address #2                            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
:                         Address #3                            :
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
:                               |       Sequence Control        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                         Address #4                            :
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
:                               |          Data                 :
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
:                                                               :
:                        (0-2312 octets)                        :
:                                                               :
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
                   Frame Check Sequence                         :
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
:                        (CRC-32)                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Section 17: Logical Link Control (IEEE 802.2)

11 Local Area Networks Overview

12 Ethernet (IEEE 802.3)

13 Bridges (IEEE 802.1)

14 Virtual Local Area Networks (IEEE 802.1Q)

15 Port Access Control (IEEE 802.1X)

16 Wireless LAN (IEEE 802.11)

17 Logical Link Control (IEEE 802.2)

IEEE Logical Link Control Header

```
 0                   1                   2
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| DSAP address  | SSAP address  | Control 8-bit |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+

 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| DSAP address  | SSAP address  |        Control 16-bit         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

- The LLC header follows the MAC frame header (but not always used)
- DSAP = Destination Service Access Point
- SSAP = Source Service Access Point
- Control = Various control bits, may indicate subsequent header

IEEE Logical Link Control SNAP Header (8-bit Control)

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| DSAP address  | SSAP address  | Control 8-bit |      OUI      :
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
:     (Org. Unit Identifier)    |           Protocol ID         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

- The LLC header may be followed by the Subnetwork Access Protocol (SNAP) header (if indicated by the Control field)
- The Organizational Unit Identifier (OUI) identifies an organization defining Protocol ID values
- The Protocol ID identifies the protocol in the following payload
- If the OUI is 0x000000, the Protocol ID contains an Ethernet type value

Part IV

Internet Network Layer

The Internet network layer provides a packet-oriented connection-less data exchange function. It has been designed to interconnect networks and it forms the basis of the global Internet. The main service provided by the Internet network layer is the delivery of packets from an arbitrary source interface to an arbitrary destination interface (or a set of destination addresses in the case of multicasts) in the Internet. For this to work, the Internet layer has to find paths, it has to deal with changes and errors, and it has to deal with some issues caused by crossing different types of subnetworks.

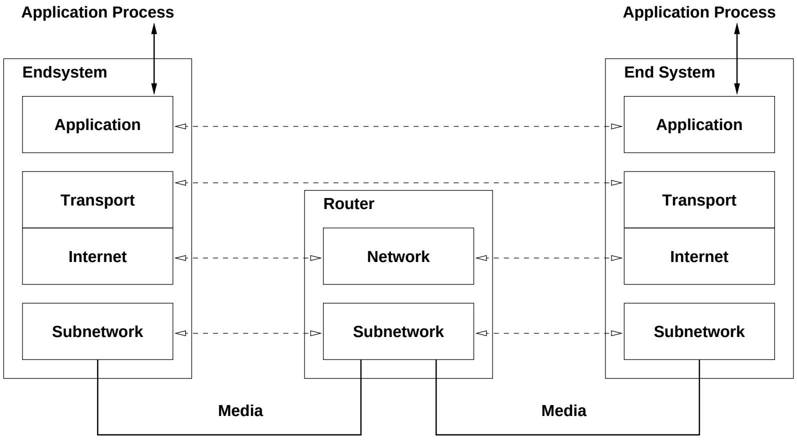
Section 18: Concepts and Terminology

18 Concepts and Terminology

19 Internet Protocol Version 6

20 Internet Protocol Version 4

Internet Reference Model



The Internet reference model is less detailed compared to the OSI reference model, but there are similarities.

- The Internet layer roughly corresponds to the OSI network layer.
- The Transport layer roughly corresponds to the OSI transport layer.

The subnetwork layer below the Internet layer is mostly seen as a black box. The idea is that the Internet layer can function over a large number of very different subnetwork layers. There are, however, a number of suggestions for designers of subnetwork layers, see [17]. Note that it will be possible to run the Internet layer over a subnetwork that is itself an Internet layer or a transport layer (this recursion leads to what networking people call tunnels).

The application layer is not broken into separate layers by the architectural model, leaving it to designers of application protocols to define how many application layer internal layers they want to have. A common trend is to divide the application layer into a secure session layer, an application protocol layer, and a content layer. But there are no strict rules and application layer designers have the freedom to do whatever is considered the best solution.

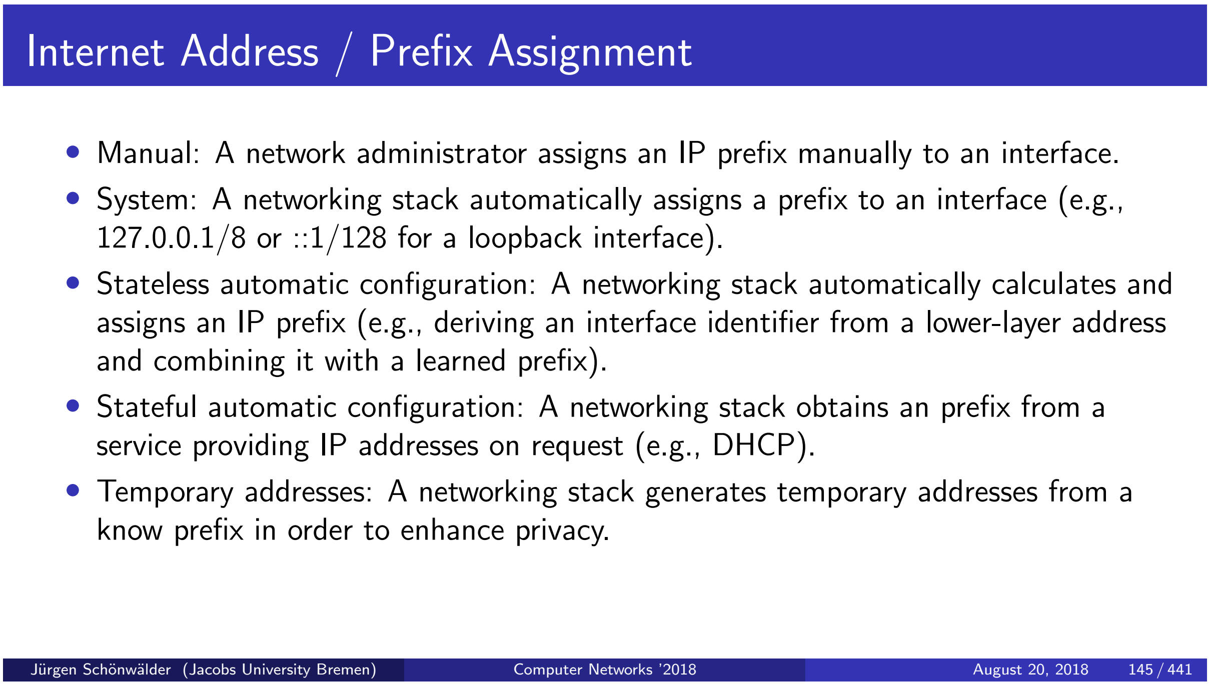
Terminology (1/2)

- A *node* is a device which implements an Internet Protocol (such as IPv4 or IPv6).
- A *router* is a node that forwards IP packets not addressed to itself.
- A *host* is any node which is not a router.
- A *link* is a communication channel below the IP layer which allows nodes to communicate with each other (e.g., an Ethernet).
- The *neighbors* is the set of all nodes attached to the same link.
- An *interface* is a node's attachement to a link.
- An *IP address* identifies an interface or a set of interfaces.

Note that a link connecting neighboring nodes can be simple or complex. A simple cable connecting exactly two nodes would be an example of a very simple link. An example for a much more complex link would be a large bridged IEEE 802 campus network involving several virtual LANs and integrating different transmission medias (e.g., fiber segments, twisted pair cobber wire segments, wireless segments).

Terminology (2/2)

- An *IP prefix* is the initial part of an IP address identifying an IP network. The IP prefix is commonly defined by the number of the initial bits of an IP address that are identifying an IP network, the so called *prefix length*.
- An *interface identifier* is the portion of an IP address that identifies an interface in a certain IP network.
- An *IP packet* is a bit sequence consisting of an IP header and the payload.
- The *link MTU* is the maximum transmission unit, i.e., maximum packet size in octets, that can be conveyed over a link.
- The *path MTU* is the the minimum link MTU of all the links in a path between a source node and a destination node.



Internet Address / Prefix Assignment

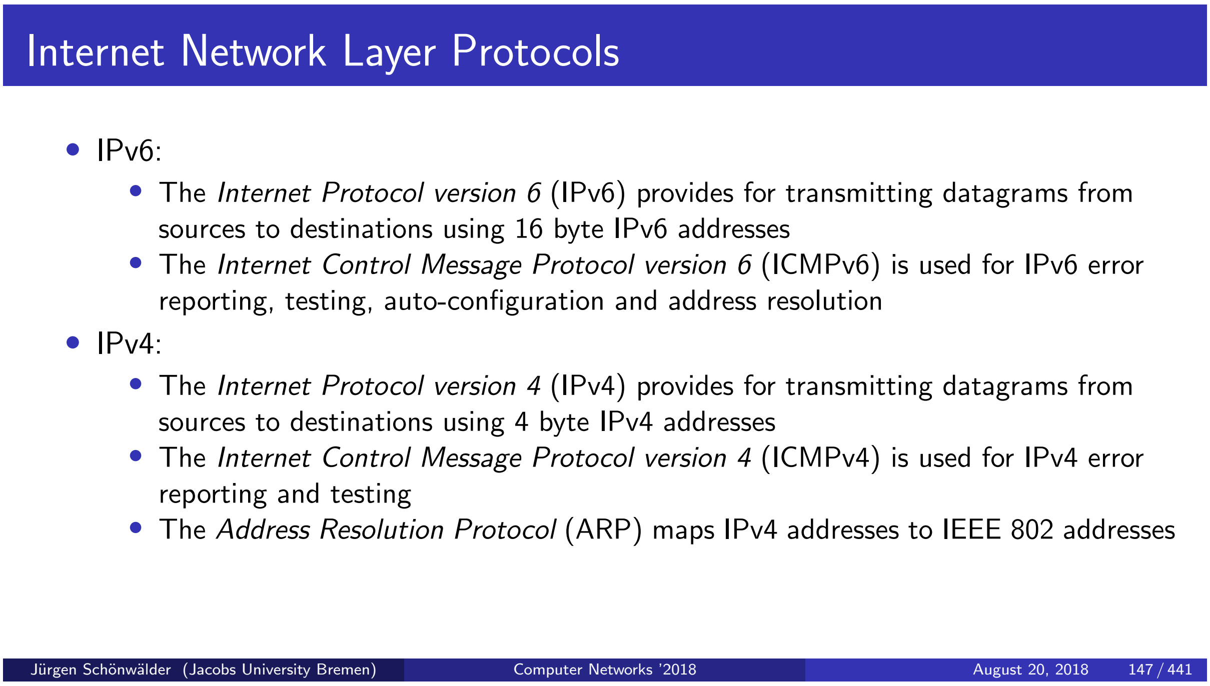
- Manual: A network administrator assigns an IP prefix manually to an interface.
- System: A networking stack automatically assigns a prefix to an interface (e.g., 127.0.0.1/8 or ::1/128 for a loopback interface).
- Stateless automatic configuration: A networking stack automatically calculates and assigns an IP prefix (e.g., deriving an interface identifier from a lower-layer address and combining it with a learned prefix).
- Stateful automatic configuration: A networking stack obtains an prefix from a service providing IP addresses on request (e.g., DHCP).
- Temporary addresses: A networking stack generates temporary addresses from a know prefix in order to enhance privacy.

There are additional address assignment techniques for specific purposes. Some examples:

- Derived addresses: A networking stack obtains an address or prefix from another valid address or prefix. This may be used in order to support address family translations.
- Cryptographic addresses: A networking stack generates an IP address from a cryptographic hash of a node's public key.

Jacobs University's IP Networks

- Jacobs University currently uses the global IPv4 address blocks 212.201.44.0/22 and 212.201.48.0/23. How many IPv4 addresses can be used in these two address spaces?
- 212.201.44.0/22: $2^{32-22} - 2 = 2^{10} - 2 = 1022$
  212.201.48.0/23: $2^{32-23} - 2 = 2^{9} - 2 = 510$
- Jacobs University currently uses the global IPv6 address block 2001:638:709::/48. How many IPv6 addresses can be used?
- 2001:638:709::/48: $2^{128-48} - 2 = 2^{80} - 2$ which is 1208925819614629174706174.
- If you equally distribute the addresses over the campus area ($30 \cdot 10^4 m^2$), what is the space covered per address?

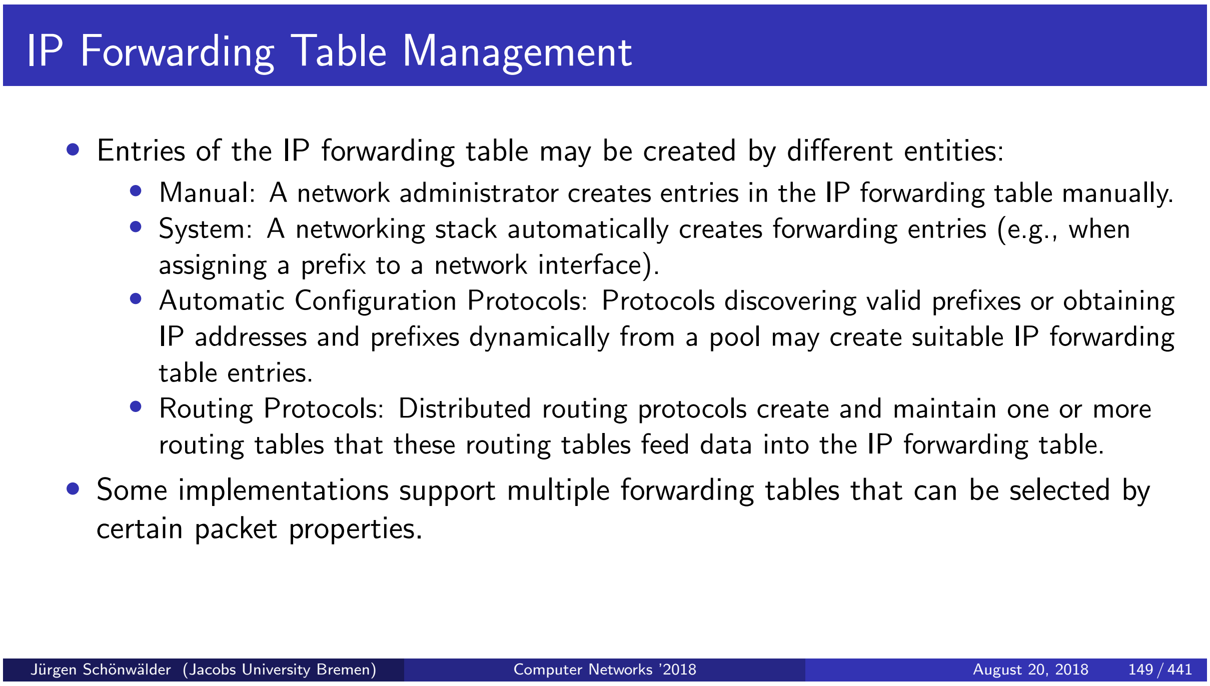


Internet Network Layer Protocols

- IPv6:
  - The *Internet Protocol version 6* (IPv6) provides for transmitting datagrams from sources to destinations using 16 byte IPv6 addresses
  - The *Internet Control Message Protocol version 6* (ICMPv6) is used for IPv6 error reporting, testing, auto-configuration and address resolution
- IPv4:
  - The *Internet Protocol version 4* (IPv4) provides for transmitting datagrams from sources to destinations using 4 byte IPv4 addresses
  - The *Internet Control Message Protocol version 4* (ICMPv4) is used for IPv4 error reporting and testing
  - The *Address Resolution Protocol* (ARP) maps IPv4 addresses to IEEE 802 addresses

IP Forwarding

- IP addresses can be devided into a part which identifies a network (the network prefix) and a part which identifies an interface of a node within that network (the interface identifier).
- The *forwarding table* realizes a mapping of the network prefix to the next node (next hop) closer to the destination and the local interface used to reach the next node.
- For every IP packet, the entry in the forwarding table with the longest matching prefix for the destination address has to be found (longest prefix match).
- A default forwarding table entry (if it exists) uses a zero-length prefix, that is either 0.0.0.0/0 (IPv4) or ::/0 (IPv6).



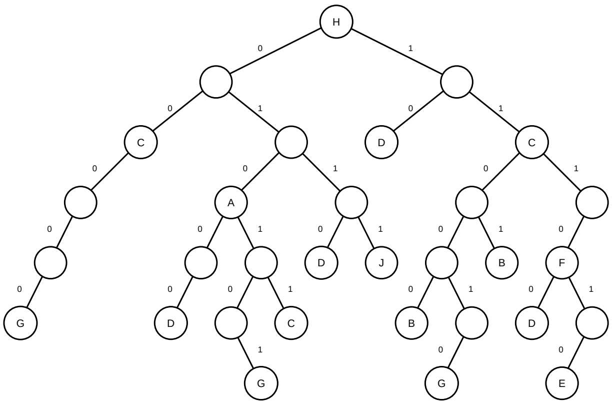
IP Forwarding Table Management

- Entries of the IP forwarding table may be created by different entities:
  - Manual: A network administrator creates entries in the IP forwarding table manually.
  - System: A networking stack automatically creates forwarding entries (e.g., when assigning a prefix to a network interface).
  - Automatic Configuration Protocols: Protocols discovering valid prefixes or obtaining IP addresses and prefixes dynamically from a pool may create suitable IP forwarding table entries.
  - Routing Protocols: Distributed routing protocols create and maintain one or more routing tables that these routing tables feed data into the IP forwarding table.
- Some implementations support multiple forwarding tables that can be selected by certain packet properties.

The terminology is often very imprecise here. People often call the forwarding table a routing table. However, people familiar with routing protocols tend to make a clear difference between the forwarding table that is actually used to forward traffic and the routing table(s) maintained by routing protocol implementations.

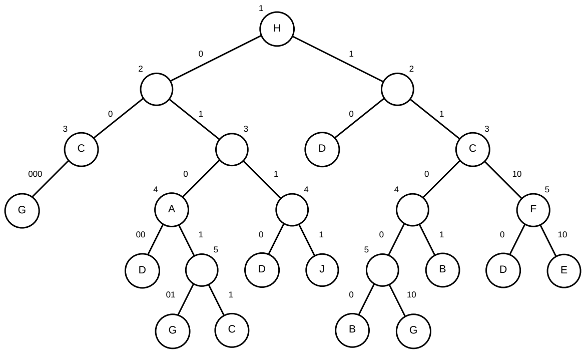
Modern routing protocol implementations usually support multiple routing protocols and they provide control mechanisms that can be used to select which entries of the various routing tables propagate to other routing tables and the forwarding table. On modern modular hardware-assisted routers, the forwarding table content is often distributed to the various line cards, with the routing protocol implementations running on a central processor, detached from the line cards forwarding traffic in hardware.

Longest Prefix Match: Binary Trie



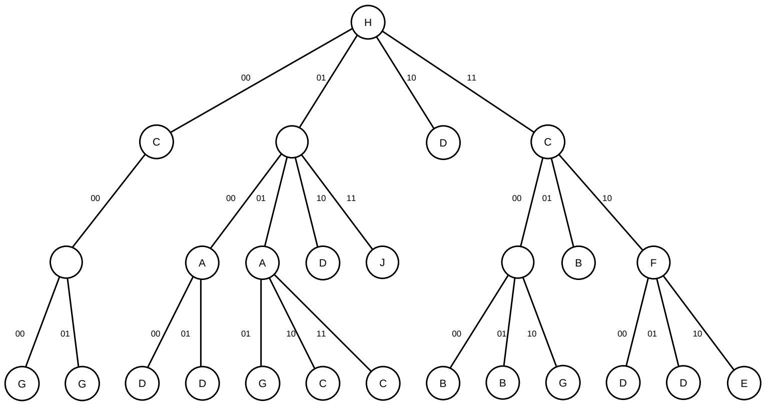
- A binary trie is the representation of the binary prefixes in a tree.

Longest Prefix Match: Path Compressed Trie

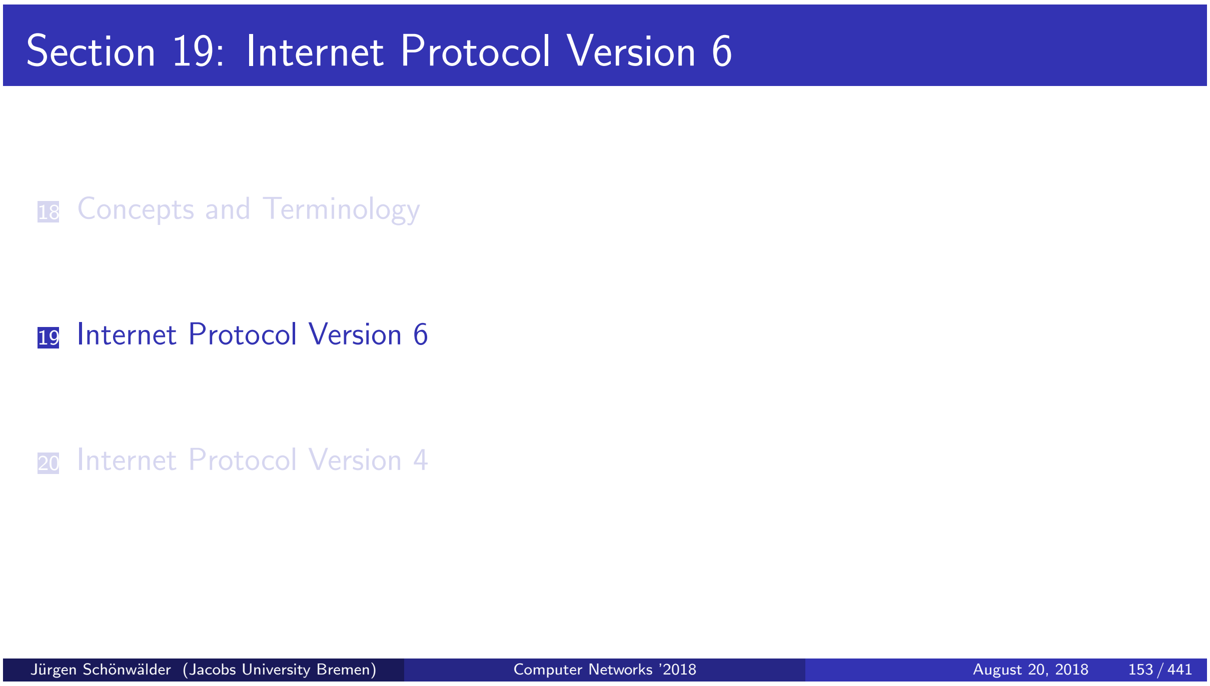


- A path compressed trie is obtained by collapsing all one-way branch nodes.
- The number attached to nodes indicates the next (absolute) bit to inspect.
- While walking down the tree, you verify in each step that the prefix still matches the prefix stored at each node.

Longest Prefix Match: Two-bit stride multibit Trie



- A two-bit multibit trie reduces the number of memory accesses.



Section 19: Internet Protocol Version 6

18 Concepts and Terminology

19 Internet Protocol Version 6

20 Internet Protocol Version 4

IPv6 Interface Identifier

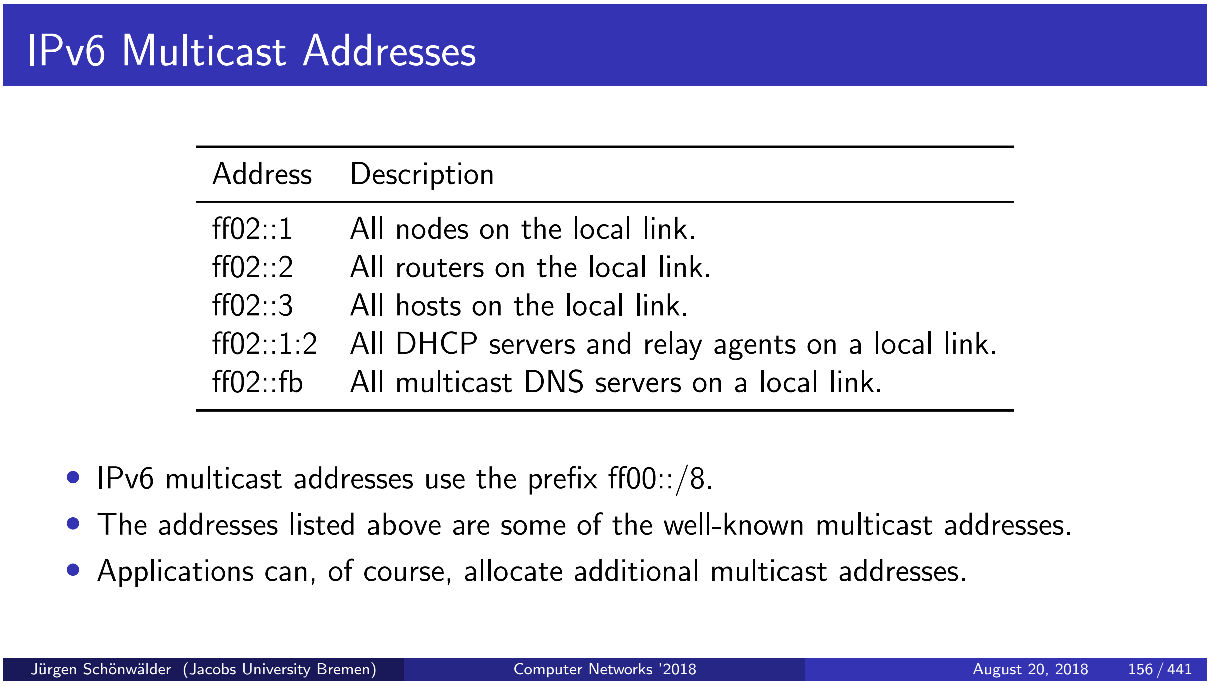
- Interface identifiers in IPv6 unicast addresses are used to uniquely indentify interfaces on a link.
- For unicast addresses, except those that start with binary 000, interface identifiers are generally required to be 64 bits long.
- Combination of the interface identifier with a network prefix leads to an IPv6 address.
- Link local unicast addresses have the prefix fe80::/10.
- Interface identifier may be obtained from an IEEE 802 MAC address using a modified EUI-64 format, but this has privacy issues.
- Alternatively, it is possible to use temporary interface identifiers that keep changing.

Modified EUI-64 Format

```
|0              1|1              3|3              4|
|0              5|6              1|2              7|
+----------------+----------------+----------------+
|cccccc0gcccccccc|ccccccccmmmmmmmm|mmmmmmmmmmmmmmmm|
+----------------+----------------+----------------+
```

