ingenieur
wissenschaften
htw saar

Hochschule für
Technik und Wirtschaft
des Saarlandes
University of
Applied Sciences

Seminararbeit

an der Hochschule für Technik und Wirtschaft des Saarlandes

im Studiengang Kommunikationsinformatik

der Fakultät für Ingenieurwissenschaften

Video- und Audiokommunikationsplattformen

vorgelegt von

Matthias Riegler
Pascal Sprenger
Kai Schultz
Benjamin Hesser
Matthias Langenfeld

betreut und begutachtet von

Prof. Dr. Horst Wieker
M.Sc. Jonas Vogt

Saarbrücken, 7. August 2020

Inhaltsverzeichnis

Einleitung

Gerade in der heutigen Zeit, bedingt durch die weltweite Corona Pandemie, ist die Nutzung und der Einsatz von Audio- und Videokonferenzen sowie Streamingdiensten gestiegen. Im Besonderen profitieren wir von der Möglichkeit, soziale Kontakte zu pflegen und Zeit miteinander zu verbringen, obwohl wir Distanz zueinander halten müssen. Audio- und Videokonferenzsystem tragen dazu bei, auch in diesen Zeiten produktiv arbeiten zu können. Im Rahmen dieser Ausarbeitung beschäftigen wir uns mit dem Aufbau und der Technik von diesen Systemen. Außerdem betrachten wir Video Streaming in den Bereichen On-Demand und Livestreaming.

1 Video- und Audiokonferenzen

1.1 Architektur

Im Bereich der Architektur von Audio- und Videokonferenzsystemen kommen hauptsächlich zwei Architekturen zum Einsatz. Dabei handelt es sich zum einen um Peer-to-Peer, zum anderen um die Client Server Architektur.

1.1.1 Peer-to-Peer

Bei der `Peer-to-Peer` Architektur handelt es sich um eine dezentrale Netzwerkstruktur, bei der alle Teilnehmer gleichberechtigt sind. Diese Teilnehmer strukturieren sich dabei vollkommen selbstständig und können direkt untereinander kommunizieren.

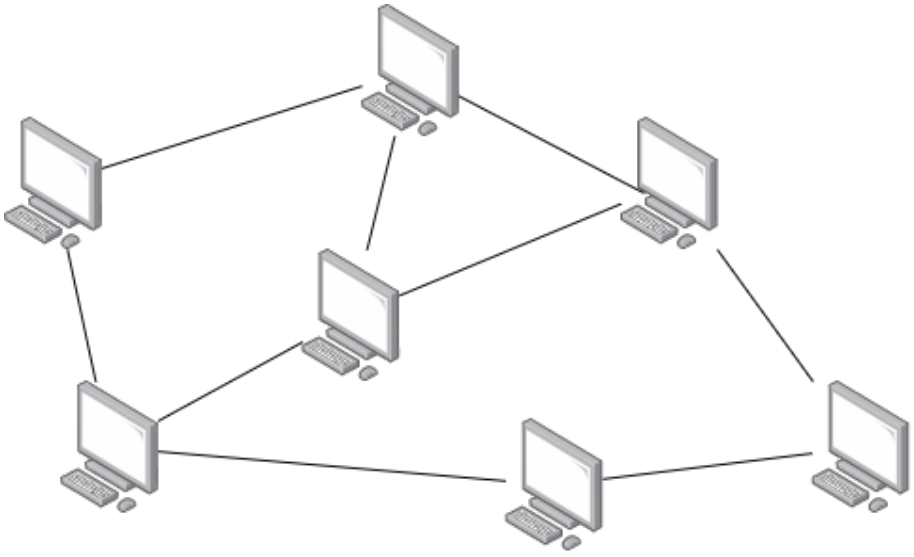


Abbildung 1.1: Peer-to-Peer Architektur

Ein `Peer-to-Peer` Netzwerk zeichnet sich durch einige wichtige Eigenschaften aus. Es handelt sich zunächst um eine dezentrale Struktur. In dieser gibt es keine Server. Das Peer-to-Peer Netzwerk besteht lediglich aus Clients als Teilnehmer. Diese Clients organisieren sich innerhalb des Netzwerks selbst und arbeiten dabei völlig selbständig. Sie sind in der Lage, die Funktionen und Ressourcen des Netzwerks sowohl zu nutzen als auch anzubieten.

