Zusammenfassung Computer Networks

Entropy

→ How much Information is needed to fully specify the micro state of a System

Level of Randomness → the amount of Information encoded

$$H(X) = -\sum_{x \in X} p(x) \log_2 p(x)$$

⇒ We only use binary

$$H_b(p) = -p \cdot \log_2(p) - (1-p) \cdot \log_2(1-p)$$

↳ Result is in Bits

$p =$ proportion of bits in a Data set

↳ z.B $\frac{\text{number of 1}}{\text{number of 1 and 0}}$

Channel Capacity

noiseless Channel: Every sent Bit is received → capacity 1 Bit

binary Symmetric Channel: Every sent Bit has a probability of p to be received correctly and a probability of $1-p$ to be flipped

⇒ capacity $C = 1 - H_b(p)$ bits

Circuit switching

→ direct link between sender and receiver

↳ a direct channel like a old telephone switchboard

It is dedicated and as long as it is open only the two can send data

⇒ Not ideal for computers as they don't send data most of the time but when they do it is a big load

Packet switching

→ breaks messages into pieces and sends them through whatever channel has space, independently

↳ It is reassembled at the destination.

Packets can arrive out of order.

Internet

→ System of interconnected computer networks

↳ It uses the two main Protocols IP & TCP

IP (Internet Protocol) → addressing and routing

TCP (Transmission Control Protocol) → reliable data delivery

World Wide Web

→ built on top of the Internet

↳ allows resource sharing

Resources are identified by URI's (Uniform Resource Identifiers)

and can be linked with Hyperlinks

It is built on multiple standards like:

URL (Uniform Resource Locator)

HTTP (Hypertext Transfer Protocol)

HTML (Hypertext Markup Language)

originally URI was just UDI (Uniform Document Identifiers)

but with the addition of The Web of Things, it become URI

API

→ Application Programming Interface

↳ a set of definition and protocol for building and integrating application software without having to know how they are implemented.

Physical Layer

→ transmits Bits from sender to receiver wired or wireless

Wired Carriers

Fibre optic → average speed: 30-40 Gbit/s
↳ carries data using lasers

IEEE 802.3 Ethernet → average speed: 100MBit/s - 1Gbit/s
↳ carries data on a 13v connection

1 Wire Protocol → average Speed: 16kBit/s
↳ only one wire, used for sensors etc.
↳ carries data on a 3.3/5v connection

Wireless Carriers

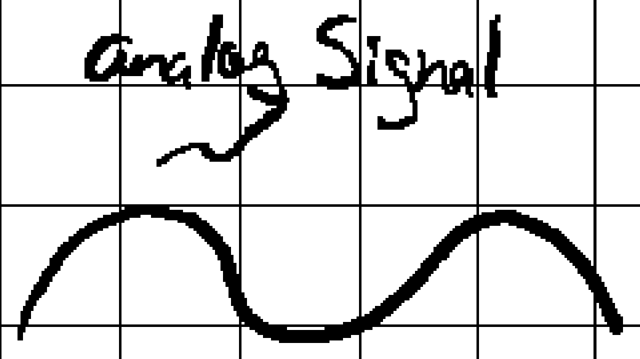
→ uses electromagnetic waves to transmit

Visible Ligh Communication → really fast blinking LEDs

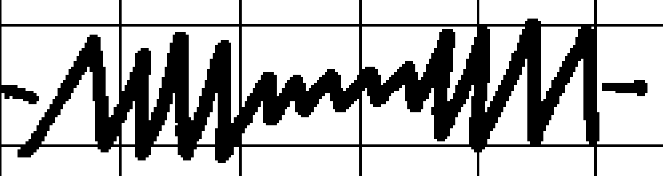
NEC InfraRed → wave length 870nm / 930-950nm

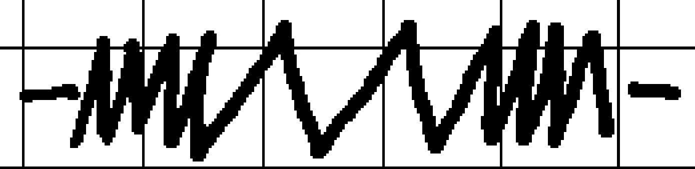
wireless Radio
↳ **Bluetooth Low Energy** → 2.4 GHz, 40 channels à 2 MHz width
↳ **IEEE 802.11 (wifi)** → 14 channel 2.4GHz
→ others in 3.65, 4.9, 5.0, 5.9 GHz

