Verteilte Sensornetzwerke

Mit Datenaggregation und Sensorfusion

PD Stefan Bosse

Universität Bremen - FB Mathematik und Informatik

Kommunikationstechnologien und Protokolle im IoT

Wie kann praktisch Kommunikation von Sensorknoten untereinander und mit Berechnungsknoten stattfinden?

Welche Protokolle können verwendet werden?

Es sollen das Internet der Dinge (IoT) betrachtet werden, d.h. Einbindung von Sensorknoten in das Internet

Welche physikalischen Übertragungstechnologien sind nutzbar?

Kommunikationtechnologien

Klassifizierung

Drahtlose Kommunikation

Drahtlose Kommunikation ist durch 1:N Kommunikation (Broadcasting) gekennzeichnet, d.h., i.A. findet keine direkte Kommunikation mit anderen Knoten (Peer-to-Peer) statt. Z.B. Radiowellen können in einem Umkreis von vielen Knoten empfangen werden.

Drahtgebundene Kommunikation

Hier findet immer gerichtete Kommunikation zwischen zwei Knoten statt. Aber: Auch Bussysteme sind 1:1 Kommunikationsnetze, wo auch Broadcasting stattfindet!

Physikalischen Medien

Drahtlose Kommunikation

Radiowellen (Elektromagnetische Felder) in unterschiedlichen Frequenzbereichen, Lichtwellen, quantenmechanische Interaktion (Quantencomputer!), magnetische Felder,

Drahtgebundene Kommunikation

Elektrische Leitungen mit elektrischen Strom, aber auch optische Leitungen wie Lichtwellenleiter, oder auch mittels Laserdioden gerichtete Lichtwellenausstrahlung, geht auch mit Radiowellen und Antenntenarrays (MIMO) → 5G Netze!!

Technologien

WLAN

Radiowellen im frequenzbereich 2.4GHz und 5GHz, Reichweite ca. 100m, über Zeit- und Kanalmultiplexing können 1:N Netzwerke aufgebaut werden;

Bluetooth

Radiowellen im frequenzbereich 2.4GHz, Reichweite ca. 1-10m, i.A. nur durch Kanalmultiplexing können 1:N Übertragungskanäle ermöglicht werden

RFID

Radiowellen im Frequenzbereich 1MHZ/14MHz, Reichweite ca. 10cm (durch gleichzeitige Energieübertragung limitiert) , nur 1:1 Kommunikation

UHF

Einfache Radiowellenübertragung im Frequenzbereich 400 MHz, Reichweite ca. 10m, nur 1:1 Übertragung (ansonsten Interferenz)

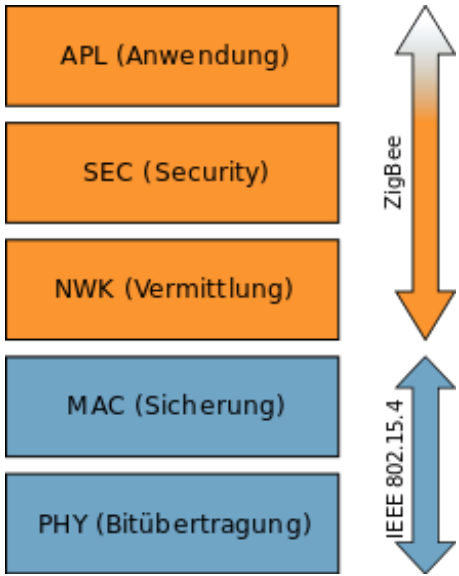
LoRa

LoRa ist eine Funktechnologie, die das Senden kurzer Datenpakete über eine große Reichweite (in der Regel einige Kilometer) bei einer UHF Frequenz von 915 MHz ermöglicht. Long Range Wide Area Network (LoRaWAN) ist ein Low-Power-Wireless-Netzprotokoll auf der Ebene der Vermittlungsschicht. Endgeräte und Gateways im LoRaWAN nutzen ein proprietäres und patentiertes Übertragungsverfahren.

ZigBee

Zigbee ist ein Funkstandard der unterschiedliche Geräte auf kurzen Strecken miteinander verbindet – die Funkreichweite beträgt je nach Leistung und baulicher Umgebung zwischen 10 und 20 Metern, unter idealen Bedingungen auch bis zu 100 Meter. Im Gegensatz zu bekannteren Standards wie WLAN oder Bluetooth zeichnet sich Zigbee durch Energieeffizienz, Einfachheit in der Anwendung und hohe Flexibilität aus. Die Endgeräte verwalten das Netzwerk autark. Das bedeutet, die Geräte treten dem Netzwerk nach einer ersten Einbindung selbständig bei, werden gefunden und kommunizieren darüber. Bei ZigBee ist das 2,4-GHz-Frquenzband unterteilt in 16 Kanäle mit einem Kanalraster von 5 MHz.