```
|0              1|1              3|3              4|4              6|
|0              5|6              1|2              7|8              3|
+----------------+----------------+----------------+----------------+
|cccccc1gcccccccc|cccccccc11111111|11111110mmmmmmmm|mmmmmmmmmmmmmmmm|
+----------------+----------------+----------------+----------------+
```

- Modified EUI-64 format can be obtained from IEEE 802 MAC addresses.
- Warning: Ipv6 addresses derived from MAC addresses can be used to track mobile nodes used in different networks.
- Solution: Temporary addresses with interface identifiers based on time-varying random bit strings and relatively short lifetimes.



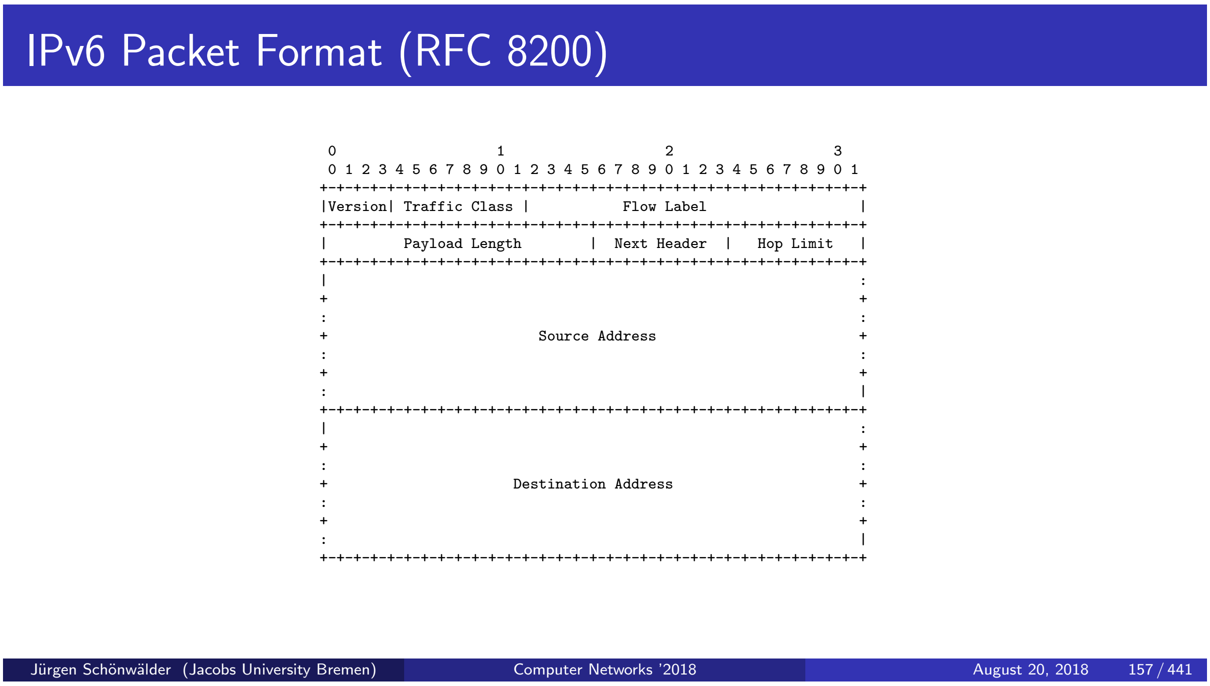
IPv6 Multicast Addresses

Address	Description
ff02::1	All nodes on the local link.
ff02::2	All routers on the local link.
ff02::3	All hosts on the local link.
ff02::1:2	All DHCP servers and relay agents on a local link.
ff02::fb	All multicast DNS servers on a local link.

- IPv6 multicast addresses use the prefix ff00::/8.
- The addresses listed above are some of the well-known multicast addresses.
- Applications can, of course, allocate additional multicast addresses.

Jürgen Schönwälder (Jacobs University Bremen) Computer Networks '2018 August 20, 2018 156 / 441

IPv6 uses multicast addresses in situations where IPv4 used broadcast addresses. This has certain advantages. For example, a request to find link local routers can be multicasted to all routers instead of sending a broadcast to all nodes.



IPv6 Packet Format (RFC 8200)

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|Version| Traffic Class |           Flow Label                  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|         Payload Length        |  Next Header  |   Hop Limit   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               :
+                                                               +
:                                                               :
+                         Source Address                        +
:                                                               :
+                                                               +
:                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               :
+                                                               +
:                                                               :
+                      Destination Address                      +
:                                                               :
+                                                               +
:                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

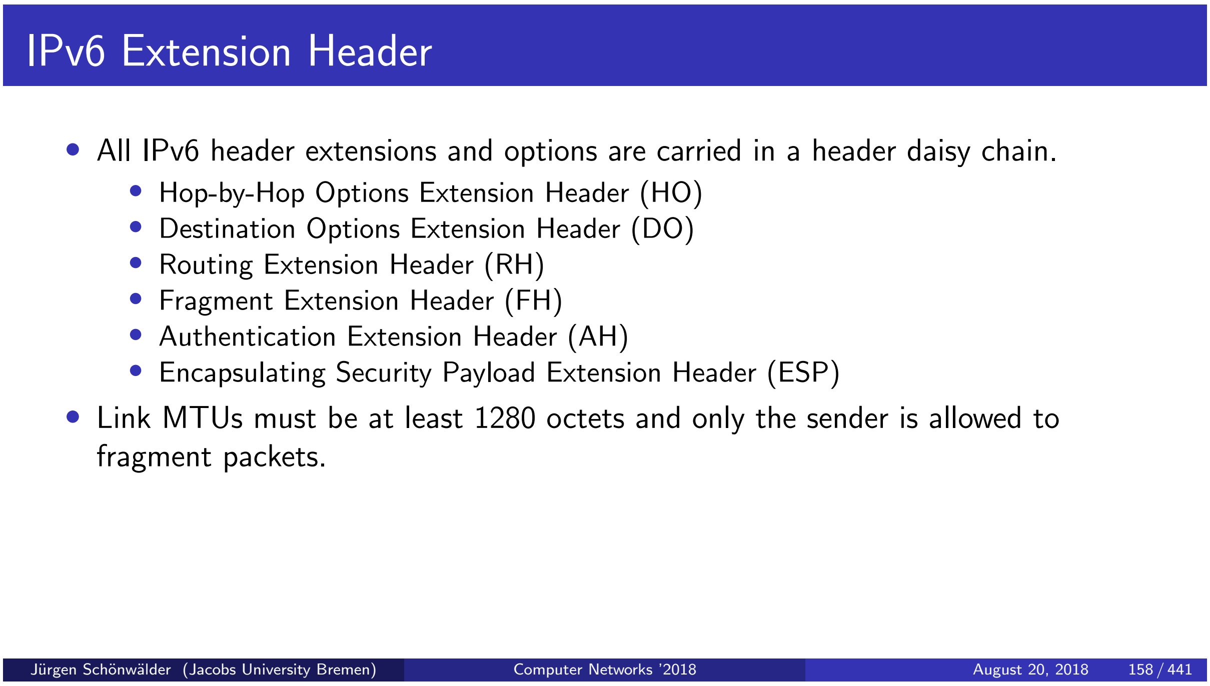
The IPv6 packet format is defined in RFC 8200 [8]. The packet fields have the following meaning:

- Version: 4-bit Internet Protocol version number (= 6).
- Traffic Class: 8-bit Traffic Class field. Used to carry Differentiated Services code points and Explicit Congestion Notification bits.
- Flow Label: 20-bit flow label. Can be used to tag IPv6 packets that belong to a flow. For details, see RFC 6437 [2].
- Payload Length: Length of the IPv6 payload, i.e., the rest of the packet following this IPv6 header, in octets.
- Next Header: 8-bit selector identifying the type of header immediately following the IPv6 header.
- Hop Limit: 8-bit unsigned integer, decremented by 1 by each node that forwards the packet. When forwarding, the packet is discarded if Hop Limit was zero when received or is decremented to zero.
- Source Address: 128-bit address of the originator of the packet.
- Destination Address: 128-bit address of the intended recipient of the packet.

Note that this is a header with a fixed size, there are no optional elements in the header itself.

The hop-limit field can be used to trace routes to hosts:

- Send an ICMPv6 Echo Request messages with increasing hop-limit values starting with one.
- A router counting the hop-limit down to zero will send a time exceeded ICMPv6 error message.
- The final destination will send an ICMPv6 Echo Reply message.



IPv6 Extension Header

- All IPv6 header extensions and options are carried in a header daisy chain.
  - Hop-by-Hop Options Extension Header (HO)
  - Destination Options Extension Header (DO)
  - Routing Extension Header (RH)
  - Fragment Extension Header (FH)
  - Authentication Extension Header (AH)
  - Encapsulating Security Payload Extension Header (ESP)
- Link MTUs must be at least 1280 octets and only the sender is allowed to fragment packets.

The basic packet forwarding functionality provided by the IPv6 header can be extended by using extension headers or a chain of extension headers.

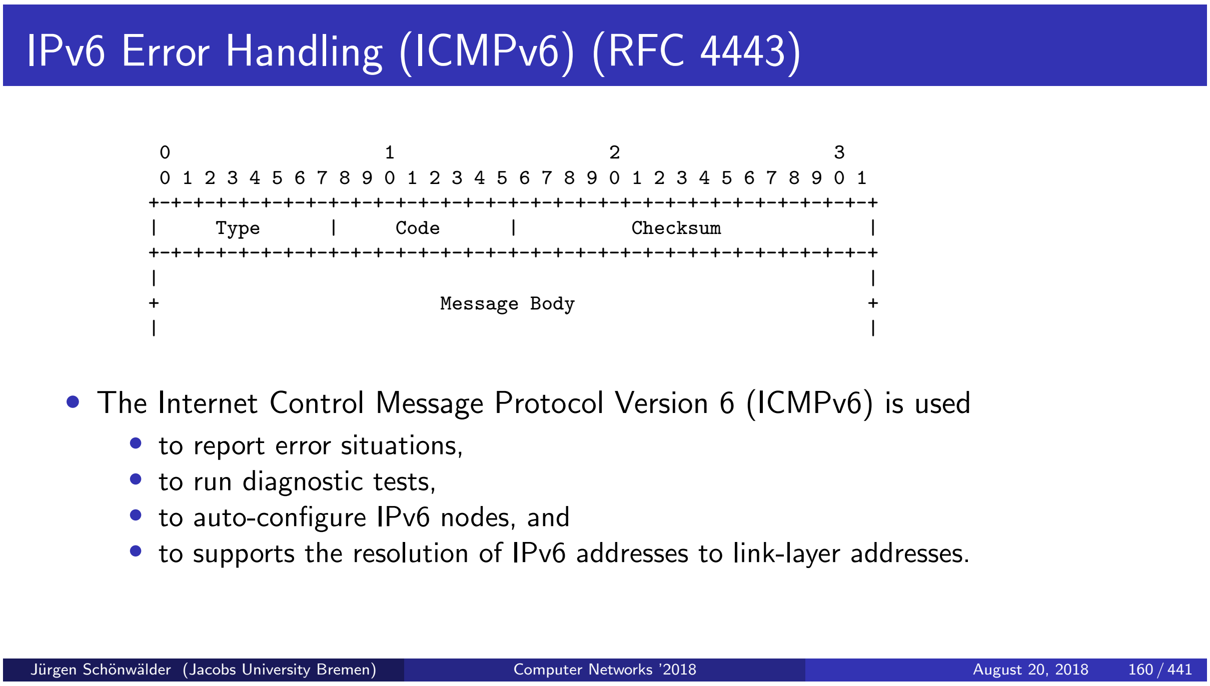
- Hop-by-Hop Options Extension Header (HO): Options that need to be examined by all devices on the path.
- Destination Options Extension Header (DO): Options that need to examined only by the destination of the packet.
- Routing Extension Header (RH): Used to direct a packet to one or more intermediate nodes before being sent to its destination
- Fragment Extension Header (FH): Used to send packets that are larger than the path MTU; only the originator of a packet is allowed to fragment the packet.
- Authentication Extension Header (AH): Authenticates the originator of a packet and ensures data integrity by using a hash function and a secret shared key. A sequence number can protect the IP packet's contents against replay attacks.
- Encapsulating Security Payload Extension Header (ESP): Authenticates the originator of a packet and ensures data integrity by using a hash functions and a secret shared key. Confidentiality is provided by encrypting portions of IP packets.

The general idea is that extension headers are used rarely. In fact, on some routing platforms, packets with extension headers are processed in software while packets without extension headers are processed entirely in hardware.

IPv6 Forwarding

- IPv6 packets are forwarded using the longest prefix match algorithm.
- IPv6 addresses have relatively long prefixes, which allows network operators to achieve better address aggregation, which reduces the number of forwarding table entries needed in the backbone infrastructure.
- Due to the length of the prefixes, it is crucial to use a longest prefix match algorithm whose complexity does not dependent on the number of entries in the forwarding table or the average prefix length.
- Due to better aggregation possibilities, IPv6 forwarding tables can be expected to be shorter than IPv4 forwarding tables

On a Linux system, it is possible to look at the IPv6 forwarding table using the `ip -6 route show` command. The `ip -6 route get` command can be used to query the IPv6 forwarding table for the forwarding entry that would be used for a given address. The `ip -6 route add` and `ip -6 route del` commands can be used to add or remove forwarding table entries. For all command options (and there are many), look at `ip -6 route help`.



IPv6 Error Handling (ICMPv6) (RFC 4443)

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     Type      |     Code      |          Checksum             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
+                         Message Body                          +
|                                                               |
```

- The Internet Control Message Protocol Version 6 (ICMPv6) is used
  - to report error situations,
  - to run diagnostic tests,
  - to auto-configure IPv6 nodes, and
  - to supports the resolution of IPv6 addresses to link-layer addresses.

The ICMPv6 protocol is defined in RFC 4443 [5]. It uses a generic header that is followed by a message type specific message body. The fields of the generic ICMPv6 header are:

- Type: The type field indicates the type of ICMPv6 message.
- Code: The code field provides further details about the message type and the value spaces is scoped by the type field.
- Checksum: The checksum field is used to detect corruption in the ICMPv6 header and parts of the IPv6 header preceding the ICMPv6 header.

Some of the allocated type values are listed in the following table:

Type	Description	
1	Destination Unreachable	RFC 4443
2	Packet Too Big	RFC 4443
3	Time Exceeded	RFC 4443
4	Parameter Problem	RFC 4443
128	Echo Request	RFC 4443
129	Echo Reply	RFC 4443
133	Router Solicitation	RFC 4861
134	Router Advertisement	RFC 4861
135	Neighbor Solicitation	RFC 4861
136	Neighbor Advertisement	RFC 4861

The following code values are used by a Destination Unreachable ICMPv6 message:

Code	Description
0	No route to destination
1	Communication with destination administratively prohibited
2	Beyond scope of source address
3	Address unreachable
4	Port unreachable
5	Source address failed ingress/egress policy
6	Reject route to destination

IPv6 Neighbor Discovery (RFC 4861)

- Discovery of the routers attached to a link.
- Discovery of the prefixes used on a link.
- Discovery of parameters such as the link MTU or the hop limit for outgoing packets.
- Automatic configuration of IPv6 addresses.
- Resolution of IPv6 addresses to link-layer addresses.
- Determination of next-hop addresses for IPv6 destination addresses.
- Detection of unreachable nodes which are attached to the same link.
- Detection of conflicts during address generation.
- Discovery of better alternatives to forward packets.

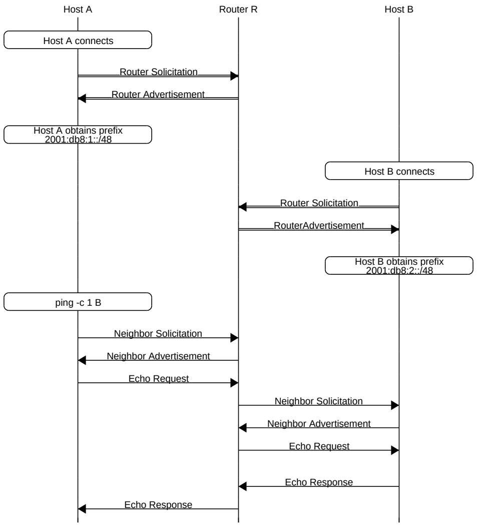
Jürgen Schönwälder (Jacobs University Bremen) Computer Networks '2018 August 20, 2018 161 / 441

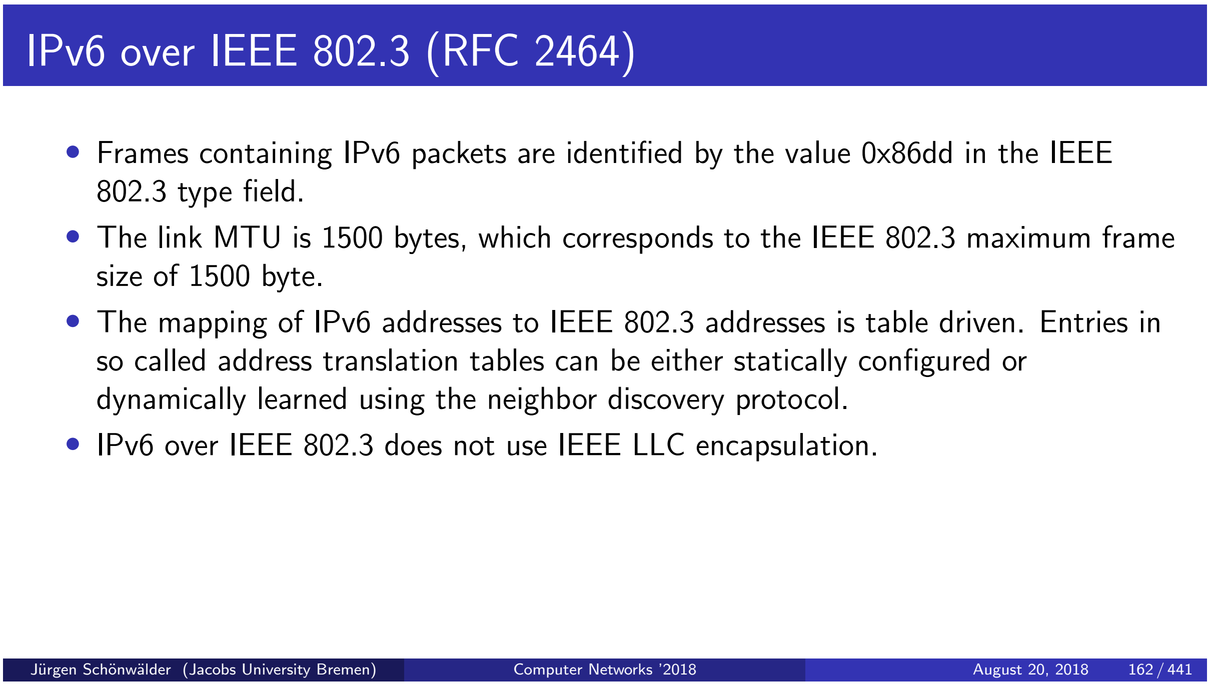
An IPv6 router may send periodically a Router Advertisement ICMPv6 message to the all nodes multicast address in order to announce its existence and to announce a prefix that is valid on a link. Nodes may also request that routers identify themselves by sending a Router Solicitation message to the all routers multicast address.

An IPv6 node may send a Neighbor Solicitations ICMPv6 message to the solicited-node multicast address in order to obtain the link-layer address of a target address contained in the IPv6 message body. A node responds by sending a Neighbor Advertisement message to the all nodes multicast address which adds the link-layer address to the target address in the message body.

Nodes generally cache neighborhood information and information about advertised routers and prefixes. On a Linux system, it is possible to look at the neighbor cache using the `ip -6 neigh show` command. Additional sub-commands can be used to add or delete neighbor cache entries. See `ip -6 neigh help` for more details.

The following time sequence diagram shows the usage of router and neighbor discovery messages.





IPv6 over IEEE 802.3 (RFC 2464)

- Frames containing IPv6 packets are identified by the value 0x86dd in the IEEE 802.3 type field.
- The link MTU is 1500 bytes, which corresponds to the IEEE 802.3 maximum frame size of 1500 byte.
- The mapping of IPv6 addresses to IEEE 802.3 addresses is table driven. Entries in so called address translation tables can be either statically configured or dynamically learned using the neighbor discovery protocol.
- IPv6 over IEEE 802.3 does not use IEEE LLC encapsulation.

IPv6 packets are easily sent over IEEE 802.3 links. RFC 2464 [6] allocates a suitable type code for the IEEE 802.3 length/type field.

IPv6 multicast addresses are mapped to IEEE 802.3 multicast addresses. Note that the mapping potentially maps multiple IPv6 multicast addresses to the same IEEE 802.3 multicast address.

A router solicitation / advertisement exchange may look as follows:

```
{ eth-dst(all-router) eth-src(A) eth-type(ipv6)
  { ipv6-src(A) ipv6-dst(all-router) ipv6-next(icmpv6) ipv6-hop-limit(255)
    { icmpv6-type(133) icmpv6-code(0) }}}

{ eth-dst(all-nodes) eth-src(R) eth-type(ipv6)
  { ipv6-src(R) ipv6-dst(all-nodes) ipv6-next(icmpv6) ipv6-hop-limit(255)
    { icmpv6-type(134) icmpv6-code(0) { ... }}}}
```

The ICMPv6 echo request / response packets may look as follows:

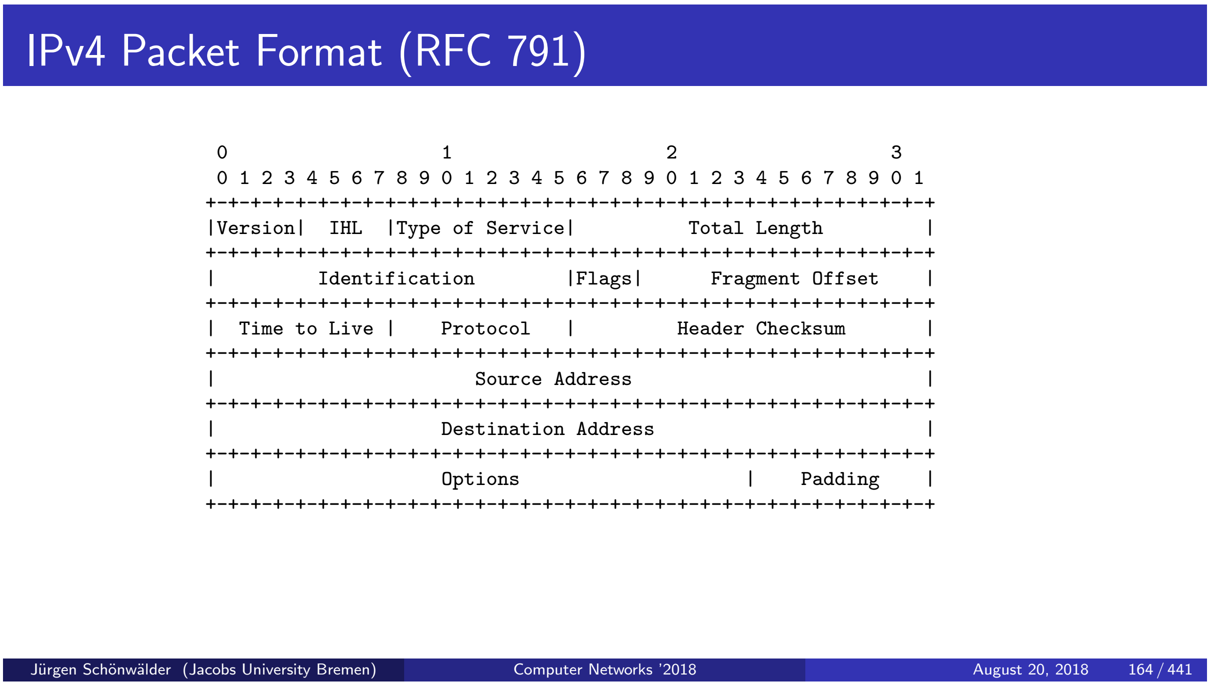
```
{ eth-dst(R) eth-src(A) eth-type(ipv6)
  { ipv6-src(A) ipv6-dst(B) ipv6-next(icmpv6) ipv6-hop-limit(42)
    { icmpv6-type(128) icmpv6-code(0) }}}

{ eth-dst(B) eth-src(R) eth-type(ipv6)
  { ipv6-src(A) ipv6-dst(B) ipv6-next(icmpv6) ipv6-hop-limit(41)
    { icmpv6-type(128) icmpv6-code(0) }}}

{ eth-dst(R) eth-src(B) eth-type(ipv6)
  { ipv6-src(B) ipv6-dst(A) ipv6-next(icmpv6) ipv6-hop-count(42)
    { icmpv6-type(129) icmpv6-code(0) }}}

{ eth-dst(A) eth-src(R) eth-type(ipv6)
  { ipv6-src(B) ipv6-dst(A) ipv6-next(icmpv6) ipv6-hop-count(41)
    { icmpv6-type(129) icmpv6-code(0) }}}
```

Section 20: Internet Protocol Version 4

18 Concepts and Terminology

19 Internet Protocol Version 6

20 Internet Protocol Version 4



IPv4 Packet Format (RFC 791)

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|Version|  IHL  |Type of Service|          Total Length         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|         Identification        |Flags|      Fragment Offset    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  Time to Live |    Protocol   |         Header Checksum       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                       Source Address                          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                    Destination Address                        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                    Options                    |    Padding    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

The IPv4 packet format is defined in RFC 791 [28]. The packet fields have the following meaning:

- Version: 4-bit Internet Protocol version number (= 4)
- IHL: 4-bit Internet Header Length in 32-bit words
- Type of Service: 8-bit Type of Service field. Used to carry Differentiated Services code points and Explicit Congestion Notification bits.
- Total Length: 16-bit total length of the packet in octets, including the IPv4 header and the data.
- Identification: 16-bit identification field used by the receiver to reassemble fragments.
- Flags: 3-bit control flags, including Don't Fragment (DF) and More Fragments (MF).
- Fragment Offset: 13-bit Fragment Offset indicates where a fragment belongs. The offset is measured in units of 8 octets (64 bits).
- Time to Live: 8-bit unsigned integer hop count field, decremented by 1 by each node that forwards the packet. When forwarding, the packet is discarded if Time to Live was zero when received or is decremented to zero.
- Protocol: 8-bit selector identifying the type of header immediately following the IPv4 header.
- Header Checksum: 16-bit checksum computed over the header. Since it includes fields that are modified by routers, it must be recomputed by every hop on a path.
- Source Address: 32-bit address of the originator of the packet.
- Destination Address: 32-bit address of the intended recipient of the packet.
- Options: Variable length list of header options.
- Padding: Data to align the header to a 32-bit boundary.