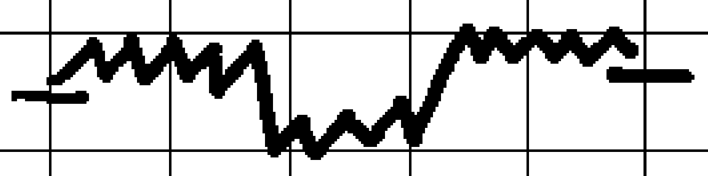
Digital to Analog



→ we modulate analog carries Signals

Amplitude Shift keying → modulate Amplitude → 

Frequnzy Shift keying → modulate Frequency → 

Phase Shift keying → modulate Phase → 

Data Link Layer

Medium Access Control (MAC)

→ MAC addresses are unique identifiers assigned to device network inteface controller (NIC)

MAC addresses are assigned by manufactues as Univesally Administred Addresses (UAA)

They can be manually chenged

↳ They get flagged as Locally Administred Addresses (LAA)

↳ Second-least significant Bit of the first octed is set to 1

↳ 0A:1B:44:11:3A:B7

0000 1010

Collision Domain

→ Devices that share the same physical trasmission Medium

↳ communication collides if it is sent at the same time

Broadcast Domain

→ Devices that share the same data Link

↳ manged by e.g. a switch

↳ Broadcaster sends signall all receiver check if it is ment for them, ignore if not, process if is

Address Resolution Protocol (ARP)

→ Protocol to discover Link-Layer addresses

↳ establishes connection between Link- and network-Layer

⇒ addresses are temporary stored on a devices ARP-Table

Token Ring

→ "talking" Stick moves arround, holder can send data

Carrier Sense Multiple Access with Collision Detection (CSMA/CD)

→ waits with transmitting until channel is free

→ on collision → jam colliding signal and restart

⇒ obsolete, cause send/receive doesn't use the same channel anymore

⇒ does not work in wireless connection, because not all devices see each other

Carrier Sense Multiple Access with Collision Avoidance (CSMA/CA)

→ Handshake idea

↳ Sender asks receiver if channel is free using Request to Send (RTS)

→ Receiver checks if free and sends Clear to Send (CTS)

⇒ used in Wifi

Exponential Backoff

→ Finite Patience, infinite Mercy

↳ network use this to avoid overloading servers

↳ clients never give up completly

Network Layer

→ internetwork identification for addressing across broadcast domains

IP Addresses

→ nummeric Labels assigned to connected devices

IPv4: 32 Bits → 2^{32} possible Addresses (~ 4.3 bio)

- manged by Internet Assigned Numbers Authority (IANA) and five regional Internet registries (RIRs)

IPv6: 128 Bits → 2^{128} (more Atoms then the known universe)

→ future of IP addresses (currently ~60% penetration)

Internet Routing

→ split into Autonomos Systems that are connected

↳ manage and rout their own traffic inside

Time to live (TTL)

→ avoid ∞ loops inside a network by using TTL

↳ TTL is decremented on each hop inside the network

↳ it is dropped if TTL reaches 0

Routing between Autonomaus Systems (AS)

→ AS 1 asks connection provider (Switchlan) if thee is a path to AS 2

↳ provide returns list of paths

↳ AS 1 checks if it itself is on the path to avoid loops

⇒ Date can be sent without AS1 knowing the structure of AS 2 and vice versa

Subnet Mask

→ defines how many bits in a IP address can be used for host addesses

e.g. $\underbrace{192.168.0.1}_{\text{in binary}}$ /24 ~ number of Bits that can't be used (network Bits)

⇒ $\underbrace{11000000.10101000.00000000}_{\text{network Bits}}.\underbrace{00000001}_{\text{Host bits}}$

Subnet Mask = all network bits to 1, unused to 0

e.g. ⇒ 255.255.255.0

network Id = all host bits to 0

broadcast Id = all host bits to 1

⇒ total available IP addresses = $2^{\text{hostbits}} - 2$

e.g. ⇒ $2^8 - 2 = 254$

IPv6 Structure

- lowe 64 Bits → identify host intofaces
- upper 64 Bits → depends on routing Methodology

Unicast

- id individual recipients

Anycast

- id group of recipients any receive message

Multicast

- id group of recipients, each receive message

(⇒ broadcast is a type of multicast)