Protokollschichten



[https://de.wikipedia.org/wiki/ZigBee]

Abb. 1. Verschiedene Protokollschichten in Sensornetzwerken (ähnlich ISO-OSI Schichtenmodell)

Netzwerke und Protokolle

- Häufig werden Sensorknoten über vorhandene Netzwerke eingebunden
  - WLAN (Intra- und Internet)
  - 4G/5G Mobiles Internet
- Dann werden gängige Kommunikationsprotokolle verwendet, gegeben durch das ISO Schichtenmodell der Netzwerkkommunikation und der physikalischen Technologie:
  - User Datagram Protocol (UDP)
  - Transmission Control Protocol (TCP)
  - Hypertext Transfer Protocol (HTTP/HTTPS)
  - WebSockets
  - Web Real Time Communication (WebRTC)

Man unterscheidet:

Klienten-Server Kommunikation

Es gibt ein oder mehrere Klientenknoten die mit einem Serverknoten kommunizieren (1:N Master-Slave Architektur). Die Klientenknoten können nicht untereinander kommunizieren. Die Kommunikation kann unidirektional, bidirektional, oder semibidirektional sein (d.h. nur der Klient kann einen Kommunikationskanal aufbauen).

Punkt-zu-Punkt Kommunikation

Ein Knoten A kann direkt mit einem Knoten B bidirektional kommunizieren.

Der Kommunikationsendpunkt muss durch den Kommunikationsstartpunkt erreichbar und adressierbar sein! D.h. der Endpunkt muss öffentlich sichtbar sein wenn sich *A* und *B* nicht kennen.

Hier fangen jetzt die Problem an:

Welches Adressierungsschema soll für die Knoten verwendet werden? Gibt es öffentliche oder private Adressen? Wann und wo müssen die Adressen eindeutig sein (keine Dopplung)?

Sind zwei Knoten *A* und *B* nicht direkt (in einem Netzwerk) verbunden wird Routing (Pfadfindung) benötigt

IP Kommunikation

- Internet Kommunikation mit dem Internet Protocol (IP)
  - Netzwerkknoten benötigen eindeutige IP Adresse

IP-v4

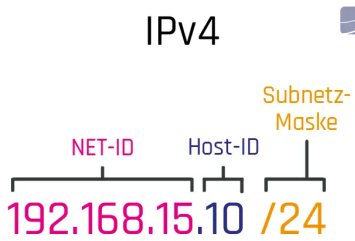
- 32 Bit Adresse
- 4 Oktetts (8 Bit, vorzeichenlose Zahl)
- Vorderer Teil: Netzwerk (Network)
- Hinterer Teil: Gerät (Host)
- Benötigt NAT
- Geräteadresse: Zugeteilt pro Netzwerkgerät/Rechner

IP-v6

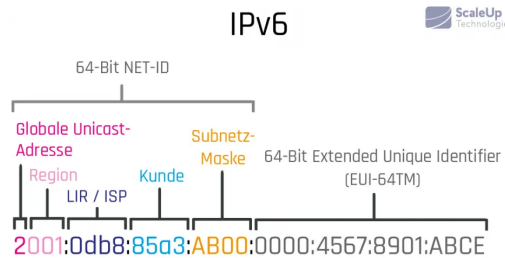
- 128 Bit Adresse
- 8 × 16 Bit Oktetts (variierende Bedeutung)
- Netzwerk (Site) und Geräteadressen (Device) gemischt
- Geräteadresse: Netzwerkgerät (MAC) ID vom Hersteller vergeben
- Es gibt Aggregatoren

IP Kommunikation

IP-v4



IP-v6



Das Internet ist ein Netz aus Netzen (hierarchisch und verteilt)

- Netzwerkadressen werden in räumlichen Domains zusammengefasst (IP4)
- Geräte benötigen daher lokale IP Adressen (u.U. wechselnd), Auflösung über Namen und Domain Name Service (DNS)
- IP6 ordnet Geräten eindeutige Adressen zu (mobil)

Die Anzahl der Geräte (gerade durch IoT und Sensornetzwerke) übersteigt die tatsächlich möglichen IP4 Adressen. IP6 ist derzeit davon noch nicht betroffen.

IP Kommunikation

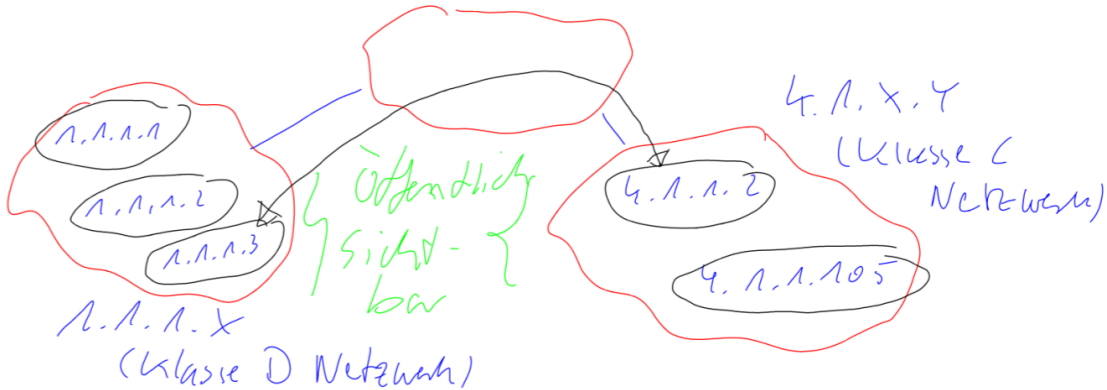


Abb. 2. Domainen im Internet und Netze aus Netzen

IP Kommunikation

- Private Netzwerke und die immer größer werdende Anzahl von Geräten erfordern bei IP-v4 NAT (nicht bei IP-v6)
- NAT: Network Address Translation