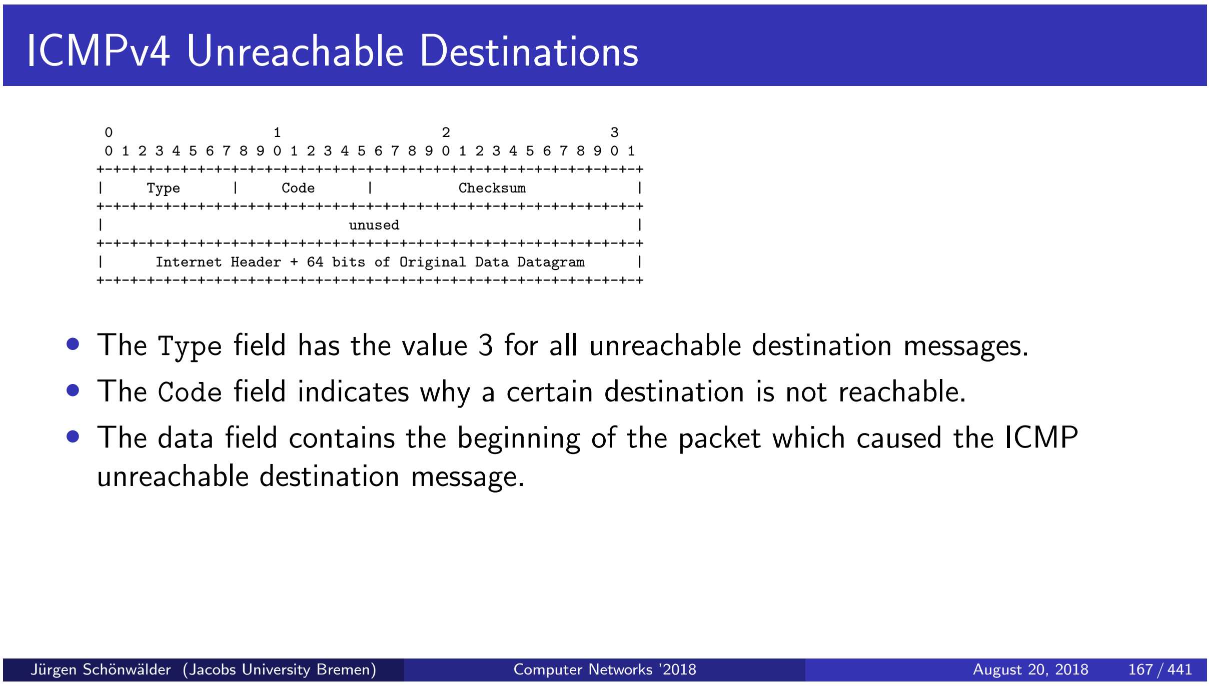
IPv4 Error Handling (ICMPv4)

- The Internet Control Message Protocol (ICMP) is used to inform nodes about problems encountered while forwarding IP packets.
  - Echo Request/Reply messages are used to test connectivity.
  - Unreachable Destination messages are used to report why a destination is not reachable.
  - Redirect messages are used to inform the sender of a better (shorter) path.
- Can be used to trace routes to hosts:
  - Send messages with increasing TTL starting with one and interpret the ICMP response message.
  - Pack additional data into the request to measure latency.
- ICMPv4 is an integral part of IPv4 (even though it is a different protocol).

ICMPv4 Echo Request/Reply

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     Type      |     Code      |          Checksum             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|           Identifier          |        Sequence Number        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     Data ...
+-+-+-+-+-
```

- The ICMP echo request message (type = 8, code = 0) asks the destination node to return an echo reply message (type = 0, code = 0).
- The `Identifier` and `Sequence Number` fields are used to correlate incoming replies with outstanding requests.
- The data field may contain any additional data.



ICMPv4 Unreachable Destinations

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     Type      |     Code      |          Checksum             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                             unused                            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|      Internet Header + 64 bits of Original Data Datagram      |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

- The `Type` field has the value 3 for all unreachable destination messages.
- The `Code` field indicates why a certain destination is not reachable.
- The data field contains the beginning of the packet which caused the ICMP unreachable destination message.

Jürgen Schönwälder (Jacobs University Bremen) Computer Networks '2018 August 20, 2018 167 / 441

Unreachable destination codes:

Code	Description
0	Network Unreachable
1	Host Unreachable
2	Protocol Unreachable
3	Port Unreachable
4	Fragmentation Needed and Don't Fragment was Set
5	Source Route Failed
6	Destination Network Unknown
7	Destination Host Unknown
8	Source Host Isolated
9	Communication with Destination Network is Administratively Prohibited
10	Communication with Destination Host is Administratively Prohibited
11	Destination Network Unreachable for Type of Service
12	Destination Host Unreachable for Type of Service
13	Communication Administratively Prohibited
14	Host Precedence Violation
15	Precedence cutoff in effect

Note that some unreachable destination messages are crucial for the Internet to function correctly. It is therefore crucial that delivery of unreachable destination messages is not generally administratively blocked.

ICMPv4 Redirect

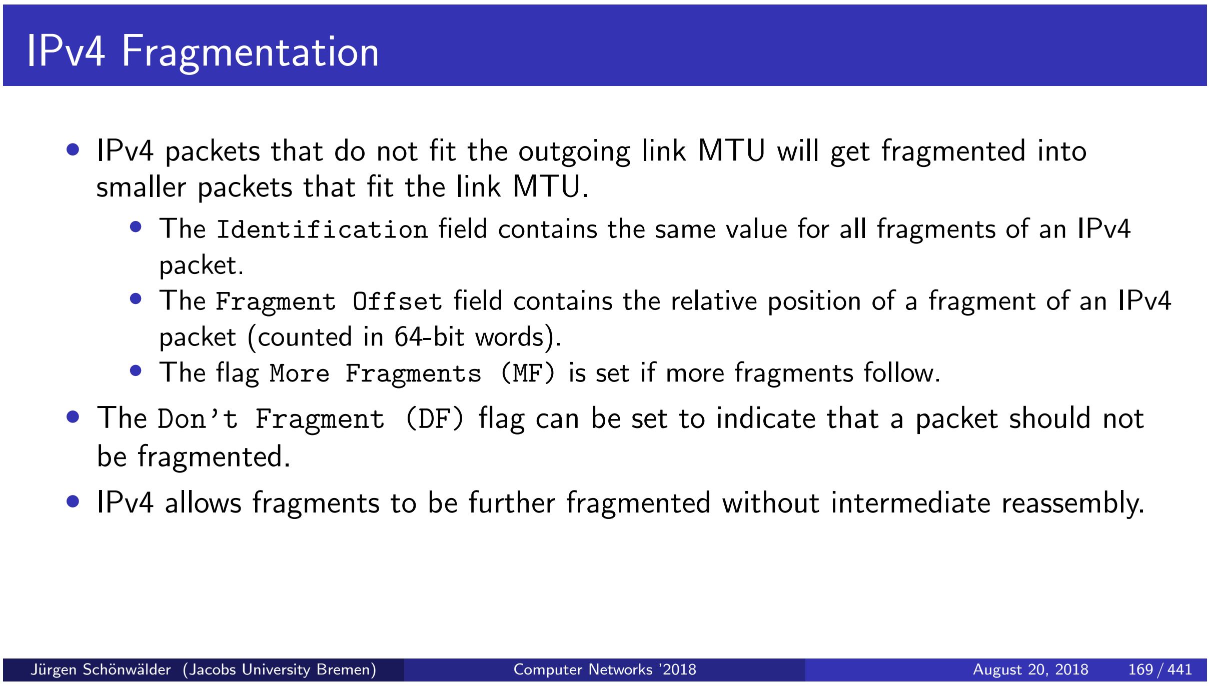
```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     Type      |     Code      |          Checksum             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                 Router Internet Address                       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|      Internet Header + 64 bits of Original Data Datagram      |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

- The `Type` field has the value 5 for redirect messages.
- The `Code` field indicates which type of packets should be redirected.
- The `Router Internet Address` field identifies the IP router to which packets should be redirected.
- The data field contains the beginning of the packet which caused the ICMP redirect message.

Redirect codes:

Code	Description
0	Redirect datagrams for the Network
1	Redirect datagrams for the Host
2	Redirect datagrams for the Type of Service and Network
3	Redirect datagrams for the Type of Service and Host

Note that redirect messages may be somewhat “dangerous” if the sender of the message cannot be authenticated. A malicious node may simply inject fake redirect messages in order to redirect traffic to a certain node.



IPv4 Fragmentation

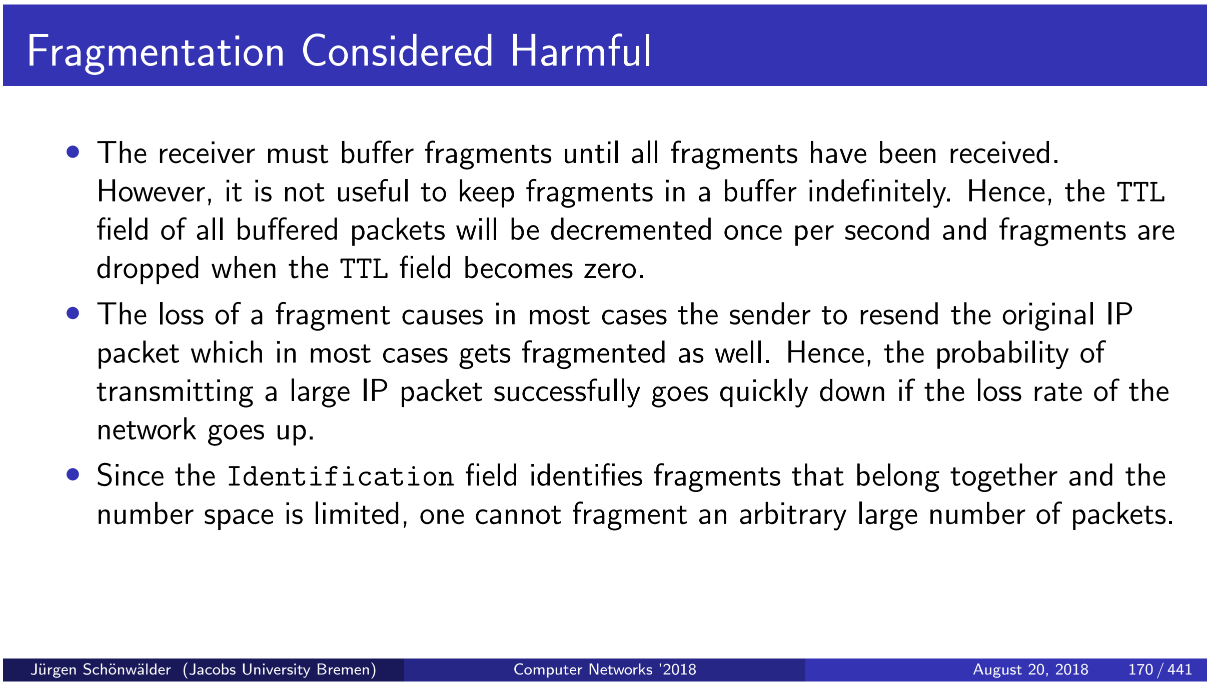
- IPv4 packets that do not fit the outgoing link MTU will get fragmented into smaller packets that fit the link MTU.
  - The `Identification` field contains the same value for all fragments of an IPv4 packet.
  - The `Fragment Offset` field contains the relative position of a fragment of an IPv4 packet (counted in 64-bit words).
  - The flag `More Fragments (MF)` is set if more fragments follow.
- The `Don’t Fragment (DF)` flag can be set to indicate that a packet should not be fragmented.
- IPv4 allows fragments to be further fragmented without intermediate reassembly.

IPv4 allows routers on the path to fragment IPv4 packets. Reassembly of the fragments is done by the final receiver. This is inline with the general design idea that functionality that can be provided at the end points should be provided at the end points. Performing reassembly by the final receiver means that there is not fragmentation / reassembly state kept anywhere in the network.

Since IPv4 allows fragmentation on the path, it can happen that fragments get further fragmented. Here is a possible scenario:

```
A -------------- R1 -------------- R2 ------------- R3 -------------- B
     MTU 1500         MTU 1200          MTU 800          MTU 1500
```

If node A sends a 1500 octet IPv4 packet to B, it will be fragmented by R1 into a 1200 octet IPv4 packet and an 320 octet IPv4 fragment packet (assuming a standard IPv4 header with a size of 20 octets). The 1200 octet IPv4 packet will be further fragmented by R2 into a 800 octet IPv4 packet and a new 420 octet fragment. R3 will simply forward the received packets and fragments. Hence, node B will receive three packets (all with the same identification value): 800 octets (fragment-offset=0, MF=1), 420 octets (fragment-offset=800, MF = 1), 320 octets (fragment-offset=1200, MF = 0). Note that these packets may arrive in a different order and it is of course possible that some packets never arrive (hence an attempt to reassemble a packet must eventually time out).

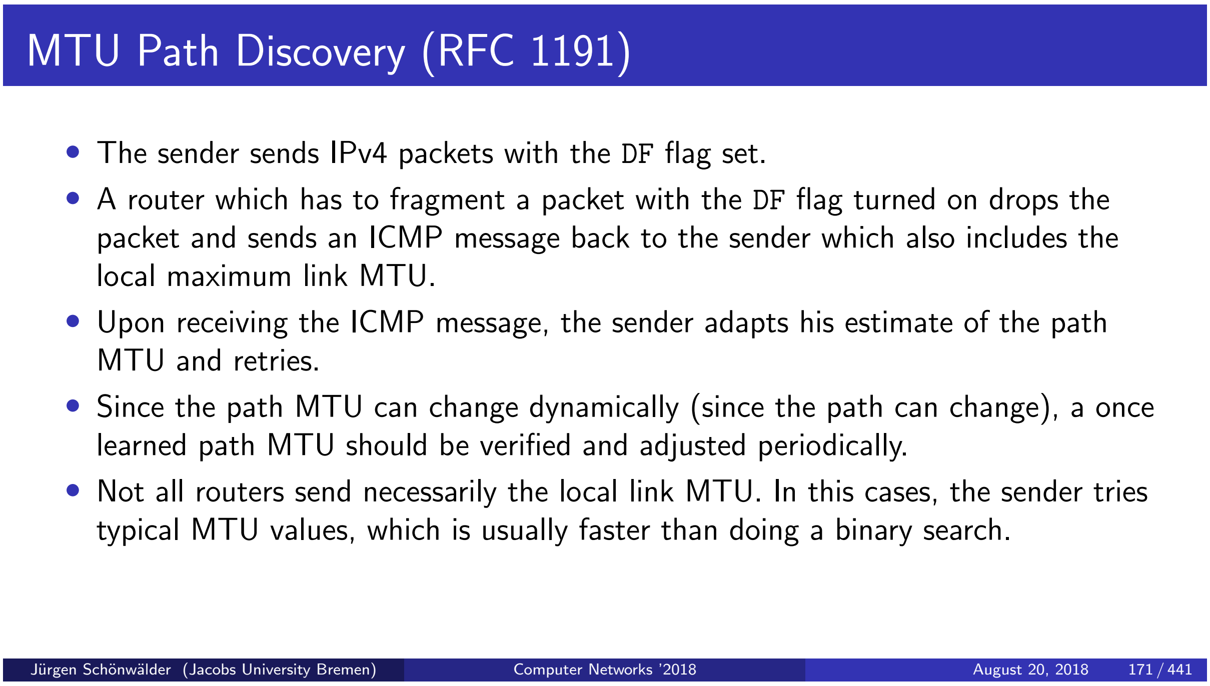


Fragmentation Considered Harmful

- The receiver must buffer fragments until all fragments have been received. However, it is not useful to keep fragments in a buffer indefinitely. Hence, the `TTL` field of all buffered packets will be decremented once per second and fragments are dropped when the `TTL` field becomes zero.
- The loss of a fragment causes in most cases the sender to resend the original IP packet which in most cases gets fragmented as well. Hence, the probability of transmitting a large IP packet successfully goes quickly down if the loss rate of the network goes up.
- Since the `Identification` field identifies fragments that belong together and the number space is limited, one cannot fragment an arbitrary large number of packets.

In a classic paper [18], Christopher A. Kent and Jeffrey C. Mogul argued that fragmentation in the network can lead to poor performance since (i) it causes inefficient use of resources, (ii) lost fragments lead to degenerated performance, and (iii) efficient reassembly is difficult. This paper had a big impact and mechanisms were invented to discover the path MTU in order to avoid fragmentation within the network.

During the design of IPv6, a decision was made that links have to support a minimum MTU of 1280 octets and that IPv6 routers never fragment packets. The later, however, has serious consequences if ICMPv6 error messages are blocked or lost since routers that drop packets that are too big essentially become blackholes, causing higher layers to timeout.

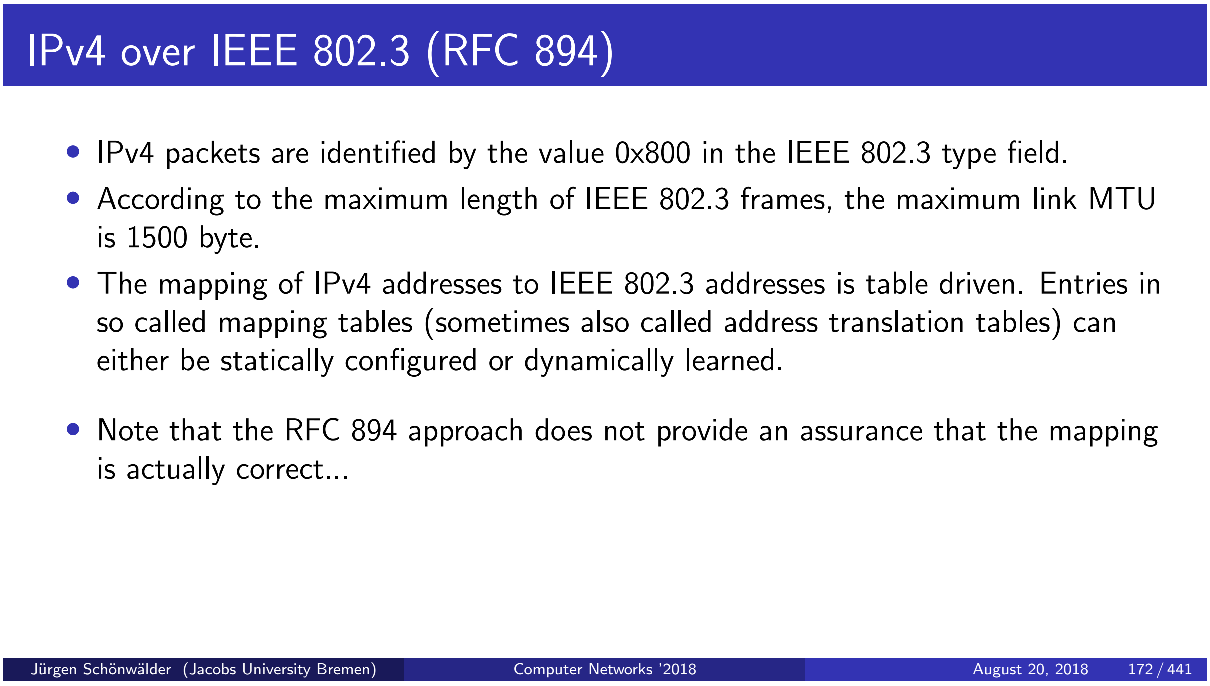


MTU Path Discovery (RFC 1191)

- The sender sends IPv4 packets with the `DF` flag set.
- A router which has to fragment a packet with the `DF` flag turned on drops the packet and sends an ICMP message back to the sender which also includes the local maximum link MTU.
- Upon receiving the ICMP message, the sender adapts his estimate of the path MTU and retries.
- Since the path MTU can change dynamically (since the path can change), a once learned path MTU should be verified and adjusted periodically.
- Not all routers send necessarily the local link MTU. In this cases, the sender tries typical MTU values, which is usually faster than doing a binary search.

In IPv4 networks, path MTU discovery [25] is a mechanism that should be used to avoid fragmentation on the path. However, if path MTU discovery fails, IPv4 routers will still take care of things by fragmenting packets.

In IPv6 networks, path MTU discovery has to be used if the sender uses a link with an MTU larger than the minimum IPv6 link MTU of 1280 octets. If path MTU discovery fails in IPv6 networks, then large packets will simply disappear from the network, typically causing higher layers in the protocol stack to experience timeouts.



IPv4 over IEEE 802.3 (RFC 894)

- IPv4 packets are identified by the value 0x800 in the IEEE 802.3 type field.
- According to the maximum length of IEEE 802.3 frames, the maximum link MTU is 1500 byte.
- The mapping of IPv4 addresses to IEEE 802.3 addresses is table driven. Entries in so called mapping tables (sometimes also called address translation tables) can either be statically configured or dynamically learned.
- Note that the RFC 894 approach does not provide an assurance that the mapping is actually correct...

RFC 894 [15] defines how IPv4 packets are commonly encapsulated in Ethernet frames.

IPv4 Address Translation (RFC 826)