Durch die selbständige Organisation entsteht ein unstrukturiertes Netzwerk, in dem die Teilnehmer eher zufällig miteinander verbunden sind. Über den Aufbau und die angebotenen Funktionen ist dadurch keine Information bekannt. Da lediglich der Zielcomputer weiß, dass er gemeint ist, wird zur Suche nach den entsprechenden Ressourcen oder Teilnehmern innerhalb des Netzwerks Flooding verwendet. Dies geschieht entlang der Verbindungen im Netzwerk. Die eigentliche Kommunikation zwischen den Teilnehmer geschieht dann jedoch direkt ohne Zwischeninstanz.
Außerdem sind die Clients im Peer-to-Peer Netzwerk alle gleichberechtigt. Dadurch können alle unter identischen Bedingungen am Netzwerk teilnehmen und niemand wird bevorzugt oder ausgeschlossen [28] [27].

Ein Peer-to-Peer System bietet so einige Möglichkeiten. Insgesamt ist das System leicht zu skalieren. Durch die Selbststrukturierung des Netzwerks können neue Teilnehmer einfach hinzugefügt werden. Ebenso existiert dadurch auch nur ein geringer Verwaltungsaufwand. Durch den dezentralen Ansatz des Netzwerks steigt zum einen mit jedem neuen Teilnehmer die Leistungsfähigkeit, zum anderen ist es weniger anfällig für Ausfälle, da die Funktionen mehrfach von unterschiedlichen Teilnehmern angeboten werden können. Auf der anderen Seite hat ein Peer-to-Peer Netzwerk auch Grenzen. Mit steigender Zahl der Teilnehmer wächst auch der Aufwand, der bei der Suche über Flooding betrieben werden muss. Für die selbstständige Organisation werden durchgehend Ressourcen der Clients verwendet, da diese sich ständig anpasst und neue Teilnehmer aufgenommen beziehungsweise bestehende wegfallen. Hinzu kommt, dass die Rechenleistung der Clients stark variieren kann und somit nicht jeder gleich zum Netzwerk beiträgt. Zusätzlich lässt sich auch die Verfügbarkeit bestimmter Clients nicht garantieren, wodurch es dazu kommen kann, das bestimmte Funktionen oder Ressourcen zeitweise nicht verfügbar sind [28] [27].

Zusätzlich zum beschriebenen Peer-to-Peer Modell gibt es verschiedene Varianten, die sich leicht unterscheiden. So gibt es Varianten, die einen strukturierteren Ansatz verfolgen oder sich im Grad der Dezentralität und Gleichberechtigung der Clients unterscheiden.

Peer-to-Peer Netzwerke sind hauptsächlich durch Filesharing Plattformen wie eMule bekannt geworden. Im Bereich der Audio- und Videokonferenzsystemen ist der bekannteste Einsatz von Peer-to-Peer bei Skype. Heute kommen solche Netzwerke beispielsweise beim Grid Computing zum Einsatz.

1.1.1.1 Peer-to-Peer bei Skype

Bei Skype wurde eine Variante von Peer-to-Peer benutzt, um sich besser an die Eigenheiten von Audio- und Videokonferenzen anzupassen. Insgesamt ist die verwendete Variante etwas strukturierter und zentralisierter.