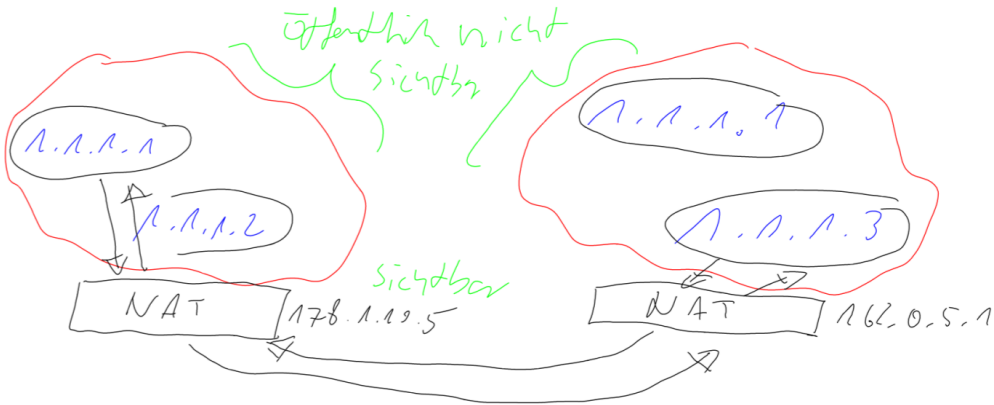


Abb. 3. Kommunikation zwischen privaten IP Netzwerken über NAT

Network Address Translation

- Zwischen einem Rechner A und B befindet sich ein NAT Gerät
- Dieses NAT Gerät ist nach außen durch eine IP Adresse $ipaddr_{\text{pub}}$ öffentlich sichtbar (wobei es auch hierarchische Netze mit mehreren NAT Ebenen geben kann)
- Rechner in dem NAT Netzwerk haben verschiedene private IP Adressen $ipaddr_{\text{prv,i}}$.
- Wenn nun Kommunikation unter Verwendung der öffentlichen IP Adresse stattfindet muss das NAT die Nachrichten verändern (Austausch privater Adresse mit öffentlicher, $ipaddr_{\text{prv,i}} \rightarrow ipaddr_{\text{pub}}$)
- Von außen empfangene Nachrichten müssen ebenfalls wieder umgerechnet werden ($ipaddr_{\text{pub}} \rightarrow ipaddr_{\text{prv,i}}$) und dem richtigen Rechner zugeführt werden!

Daher können in privaten Netzwerken keine öffentlich sichtbaren Services angeboten werden (also z.B. HTTP Server)!

- Neben der Adressumrechnung sind weitere Umrechnungen über Zuordnungstabellen erforderlich, z.B. ein Kommunikationsport *port*
  - Port-preserving NATs rechnen nur die Adressen um, aber nicht die Portnummern
  - Not-Port-preserving NATs rechnen auch die Portnummer um (in mobilen Netzwerken immer der Fall, da dort auch verschiedene Geräte die gleiche IP Adresse besitzen können)
- Man unterscheidet:
  - Symmetrische NATs (Not-Port-preserving NATs); jeder ausgehende Verbindung ist unterscheidbar, d.h., es gibt nicht zwei gleichzeitige ausgehende Verbindungen (auch zu unterschiedlichen Servern) mit der gleichen Portnnumer
  - Asymmetrische NATs (Eventuell Port-preserving)

Symmetrische NAT Netzwerke sind eine echte Hürde für direkte 1:1 Kommunikation von Sensorknoten (und sonstigen Rechnern), vor allem wenn A und B durch symmetrische NATs getrennt sind

Eine Methode um 1:1 Verbindungen durch NATs herzustellen (zumindest via UDP) ist das sogennante Hole-Punshing Verfahren

IPv6 Kommunikation

Effizienteres Routing [https://www.scaleuptech.com/blog/ipv4-vs-ipv6]

IPv6 reduziert die Größe von Routing-Tabellen und macht das Routing effizienter und hierarchischer.

Effizientere Paketverarbeitung

Der vereinfachte Paket-Header von IPv6 macht die Paketverarbeitung effizienter.

Gerichtete Datenflüsse

Positiv auf die Netzwerkbandbreite wirkt sich hingegen die Multicast-Unterstützung von IPv6 aus. Im Gegensatz zu Broadcast können mit Multicast bandbreitenintensive Datenpaketflüsse (wie Multimedia-Streams) gleichzeitig an mehrere Ziele gesendet werden

Vereinfachte Netzwerkkonfiguration

Die automatische Adresskonfiguration (Adresszuweisung) ist wie bereits erwähnt in IPv6 schon integriert. Ein Router sendet das Präfix der lokalen Verbindung in seinen Routerankündigungen. Ein Host kann seine eigene IP-Adresse generieren, indem er seine MAC-Adresse (Link-Layer), die in das 64-Bit-Format Extended Universal Identifier (EUI) konvertiert wurde, an die 64 Bit des lokalen Verbindungspräfixes anfügt.