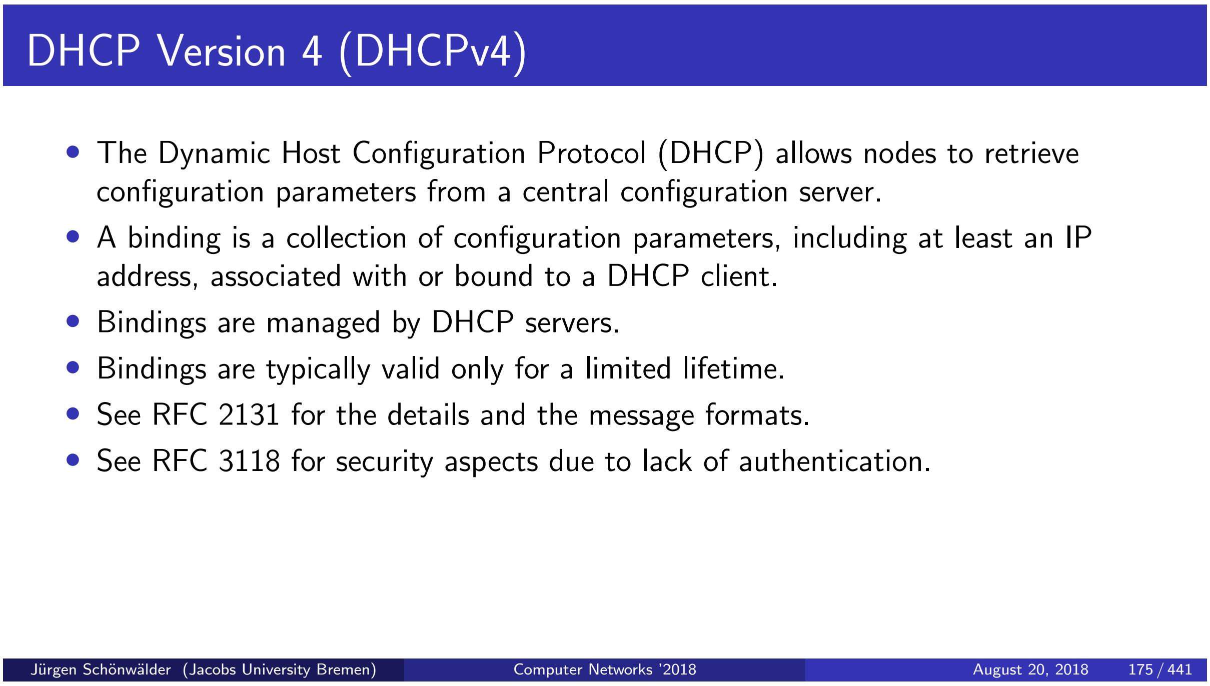
```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|         Hardware Type         |        Protocol Type          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|       HLEN    |      PLEN     |          Operation            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                 Sender Hardware Address (SHA)                 =
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
= Sender Hardware Address (SHA) |     Sender IP Address (SIP)   =
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
=    Sender IP Address (SIP)    | Target Hardware Address (THA) =
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
=                 Target Hardware Address (THA)                 |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                       Target IP Address                       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

- The Address Resolution Protocol (ARP) resolved IPv4 addresses to link-layer addresses of neighboring nodes.

RFC 826 [26] defines how IPv4 addresses can be resolved to so called “hardware” addresses. RFC 903 [11] defines how a reverse resolution of a “hardware” address to an IPv4 address can be performed. In the common case, a “hardware address” is an Ethernet MAC address.

ARP and RARP

- The `Hardware Type` field identifies the address type used on the link-layer (the value 1 is used for IEEE 802.3 MAC addresses).
- The `Protocol Type` field identifies the network layer address type (the value 0x800 is used for IPv4).
- ARP/RARP packets use the 802.3 type value 0x806.
- The `Operation` field contains the message type: ARP Request (1), ARP Response (2), RARP Request (3), RARP Response (4).
- The sender fills, depending on the request type, either the `Target IP Address` field (ARP) or the `Target Hardware Address` field (RARP).
- The responding node swaps the Sender/Target fields and fills the empty fields with the requested information.



DHCP Version 4 (DHCPv4)

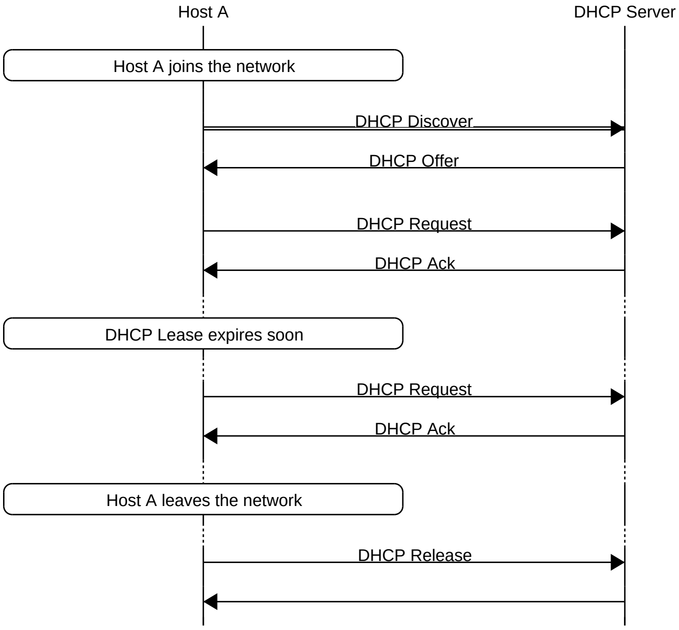
- The Dynamic Host Configuration Protocol (DHCP) allows nodes to retrieve configuration parameters from a central configuration server.
- A binding is a collection of configuration parameters, including at least an IP address, associated with or bound to a DHCP client.
- Bindings are managed by DHCP servers.
- Bindings are typically valid only for a limited lifetime.
- See RFC 2131 for the details and the message formats.
- See RFC 3118 for security aspects due to lack of authentication.

DHCPv4 is widely used to supply hosts with IPv4 network configuration parameters (e.g., an IPv4 address, a default router address, a DNS server address). RFC 2131 [9] defines the DHCPv4 protocol and message formats while RFC 3118 [10] discusses security aspects of DHCPv4 while RFC 7819 [16] discusses privacy issues of DHCPv4.

DHCPv4 Message Types

- The DHCPDISCOVER message is a broadcast message which is sent by DHCP clients to locate DHCP servers.
- The DHCPOFFER message is sent from a DHCP server to offer a client a set of configuration parameters.
- The DHCPREQUEST is sent from the client to a DHCP server as a response to a previous DHCPOFFER message, to verify a previously allocated binding or to extend the lease of a binding.
- The DHCPACK message is sent by a DHCP server with some additional parameters to the client as a positive acknowledgement to a DHCPREQUEST.
- The DHCPNAK message is sent by a DHCP server to indicate that the client's notion of a configuration binding is incorrect.

The following time sequence diagram shows a typical DHCPv4 exchange. Note that DHCP leases configuration parameters and hosts are expected to renew leases before they expire. Furthermore, it is appreciated by DHCP servers if hosts return their lease when leaving a network.



DHCPv4 Message Types (cont.)

- The `DHCPDECLINE` message is sent by a DHCP client to indicate that parameters are already in use.
- The `DHCPRELEASE` message is sent by a DHCP client to inform the DHCP server that configuration parameters are no longer used.
- The `DHCPINFORM` message is sent from the DHCP client to inform the DHCP server that only local configuration parameters are needed.

Part V

Internet Routing

IP forwarding tables determine how traffic is moved through the Internet. For the global Internet to function, it is important that IP forwarding tables are setup “correctly” so that every node can reach any other node on the Internet. (We ignore for a moment situations where nodes are on purpose not globally reachable.) Of course, in a large scale network, everything is constantly changing since links and systems fail, business relationships change, or organizations simply decide to optimize traffic flows for different priorities. Hence, a global routing system should be (i) flexible, (ii) robust, and (iii) converge fast.

In the following, we will focus on routing for unicast communication. Multicast routing is a very different problem and the majority of wide-area traffic happens to be unicast traffic. We will look into the following three routing protocols since they use different techniques

- The *Routing Information Protocol* (RIP) is a simple *distance vector routing protocol*. It uses the distributed Bellman-Ford shortest path algorithm.
- The *Open Shortest Path First* (OSPF) routing protocol floods *link state* information so that every node can compute shortest paths using Dijkstra’s shortest path algorithm. It is an example of a *link state routing protocol*.
- The *Border Gateway Protocol* (BGP) routing protocol propagates reachability information between Autonomous Systems. This information is used to make policy-based routing decisions.

The first two protocols (RIP and OSPF) were designed to distribute routing information within an Autonomous Systems. The BGP protocol is the standard protocol to exchange routing information between Autonomous Systems.

We will not cover the usage of BGP as an internal routing protocol of an Autonomous Systems and we will not cover other alternatives such as the *Intermediate System to Intermediate System* (IS-IS) routing protocol, another link state routing protocol adopted from the ISO/OSI protocol standards. We will also not cover routing protocols for special kinds of networks, e.g., the Routing Protocol for Low-Power and Lossy Networks (RPL), which is essentially establishing routing trees.

Section 21: Distance Vector Routing (RIP)

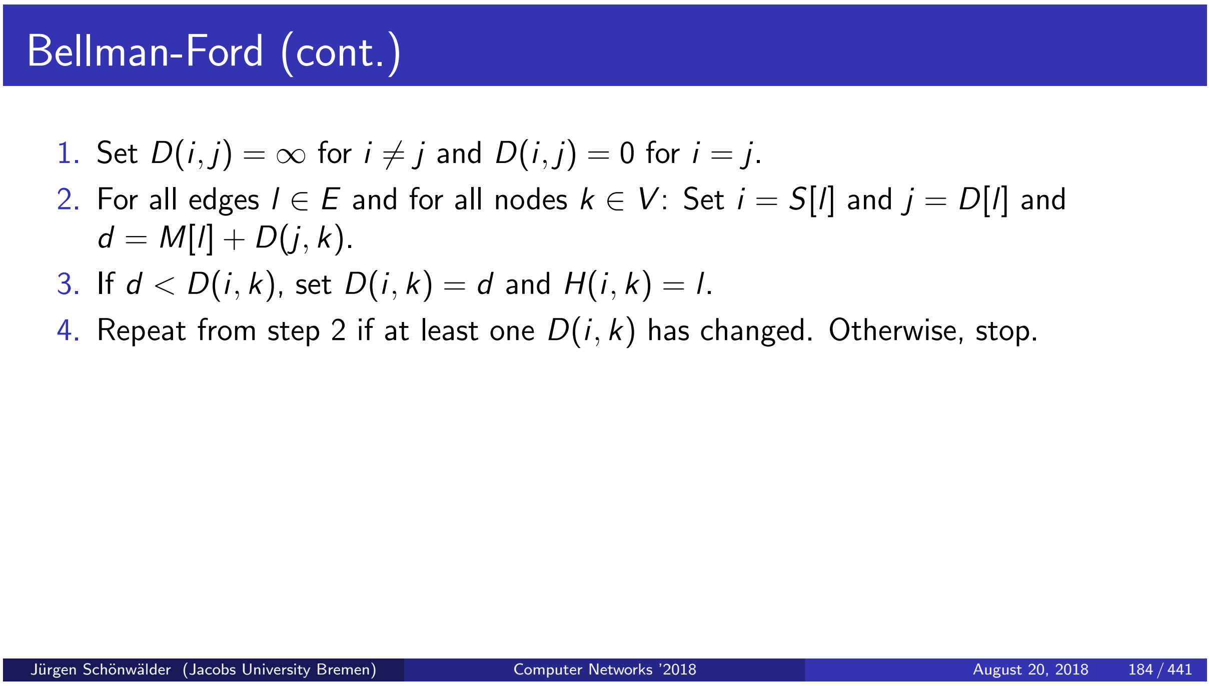
21 Distance Vector Routing (RIP)

22 Link State Routing (OSPF)

23 Path Vector Policy Routing (BGP)

Bellman-Ford

- Let $G = (V, E)$ be a graph with the vertices V and the edges E with $n = |V|$ and $m = |E|$.
- Let D be an $n \times n$ distance matrix in which $D(i, j)$ denotes the distance from node $i \in V$ to the node $j \in V$.
- Let H be an $n \times n$ matrix in which $H(i, j) \in E$ denotes the edge on which node $i \in V$ forwards a message to node $j \in V$.
- Let M be a vector with the link metrics, S a vector with the start node of the links and D a vector with the end nodes of the links.



Bellman-Ford (cont.)

1. Set $D(i,j) = \infty$ for $i \neq j$ and $D(i,j) = 0$ for $i = j$.
2. For all edges $l \in E$ and for all nodes $k \in V$: Set $i = S[l]$ and $j = D[l]$ and $d = M[l] + D(j,k)$.
3. If $d < D(i,k)$, set $D(i,k) = d$ and $H(i,k) = l$.
4. Repeat from step 2 if at least one $D(i,k)$ has changed. Otherwise, stop.

Section 22: Link State Routing (OSPF)

21 Distance Vector Routing (RIP)

22 Link State Routing (OSPF)

23 Path Vector Policy Routing (BGP)

Section 23: Path Vector Policy Routing (BGP)

21 Distance Vector Routing (RIP)

22 Link State Routing (OSPF)

23 Path Vector Policy Routing (BGP)

Part VI

Internet Transport Layer (UDP, TCP)

The transport layer provides communication services for applications running on hosts. The transport layer is responsible for delivering data to the appropriate application process on the host computers and hence it has a multiplexing and demultiplexing function. The transport layer often is also in charge to segment data received from applications into suitable packets. The services provided by the transport layer to applications can differ since applications may have very different requirements. A file transfer application likely prefers a transport that guarantees that data is delivered in sequence and without any gaps. On the other hand, an online gaming application may prefer timeliness of data over completeness and may tolerate data arriving out of order.

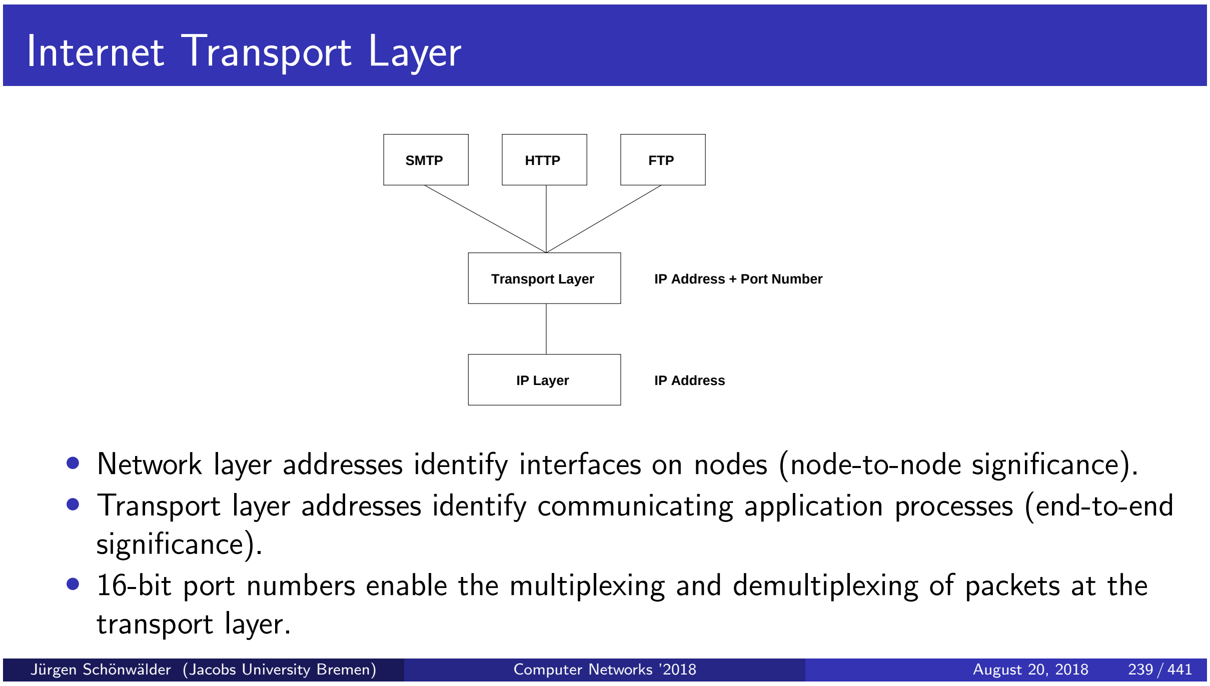
Traditionally, transport layer protocols have been implemented in operating system kernels with applications protocols residing in user space. Hence, the services of the transport layer tend to match the communication specific interface abstraction between user space and kernel space (such as the socket API).

Section 24: Transport Layer Overview

24 Transport Layer Overview

25 User Datagram Protocol (UDP)

26 Transmission Control Protocol (TCP)

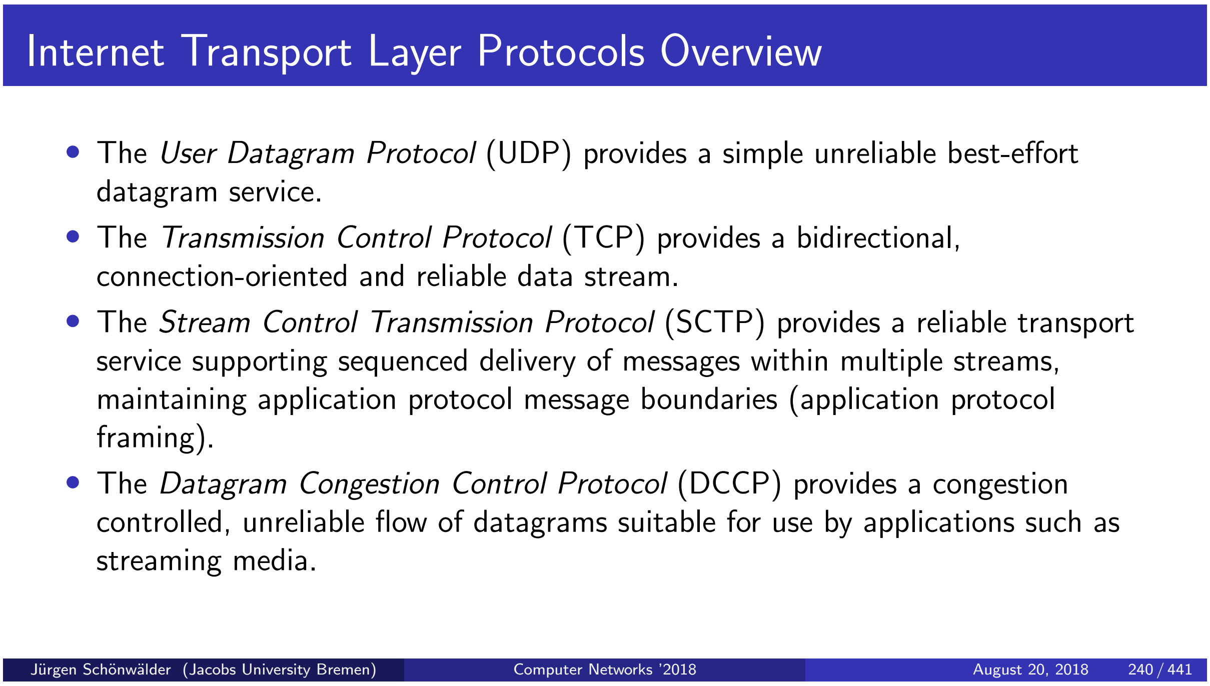


Internet Transport Layer

- Network layer addresses identify interfaces on nodes (node-to-node significance).
- Transport layer addresses identify communicating application processes (end-to-end significance).
- 16-bit port numbers enable the multiplexing and demultiplexing of packets at the transport layer.

In a properly layered architecture, protocol layers would have independent addresses and there would be a flexible mapping function mapping the addresses used by the different layers. The Internet design is not a properly layered architecture since network layer addresses are simply extended to provide transport layer addresses. While this design makes simplifies the mapping function between the layers, it has a significant drawback: Changing the network layer endpoint causes all transport layer endpoints to become invalid.

This is a significant drawback for mobile systems that can easily move between networks. When a node moves from one network to another, the node has to obtain a new IP address, which is topologically consistent with the new network (i.e., uses the prefix of the new network). This means the old address, that was topologically consistent with the old network (i.e., used the prefix of the old network), becomes unusable. A lot of work has been spent to address this problem, for example by dynamically establishing tunnels that allow a system to continue to use the old now invalid address or by creating a new slim layer that provides a proper separation of transport layer address from network layer addresses. So far, none of the solutions gained wide deployment. Instead, applications learned how to deal gracefully with errors and interruptions caused by changes of a nodes network attachment.



Internet Transport Layer Protocols Overview

- The *User Datagram Protocol* (UDP) provides a simple unreliable best-effort datagram service.
- The *Transmission Control Protocol* (TCP) provides a bidirectional, connection-oriented and reliable data stream.
- The *Stream Control Transmission Protocol* (SCTP) provides a reliable transport service supporting sequenced delivery of messages within multiple streams, maintaining application protocol message boundaries (application protocol framing).
- The *Datagram Congestion Control Protocol* (DCCP) provides a congestion controlled, unreliable flow of datagrams suitable for use by applications such as streaming media.

Jürgen Schönwälder (Jacobs University Bremen) Computer Networks '2018 August 20, 2018 240 / 441

The UDP protocol is defined in RFC 768 [27] and the TCP protocol is defined in RFC 793 [29]. Both documents appeared in the early 1980s. While the core of the specifications are still used today, there have been a number of updates and extensions on certain aspects, in particular related to congestion control.

The SCTP protocol first appeared as RFC 2960 [34] in 2000 and a major revision was published as RFC 4960 [33] in 2007. SCTP provides multiple independent streams within an association and it can provide different types of services.

The DCCP protocol first appeared in RFC 4340 [20]. There is a nice paper describing the design decisions that led to DCCP [21].

Unfortunately, both SCTP and DCCP have failed so far to gain wide-spread adoption. One key factor is a chicked-or-egg problem: Unless SCTP and DCCP are available everywhere (means all relevant operating system kernels but also middleboxes), there won't be applications using it. Without applications using SCTP and DCCP, they will not be implemented everywhere.

Google started a new effort to create new transport protocol called QUIC [3], which aims to minimize latency while at the same time mandating security. In order to avoid the chicked-or-egg problem, they designed QUIC (a transport protocol) over UDP (another transport protocol). This somewhat surprising design (from an architectural point of view) has the benefit that a QUIC implementation can be conveniently be shipped with applications. This of courses raises questions to what extend networking protocols should be implemented in kernel space at all and how we deal with applications that all come with their own networking stacks.

From an academic perspective, it is highly relevant to study the different protocol designs. From a practical perspective, most traffic today is running over TCP. This may be changed by QUIC eventually but as of today, it is unlikely that SCTP or DCCP will play a significant role at the transport layer. But it is important that this is not due to significant technical deficiencies but to the economic system around the Internet [4, 14].

IPv4 Pseudo Header

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                       Source Address                          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                    Destination Address                        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  unused (0)   |    Protocol   |            Length             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

- Pseudo headers are used during checksum computation.
- A pseudo header excludes header fields that are modified by routers.

IPv6 Pseudo Header

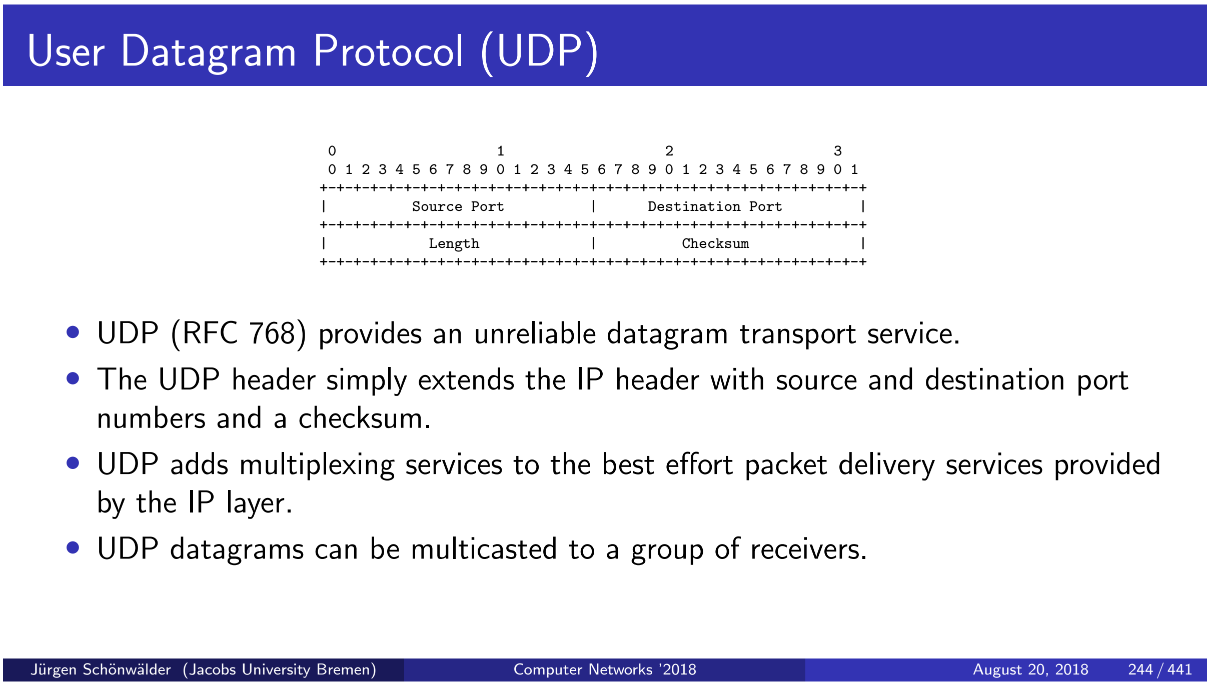
```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
+                                                               +
|                                                               |
+                         Source Address                        +
|                                                               |
+                                                               +
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
+                                                               +
|                                                               |
+                      Destination Address                      +
|                                                               |
+                                                               +
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                   Upper-Layer Packet Length                   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                      zero                     |  Next Header  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Section 25: User Datagram Protocol (UDP)

24 Transport Layer Overview

25 User Datagram Protocol (UDP)

26 Transmission Control Protocol (TCP)



User Datagram Protocol (UDP)

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|          Source Port          |       Destination Port        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|            Length             |           Checksum            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

- UDP (RFC 768) provides an unreliable datagram transport service.
- The UDP header simply extends the IP header with source and destination port numbers and a checksum.
- UDP adds multiplexing services to the best effort packet delivery services provided by the IP layer.
- UDP datagrams can be multicasted to a group of receivers.

The User Datagram Protocol (UDP) is defined in RFC 768 [27].

User Datagram Protocol (UDP)

- The `Source Port` field contains the port number used by the sending application layer process.
- The `Destination Port` field contains the port number used by the receiving application layer process.
- The `Length` field contains the length of the UDP datagram including the UDP header counted in bytes.
- The `Checksum` field contains the Internet checksum computed over the pseudo header, the UDP header and the payload contained in the UDP packet.

Section 26: Transmission Control Protocol (TCP)

24 Transport Layer Overview

25 User Datagram Protocol (UDP)

26 Transmission Control Protocol (TCP)

Transmission Control Protocol (TCP)

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|          Source Port          |       Destination Port        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                        Sequence Number                        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                    Acknowledgment Number                      |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| Offset| Reserved  |   Flags   |            Window             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|           Checksum            |         Urgent Pointer        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                    Options                    |    Padding    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

- TCP (RFC 793) provides a bidirectional connection-oriented and reliable data stream over an unreliable connection-less network protocol.

The Transmission Control Protocol (TCP) is defined in RFC 793 [29].

Transmission Control Protocol (TCP)

- The `Source Port` field contains the port number used by the sending application layer process.
- The `Destination Port` field contains the port number used by the receiving application layer process.
- The `Sequence Number` field contains the sequence number of the first data byte in the segment. During connection establishment, this field is used to establish the initial sequence number.
- The `Acknowledgment Number` field contains the next sequence number which the sender of the acknowledgement expects.
- The `Offset` field contains the length of the TCP header including any options, counted in 32-bit words.

Transmission Control Protocol (TCP)

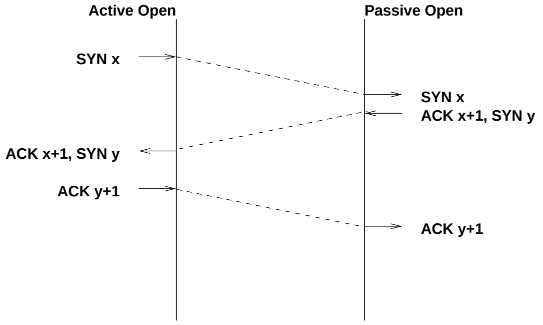
- The Flags field contains a set of binary flags:

Flag	Description
URG	Indicates that the Urgent Pointer field is significant
ACK	Indicates that the Acknowledgment Number field is significant
PSH	Data should be pushed to the application as quickly as possible
RST	Reset of the connection
SYN	Synchronization of sequence numbers
FIN	No more data from the sender

Transmission Control Protocol (TCP)

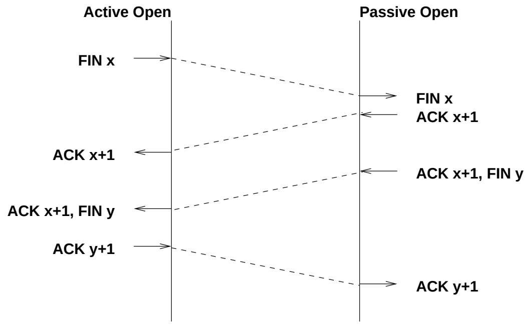
- The `Window` field indicates the number of data bytes which the sender of the segment is willing to receive.
- The `Checksum` field contains the Internet checksum computed over the pseudo header, the TCP header and the data contained in the TCP segment.
- The `Urgent Pointer` field points, relative to the actual segment number, to important data if the `URG` flag is set.
- The `Options` field can contain additional options.

TCP Connection Establishment



- Handshake protocol establishes TCP connection parameters and announces options.
- Guarantees correct connection establishment, even if TCP packets are lost or duplicated.

TCP Connection Tear-down



- TCP provides initially a bidirectional data stream.
- A TCP connection is terminated when both unidirectional connections have been closed. (It is possible to close only one half of a connection.)

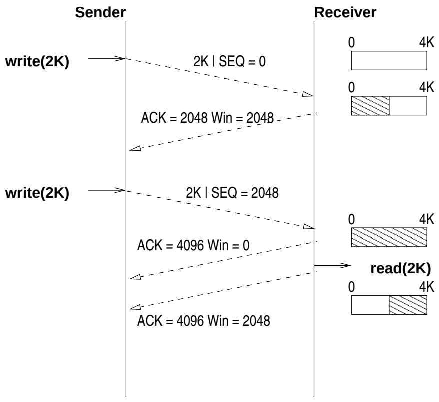
TCP State Machine (Part #1)

```
                             +---------+ ---------\      active OPEN
                             |  CLOSED |            \    -----------
                             +---------+<---------\   \   create TCB
                               |     ^              \   \  snd SYN
                  passive OPEN |     |   CLOSE        \   \
                  ------------ |     | ----------       \   \
                   create TCB  |     | delete TCB         \   \
                               V     |                      \   \
                             +---------+            CLOSE    |    \
                             |  LISTEN |          ---------- |     |
                             +---------+          delete TCB |     |
                  rcv SYN      |     |     SEND              |     |
                 -----------   |     |    -------            |     V
+---------+      snd SYN,ACK  /       \   snd SYN          +---------+
|         |<-----------------           ------------------>|         |
|   SYN   |                    rcv SYN                     |   SYN   |
|   RCVD  |<-----------------------------------------------|   SENT  |
|         |                snd SYN, ACK                    |         |
|         |------------------           -------------------|         |
+---------+   rcv ACK of SYN  \       /  rcv SYN,ACK       +---------+
  |           --------------   |     |   -----------
  |                  x         |     |     snd ACK
  |                            V     V
  |  CLOSE                   +---------+
  | -------                  |  ESTAB  |
  | snd FIN                  +---------+
  :                   CLOSE    :     :    rcv FIN
```

TCP State Machine (Part #2)

```
  :                           :     :
  |                           V     V
  |  CLOSE                  +---------+
  | -------                 |  ESTAB  |
  | snd FIN                 +---------+
  |                   CLOSE   |     |    rcv FIN
  V                  -------  |     |    -------
+---------+          snd FIN /       \   snd ACK         +---------+
|  FIN    |<----------------           ----------------->|  CLOSE  |
| WAIT-1  |-----------------                             |   WAIT  |
+---------+          rcv FIN \                           +---------+
  | rcv ACK of FIN   -------  |                           CLOSE  |
  | --------------   snd ACK  |                          ------- |
  V        x                  V                          snd FIN V
+---------+                 +---------+                  +---------+
|FINWAIT-2|                 | CLOSING |                  | LAST-ACK|
+---------+                 +---------+                  +---------+
  |               rcv ACK of FIN |                rcv ACK of FIN |
  |  rcv FIN      -------------- |   Timeout=2MSL -------------- |
  |  -------             x       V   ------------        x       V
   \ snd ACK                +---------+delete TCB        +---------+
    ----------------------->|TIME WAIT|----------------->| CLOSED  |
                            +---------+                  +---------+
```

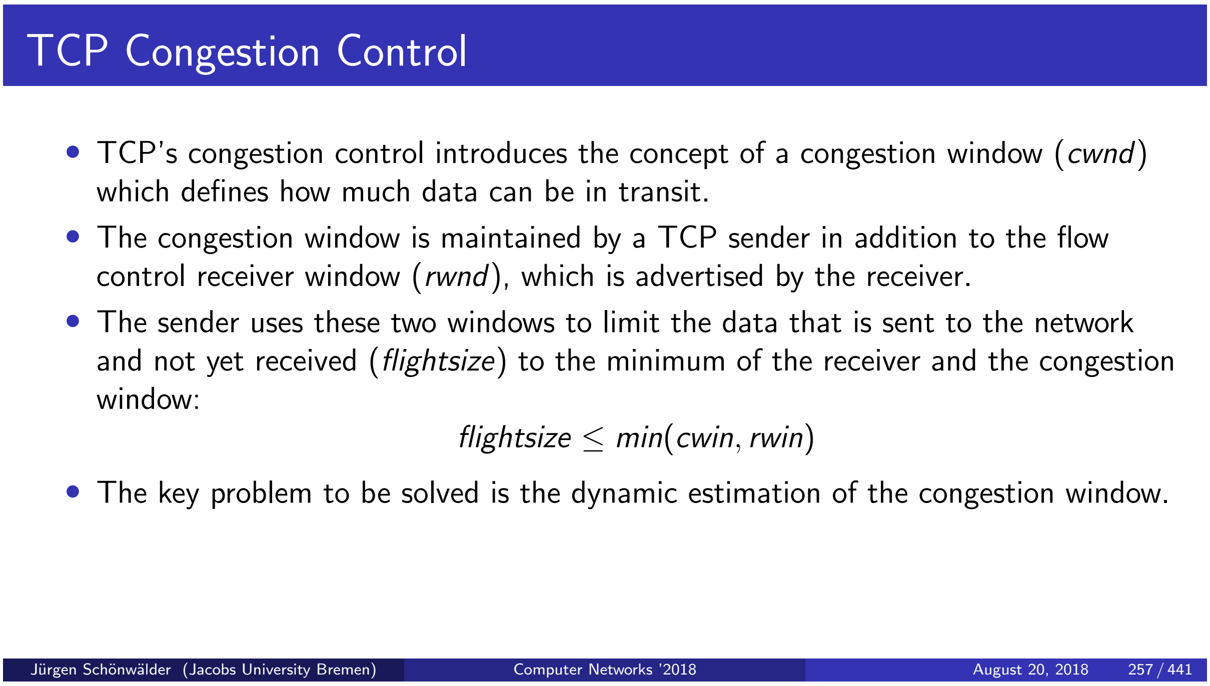
TCP Flow Control



- Both TCP engines advertise their buffer sizes during connection establishment.
- The available space left in the receiving buffer is advertised as part of the acknowledgements.