IP Datagram checksum Calculation

→ 16-bit one's complement sum of all 16-bit words in header

⇒ Sum up all 16-bit numbers

⇒ get the 1-complement of the result

⇒ Bsp Header = 4500 603c 1c46 4060 4006 (Hex)

= sum = $E188_{16}$

↳ in Binary = 1110000110001000

1-complement = 0001111001110111

⇒ $1E77_{16}$

Transport Layer

→ machine to machine process connection via ports

Transmission Control Protocol (TCP)

→ protocol to ensure data is transmitted and received in the correct order and without errors

1. establishes connection in a 3-Way Handshake:

   SYN, SYN-ACK, ACK

2. closes connection in a 4-Way Handshake:

   Fin, ACK, Fin, Ack

3. uses numbers to keep track of packages and receive confermation
   ↳ request to resubmitt if number never arrived

4. manages data flow to not overwhelme receiver (window size)

5. verifies package with a checksum

6. regulate data transmission rate and ensures efficient use of resources

Uso Datagram Protocol (UDP)

→ fast, unreliable, connectionless transmissions

1. it just sends data to the receiver without notification
2. fast delivery, but no error correction, no flow control, no connection establishing
3. little overhead
4. no guarenteed delivery
5. multicast and broadcast capable

Evolving the Protocols

→ TCP is used for websites and Transport Layer Security (TLS) is added for security, but is not integrated in TCP

→ Lacks some features like multiplexing, congestion control, connection migration

⇒ Now we could eighter upgrad TCP or build a new protocol next to TCP but based on UDP

⇒ QUIC

Quick UDP Internet Connection (QUIC)

→ new transport Layer Protocol that combines features of UDP & TCP and addresses its limitations

1. Built on UDP → low overhead, multicast support → fast
2. Combines crypto Handshake and Transport Layer
   ↳ 0-RTT zero round Trip Time connection
   ↳ instant data transmission
3. multiplex streams → simultanous data streams
4. Foreward Error Correction (FEC) → recover lost data
   ↳ no need for retransmission
5. Congestion control → manage load for effizient transmission (like TCP)
6. Built in encryption

Application Layer

→ Interface between users and network
It enables data exchange, sharing of resourses and other services like authentication, synchronisation, compression with different Protocols

Hypertext Transfer Protocol (HTTP)

→ communication between Web-server and browser by transfering data

Hypertext Transfer Protocol Secure (HTTPS)

→ encrypted HTTP with SSL/TLS

File Transfer Protocol (FTP)

→ file transfer between clients and server

Simple Mail Transfer Protocol (SMTP)

→ email transmission Protocol

Internet Message Access Protocol (IMAP)

→ email retrieval protocol to manage mails on a server, remote

Domain Name Systems (DNS)

→ IP Addres to domain name translator

World Wide Web

→ information system enabelig document and other resauces to be accessed over the Internet

Invented by Tim Berners-Lee at CERN

Representational State Transfer (REST)

→ architectural style to guide design and interaction in a distributed System

- It can be any identifyable entity it has a 2-part representation (text/html, application/json) ⇒ type /subtype
- URLs follow the Structure authority/path/fragment
  ↳ xy.com/example/#resource
- use of HTTP methods → GET, POST, PUT, DELETE...
- It is Stateless → client request has all needed data so that the server doesn't have to Store anything

Web of Things (WoT)

A "Thing" has a machine readable API Documentation:

- metadata → what is it
  - → who is the manufacturer
- Interaction → properties → what can it do
  - → actions → how can one do it (manipulate state)
  - → events → current state of the property
- hypermedia controls → info for user on how to use

Network Security

Consider when building a secure system:

- Design objective → what should the system do
- Threat Model → Assumptions about Attacks
- Policy → Rules to achive design objective
- Mechanism → Software / hardware to enforce Policy

Encryptions

Ceasar Cipher

→ shift the letters in the Alphabet by x

↳ only 25 possible shifts

↳ letter frequenzy stays the Same

Symmetric key Cryptography

→ Same key for Sender and receiver to encrypt / decrypt

Advanced Encryption Standard (AES)

→ Symmetric encryption Algorithm

↳ key size: 128, 192, 256 bits

↳ brute force would take longer than our universe exists

Diffie-Hellman key Exchange

→ Securely share key over unsecure Network

Process

- Both users agree on a large prime p and base g
- Each user chooses a secret value a & b and calculate a public value A & B with the formulas

  $A = g^a \mod p$ $\qquad$ $B = g^b \mod p$
- The users Exchange A & B
- Each user generats the same shared secret