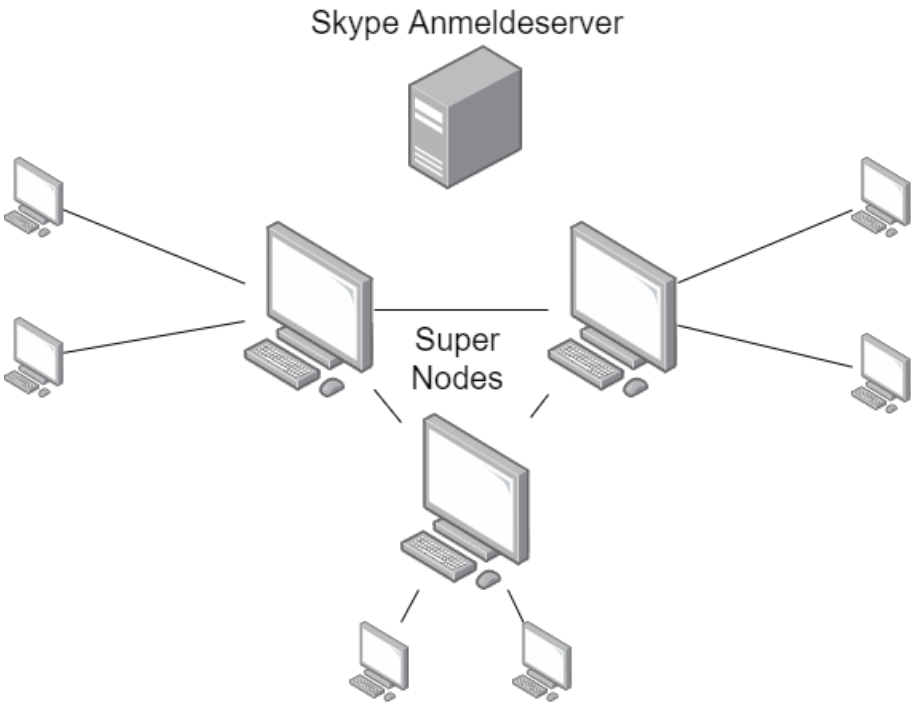


Abbildung 1.2: Peer-to-Peer Architektur bei Skype [3]

Skype hat das Peer-to-Peer Modell zunächst um einen Anmeldeserver erweitert. An diesem mussten sich die Clients authentifizieren bevor sie ins Netzwerk eintreten durften.

So wurde auch sichergestellt, dass Nutzernamen eindeutig und einmalig im ganzen System sind. Dieser Anmeldeserver ist jedoch der einzige zentrale Server im Netzwerk.
Zusätzlich zu den normalen Clients gab es im Skype Netzwerk Super Nodes. Diese Super Nodes übernahmen Übertragungen von Nachrichten bei Audio- und Videokonferenzen für einige Clients. Vor allem dann, wenn diese Aufgrund einer Firewall oder der Verwendung von Network Adress Translation (NAT) keine direkten Peer-to-Peer Verbindungen aufbauen konnten. Diese Auswahl, ob man ein Super Node wurde oder nicht, konnte nicht beeinflusst werden. Jeder Client konnte ein Super Node werden. Ausschlaggebend dafür war die zur Verfügung stehende Bandbreite, ausreichend CPU Power und Arbeitsspeicherkapazität sowie das keine Netzwerkrestriktionen für den Client bestehen und er eine öffentliche IP-Adresse besitzt.
Um die Clients mit Super Nodes zu verbinden, erstellte der Skype Client eine Liste mit Kombinationen aus IP-Adresse und Port von Super Nodes. Diese als Host Cache bezeichnete Liste beinhaltete bis zu 200 Einträge. Sie wurde in regelmäßigen Abständen erneuert, während man sich im Netzwerk befunden hat. So wurde sichergestellt, dass man sich zu einem neuen Super Node verbinden kann, wenn der aktuelle aus dem Netzwerk ausscheidet. Wichtig war dabei, dass immer mindestens ein korrekter Eintrag in der Liste aufzufinden war, um sich überhaupt mit dem Netzwerk verbinden zu können.
Grundlegende Eigenschaften von Peer-to-Peer wurden jedoch weiterhin verwendet. Für Audio- und Videokonferenzen haben die Clients, wenn möglich, direkt miteinander kommuniziert. Bei vorhandenen Restriktionen eines Clients lief die Kommunikation entsprechend über einen Super Node. Jedoch hat auch dieser mit der Gegenstelle direkt kommuniziert. Bei Gesprächen mit mehreren Teilnehmern gab es jeweils eine Verbindung zu einem der Teilnehmer der die Nachrichten gebündelt mit seinen eigenen verschickt hat [3].

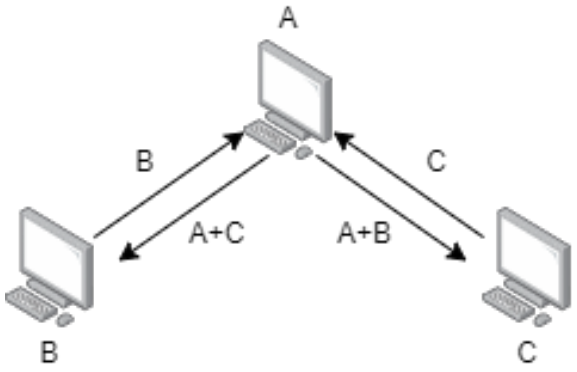


Abbildung 1.3: Ablauf einer Telefonkonferenz [3]

Auch der dezentrale Ansatz von Peer-to-Peer wurde, abgesehen vom Anmeldeserver, beibehalten. Informationen über den Onlinestatus wurden von den Super Nodes erfasst, zwischen ihnen propagiert und so an die Clients weitergegeben. Informationen wie der Host Cache waren clientbezogene Daten und nur Lokal gespeichert. Zusätzlich waren auch Freundeslisten nur lokale Informationen und wurden bei einer Anmeldung an einem anderen Client nicht übernommen [3].