TCP Flow Control Optimizations

- Nagle's Algorithm
  - When data comes into the sender one byte at a time, just send the first byte and buffer all the rest until the byte in flight has been acknowledgement.
  - This algorithm provides noticeable improvements especially for interactive traffic where a quickly typing user is connected over a rather slow network.
- Clark's Algorithm
  - The receiver should not send a window update until it can handle the maximum segment size it advertised when the connection was established or until its buffer is half empty.
  - Prevents the receiver from sending a very small window updates (such as a single byte).



TCP Congestion Control

- TCP's congestion control introduces the concept of a congestion window (*cwnd*) which defines how much data can be in transit.
- The congestion window is maintained by a TCP sender in addition to the flow control receiver window (*rwnd*), which is advertised by the receiver.
- The sender uses these two windows to limit the data that is sent to the network and not yet received (*flightsize*) to the minimum of the receiver and the congestion window:

$$flightsize \leq min(cwin, rwin)$$

- The key problem to be solved is the dynamic estimation of the congestion window.

Congestion control is of fundamental importance to avoid a collapse of the Internet. Many RFCs have been written on topics related to congestion control. RFC 5783 [**?**] provides an overview (as of Spring 2010). RFC 5681 [1] discusses the details of TCP congestion control.

TCP Congestion Control (cont.)

- The initial window (IW) is usually initialized using the following formula:

  $$IW = min(4 \cdot SMSS, max(2 \cdot SMSS, 4380 bytes))$$

  SMSS is the sender maximum segment size, the size of the largest sement that the sender can transmit (excluding TCP/IP headers and options).
- During slow start, the congestion window *cwnd* increases by at most *SMSS* bytes for every received acknowledgement that acknowledges data. Slow start ends when *cwnd* exceeds *ssthresh* or when congestion is observed.
- Note that this algorithm leads to an exponential increase if there are multiple segments acknowledged in the *cwnd*.

TCP Congestion Control (cont.)

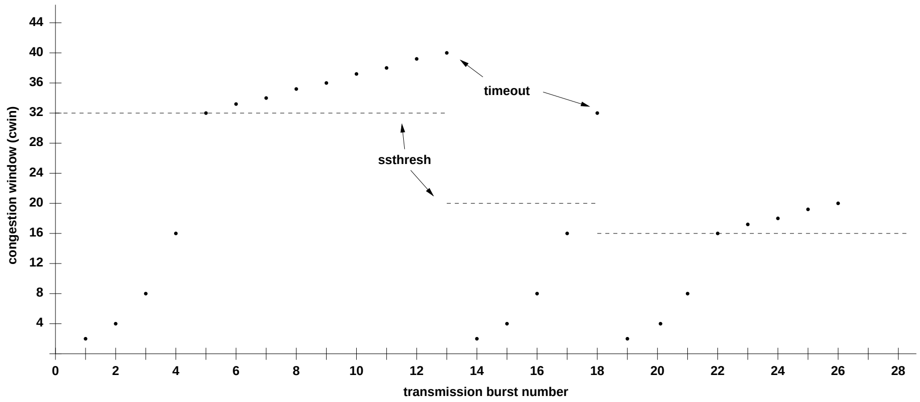
- During congestion avoidance, *cwnd* is incremented by one full-sized segment per round-trip time (RTT). Congestion avoidance continues until congestion is detected. One formula commonly used to update *cwnd* during congestion avoidance is given by the following equation:

  $$cwnd = cwnd + (SMSS * SMSS/cwnd)$$

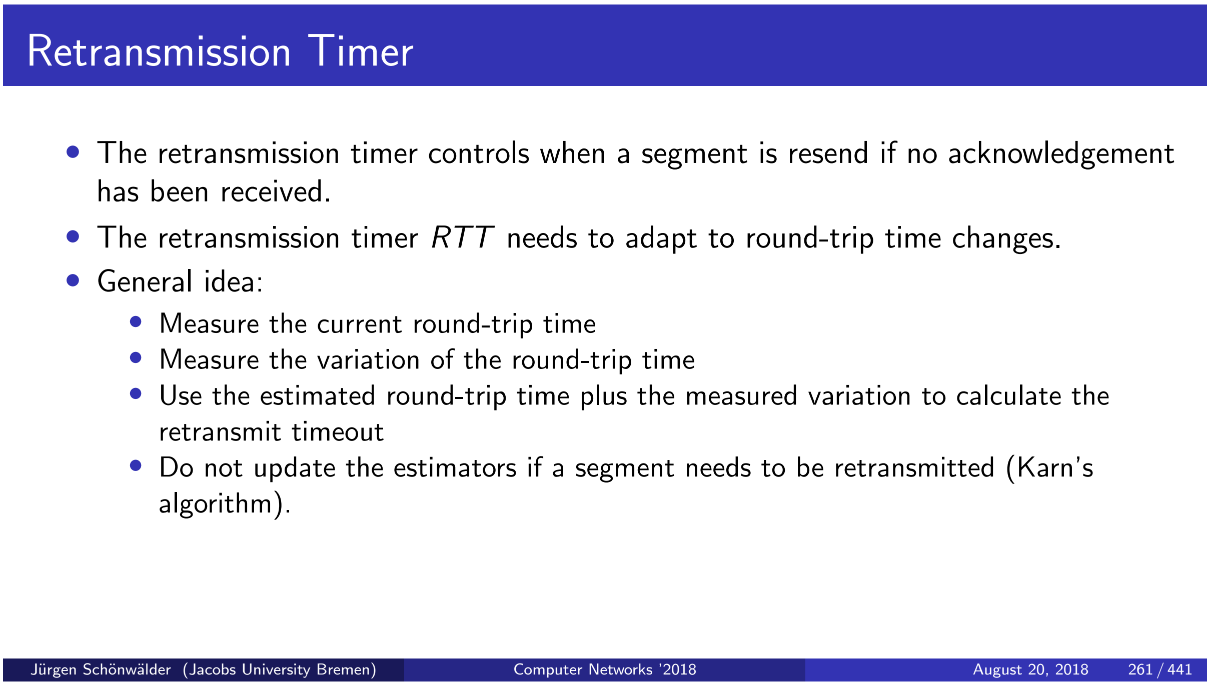
  This adjustment is executed on every incoming non-duplicate ACK.
- When congestion is noticed (the retransmission timer expires), then *cwnd*v is reset to one full-sized segment and the slow start threshold *ssthresh* is updated as follows:

  $$ssthresh = max(flightsize/2, 2 \cdot SMSS)$$

TCP Congestion Control (cont.)



- Congestion control with an initial window size of 2K.



Retransmission Timer

- The retransmission timer controls when a segment is resend if no acknowledgement has been received.
- The retransmission timer *RTT* needs to adapt to round-trip time changes.
- General idea:
    - Measure the current round-trip time
    - Measure the variation of the round-trip time
    - Use the estimated round-trip time plus the measured variation to calculate the retransmit timeout
    - Do not update the estimators if a segment needs to be retransmitted (Karn's algorithm).

Retransmission Timer

- If an acknowledgement is received for a segment before the associated retransmission timer expires:

  $$RTT = \alpha \cdot RTT + (1 - \alpha)M$$

  M is the measured round-trip time; α is typicall $\frac{7}{8}$.
- The standard deviation is estimated using:

  $$D = \alpha \cdot D + (1 - \alpha)|RTT - M|$$

  α is a smoothing factor.
- The retransmission timeout RTO is determined as follows:

  $$RTO = RTT + 4 \cdot D$$

  The factor 4 has been choosen empirically.

Fast Retransmit / Fast Recovery

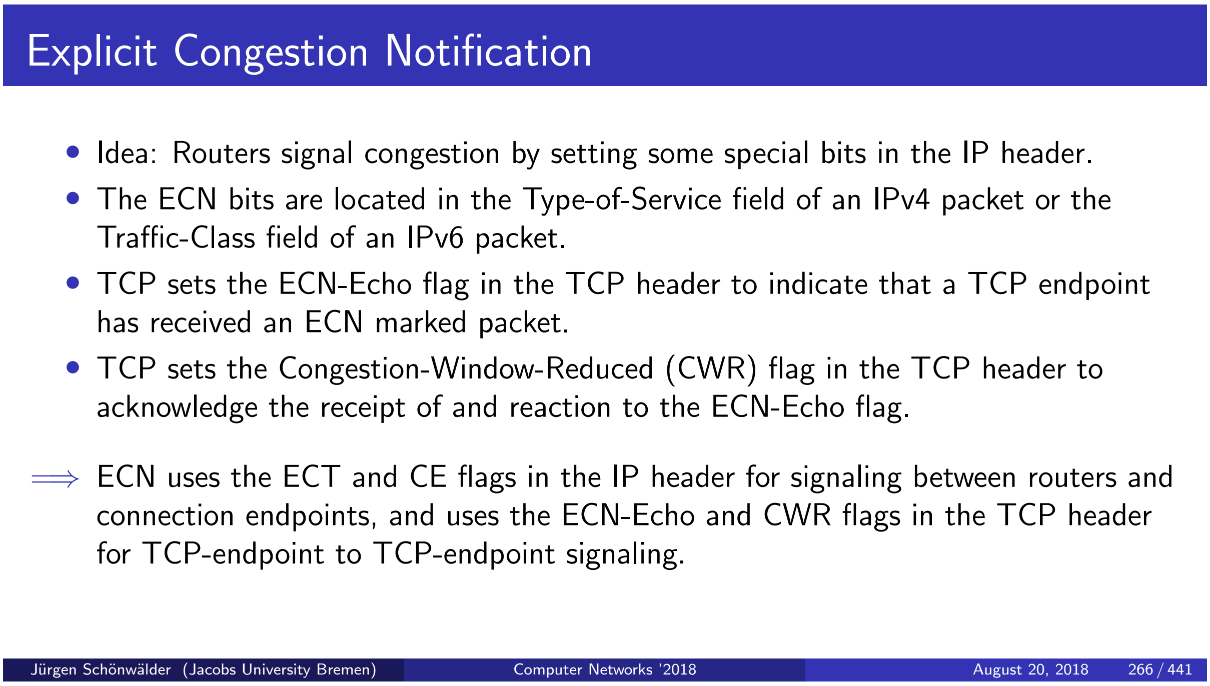
- TCP receivers should send an immediate duplicate acknowledgement when an out-of-order segment arrives.
- The arrival of four identical acknowledgements without the arrival of any other intervening packets is an indication that a segment has been lost.
- The sender performs a fast retransmission of what appears to be the missing segment, without waiting for the retransmission timer to expire.
- Upon a fast retransmission, the sender does not exercise the normal congestion reaction with a full slow start since acknowledgements are still flowing.
- See RFC 2581 section 3.1 for details.

Karn's Algorithm

- The dynamic estimation of the *RTT* has a problem if a timeout occurs and the segment is retransmitted.
- A subsequent acknowledgement might acknowledge the receipt of the first packet which contained that segment or any of the retransmissions.
- Karn suggested that the *RTT* estimation is not updated for any segments which were retransmitted and that the *RTO* is doubled on each failure until the segment gets through.
- The doubling of the *RTO* leads to an exponential back-off for each consecutive attempt.

Selected TCP Options

- Maximum Segment Size (MSS):
  - Communicates the maximum receive segment size of the sender of this option during connection establishment
- Window Scale (WS):
  - The number carried in the 16-bit `Window` field of a TCP header is scaled (shifted) by a certain constant to enable windows larger than 2^{16} octets
- TimeStamps (TS):
  - Timestamps exchanged in every TCP header to deal with the 32-bit sequence number space limitation (and to enhance round-trip time measurements)
- Selective Acknowledgment (SACK):
  - Indicate which blocks of the sequence number space are missing and which blocks are not (to improve cummulative acknowledgements)



Explicit Congestion Notification

- Idea: Routers signal congestion by setting some special bits in the IP header.
- The ECN bits are located in the Type-of-Service field of an IPv4 packet or the Traffic-Class field of an IPv6 packet.
- TCP sets the ECN-Echo flag in the TCP header to indicate that a TCP endpoint has received an ECN marked packet.
- TCP sets the Congestion-Window-Reduced (CWR) flag in the TCP header to acknowledge the receipt of and reaction to the ECN-Echo flag.

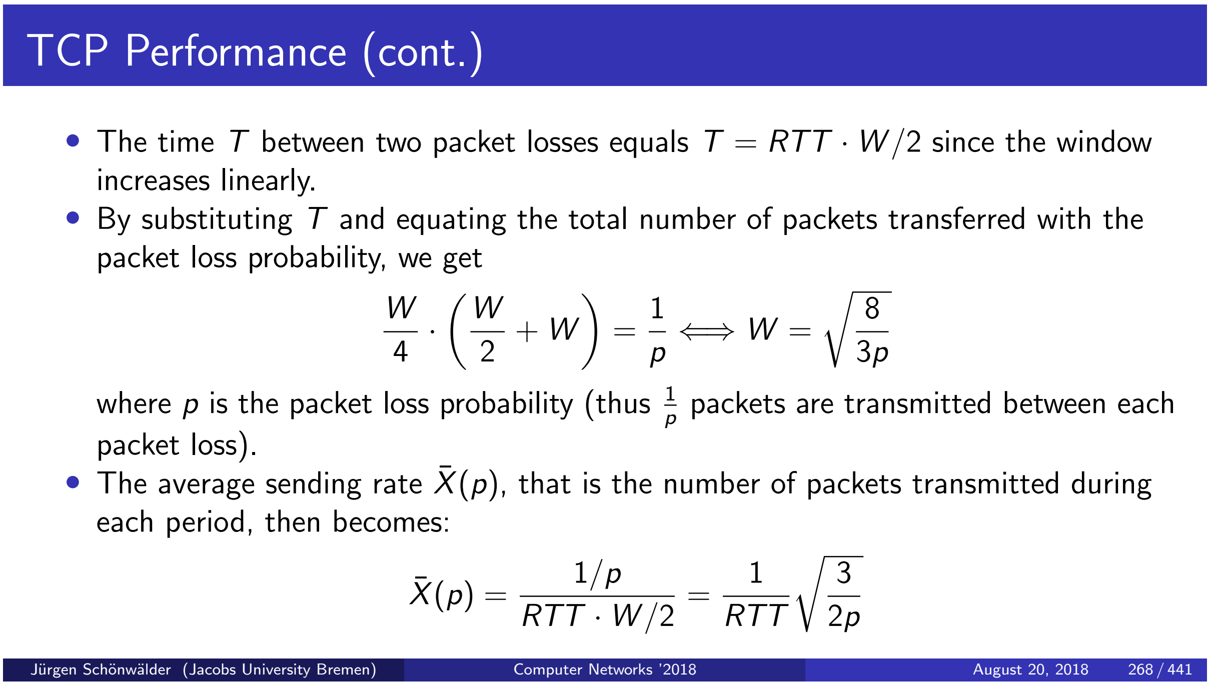
⟹ ECN uses the ECT and CE flags in the IP header for signaling between routers and connection endpoints, and uses the ECN-Echo and CWR flags in the TCP header for TCP-endpoint to TCP-endpoint signaling.

Explicit Congestion Notification (ECN) was introduced in RFC 3168 [31].

TCP Performance

- Goal: Simple analytic model for steady state TCP behavior.
- We only consider congestion avoidance (no slow start).
- $W(t)$ denotes the congestion window size at time t.
- In steady state, $W(t)$ increases to a maximum value W where it experiences congestion. As a reaction, the sender sets the congestion window to $\frac{1}{2}W$.
- The time interval needed to go from $\frac{1}{2}W$ to W is T and we can send a window size of packets every RTT.
- Hence, the number N of packets is:

$$N = \frac{1}{2}\frac{T}{RTT}\left(\frac{W}{2} + W\right)$$



TCP Performance (cont.)

- The time T between two packet losses equals $T = RTT \cdot W/2$ since the window increases linearly.
- By substituting T and equating the total number of packets transferred with the packet loss probability, we get

$$\frac{W}{4} \cdot \left(\frac{W}{2} + W\right) = \frac{1}{p} \Longleftrightarrow W = \sqrt{\frac{8}{3p}}$$

  where p is the packet loss probability (thus $\frac{1}{p}$ packets are transmitted between each packet loss).
- The average sending rate $\bar{X}(p)$, that is the number of packets transmitted during each period, then becomes:

$$\bar{X}(p) = \frac{1/p}{RTT \cdot W/2} = \frac{1}{RTT}\sqrt{\frac{3}{2p}}$$

Jürgen Schönwälder (Jacobs University Bremen) Computer Networks '2018 August 20, 2018 268 / 441

Example:

- $RTT = 200ms$, $p = 0.05$
- $\bar{X}(p) \approx 27.4pps$
- With 1500 byte segments, the data rate becomes $\approx 32Kbps$

Given this formula, it becomes clear that high data rates (say 10Gbps or 1 Tbps) require an extremely and unrealistic small packet error rate. TCP extensions to address this problem can be found in RFC 3649 [12].

Part VII

Domain Name System (DNS)

Application protocols primarily use names to refer to systems providing application specific services. A common service naming scheme are universal resource identifiers, which are build on the notion of names, or more specifically domain names. The term “domain name” indicates that a certain authority has control over the names in their “domain”.

The Domain Name System (DNS) provides a name resolution service. It was defined in the 1980s when manual maintenance of names had to be replaced by a more scalable solution.

Domain names can be though of as (components of) application layer addresses and the DNS protocol provides the mapping of these addresses to lower layers. The mapping as it was defined originally is kind of violating layering since application layer names are commonly resolved to network layer addresses and not to transport layer addresses.

Section 31: Overview and Features

31 Overview and Features

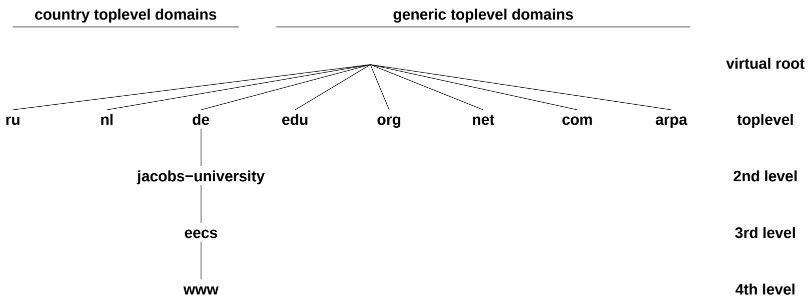
32 Resource Records

33 Message Formats

34 Security and Dynamic Updates

35 Creative Usage

Domain Name System (DNS)

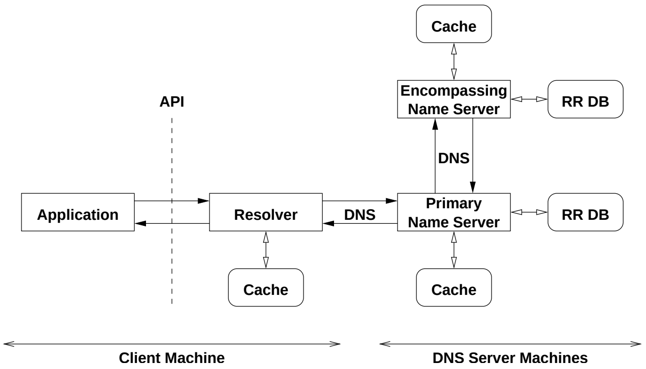


- The Domain Name System (DNS) provides a global infrastructure to map human friendly domain names into addresses (and other data).
- The DNS is a critical resource since most Internet users depend on name resolution services provided by the DNS.

The core of the domain name system is defined in RFC 1034 [23] and RFC 1035 [24]. Paul Mockapetris has received several awards for the definition of the domain name system.

While we often write `google.com`, what we really mean is `google.com.` since the trailing dot refers to the root of the DNS namespace.

Resolver and Name Resolution



- The resolver is typically tightly integrated into the operating system (or more precisely standard libraries).

The common high-level C library API for name resolution consists of the functions `getaddrinfo()` and `getnameinfo()`. To access the resolver directly, it is necessary to use resolver specific functions. The BIND resolver API is commonly available on Unix systems. There are also several alternative resolver library implementations that provide asynchronous (non-blocking) APIs. Hence, some applications may not use the system's resolver library but their own implementations.

The DNS has been designed to scale by supporting caching well. Data obtained from the DNS can be cached in several places in order to reduce the amount of DNS queries.

Most hosts on the Internet do not resolve names themself but instead the resolver is sending recursive queries to a domain name resolver (often part of the local network). The (local) domain name resolver, upon receipt of a recursive query, sends multiple queries to walk the tree from the root down to the domain name server that has an authoritative answer. Multiple users using a common domain name resolver provides caching advantages and a common domain name resolver may also hide where specific name resolution requests are originating from. On the other hand, the domain name resolver does get to see all domain name resolution requests, which can reveal quite a bit of sensitive information. (Web browsers tend to start the resolution of all embedded links as soon as a page is renders and hence pages can have certain name resolution signatures.)

Hosts typically learn the address of a domain name resolver when they configure an interface, i.e., via auto-configuration or DHCP. There are also publically known domain name resolvers:

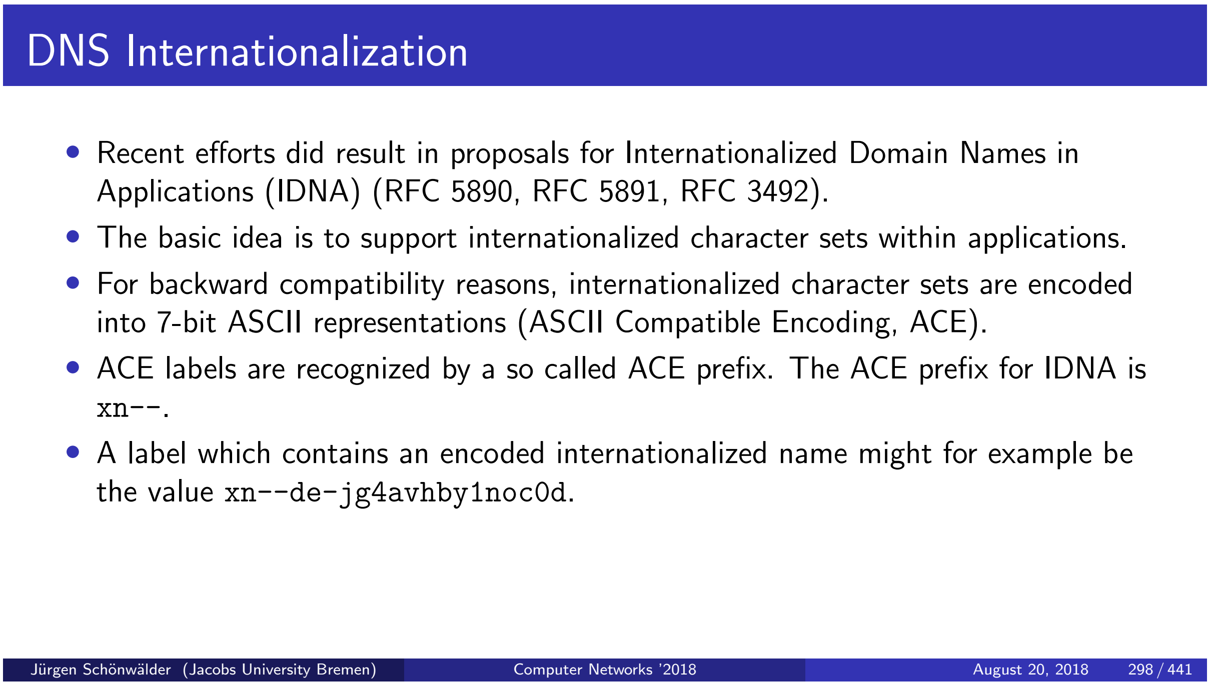
- Google is operating 8.8.8.8 and 8.8.4.4 and 2001:4860:4860::8888 and 2001:4860:4860::8844.
- Cloudflare is operating 1.1.1.1 and 1.0.0.1 and 2606:4700:4700::1111. (Cloudflare claims that they do not collect and sell data.)

DNS Characteristics

- Hierarchical name space with a virtual root.
- Administration of the name space can be delegated along the path starting from the virtual root.
- A DNS server knows a part (a zone) of the global name space and its position within the global name space.
- Name resolution queries can in principle be sent to arbitrary DNS servers. However, it is good practice to use a local DNS server as the primary DNS server.
- Recursive queries cause the queried DNS server to contact other DNS servers as needed in order to obtain a response to the query.
- The original DNS protocol does not provide sufficient security. There is usually no reason to trust DNS responses.