  $S = B^a \mod p = A^b \mod p$

Asymmetric key Cryptography

→ public and private key encryption

Rivest - Shamir - Adleman (RSA)

→ asymmetric block cypher with key length= 1024/2048/4096 bits

↳ uses the fact that it's hard to factorice prime multiplication

Process

- Each use calculates private and public key
  - ↳ choose 2 large primes q, p and calculate $n = qp$
  - ↳ choose e, so that e is relative prime to $(p-1)(q-1)$
  - ↳ calculate d = inverse of e modulo $(p-1)(q-1)$

  ⇒ public key = (n,e) private key = (n,d)

- encrypt message $m \rightarrow c = m^e \bmod n$

  ↳ (n,e) = public key of receiver

- decrypt message $c \rightarrow m = c^d \bmod n$

  ↳ (n,d) = private key of receiver

TLS with Diffie Hellman

- 1. agree on parameters
- 2. establish shared key
- 3. send encrypted data

→ Vulnrable to man in the middle
  - ↳ Solution → certificate Authority (CA)
    - ↳ bind server public key to server name
      - ↳ verifiable

TLS with RSA

- 1. agree on parameters
- 2. exchange keys ← Calculate secret symmetric key
- 3. send messages

⇒ Symmetric is faster so asymmetric is used to generate symmetric key

Wireless Communication & Constrained Networks

Extremely Resource constraint devices

⇒ battoyless object collect data and send it via Bluetooth low energy (BLE)

↳ uses solar to generate energy

⇒ more time asleep, then awake

↳ saves energy

LoRa WAN Architecture

→ Sender decides when to send and has a fixed reply time (1, 2 s) after that, it goes to sleep again

⇒ uses gateway to listen to devizes

↳ 48 simulteneous connections

↳ gateway sends to LoRa WAN server → sends to device

Range:

theoretical → 850 km

meassured → 832 km

Deployment

→ use a "gateway - provider" (swisscom)

→ build your own gateways (open source)

Wireless Communication

→ the higher the data Rate, the higher the power consumption

→ Ranging from 802.15.4 - 82.11n

↳ z.B Zigbee

↳ low Rate wireless personal area Network

↳ WLAN

RFC 7228

→ Terminology for Constrained-Node Networks

- **Constrained Node:** Devices that have signifiant limitations on processing power, memory and power availability, etc. (z.B RFID-Tag)
- **Constrained Networks:** Network of constrained Nodes

⇒ Constrained Devices can be categorized in classes, based on their RAM and flash memory capability

- **Class 0:** not able to run standard Internet Protocols
  ↳ only minimalistic protocol to connect to Internet
  - < 10 KiB of RAM
  - < 100 KiB of flash Memory

Class 1: most can connect to Internet
↳ not enough memory to run HTTP & TLS

- ~ 10 KiB of RAM
- ~ 100 kiB of flash Memory

Class 2: close to Internet Nodes like Smartphones etc.
↳ for demanding IoT Applications

- ~ 50 kiB of RAM
- ~ 250 kiB of flash Memory

IEE 802.15.4

→ standard for LoWPANs

- Low Power consumption
- Low Data Rate
- Short Range
- Simple & cheap
- Security through AES - 128 encryption
- supports P2P and star topologies

6LoWAN

→ IPv6 for Low-Power Wireless PAN

↳ The IP bases Internet of Things

challenges for IP-based & Lossy wireless networks

- Duty-cycle → assumption of IP is, that it's always connected
- Reliability → standard Internet Protocols are not optimized for Low-Power Lossy wireless networks
- Low Data Rate and small frame size
- No native multicast support
- Mesh topologies → wireless radio benefits from multi-Hop
  → Ip routing not easy appliable to such networks