Mit der Übernahme durch Microsoft begann für Skype die Abkehr vom Peer-to-Peer Modell. Microsoft führte in einem der ersten Schritte permanente Super Nodes ein. Dadurch wurden die Clients nicht mehr selbst zu Super Nodes sondern es gab eine Struktur aus Servern, die diese Aufgabe übernahmen. Besonders durch die steigende Anzahl mobiler Endgeräte ist dies aber auch verständlich. Vor allem durch begrenzte Bandbreite und Datenvolumen sowie Techniken wie Carrier-grade NAT ist es schwierig genügend Super Nodes für den Betrieb des Netzwerks zu finden. Seit 2016 läuft Skype nun auf einer Cloud-Architektur von Microsoft und verwendet somit das Client-Server Modell [29].

1.1.2 Client-Server

Das Client-Server Modell ist der Gegenentwurf zum Peer-to-Peer Modell. Es verfolgt einen zentralisierten Ansatz und hat eine klare Rollenverteilung und Struktur.

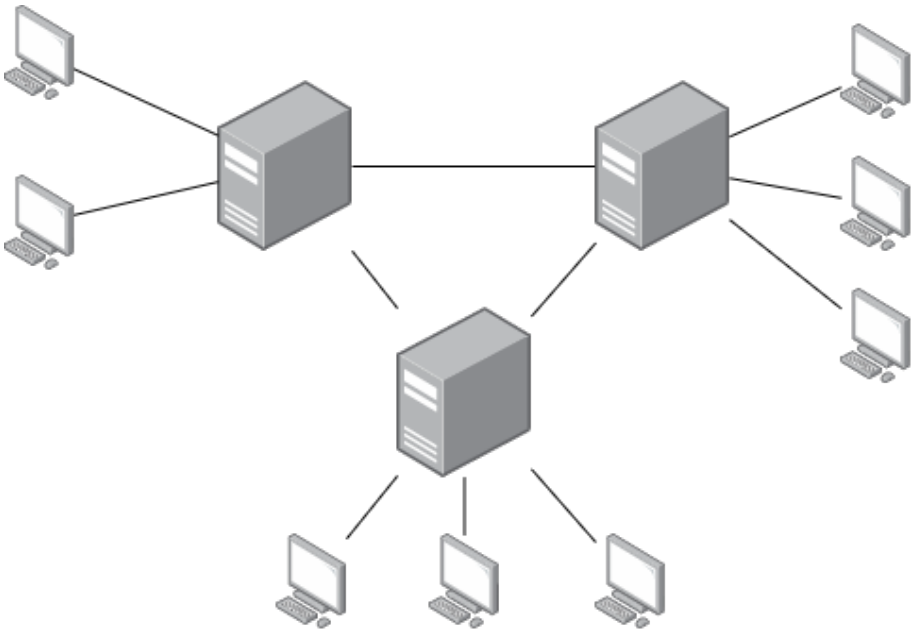


Abbildung 1.4: Client-Server Architektur

Bei der Client-Server Architektur unterscheidet man bei den Geräten zwei Rollen. Zum einen gibt es die Server, die unterschiedliche Dienste und Ressourcen zur Verfügung stellen. Zum anderen gibt es die Clients, die wiederum auf diese Dienste und Ressourcen zugreifen. Die Unterscheidung der Rollen bezieht sich dabei jedoch nicht zwingend auf physische Hardware, sondern auf gesonderte Programme. So können die Server und Client Software auch parallel auf einem Rechner laufen.

Ein Server ist in der Lage, entsprechend der ihm zur Verfügung stehenden Rechenleistung einen oder mehrere Clients zu bedienen. Für die Clients ist dabei meist nicht ersichtlich mit welchem Server sie eigentlich verbunden sind. Dies ist aus Clientsicht aber unerheblich für die Nutzung der zur Verfügung gestellten Dienste. Die Anfrage dieser Dienste geschieht immer vom Client aus. Die Kommunikation kann so entweder Lokal auf einem Rechner oder über das Netzwerk erfolgen. Hierfür werden in der Regel anwendungsspezifische Protokolle verwendet [26].

Das Client-Server Modell bietet somit vor allem die Möglichkeit der zentralen Verwaltung. Sämtliche Dienste und Ressourcen, aber auch beispielsweise das Management von Berechtigungen können so zentral verwaltet werden. Zusätzlich können über wenige Server viele Clients mit den Diensten versorgt werden. Für diese Server ist es dabei unerheblich, wo sie sich befinden, solange sie über ein Netzwerk erreichbar sind.