DNS Labels and Names

- The names (labels) on a certain level of the tree must be unique and may not exceed 63 byte in length. The character set for the labels is historically 7-bit ASCII. Comparisons are done in a case-insensitive manner.
- Labels must begin with a letter and end with a letter or decimal digit. The characters between the first and last character must be letters, digits or hyphens.
- Labels can be concatenated with dots to form paths within the name space. Absolute paths, ending at the virtual root node, end with a trailing dot. All other paths which do not end with a trailing dot are relative paths.
- The overall length of a domain name is limited to 255 bytes.



DNS Internationalization

- Recent efforts did result in proposals for Internationalized Domain Names in Applications (IDNA) (RFC 5890, RFC 5891, RFC 3492).
- The basic idea is to support internationalized character sets within applications.
- For backward compatibility reasons, internationalized character sets are encoded into 7-bit ASCII representations (ASCII Compatible Encoding, ACE).
- ACE labels are recognized by a so called ACE prefix. The ACE prefix for IDNA is `xn--`.
- A label which contains an encoded internationalized name might for example be the value `xn--de-jg4avhby1noc0d`.

Section 32: Resource Records

31 Overview and Features

32 Resource Records

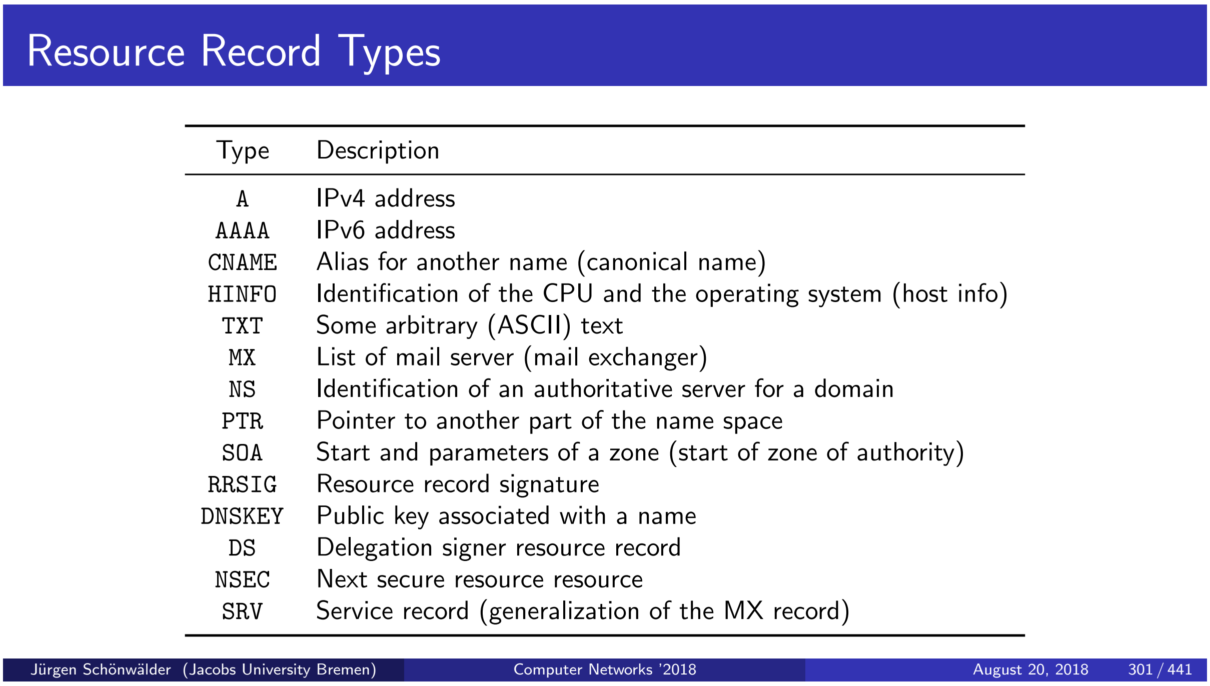
33 Message Formats

34 Security and Dynamic Updates

35 Creative Usage

Resource Records

- Resource Records (RRs) hold typed information for a given name.
- Resource records have the following components:
  - The *owner* is the domain name which identifies a resource record.
  - The *type* indicates the kind of information that is stored in a resource record.
  - The *class* indicates the protocol specific name space, normally `IN` for the Internet.
  - The *time to life* (TTL) defines how many seconds information from a resource record can be stored in a local cache.
  - The data format (RDATA) of a resource records depends on the type of the resource record.



Resource Record Types

Type	Description
A	IPv4 address
AAAA	IPv6 address
CNAME	Alias for another name (canonical name)
HINFO	Identification of the CPU and the operating system (host info)
TXT	Some arbitrary (ASCII) text
MX	List of mail server (mail exchanger)
NS	Identification of an authoritative server for a domain
PTR	Pointer to another part of the name space
SOA	Start and parameters of a zone (start of zone of authority)
RRSIG	Resource record signature
DNSKEY	Public key associated with a name
DS	Delegation signer resource record
NSEC	Next secure resource resource
SRV	Service record (generalization of the MX record)

Jürgen Schönwälder (Jacobs University Bremen) Computer Networks '2018 August 20, 2018 301 / 441

The `dig` command line utility can be used to query the DNS. The following query retrieves any (all) resource records for `instagram.com.` and it is send to the DNS server at the IPv6 address 2001:4860:4860::8888:

```
$ dig @2001:4860:4860::8888 any instagram.com.
; <<>> DiG 9.6-ESV-R4-P3 <<>> @2001:4860:4860::8888 any instagram.com
; (1 server found)
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 46252
;; flags: qr rd ra; QUERY: 1, ANSWER: 28, AUTHORITY: 0, ADDITIONAL: 0

;; QUESTION SECTION:
;instagram.com. IN ANY

;; ANSWER SECTION:
instagram.com. 59 IN A 52.204.89.224
instagram.com. 59 IN A 34.196.158.17
instagram.com. 59 IN A 54.86.54.191
instagram.com. 59 IN A 34.192.220.89
instagram.com. 59 IN A 34.233.246.5
instagram.com. 59 IN A 34.229.8.3
instagram.com. 59 IN A 54.209.115.128
instagram.com. 59 IN A 34.198.56.218
instagram.com. 21599 IN NS ns-1349.awsdns-40.org.
instagram.com. 21599 IN NS ns-2016.awsdns-60.co.uk.
instagram.com. 21599 IN NS ns-384.awsdns-48.com.
instagram.com. 21599 IN NS ns-868.awsdns-44.net.
instagram.com. 899 IN SOA ns-384.awsdns-48.com. awsdns-hostmaster.amazon.com. 3 7200 900 1209600 3600
instagram.com. 299 IN MX 10 mxa-00082601.gslb.pphosted.com.
instagram.com. 299 IN MX 10 mxb-00082601.gslb.pphosted.com.
instagram.com. 299 IN TXT "adobe-idp-site-verification=367fda82-a8bb-46cf-9cff-0062d452d229"
instagram.com. 299 IN TXT "google-site-verification=GGtId51KFyq0hqX2xNvt1u0P9Xp0C7k6pp9do49fCNw"
instagram.com. 299 IN TXT "hyWdekepiNsp/V9b1JCR+wZDdzbESurl4GqY+FLMfiN+7aeFaway0Art+kNDHeL50nGZipNeV/iIC+lOONSQVQ=="
instagram.com. 299 IN TXT "ms=ms86975275"
instagram.com. 299 IN TXT "v=spf1 include:spf.mtasv.net include:facebookmail.com include:amazonses.com ip4:199.201.64.23 ip4:199.201.65.23 in
instagram.com. 59 IN AAAA 2406:da00:ff00::3416:a694
instagram.com. 59 IN AAAA 2406:da00:ff00::3416:79b3
instagram.com. 59 IN AAAA 2406:da00:ff00::22e3:8da6
instagram.com. 59 IN AAAA 2406:da00:ff00::3414:dcb1
instagram.com. 59 IN AAAA 2406:da00:ff00::3415:5ced
instagram.com. 59 IN AAAA 2406:da00:ff00::22e1:dd34
instagram.com. 59 IN AAAA 2406:da00:ff00::3416:705d
instagram.com. 59 IN AAAA 2406:da00:ff00::22e0:b7a

;; Query time: 24 msec
;; SERVER: 2001:4860:4860::8888#53(2001:4860:4860::8888)
;; WHEN: Wed May  2 09:24:17 2018
;; MSG SIZE  rcvd: 1101
```

Section 33: Message Formats

31 Overview and Features

32 Resource Records

33 Message Formats

34 Security and Dynamic Updates

35 Creative Usage

DNS Message Formats

- A DNS message starts with a protocol header. It indicates which of the following four parts is present and whether the message is a query or a response.
- The header is followed by a list of questions.
- The list of questions is followed by a list of answers (resource records).
- The list of answers is followed by a list of pointers to authorities (also in the form of resource records).
- The list of pointers to authorities is followed by a list of additional information (also in the form of resource records). This list may contain for example `A` resource records for names in a response to an `MX` query.

DNS Message Header

Header Format

```
  0                             1
  0  1  2  3  4  5  6  7  8  9  0  1  2  3  4  5
+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+
|                      ID                       |
+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+
|QR|   Opcode  |AA|TC|RD|RA| Z|AD|CD|   RCODE   |
+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+
|                    QDCOUNT                    |
+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+
|                    ANCOUNT                    |
+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+
|                    NSCOUNT                    |
+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+
|                    ARCOUNT                    |
+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+
```

- Simple DNS queries usually use UDP as a transport.
- For larger data transfers (e.g., zone transfers), DNS may utilize TCP.

DNS Message Formats

DNS Query Format

```
  0                             1
  0  1  2  3  4  5  6  7  8  9  0  1  2  3  4  5
+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+
:                    QNAME                      :
+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+
|                    QTYPE                      |
+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+
|                    QCLASS                     |
+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+
```

DNS Response Format

```
  0                             1
  0  1  2  3  4  5  6  7  8  9  0  1  2  3  4  5
+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+
:                     NAME                      :
+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+
|                     TYPE                      |
+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+
|                    CLASS                      |
+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+
|                     TTL                       |
|                                               |
+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+
|                   RDLENGTH                    |
+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--|
:                    RDATA                      :
+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+
```

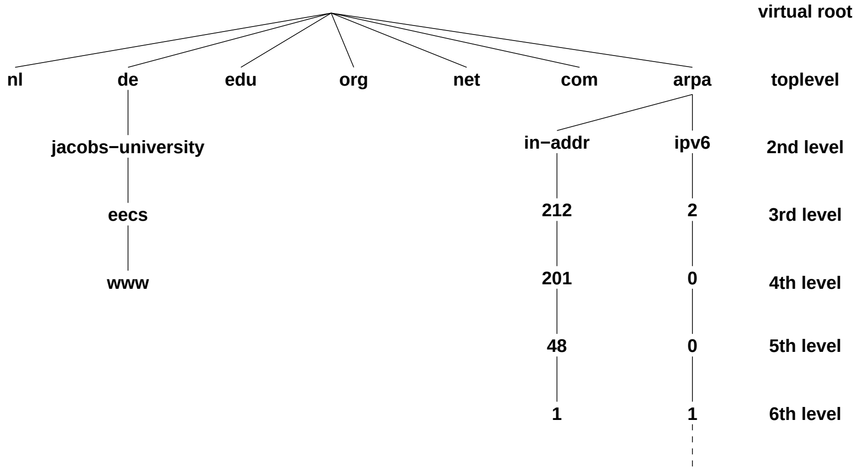
Resource Record Formats

- An A resource record contains an IPv4 address encoded in 4 bytes in network byte order.
- An AAAA resource record contains an IPv6 address encoded in 16 bytes in network byte order.
- A CNAME resource record contains a character string preceded by the length of the string encoded in the first byte.
- A HINFO resource record contains two character strings, each prefixed with a length byte. The first character string describes the CPU and the second string the operating system.
- A MX resource record contains a 16-bit preference number followed by a character string prefixed with a length bytes which contains the DNS name of a mail exchanger.

Resource Record Formats

- A `NS` resource record contains a character string prefixed by a length byte which contains the name of an authoritative DNS server.
- A `PTR` resource record contains a character string prefixed with a length byte which contains the name of another DNS server. `PTR` records are used to map IP addresses to names (so called reverse lookups). For an IPv4 address of the form $d_1.d_2.d_3.d_4$, a PTR resource record is created for the pseudo domain name $d_4.d_3.d_2.d_1.in - addr.arpa$. For an IPv6 address of the form $h_1h_2h_3h_4 : \ldots : h_{13}h_{14}h_{15}h_{16}$, a PTR resource record is created for the pseudo domain name $h_{16}.h_{15}.h_{14}.h_{13}.\ldots.h_4.h_3.h_2.h_1.ip6.arpa$

DNS Reverse Trees



Resource Record Formats

- A SOA resource record contains two character strings, each prefixed by a length byte, and five 32-bit numbers:
  - Name of the DNS server responsible for a zone.
  - Email address of the administrator responsible for the management of the zone.
  - Serial number (SERIAL) (must be incremented whenever the zone database changes).
  - Time which may elapse before cached zone information must be updated (REFRESH).
  - Time after which to retry a failed refresh (RETRY).
  - Time interval after which zone information is considered not current anymore (EXPIRE).
  - Minimum lifetime for resource records (MINIMUM).

Section 34: Security and Dynamic Updates

31 Overview and Features

32 Resource Records

33 Message Formats

34 Security and Dynamic Updates

35 Creative Usage

DNS Security

- DNS security (DNSSEC) provides data integrity and authentication to security aware resolvers and applications through the use of cryptographic digital signatures.
- The Resource Record Signature (RRSIG) resource record stores digital signatures.
- The DNS Public Key (DNSKEY) resource record can be used to store public keys in the DNS.
- The Delegation Signer (DS) resource record simplifies some of the administrative tasks involved in signing delegations across organizational boundaries.
- The Next Secure (NSEC) resource record allows a security-aware resolver to authenticate a negative reply for either name or type non-existence.

Dynamic DNS Updates

- RFC 2136 / RFC 3007 define a mechanism which allows to dynamically update RRs on name server.
- This is especially useful in environments which use dynamic IP address assignments.
- The payload of a DNS update message contains
  - the zone section,
  - the prerequisite section (supporting conditional updates),
  - the update section, and
  - an additional data section.
- The `nsupdate` command line utility can be used to make manual updates. Some DHCP servers perform automatic updates when they hand out an IP address.

Section 35: Creative Usage

31 Overview and Features

32 Resource Records

33 Message Formats

34 Security and Dynamic Updates

35 Creative Usage

DNS and Anycasting

- DNS servers often make use of IP anycasting in order to improve availability and performance
  - Several DNS service instances with the same IP address are deployed
  - The IP routing system determines to which service instance a specific DNS request is routed
- Many of the DNS root servers ([a-m].root-servers.org) use anycasts; the number of DNS service instances was reported to be above 600 in October 2016
- Anycasting works well for a simple stateless request / response protocol like DNS, see RFC 7094 and RFC 4786 for further details on IP anycasts

DNS and Service Load Balancing

- Content Delivery Networks (CDN) sometimes use short lived DNS answers to direct requests to servers close to the requester.
- An underlying assumption is that the recursive resolver used by a host is located close to the host (in terms of network topology).
- This assumption is not generally true, for example, if hosts use generic recursive resolvers like Google's public DNS resolver (8.8.8.8 or 2001:4860:4860::8888), see also `https://www.xkcd.com/1361/`.
- DNS extensions have been defined to allow a recursive resolver to indicate a client subnet in a DNS request so that DNS servers can provide responses that match the location of the host, see RFC 7871 for further details.

Kaminsky DNS Attack

- Cache poisoning attack ('2008):
  - Cause applications to generate queries for non-existing names such as aaa.example.net, aab.example.net, etc.
  - Send fake responses quickly, trying to guess the 16-bit query ID number.
  - In the fake responses, include additional records that overwrite A records for lets say example.net.
- Counter measure:
  - Updated DNS libraries use random port numbers.
  - An attacker has to guess a 16-bit ID number and in addition the 16-bit port number.
- The real solution is DNSSEC ...

DNS as DDoS Amplifier

- DNS queries with a spoofed source address can be used to direct responses from open resolvers to a certain attack target; the DNS resolver thus helps to hide (to some extend) the source of the attack.
- Since DNS responses are typically larger than DNS queries, a DNS resolver also acts as an amplifier, turning, for example, 100Mbps query traffic into 1Gbps attack traffic.
- DNS security makes amplification significantly more effective if cryptographic algorithms are used that require relatively long keys.
- It has been shown that elliptic curve algorithms tend to be way more space efficient than traditional RSA algorithms.

DNS Blacklists

- DNS Blacklists store information about bad behaving hosts.
- Originally used to publish information about sites that originated unsolicited email (spam).
- If the IP address 192.0.2.99 is found guilty to emit spam, a DNS Blacklists at bad.example.com will add the following DNS records:

```
99.2.0.192.bad.example.com    IN  A      127.0.0.2
99.2.0.192.bad.example.com    IN  TXT    "Spam received."
```

- A mail server receiving a connection from 192.0.2.99 may lookup the A record of 99.2.0.192.bad.example.com and if it has the value 127.0.0.2 decline to serve the client.
- For more details, see RFC 5782.

DNS Backscatter, Stalking, Tunnels, Command and Control, . . .

- Kensuke Fukuda has analyzed the DNS traffic generated by middleboxes when they perform reverse lookups on IP addresses that they see (so called DNS backscatter).
- Geoff Huston placed advertisements on web pages that include one-time valid DNS names to drive certain measurements and they found that these one-time DNS names sometimes enjoy additional lookups from very different locations in the network weeks later (which he called stalking).
- Since DNS traffic is often not filtered, people have created many different techniques and tools to tunnel IP traffic over DNS.
- It has been reported that DNS has seen some usage as a command and control channel for malware and botnets.

Part VIII

Augmented Backus Naur Form (ABNF)

Several application protocols use a textual encoding of protocol messages. A text-based encoding of protocol messages has the advantage that messages are “programmer readable”. The downside of text-based encodings of protocol messages is that encodings usually are less efficient (both in terms of the size of the messages as well as the encoding/decoding processing time).

For the text-based encoding of protocol messages, it is useful to formally specify the format of well-formed protocol messages. This is commonly done by specifying a grammar for the set of well-formed protocol messages. ABNF is a formalism for writing down such grammars.

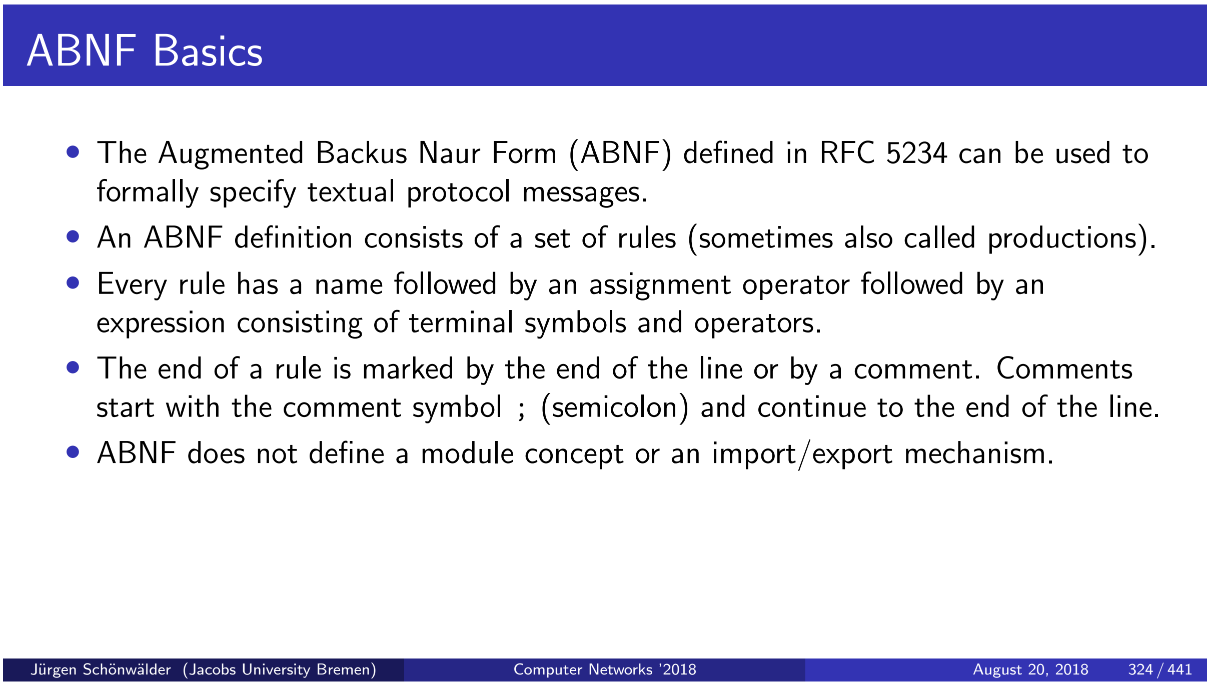
Section 36: Basics, Rule Names, Terminal Symbols

36 Basics, Rule Names, Terminal Symbols

37 Operators

38 Core Definitions

39 ABNF in ABNF



ABNF Basics

- The Augmented Backus Naur Form (ABNF) defined in RFC 5234 can be used to formally specify textual protocol messages.
- An ABNF definition consists of a set of rules (sometimes also called productions).
- Every rule has a name followed by an assignment operator followed by an expression consisting of terminal symbols and operators.
- The end of a rule is marked by the end of the line or by a comment. Comments start with the comment symbol ; (semicolon) and continue to the end of the line.
- ABNF does not define a module concept or an import/export mechanism.

ABNF as defined in RFC 5234 [7] was originally part of the Internet email standards. It has been later factored out since ABNF is useful for many other protocols that use textual message formats.

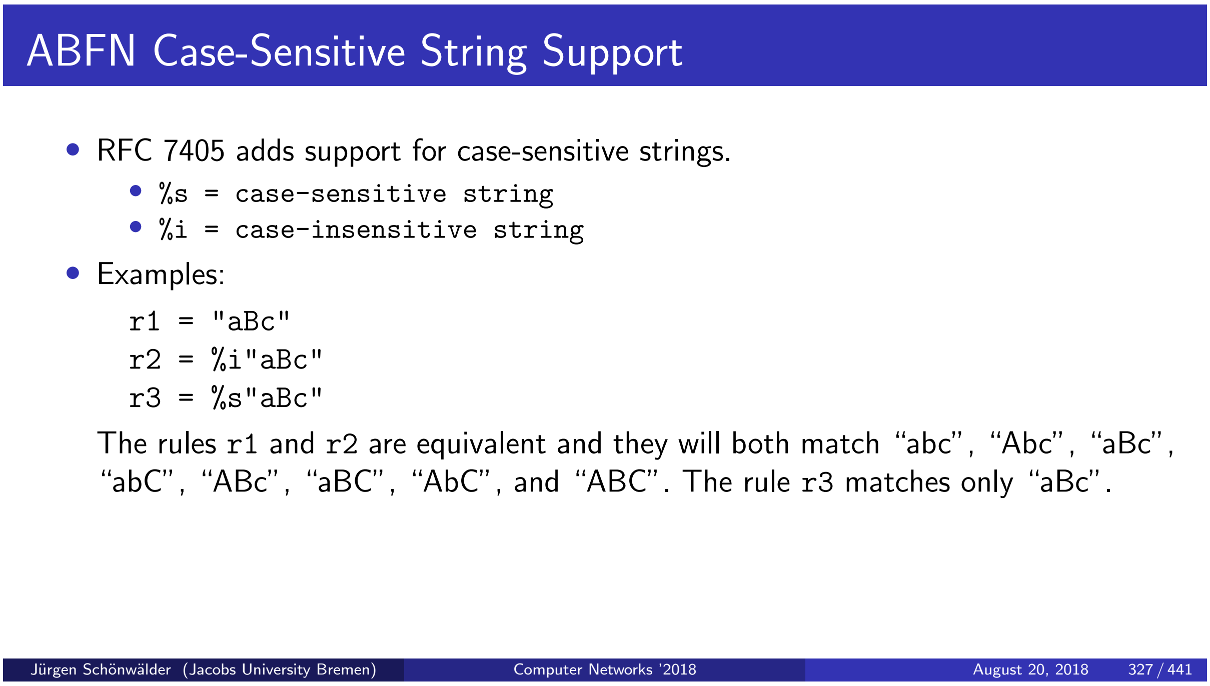
ABNF is a special version of a Backus Naur Form for describing context-free grammars. A specific difference of ABNF to more regular Backus–Naur forms is that ABNF is specific how terminal symbols are encoded.

Rule Names and Terminal Symbols

- The name of a rule must start with an alphabetic character followed by a combination of alphabetics, digits and hyphens. The case of a rule name is not significant.
- Terminal symbols are non-negative numbers. The basis of these numbers can be binary (`b`), decimal (`d`) or hexadecimal (`x`). Multiple values can be concatenated by using the dot `.` as a value concatenation operator. It is also possible to define ranges of consecutive values by using the hyphen `-` as a value range operator.
- Terminal symbols can also be defined by using literal text strings containing US ASCII characters enclosed in double quotes. Note that these literal text strings are case-insensitive.

Simple ABNF Examples

```
CR    = %d13            ; ASCII carriage return code in decimal

CRLF  = %d13.10         ; ASCII carriage return and linefeed code sequence

DIGIT = %x30-39         ; ASCCI digits (0 - 9)

ABA   = "aba"           ; ASCII string "aba" or "ABA" or "Aba" or ...

abba  = %x61.62.62.61   ; ASCII string "abba"
```



ABFN Case-Sensitive String Support

- RFC 7405 adds support for case-sensitive strings.
  - `%s = case-sensitive string`
  - `%i = case-insensitive string`
- Examples:

```
r1 = "aBc"
r2 = %i"aBc"
r3 = %s"aBc"
```

The rules `r1` and `r2` are equivalent and they will both match “abc”, “Abc”, “aBc”, “abC”, “ABc”, “aBC”, “AbC”, and “ABC”. The rule `r3` matches only “aBc”.

The case sensitive string support defined in RFC 7405 [22] is useful in situations where case insensitive string constants are not desired. (Essentially newer protocols that will never run on hardware that only supports uppercase or lowercase characters.)