IPv6 Kommunikation

[https://ip.engineering/ipv6-subnetting-overview-and-case-study/]

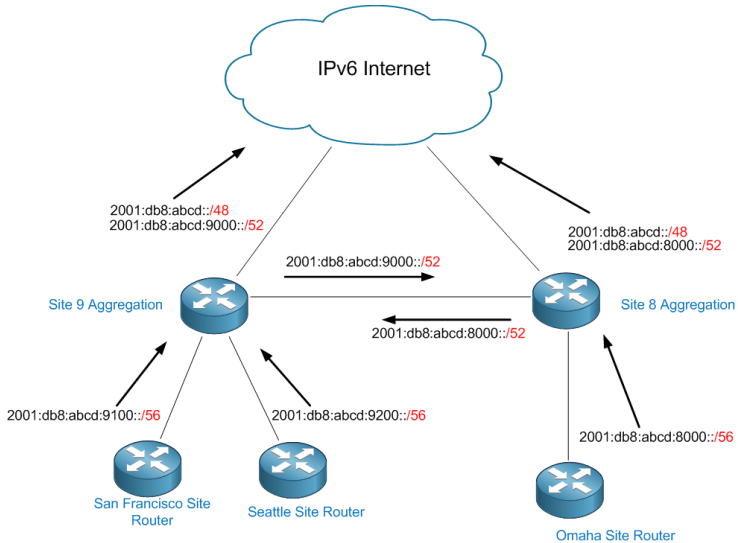


Abb. 4. Neben Endroutern gibt es Aggregatoren (beide Sites), die eine hierarchische Graphenstruktur aufbauen. Die Abbildung zeigt die Adresskodierung der einzelnen Netzwerkknoten

IPv6 Kommunikation

IPv6 ist attraktiv für IoT Netzwerke: Einfache Konfiguration (via HW MAC Adressen), Nachbarschaftssuche via dem SEND Protokoll, kein NAT, jedes Geräte ist praktisch global sichtbar und erreichbar.

IPv6 Kommunikation

IPv6 ist attraktiv für IoT Netzwerke: Einfache Konfiguration (via HW MAC Adressen), Nachbarschaftssuche via dem SEND Protokoll, kein NAT, jedes Geräte ist praktisch global sichtbar und erreichbar.

Aber auf IPv6 basiertes IoT hat auch Nachteile: Adressen sind menschlich nicht mehr "merkbar" und transferrierbar, und entgegen weit verbreiteter Behauptung dass IPv& mehr Sicherheit bietet kann genau das Gegenteil eintreten.

IPv6 Kommunikation

Nach Said Jawad Saidi (10 Jun 2022, APNIC) werden folgende Nachteile (Sicherheit und Privatsphäre) von IPv6-IoT untersucht und genannt:

[https://blog.apnic.net/2022/06/10/iot-devices-endanger-ipv6-privacy/]

1. Die EUI-64-Adressierung beeinträchtigt die Privatsphäre der Benutzer und wird daher durch IPv6-Datenschutzerweiterungen überlagert.
2. **Ein einzelnes mit dem Internet verbundenes Gerät, das EUI-64 verwendet, kann die Verfolgung anderer Geräte ermöglichen, selbst wenn alle anderen Geräte Datenschutzmechanismen wie IPv6-Datenschutzerweiterungen verwenden und der ISP Präfixrotation verwendet**.
3. 14,4 Millionen Geräte verwenden das alte EUI-64-Adressierungsschema, wodurch 19% der Teilnehmer anfällig für Datenschutzverluste sind.
4. Mehr als 2 Millionen Teilnehmeranschlüsse sind von Datenschutzlecks bei CDNs betroffen. IoT-Geräte spielen eine große Rolle bei EUI-64-Datenschutzverlusten.

ISP: Internet Service Provider, CDN: Content Deliver Network

IP Kommunikation

Transportprotokolle

UDP

User Datagram Protocol. Es wird für die Übertragung einzelner unabhängiger Datenpakete von einem Senderknoten A das Tupel $\langle ipaddr_B, port_B \rangle$ des Empfängerknoten B benötigt. Es wird Routing benötigt. Der Sender muss aber nicht auffindbar sein! Der Port ist einem Verarbeitungsprozess zugeordnet.

TCP

Transmission Control Protocol. Es findet ein Verbindungsaufbau statt mit zwei unidirektionalen Kommunikationskanälen zwischen A und B. Sowohl das Tupel $\langle ipaddr_A, port_A \rangle$ als auch das Tupel $\langle ipaddr_B, port_B \rangle$ müssen bekannt sein

Anwendungsprotokolle

HTTP

Hypertext Transfer Protocol. Benutzt TCP, nur 1:N Serverkommunikation, wobei auch jeder Sensorknoten mit einem $\langle ipaddr, port \rangle$ Tupel einen Service anbieten kann. Alle HTTP Anfragen sind bidirektional (GET/PUT)

WebSockets

Benutzt initial HTTP(S) und dann TCP. Reine bidirektionale Datenübertragung (Datenkanal)

WebRTC

Benutzt nur UDP. Echtzeitfähige Datenstromübertragung (Audio, Video, Daten)

HTTP

GET

Es können nur Daten vom Server (*B*) zum Klienten (*A*) auf Anfrage gesendet werden. Eine Parametrisierung einer Anfrage ist durch URL Parameter eingeschränkt möglich.