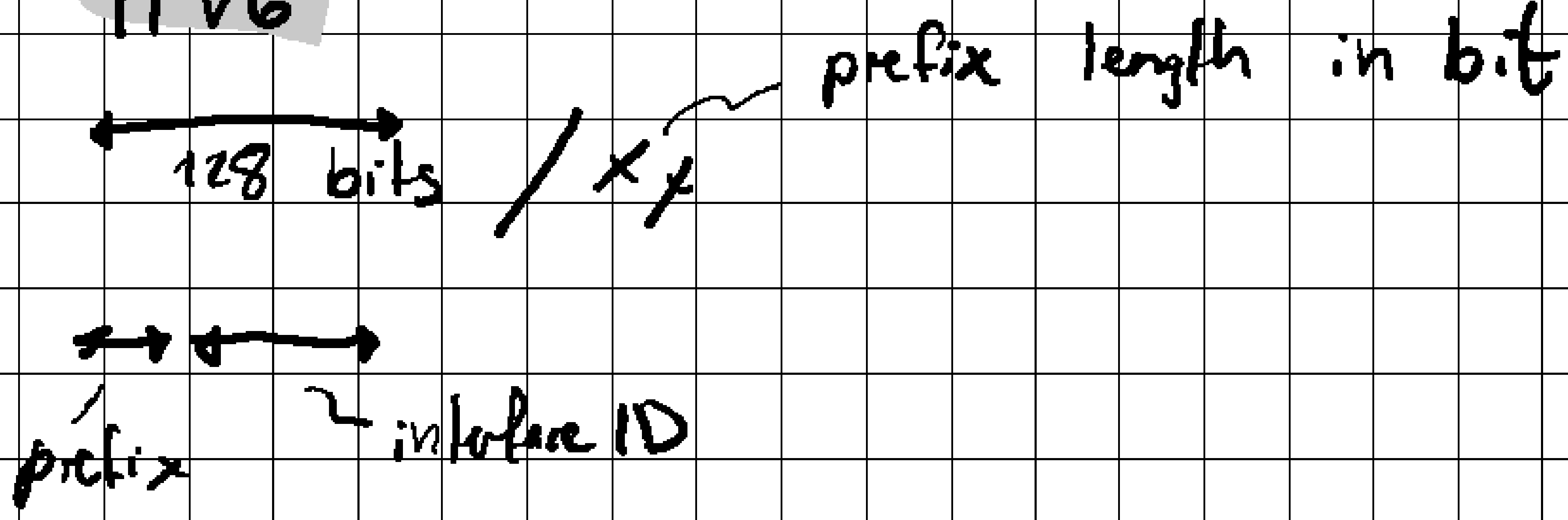
6LoWPAN Architectue

- Stub network → collection of nodes that share a common IPv6 prefix

Types of LoWPAN

- Simple : one edge Router that connects all devices with eg the Internet
- extended : mesh like Network with multiple edge Routers → multiple possible package delivery paths
- Ad-hoc : decentralized and flexible no edge Router → all devices can connect → self formating

IPv6



compression

- leading 0s can be ignored
- 0-block can be ignored (only once)

⇒ 2001:300a:0000:0010:0000:0000:0000:0001 /64

= 2001:300a:0:10::1 /64

Neighbor Discovery

- Edge Router mantain a whiteboard (List of Nodes in 6LoWPAN)
  ↳ extended 6LoWPAN : whiteboard is synchronized with edgerouters
- Devices send Router Solicitation (RS) after wakeup
  ↳ Router responds with Router Advertisement (RA)

↳ This is Node Registration and Node Confirmation

6LoWPAN Routing

→ directed graph towards root (DAG - directed acyclid grap)

→ rank characterizes distance (avoids loops)

→ Leaf nodes have parents (one preferred parent)

DAG is built in 2 Phases

1. root to leafes → establishing ranks
2. leafes to root → propagate addresses using DAG

Constrained Application Protocol (CoAP)

→ targets constraints of TCP in low power wireless networks

→ efficient REST Uniform interface on top of UDP

↳ lightweight security

↳ multicast & Observe

↳ easy proxy from/into HTTP

⇒ Things watching Things

CoAP extends REST uniform interface to permit the obervation of resources

↳ it receives update on changes

↳ better than constant HTTP polling

Reliability

→ CoAP sends 4 messages to ensure reliability

→ CON - confirmable → reliable message

→ NON - Non confirmable → "fire and forget"

→ ACK - Acknowledgement → confirm CON

→ RST - Reset → sent when CON/NON is missing context

Constrained RESTful Environments (CORE)

→ main framework used for constrained devices

Components:

- CoAP
- Resource Directory (RD) → way to register resource & service
- Link format → uniform format to advertise resource
  - ↳ hypermedia driven discovery

Future of Networking

end-to-end arguments

↳ certain functions like ecc / encryption should be implemented at the endpoint of a network system, rather then intermediate Nodes, bcause of the applications contextual understanding of its own requirements.

Centeralize Personal Data

→ save personal data on a provider

↳ give selective permission to services to acess but not store

↳ give selective permission to services to change data

→ ability to revoke or change acess permissions

Centeralize ID

↳ Login data etc.