Section 37: Operators

36 Basics, Rule Names, Terminal Symbols

37 Operators

38 Core Definitions

39 ABNF in ABNF

ABFN Operators

- Concatenation
  - Concatenation operator symbol is the empty word
  - Example: `abba = %x61 %x62 %x62 %x61`
- Alternatives
  - Alternatives operator symbol is the forward slash /
  - Example: `aorb = %x61 / %x62`
  - Incremental alternatives assignment operator =/ can be used for long lists of alternatives
- Grouping
  - Expressions can be grouped using parenthesis
  - A grouped expression is treated as a single element
  - Example: `abba = %x61 (%x62 %x62) %x61`

ABFN Operators

- Repetitions
  - The repetitions operator has the format `n*m` where `n` and `m` are optional decimal values
  - The value of `n` indicates the minimum number of repetitions (defaults to 0 if not present)
  - The value `m` indicates the maximum number of repetitions (defaults to infinity if not present)
  - The format `*` indicates 0 or more repetitions
  - Example: `abba = %x61 2 %x62 %x61`
- Optional
  - Square brackets enclose an optional element
  - Example: `[ab] ; equivalent to *1(ab)`

Section 38: Core Definitions

36 Basics, Rule Names, Terminal Symbols

37 Operators

38 Core Definitions

39 ABNF in ABNF

ABNF Core Definitions

```
ALPHA          =  %x41-5A / %x61-7A     ; A-Z / a-z
BIT            =  "0" / "1"
CHAR           =  %x01-7F               ; any 7-bit US-ASCII character,
                                        ; excluding NUL
CR             =  %x0D                  ; carriage return
CRLF           =  CR LF                 ; Internet standard newline
CTL            =  %x00-1F / %x7F        ; controls
DIGIT          =  %x30-39               ; 0-9
DQUOTE         =  %x22                  ; " (Double Quote)
HEXDIG         =  DIGIT / "A" / "B" / "C" / "D" / "E" / "F"
HTAB           =  %x09                  ; horizontal tab
LF             =  %x0A                  ; linefeed
LWSP           =  *(WSP / CRLF WSP)     ; linear white space (past newline)
OCTET          =  %x00-FF               ; 8 bits of data
SP             =  %x20                  ; space
VCHAR          =  %x21-7E               ; visible (printing) characters
WSP            =  SP / HTAB             ; White space
```

Section 39: ABNF in ABNF

36 Basics, Rule Names, Terminal Symbols

37 Operators

38 Core Definitions

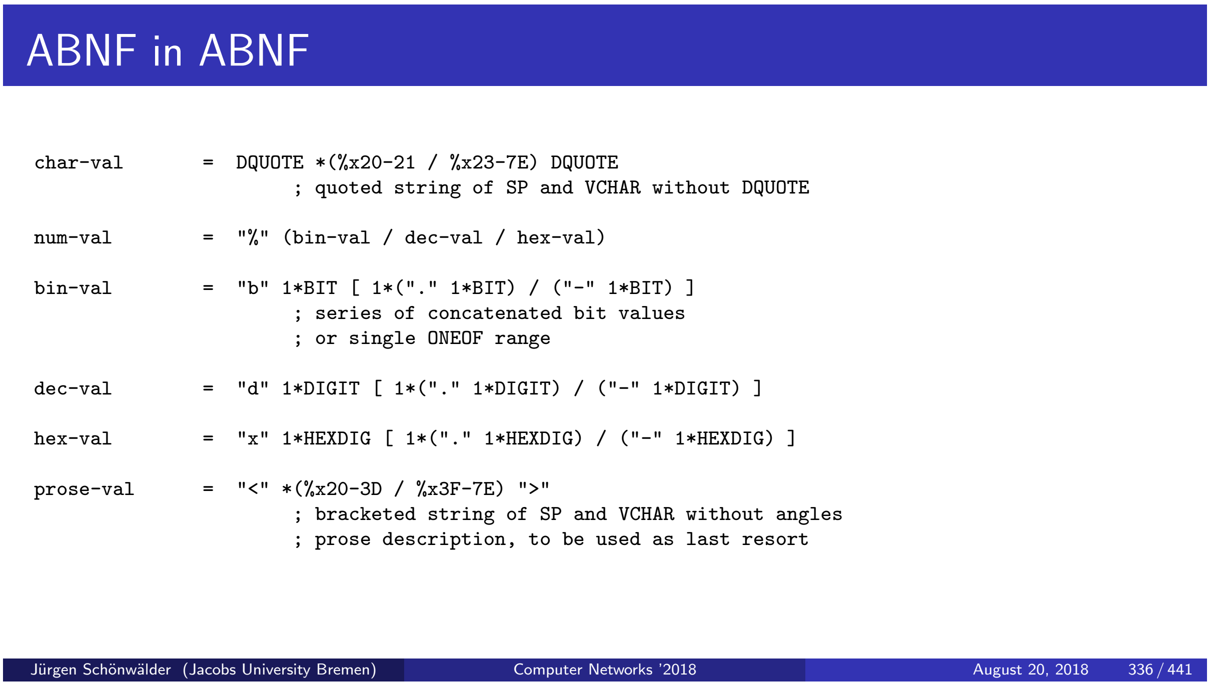
39 ABNF in ABNF

ABNF in ABNF

```
rulelist       =  1*( rule / (*c-wsp c-nl) )

rule           =  rulename defined-as elements c-nl
                       ; continues if next line starts with white space

rulename       =  ALPHA *(ALPHA / DIGIT / "-")

defined-as     =  *c-wsp ("=" / "=/") *c-wsp
                       ; basic rules definition and incremental alternatives

elements       =  alternation *c-wsp

c-wsp          =  WSP / (c-nl WSP)

c-nl           =  comment / CRLF
                       ; comment or newline

comment        =  ";" *(WSP / VCHAR) CRLF
```

ABNF in ABNF

```
alternation    =  concatenation
                  *(*c-wsp "/" *c-wsp concatenation)

concatenation  =  repetition *(1*c-wsp repetition)

repetition     =  [repeat] element

repeat         =  1*DIGIT / (*DIGIT "*" *DIGIT)

element        =  rulename / group / option /
                  char-val / num-val / prose-val

group          =  "(" *c-wsp alternation *c-wsp ")"

option         =  "[" *c-wsp alternation *c-wsp "]"
```



ABNF in ABNF

```
char-val       =  DQUOTE *(%x20-21 / %x23-7E) DQUOTE
                       ; quoted string of SP and VCHAR without DQUOTE

num-val        =  "%" (bin-val / dec-val / hex-val)

bin-val        =  "b" 1*BIT [ 1*("." 1*BIT) / ("-" 1*BIT) ]
                       ; series of concatenated bit values
                       ; or single ONEOF range

dec-val        =  "d" 1*DIGIT [ 1*("." 1*DIGIT) / ("-" 1*DIGIT) ]

hex-val        =  "x" 1*HEXDIG [ 1*("." 1*HEXDIG) / ("-" 1*HEXDIG) ]

prose-val      =  "<" *(%x20-3D / %x3F-7E) ">"
                       ; bracketed string of SP and VCHAR without angles
                       ; prose description, to be used as last resort
```

Part IX

Electronic Mail (SMTP, IMAP)

Electronic mail (email) is a method of exchanging messages between people using electronic devices. Email appeared in the 1960s and by the mid-1970s had taken the form now recognized as email. Email deliver in the early days could take hours or days. Today's email systems use a store-and-forward model where messages are repeatedly stored and forwarded until they have reached the receiver's mailbox. Transient failures of systems may cause delivery delays but usually do not lead to a loss of messages.

We focus on the Internet email system. The central protocols are the Simple Mail Transfer Protocol (SMTP) and the associated Internet Message Format. Like most protocols designed in the late 1970s and early 1980s, not much attention was given to security and privacy aspects.

Section 40: Components and Terminology

40 Components and Terminology

41 Simple Mail Transfer Protocol (SMTP)

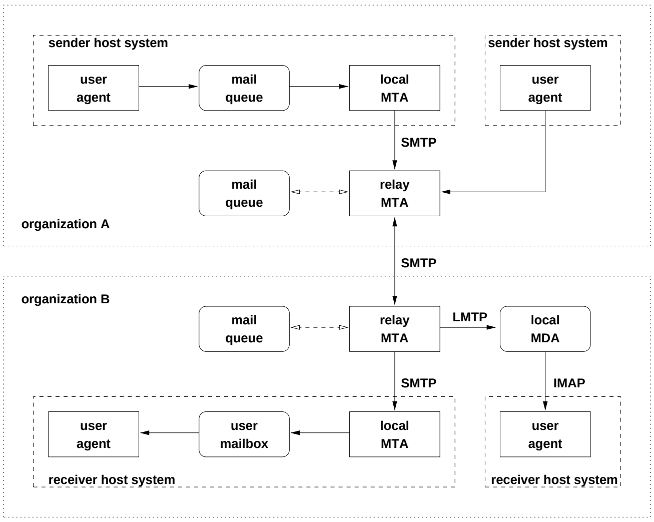
42 Multipurpose Internet Mail Extensions (MIME)

43 Internet Message Access Protocol (IMAP)

44 Filtering of Messages (SIEVE)

45 DomainKeys Identified Mail (DKIM)

Components Involved in Electronic Mail



Terminology

- Mail User Agent (MUA) - the source or targets of electronic mail
- Mail Transfer Agent (MTA) - server and clients providing mail transport service
- Mail Delivery Agent (MDA) - delivers mail messages to the receiver's mail box
- Store-and-Forward Principle - mail messages are stored and then forwarded to another system; responsibility for a message is transferred once it is stored again
- Envelop vs. Header vs. Body - mail messages consist of a header and a body; the transfer of mail message is controlled by the envelop (which might be different from the header)

Section 41: Simple Mail Transfer Protocol (SMTP)

40 Components and Terminology

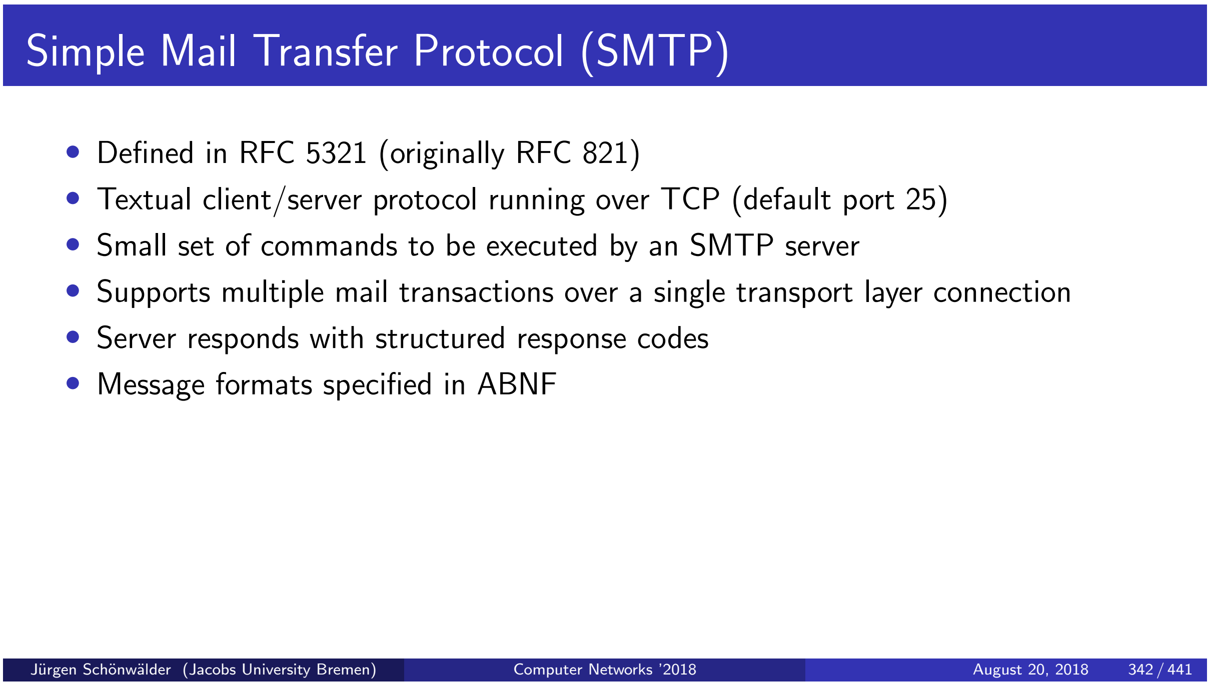
41 Simple Mail Transfer Protocol (SMTP)

42 Multipurpose Internet Mail Extensions (MIME)

43 Internet Message Access Protocol (IMAP)

44 Filtering of Messages (SIEVE)

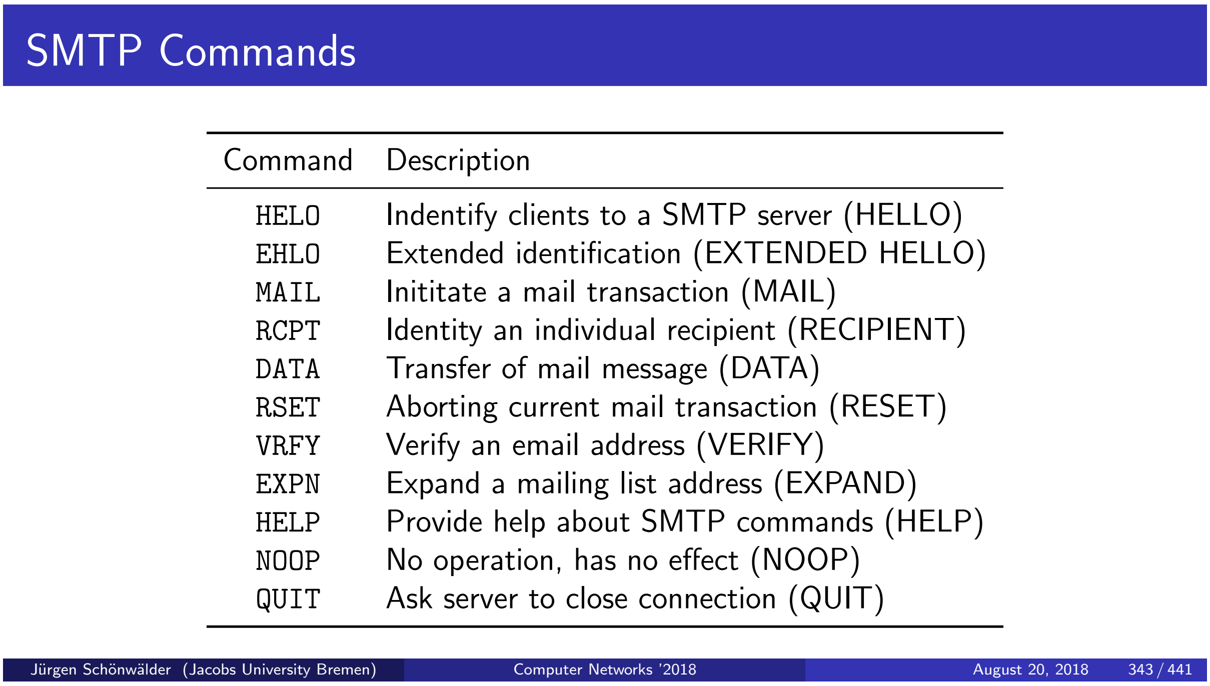
45 DomainKeys Identified Mail (DKIM)



Simple Mail Transfer Protocol (SMTP)

- Defined in RFC 5321 (originally RFC 821)
- Textual client/server protocol running over TCP (default port 25)
- Small set of commands to be executed by an SMTP server
- Supports multiple mail transactions over a single transport layer connection
- Server responds with structured response codes
- Message formats specified in ABNF

The SMTP protocol is one on the old classic Internet protocols. The first version was published in RFC 821 [30] in 1982. The current version is defined in RFC 5321 [19], which appeared in 2008. In the early days of the Internet, there were many email systems out there and there was a big need to interwork with other email systems.



SMTP Commands

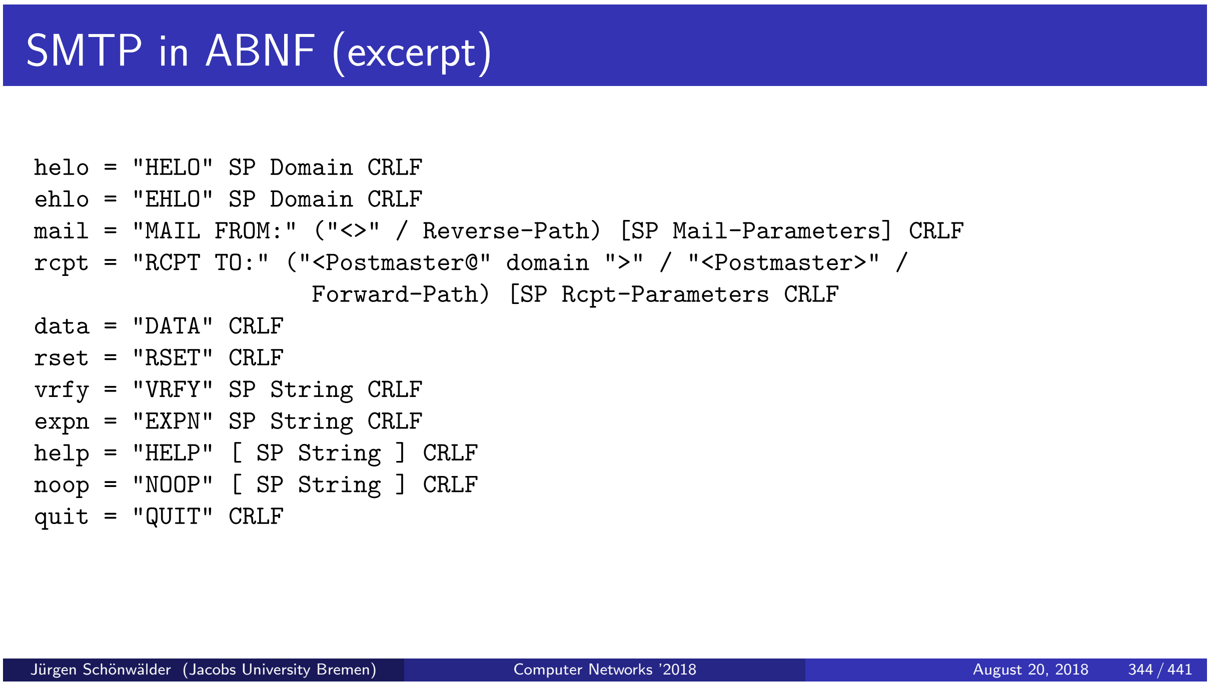
Command	Description
HELO	Indentify clients to a SMTP server (HELLO)
EHLO	Extended identification (EXTENDED HELLO)
MAIL	Inititate a mail transaction (MAIL)
RCPT	Identity an individual recipient (RECIPIENT)
DATA	Transfer of mail message (DATA)
RSET	Aborting current mail transaction (RESET)
VRFY	Verify an email address (VERIFY)
EXPN	Expand a mailing list address (EXPAND)
HELP	Provide help about SMTP commands (HELP)
NOOP	No operation, has no effect (NOOP)
QUIT	Ask server to close connection (QUIT)

Here is the transcript of an example session with a mail server (sending a fake email). We first lookup the mail exchanger for our target domain:

```
$ dig +short mx example.com.
20 mx1.example.com
20 mx2.example.com
20 mx3.example.com
```

We randomly pick an address (if there are multiple addresses with the same priority) and start a TCP connection to port 25:

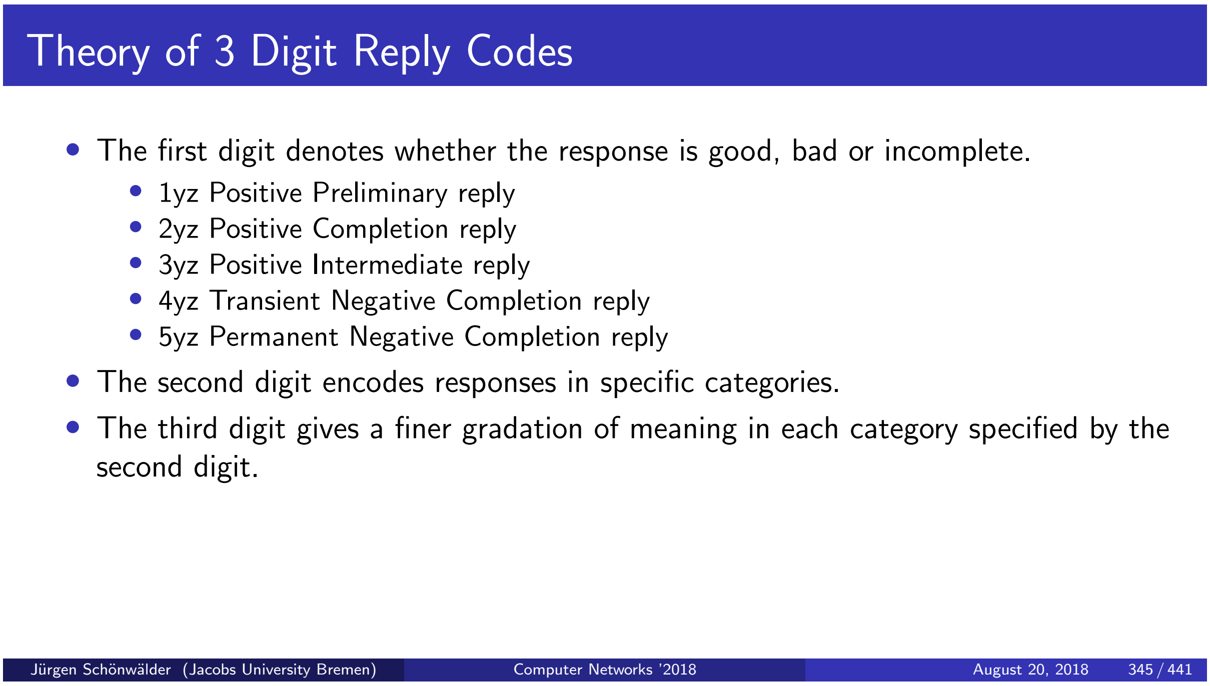
```
$ nc mx2.example.com. 25
220 mx2.example.com. ESMTP Postfix
EHLO hacker.com
250-mx2.example.com
250-PIPELINING
250-SIZE 102400000
250-VRFY
250-ETRN
250-STARTTLS
250-ENHANCEDSTATUSCODES
250-8BITMIME
250 DSN
MAIL FROM: <j.luser@fakemail.com>
250 2.1.0 Ok
RCPT TO: <joe.researcher@example.com>
250 2.1.5 Ok
DATA
354 End data with <CR><LF>.<CR><LF>
From: A. Turing <a.turing@award.com>
To: <joe.researcher@example.com>
Subject: your turing award
Date: Mon, 30 Apr 2018 10:01:04 +0200

Dear outstanding researcher,

I am delighted to announce that you will receive this year's Alan
Turing award for your outstanding research. To participate at the
ceremony, please pay 3.000 Euro for the award processing and the
travel arrangements.

Alan Turing
.
250 2.0.0 from MTA(smtp:[2001:db8::42]:10031): 250 2.0.0 Ok: queued as F39AC6CF
QUIT
221 2.0.0 Bye
```



SMTP in ABNF (excerpt)

```
helo = "HELO" SP Domain CRLF
ehlo = "EHLO" SP Domain CRLF
mail = "MAIL FROM:" ("<>" / Reverse-Path) [SP Mail-Parameters] CRLF
rcpt = "RCPT TO:" ("<Postmaster@" domain ">" / "<Postmaster>" /
                    Forward-Path) [SP Rcpt-Parameters CRLF
data = "DATA" CRLF
rset = "RSET" CRLF
vrfy = "VRFY" SP String CRLF
expn = "EXPN" SP String CRLF
help = "HELP" [ SP String ] CRLF
noop = "NOOP" [ SP String ] CRLF
quit = "QUIT" CRLF
```

Note that email addresses are written using angle brackets as delimiters. The correct writing of the email address j.user@example.org is therefore `<joe.user@example.org>`. Note that case does not matter, hence `<Joe.User@example.org>` is the same address as `<joe.user@example.org>`. (Since there is no real value in writing email addresses in mixed case, technical people tend to prefer writing email address with all characters in lowercase.)

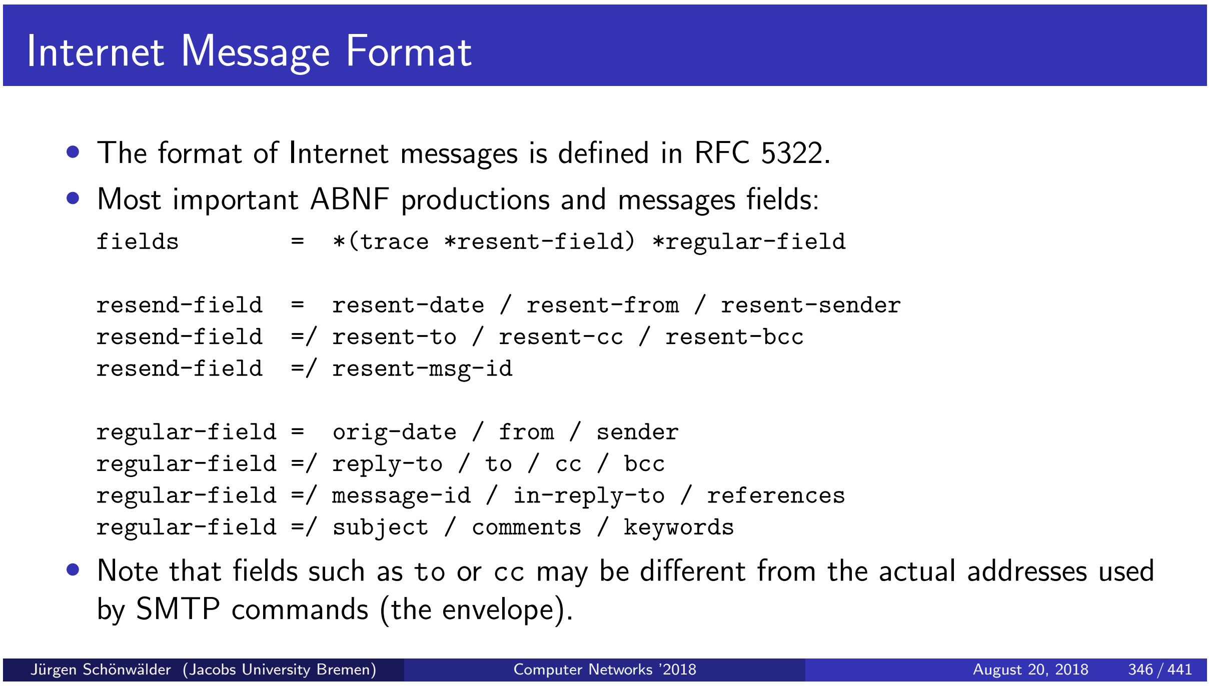


Theory of 3 Digit Reply Codes

- The first digit denotes whether the response is good, bad or incomplete.
  - 1yz Positive Preliminary reply
  - 2yz Positive Completion reply
  - 3yz Positive Intermediate reply
  - 4yz Transient Negative Completion reply
  - 5yz Permanent Negative Completion reply
- The second digit encodes responses in specific categories.
- The third digit gives a finer gradation of meaning in each category specified by the second digit.

Jürgen Schönwälder (Jacobs University Bremen) Computer Networks '2018 August 20, 2018 345 / 441

The three digit reply codes promote extensibility of the SMTP protocol. If new error codes are introduced (see for example RFC 7504 [**?**]), then old implementations that do not understand the precise semantics can still behave sensible if the semantics of the first or second digit is understood.