PUT/POST

Es können Daten von *A* nach *B* und dann von *B* nach *A* gesendet werden,

Node B

```
hs=http:new()
hs:service('a:9099',
  function (url,address,params,body)
    data = JSON.decode(body)
    return JSON.encode({3,2,1})
  end
)
```

Node A

```
hs=http:new()
hs:put('a:9099',JSON.encode({1,2,3}),
       function (text,err)
         data = JSON.decode(text)
       end)
```

[Zoltan Horvat/Israel Cidon]

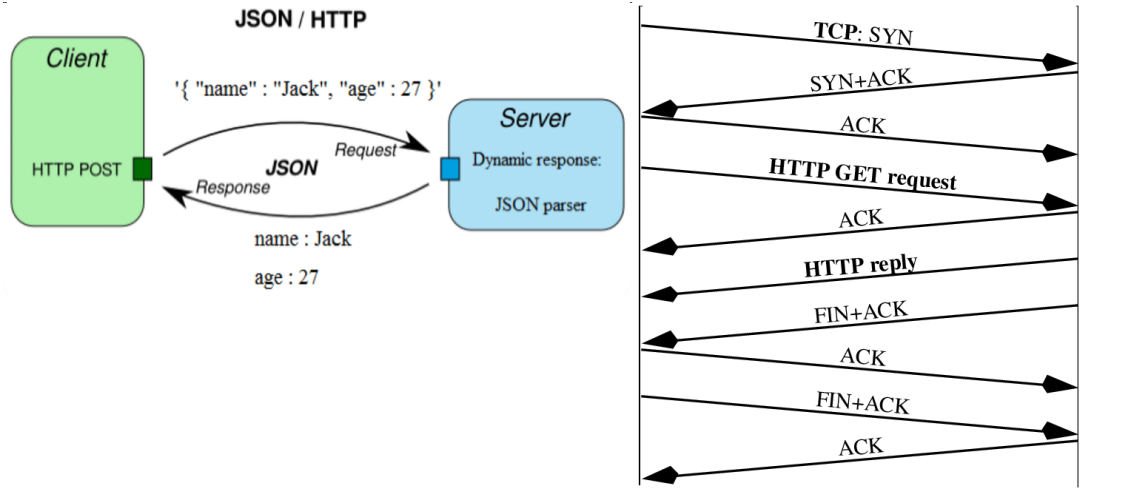


Abb. 5. (Links) HTTP POSt Anfrage (Rechts) TCP Kommunikation für eine HTTP GET Anfrage

WebSockets

- WebSockets werden ebenfalls primär für die Klienten-Server Kommunikation verwendet, sind aber bidirektional.
  - Datenübertragung mit HTTP GET/PUT Anfragen können nur vom Klienten initiiert werden, nicht vom Server
  - WebSocket Datenübertragung kann von beiden Seiten initiativ erfolgen (aber Verbindungsaufbau nur durch Klienten)
- Verbindungsaufbau eines Sockets über HTTP als eine Art "Upgrade" der Verbindung
- Übertragung von Daten mit WebSockets ist zuverlässig

[bloggeek.me/webrtc-vs-websockets]

Abb. 6. Vergleich WebSockets mit WebRTC Datenkanälen

Da der Verbindungsaufbau immer mit einem öffentlichen Server (ggfs. nur im Intranet) stattfindet kann ein Klientenknoten sich in einem privaten Netzwerk befinden (nicht öffentlich sichtbar).

Auch wenn zwei Knoten *A* und *B* über WebSockets kommunizieren wollen wird dieses nur über den öffentlichen Server stattfinden und keiner der Knoten *A* und *B* müssen öffentlich sichtbar sein → Kommunikation über Relaisknoten *C*

Eine direkte Kommunikation zwischen zwei Sensorknoten über WebSockets geht nur dann wenn wenigstens ein Knoten direkt erreichbar ist (also ohne NAT) und einen HTTP Service anbietet. Aber im WEB Browser kann kein Service angeboten werden!

[Renuka B S,2020]

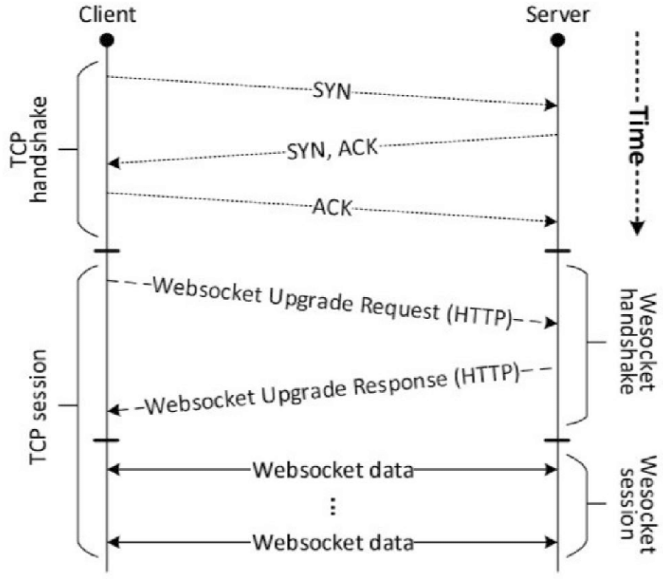


Abb. 7. Initierung von WebSocket Kanälen über eine HTTP Anfrage und schließlich direkte Kommunikation via TCP