Internet Message Format

- The format of Internet messages is defined in RFC 5322.
- Most important ABNF productions and messages fields:

```
fields        =  *(trace *resent-field) *regular-field

resend-field  =  resent-date / resent-from / resent-sender
resend-field  =/ resent-to / resent-cc / resent-bcc
resend-field  =/ resent-msg-id

regular-field =  orig-date / from / sender
regular-field =/ reply-to / to / cc / bcc
regular-field =/ message-id / in-reply-to / references
regular-field =/ subject / comments / keywords
```

- Note that fields such as `to` or `cc` may be different from the actual addresses used by SMTP commands (the envelope).

Originator Fields

- The `From:` field specifies the author(s) of the message.

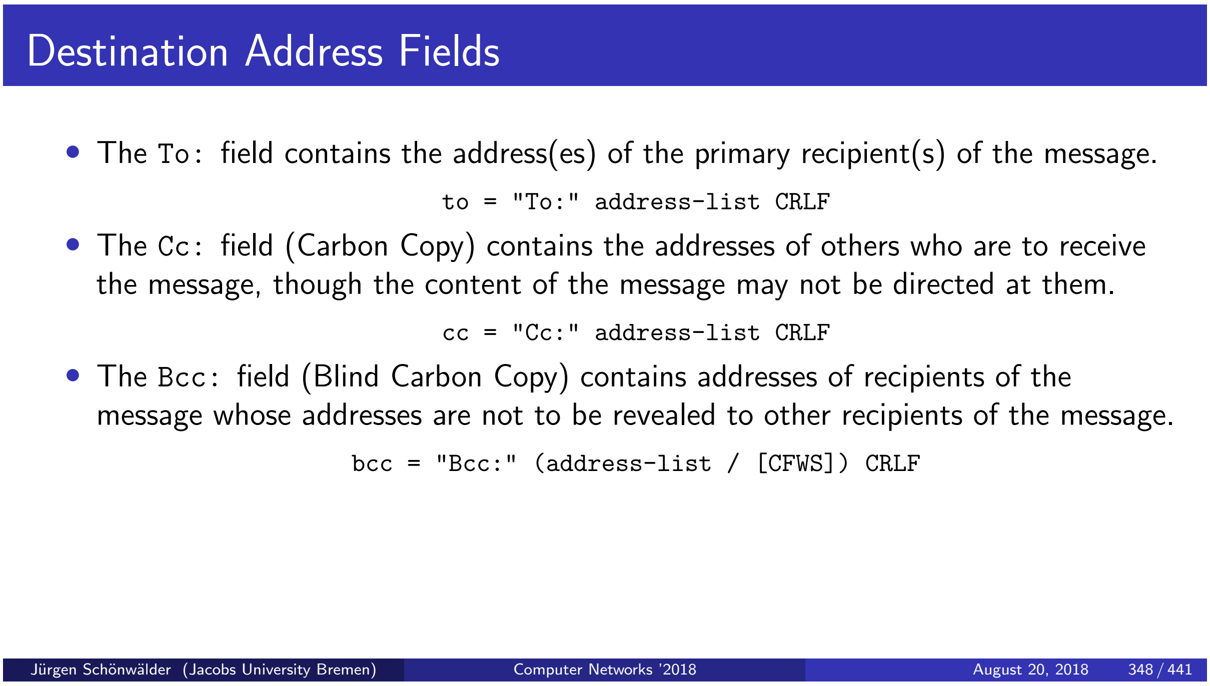
```
from = "From:" mailbox-list CRLF
```

- The `Sender:` field specifies the mailbox of the sender in cases where the actual sender is not the author (e.g., a secretary).

```
sender = "Sender:" mailbox CRLF
```

- The `Reply-To:` field indicates the mailbox(es) to which the author of the message suggests that replies be sent.

```
reply-to = "Reply-To:" address-list CRLF
```



Destination Address Fields

- The `To:` field contains the address(es) of the primary recipient(s) of the message.

```
to = "To:" address-list CRLF
```

- The `Cc:` field (Carbon Copy) contains the addresses of others who are to receive the message, though the content of the message may not be directed at them.

```
cc = "Cc:" address-list CRLF
```

- The `Bcc:` field (Blind Carbon Copy) contains addresses of recipients of the message whose addresses are not to be revealed to other recipients of the message.

```
bcc = "Bcc:" (address-list / [CFWS]) CRLF
```

Identification and Origination Date Fields

- The `Message-ID:` field provides a unique message identifier that refers to a particular version of a particular message.

```
message-id = "Message-ID:" msg-id CRLF
```

- The `In-Reply-To:` field will contain the contents of the `Message-ID:` field of the message to which this one is a reply.

```
in-reply-to = "In-Reply-To:" 1*msg-id CRLF
```

- The `References:` field will contain the contents of the parent's `References:` field (if any) followed by the contents of the parent's `Message-ID:` field (if any).

```
references = "References:" 1*msg-id CRLF
```

Informational Fields

- The `Subject:` field contains a short string identifying the topic of the message.

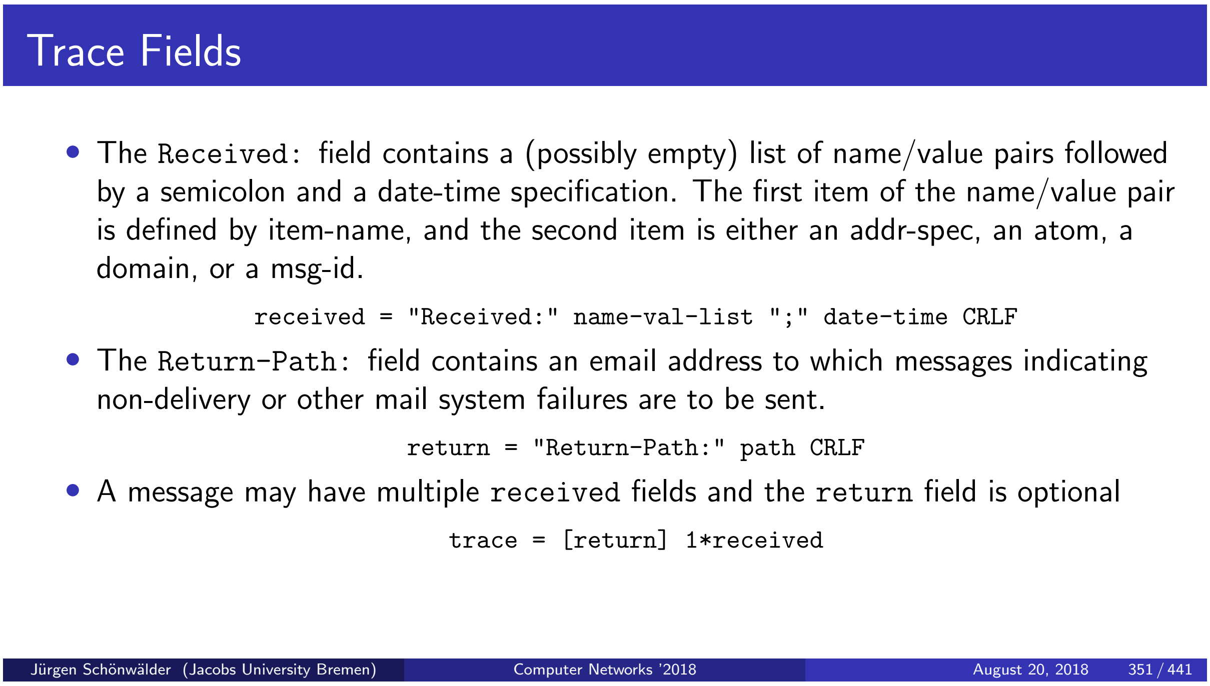
```
subject = "Subject:" unstructured CRLF
```

- The `Comments:` field contains any additional comments on the text of the body of the message.

```
comments = "Comments:" unstructured CRLF
```

- The `Keywords:` field contains a comma-separated list of important words and phrases that might be useful for the recipient.

```
keywords = "Keywords:" phrase *("," phrase) CRLF
```



Trace Fields

- The `Received:` field contains a (possibly empty) list of name/value pairs followed by a semicolon and a date-time specification. The first item of the name/value pair is defined by item-name, and the second item is either an addr-spec, an atom, a domain, or a msg-id.

```
received = "Received:" name-val-list ";" date-time CRLF
```

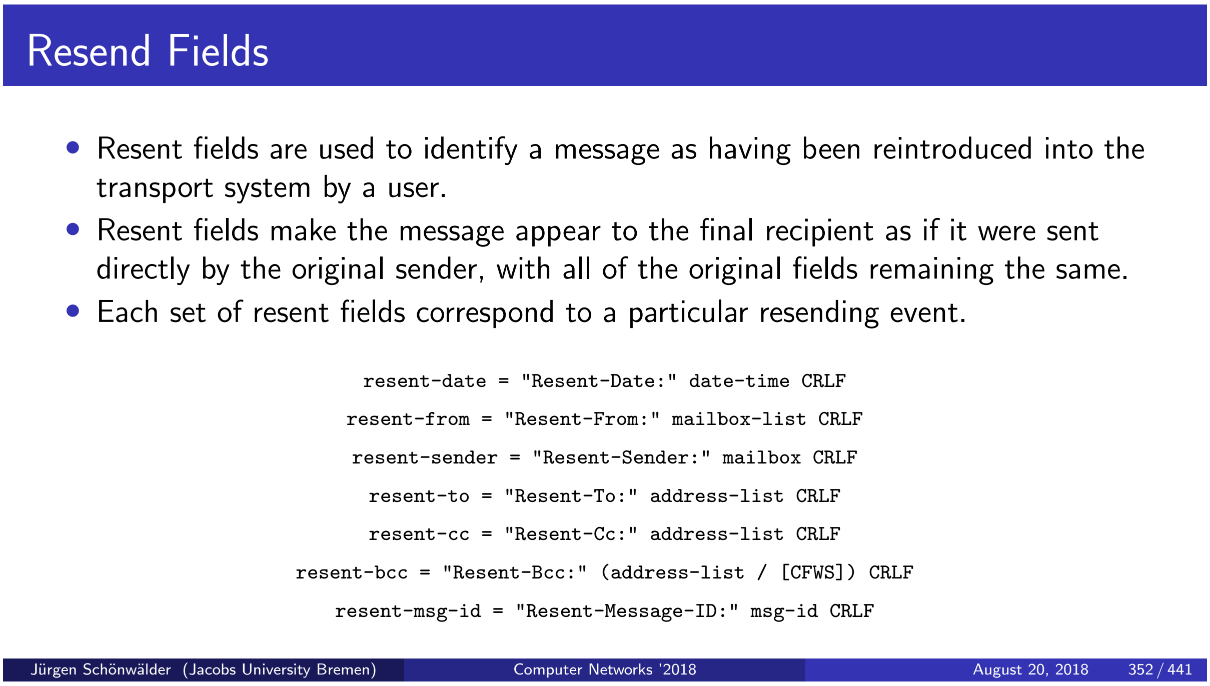
- The `Return-Path:` field contains an email address to which messages indicating non-delivery or other mail system failures are to be sent.

```
return = "Return-Path:" path CRLF
```

- A message may have multiple `received` fields and the `return` field is optional

```
trace = [return] 1*received
```

Trace fields are important for troubleshooting email delivery problems. They are usually not shown to the user (unless the user wishes to see them.)

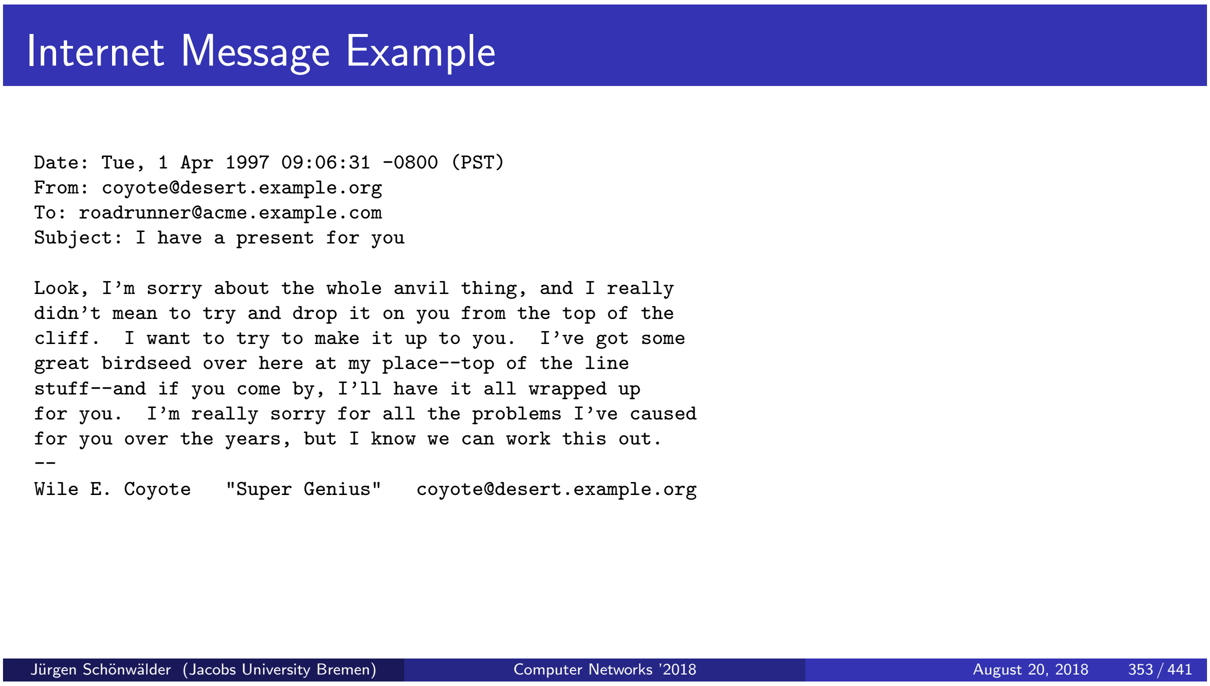


Resend Fields

- Resent fields are used to identify a message as having been reintroduced into the transport system by a user.
- Resent fields make the message appear to the final recipient as if it were sent directly by the original sender, with all of the original fields remaining the same.
- Each set of resent fields correspond to a particular resending event.

```
resent-date = "Resent-Date:" date-time CRLF
resent-from = "Resent-From:" mailbox-list CRLF
resent-sender = "Resent-Sender:" mailbox CRLF
resent-to = "Resent-To:" address-list CRLF
resent-cc = "Resent-Cc:" address-list CRLF
resent-bcc = "Resent-Bcc:" (address-list / [CFWS]) CRLF
resent-msg-id = "Resent-Message-ID:" msg-id CRLF
```

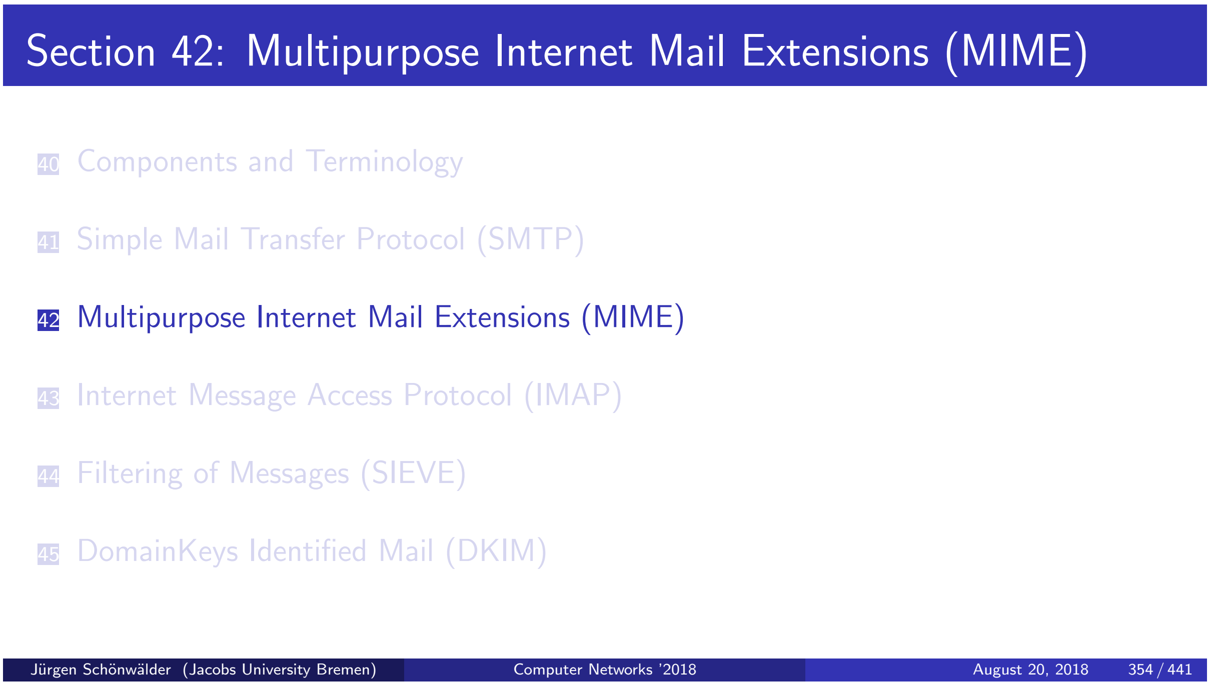
The resens mechanism, though sometimes very useful, often tends to confuse people.



Internet Message Example

```
Date: Tue, 1 Apr 1997 09:06:31 -0800 (PST)
From: coyote@desert.example.org
To: roadrunner@acme.example.com
Subject: I have a present for you

Look, I’m sorry about the whole anvil thing, and I really
didn’t mean to try and drop it on you from the top of the
cliff.  I want to try to make it up to you.  I’ve got some
great birdseed over here at my place--top of the line
stuff--and if you come by, I’ll have it all wrapped up
for you.  I’m really sorry for all the problems I’ve caused
for you over the years, but I know we can work this out.
--
Wile E. Coyote   "Super Genius"   coyote@desert.example.org
```

This example is taken from RFC 5228 [13]. Additional examples for mail messages can be found in Appendix A of RFC 5322 [32].



Section 42: Multipurpose Internet Mail Extensions (MIME)

40 Components and Terminology

41 Simple Mail Transfer Protocol (SMTP)

42 Multipurpose Internet Mail Extensions (MIME)

43 Internet Message Access Protocol (IMAP)

44 Filtering of Messages (SIEVE)

45 DomainKeys Identified Mail (DKIM)

TBD

Section 43: Internet Message Access Protocol (IMAP)

40 Components and Terminology

41 Simple Mail Transfer Protocol (SMTP)

42 Multipurpose Internet Mail Extensions (MIME)

43 Internet Message Access Protocol (IMAP)

44 Filtering of Messages (SIEVE)

45 DomainKeys Identified Mail (DKIM)

TBD

Section 44: Filtering of Messages (SIEVE)

40 Components and Terminology

41 Simple Mail Transfer Protocol (SMTP)

42 Multipurpose Internet Mail Extensions (MIME)

43 Internet Message Access Protocol (IMAP)

44 Filtering of Messages (SIEVE)

45 DomainKeys Identified Mail (DKIM)

TBD

Section 45: DomainKeys Identified Mail (DKIM)

40 Components and Terminology

41 Simple Mail Transfer Protocol (SMTP)

42 Multipurpose Internet Mail Extensions (MIME)

43 Internet Message Access Protocol (IMAP)

44 Filtering of Messages (SIEVE)

45 DomainKeys Identified Mail (DKIM)

TBD

Part X

Summary Tables

This appendix contains some summary tables that make it easier to compare some protocols or to quickly lookup how certain things are solved by a certain protocol.

Aspect	Ethernet (+ VLANs)	IP version 4	IP version 6
Layers	physical (1) and data link (2)	network (3)	network (3)
Addresses	48-bit	32-bit	128-bit (64-bit interface identifiers)
Assignment	static (vendor)	manual or dynamic	manual or dynamic
Auto Configuration	auto negotiation (802.3u)	DHCPv4 (address)	DHCPv6 (address) or SLAC (prefix)
Header	fixed length	variable length	fixed length + header chain
Checksum	CRC-32	Interent checksum (header)	none (trust other layers)
Forwarding	exact lookup and flooding	longest prefix match	longest prefix match
Routing	backward learning + spanning tree	OSPF, BGP, . . .	OSPF, BGP, . . .
Tagging	VLANs (802.1Q)		Flow Label
Priorities	3 priority bits (801.1Q)	6-bit DSCP (Type of Service field)	6-bit DSCP (Traffic Class field)
Error Reporting	none	ICMPv4	ICMPv6
Testing	none	ICMPv4 (ping / traceroute)	ICMPv6 (ping / traceroute)
Adress Mapping	not applicable	ARP	ICMPv6 (neighbor discovery)
Fragmentation		routers on the path	sender only
PMTU discovery		optional but recommended	mandatory if MTU $>$ 1280
Security	lacking	lacking	possible (AH, ESP), not widely used

Table 1: Comparison of Ethernet (+ VLANs) and IPv4 and IPv6

Aspect	UDP	TCP	SCTP	DCCP
congestion aware	no	yes	yes	yes

Table 2: Comparison of UDP, TCP, SCTP, and DCCP

Aspect	RIP	OSPF	BGP
Algorithm	Bellman Ford	Dijkstra	
Class	distance vector routing	link state routing	path vector routing
Usage	IGP	IGP	EGP
Scalability	limited	areas	
Transport			
Security			

Table 3: Comparison of Internet Routing Protocols

References

[1] M. Allman, V. Paxson, and E. Blanton. TCP Congestion Control. RFC 5681, ICSI, Purdue University, September 2009.

[2] S. Amante, B. Carpenter, S. Jiang, and J. Rajahalme. IPv6 Flow Label Specification. RFC 6437, Level 3, Univ. of Auckland, Huawei, Nokia Siemens Networks, November 2011.

[3] G. Carlucci, L. De Cicco, and S. Mascolo. HTTP over UDP: An Experimental Investigation of QUIC. In *Proceedings of the 30th Annual ACM Symposium on Applied Computing*, SAC '15, pages 609–614. ACM, April 2015.

[4] D. D. Clark, J. Wroclawski, K. R. Sollins, and R. Braden. Tussle in Cyberspace: Defining Tomorrow's Internet. In *Proc. SIGCOMM 2002*, Pittsburgh, August 2002. ACM.

[5] A. Conta, S. Deering, and M. Gupta. Internet Control Message Protocol (ICMPv6) for the Internet Protocol Version 6 (IPv6) Specification. RFC 4443, Transwitch, Cisco Systems, Tropos Networks, March 2006.

[6] M. Crawford. Transmission of IPv6 Packets over Ethernet Networks. RFC 2464, Fermilab, December 1998.

[7] D. Crocker and P. Overell. Augmented BNF for Syntax Specifications: ABNF. RFC 5234, Brandenburg InternetWorking, THUS plc., January 2008.

[8] S. Deering and R. Hinden. Internet Protocol, Version 6 (IPv6) Specification. RFC 8200, Check Point Software, July 2017.

[9] R. Droms. Dynamic Host Configuration Protocol. RFC 2131, Bucknell University, March 1997.

[10] R. Droms and W. Arbaugh. Authentication for DHCP Messages. RFC 3118, Cisco Systems, University of Maryland, June 2001.

[11] R. Finlayson, T. Mann, J. Mogul, and M. Theimer. A Reverse Address Resolution Protocol. RFC 903, Stanford University, June 1984.

[12] S. Floyd. HighSpeed TCP for Large Congestion Windows. RFC 3649, ICSI, December 2003.

[13] P. Guenther and T. Showalter. Sieve: A Mail Filtering Language. RFC 5228, Sendmail Inc., Mirapoint Inc., January 2008.

[14] M. Handley. Why the Internet only just works. *BT Technology Journal*, 24(3):119–129, July 2006.

[15] C. Hornig. A Standard for the Transmission of IP Datagrams over Ethernet Networks. RFC 894, Symbolics Cambridge Research Center, April 1984.

[16] S. Jiang, S. Krishnan, and T. Mrugalski. Privacy Considerations for DHCP. RFC 7819, Huawei Technologies, Ericsson, ISC, April 2016.

[17] P. Karn, C. Bormann, G. Fairhurst, D. Grossman, R. Ludwig, J. Mahdavi, G. Montenegro, J. Touch, and L. Wood. Advice for Internet Subnetwork Designers. RFC 3819, Qualcomm, Universitaet Bremen TZI, University of Aberdeen, Motorola, Ericsson Research, Novell, Sun Microsystems Laboratories, USC/ISI, Cisco Systems, July 2004.

[18] C. Kent and J. Mogul. Fragmentation Considered Harmful. In *Proc. SIGCOMM '87 Workshop on Frontiers in Computer Communications Technology*, August 1987.

[19] J. Klensin. Simple Mail Transfer Protocol. RFC 5321, October 2008.

[20] E. Kohler, M. Handley, and S. Floyd. Datagram Congestion Control Protocol (DCCP). RFC 4340, UCLA, UCL, ICIR, March 2006.

[21] E. Kohler, M. Handley, and S. Floyd. Designing DCCP: Congestion Control Without Reliability. In *Proc. SIGCOMM 2006*, pages 27–38, Pisa, September 2006. ACM.

[22] P. Kyzivat. Case-Sensitive String Support in ABNF. RFC 7405, December 2014.

[23] P. Mockapetris. Domain Names - Concepts and Facilities. RFC 1034, ISI, November 1987.

[24] P. Mockapetris. Domain Names - Implementation and Specification. RFC 1035, ISI, November 1987.

[25] J. Mogul and S. Deering. Path MTU Discovery. RFC 1191, DECWRL, Stanford University, November 1990.

[26] D. C. Plummer. An Ethernet Address Resolution Protocol. RFC 826, MIT, November 1982.

[27] J. Postel. User Datagram Protocol. RFC 768, ISI, August 1980.

[28] J. Postel. Internet Protocol. RFC 791, ISI, September 1981.

[29] J. Postel. Transmission Control Protocol. RFC 793, ISI, September 1981.

[30] J. Postel. Simple Mail Transfer Protocol. RFC 821, ISI, August 1982.

[31] K. Ramakrishnan, S. Floyd, and D. Black. The Addition of Explicit Congestion Notification (ECN) to IP. RFC 3168, TeraOptic Networks, ACIRI, EMC, September 2001.

[32] P. Resnick. Internet Message Format. RFC 5322, QUALCOMM Incorporated, October 2008.

[33] R. Stewart. Stream Control Transmission Protocol. RFC 4960, September 2007.

[34] R. Stewart, Q. Xie, K. Morneault, C. Sharp, H. Schwarzbauer, T. Taylor, I. Rytina, M. Kalla, L. Zhang, and V. Paxson. Stream Control Transmission Protocol. RFC 2960, Motorola, Cisco, Siemens, Nortel Networks, Ericsson, Telcordia, UCLA, ACIRI, October 2000.

[35] K. Wierenga, S. Winter, and T. Wolniewicz. The eduroam Architecture for Network Roaming. RFC 7593, Cisco Systems, RESTENA, Nicolaus Copernicus University, September 2015.