WebRTC

- WebRTC stellt echtzeitfähige Datenstromkommunikation via UDP zur Verfügung
- Übertragung von Daten mit WebRTC/UDP ist daher nicht grundsätzlich zuverlässig (obliegt der eigenen Anwendungsschicht)

[hpbn.co/webrtc]

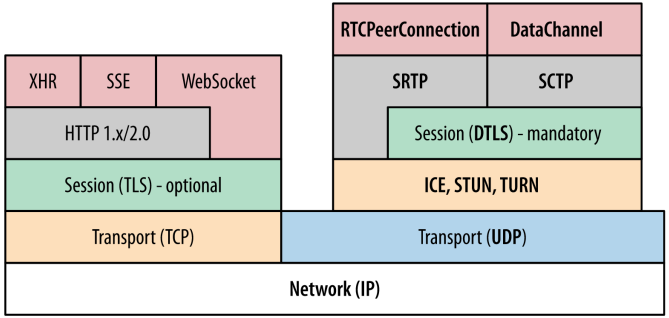


Abb. 8. Vergleich der Protokollarchitektur von WebSockets und WebRTC

- Daten: primär Audio und Videodatenströme, sekundär generische Datenkanäle
- Neben der strom- und kanalbasierten Kommunikation bietet die WebRTC Schicht auch den Aufbau und die Einbeziehung von externen Services durch den ICE Agenten

Es gibt zwei Fragestellung:

1. Wie können zwei Endgeräte (Prozesse) A und B protokollarisch und technologisch mit einander kommunizieren (Verbindung), d.h. wie finden sie sich physisch
2. Welche Knoten sollen Verbindungen aufbauen und Daten austauschen (Auktionsserver! Biete Sensor, Suche Sensor), d.h. wie finden sie sich logisch/funktional

[hpbn.co/webrtc]

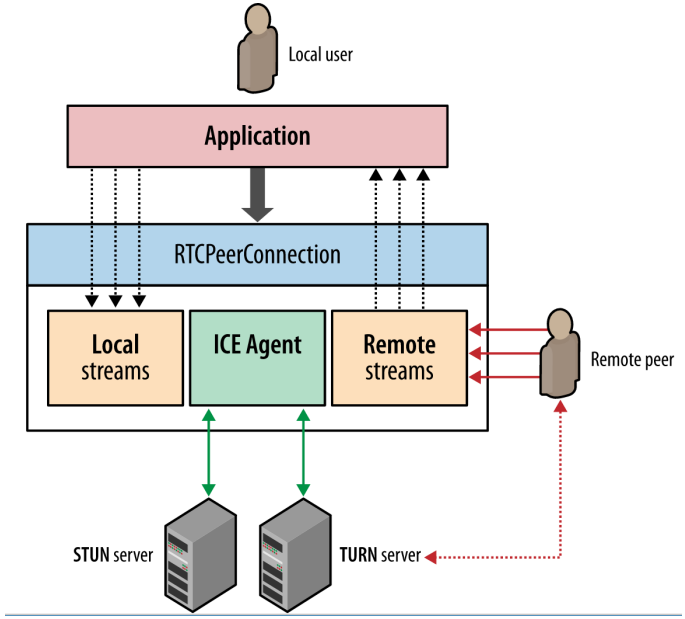


Abb. 9. RTCPeerconnection Architektur

NAT und UDP Hole Punching

https://bford.info/pub/net/p2pnat

- Das Problem: Das öffentlich sichtbare Tupel ⟨*ipaddr*,*ipport*⟩ eines Peer Ports muss auch für eingehende Datenströme verwendet werden.
- Wenn ein Klient Daten über einen Port mit NAT sendet merkt sich der NAT das Mapping und eine Rücksendung an den Senderport wird an das richtige Gerät und den Prozess vermittelt.
- Aber wie soll Klient A mit Klient B Kontakt aufnehmen wenn zuvor Klient B keinen Kontakt mit A aufgenommen hat? Es muss versucht werden den NAT von B "durchzuschalten"

Dazu werden (leere Ping) UDP Pakete zwischen den Klienten gegenseitig zugesendet.

STUN, TURN

https://help.estos.com/help/de-DE/procall/7/erestunservice/dokumentation/htm/IDD_FUNCTIONALITY.htm

Signaling Server

[help.estos.com]

Signaling Server dienen zum indirekten Austausch von Daten zwischen zwei Klienten. Dies kann ein Dienst sein, der von beiden Klienten erreichbar ist oder auch mehrere Dienste die mittels Zusammenschluss miteinander verbunden sind

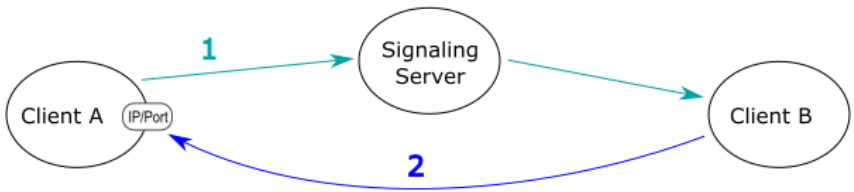


Abb. 10. Klient A ist direkt erreichbar. Klient B kann Daten direkt an Klient A senden

STUN, TURN

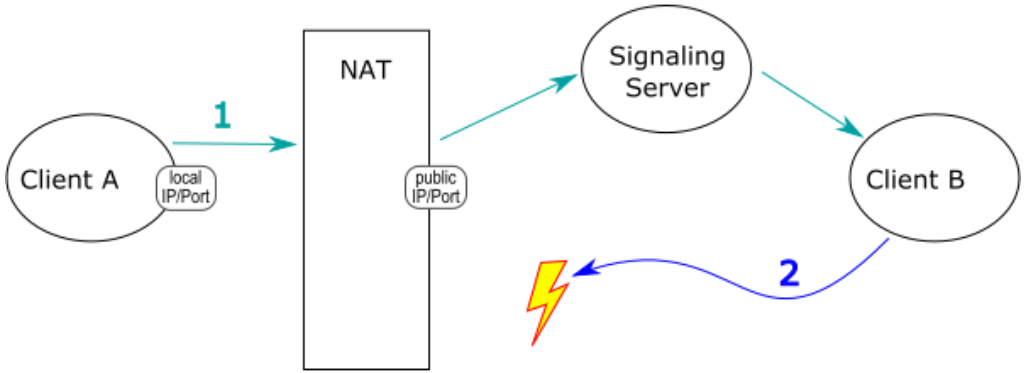


Abb. 11. Erfolgloser Verbindungsaufbau über einen NAT-Router hinweg.

STUN - Session Traversal Utilities for NAT (RFC5389)

Dieses Protokoll ermöglicht es einem Klienten in einem lokalen Netzwerk (LAN), seine eigene, öffentliche IPv4-Adresse zu ermitteln. Der rufende Client im LAN kann auf diese Weise dem angerufenen Klienten außerhalb des LAN mitteilen, welche IPv4-Adresse (und Portnummer) verwendet werden kann um eine direkte Kommunikation mit ihm zu ermöglichen ("Peer-to-Peer" Verbindung).

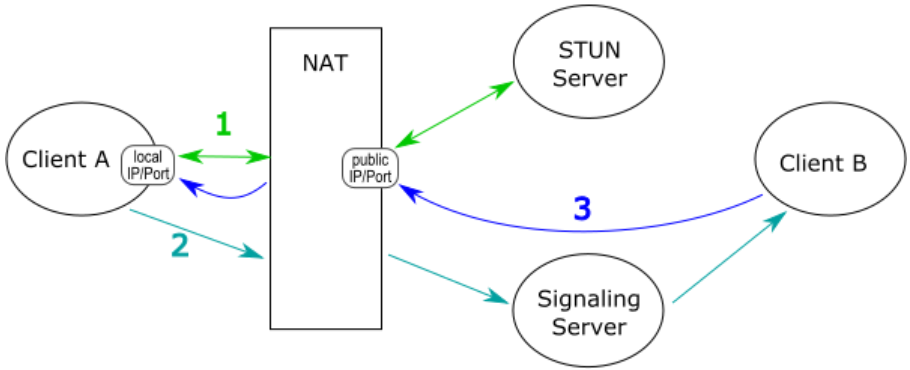


Abb. 12. Erfolgreiche Kommunikation unter Zuhilfenahme eines STUN-Servers.

STUN, TURN

TURN - Traversal Using Relays around NAT (RFC5766)

Ein Server im Internet, der das TURN-Protokoll implementiert, ermöglicht es zwei Klienten, Daten ohne eine direkte Verbindung auszutauschen ("Relais Server"). Dies wird notwendig, wenn es keine Möglichkeit gibt, eine direkte Klient-zu-Klient-Verbindung aufzubauen.

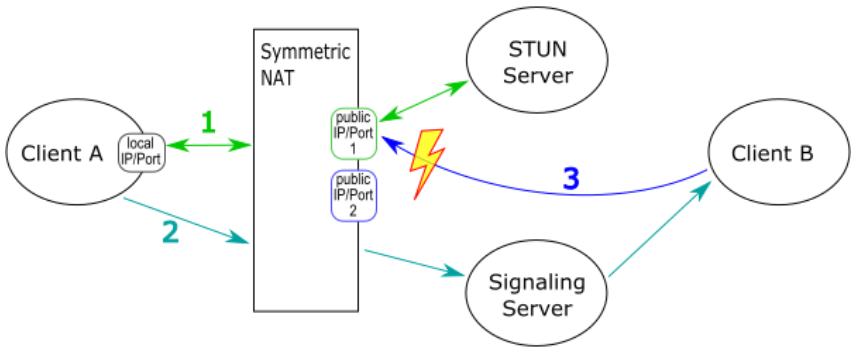


Abb. 13. Erfolgloser Kommunikationsversuch über ein "Symmetric NAT".

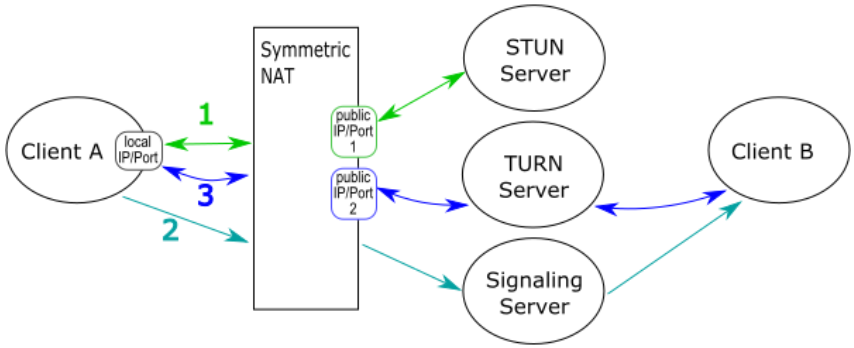


Abb. 14. Erfolgreicher Kommunikationsversuch über ein "Symmetric NAT" durch Nutzung eines TURN-Servers.

ICE - Interactive Connectivity Establishment (RFC5245)

Zwei Klienten können die mit Hilfe von STUN und TURN ermittelten Verbindungsinformationen (und andere Daten) mit Hilfe des ICE Protokolls austauschen. Die Übermittlung der Informationen muss dabei über einen eigenen Dienst erfolgen, einen sog. "Signaling Server". Dieser Dienst muss von beiden Klienten erreichbar sein.

Content Deliver Networks

Auch Sensornetzwerke sind CDN und können von einer dezentralen verteilten Architektur profitieren.

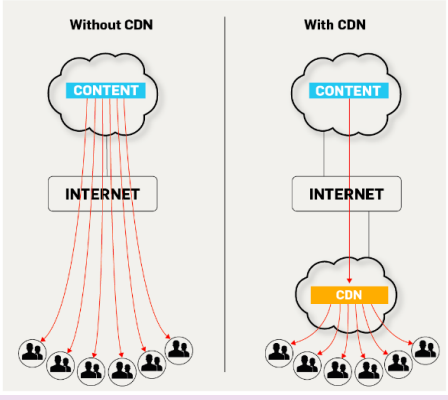


Abb. 15. CDN versa nicht CDN

[https://www.globaldots.com/resources/blog/content-delivery-network-explained/]

WEB Browser

- Im WEB Browser (als Sensor-, Aggregations-, oder Anwendundgsknoten) stehen zur Verfügung:
  - HTTP GET/PUT
  - WebSockets
  - WebRTC mit Peer Services
- Aber: Berechnung sollte in separate Threads (WEB Worker) ausgelagert werden. In einem WEB Worker stehen dann nur noch zur Verfügung:
  - HTTP GET/PUT
  - WebSockets
- Auch unsere Lua VM im WEB Browser wird in einem separaten Worker Threadprozess ausgeführt (unabhängig vom Hauptthread der nur für den Zugriff und Veränderung des DOM zuständig ist)

Vernetzung und Kommunikation in LuaOS

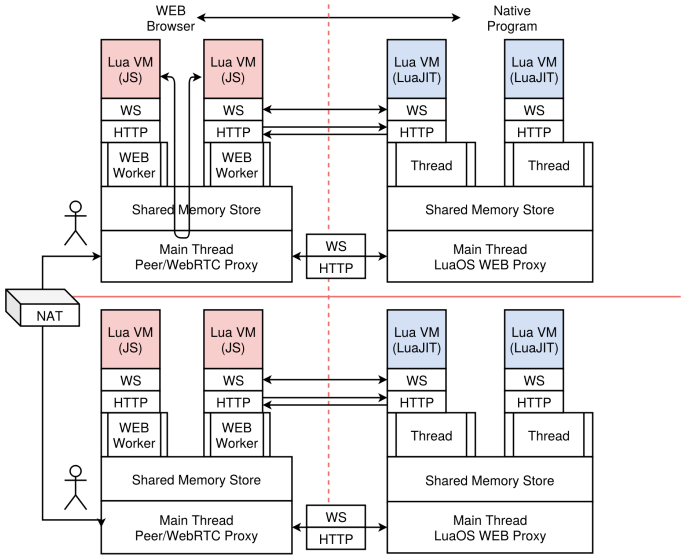
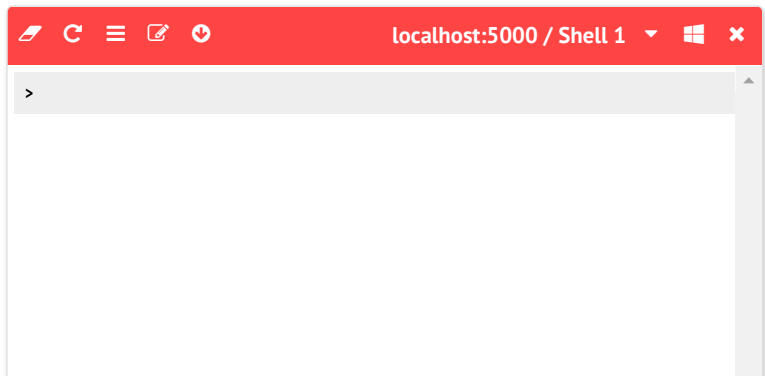


Abb. 16. Verschiedene Arten der Kommunikation und Vernetzung von Sensorknoten mit Lua VM innerhalb und außerhalb von NAT Netzwerken


Shells
Editors
Monitors
LuaOS NET WEB
Not connected!


localhost:5000 / Shell 1
>

Vertiefung

WebRTC

https://hpbn.co/webrtc

NAT/STUN

https://www.net.in.tum.de/fileadmin/TUM/teaching/masterkurs_rechnernetze/ws