Das Betreiben einer Client-Server Architektur ist jedoch mit einem erhöhten Arbeitsaufwand und hohen Kosten verbunden. Um den Ausfall der zur Verfügung gestellten Dienste zu verhindern, muss das System entsprechend redundant angelegt werden. Zusätzlich muss ausreichend Rechenkapazität und Bandbreite bereitgestellt werden, um Ressourcenengpässe zu vermeiden und alle Clients versorgen zu können [26].

Client-Server Architekturen sind heute sehr weit verbreitet. Zu den Anwendungsgebieten gehören unter anderem das World Wide Web (HTTP, HTTPS), E-Mail (SMTP, POP3, IMAP), Dateitransfer Anwendungen (FTP) oder auch die Namensauflösung im Netzwerk (DNS) [26].

Im Bereich der Audio- und Videokonferenzsysteme wird heute ebenfalls hauptsächlich ein Client-Server Modell verwendet. Während Anwendungen wie Microsoft Teams oder Google Hangouts die Funktionalität lediglich über ihre Cloud-Infrastruktur über Server zur Verfügung stellen gibt es mit BigBlueButton oder Jitsi auch kostenlose Programme, mit denen man sich seine eigene Struktur schaffen kann.

1.2 Protokolle

1.2.1 WebRTC

Web Real Time Communication (WebRTC) wird vom World Wide Web Consortium (W3C) als offener Standard definiert. Unterstützt wird er von der Mozilla Foundation, Google und der Open Software ASA. Der offene Standard, definiert eine Ansammlung von Kommunikationsprotokollen und Programmierschnittstellen (API). Diese ermögliche eine Kommunikation über eine Rechner-Rechner Verbindungen in Echtzeit. Webbrowser sind damit in der Lage Echtzeitinformationen von anderen Browsern direkt abzurufen [64]. Ermöglicht wird hierdurch:

- Videokonferenzen
- Dateiübertragungen
- Chat
- Desktopsharing

Zwischen den beiden Clients wird ein Webserver benötigt, um eine Verbindung herzustellen. Eine weitere freie Implementierung, welche als Basis GStreamer nutzt, ist Open WebRTC. Dieses unterstützt auch den H.264 Standard und ist besonders für browserunabhängige native Anwendungen geeignet.

1.2.1.1 Technik

WebRTC basiert auf JavaScript und HTML5. Es wird bei der Übertragung darauf geachtet, dass alles verschlüsselt ist. Dies ist eine Vorgabe, welche erfüllt werden muss [64]. Die Übertragung erfolgt über (S)RTP mit der XMPP-Erweiterung Jingle in Kombination mit dem JavaScript Session Establishment Protocol (JSEP). Es wird hierbei eine direkte Verbindung (Peer to Peer) aufgebaut. Die Verschlüsselung der Datenströme wird mit Hilfe von Datagram Transport Layer Security (DTLS) realisiert. Bei der Audio- und Videokommunikation verwendet man zur Sicherheit zusätzlich Secure Real Time Transport Protocol (SRTP). SRTP nutzt den Advanced Encryption Standard [64]. Als bevorzugter Audio Codec nutzt man hier Opus. Für herkömmliche Telefonsysteme wird zusätzlich A-law und μ-law unterstützt.

Dies ermöglicht es Teilnehmer an der Konferenz teilhaben zu lassen, welche sich per Telefon einwählen müssen. Der bevorzugte Video-Codec soll VP8 werden, welches Google freigekauft hat [16]. Es liegen bei der Referenzimplementierung weitere Tools, wie eine Rauschunterdrückung oder die Software Bibliothek „libjingle“ bei.

1.2.1.2 Sicherheitsrisiken

Eine Webseite kann mittels JavaScript den Simple Traversal of UDP through NAT (STUN)-Server nach der tatsächliche IP-Adresse des Rechners fragen. Dies hat zur Folge, dass ein Anonymisierungsdienst nicht mehr seinen Sinn erfüllen kann. -> IP-Leak [57] Schutz gegen diesen IP-Leak bieten beispielsweise Plugins / Ad-Ons wie „WebRTC Leak Prevent“. Diese verhindern, dass die öffentliche IP-Adresse mittels JavaScript ausgelesen werden kann.

1.2.2 XMPP

Es handelt sich bei Extensible Messeging and Presence Protocol (XMMP) ebenfalls um einen offenen Standard eines Kommunikationsprotokolls. Es wurde von der Internet Engineering Task Force (IETF) als RFC 6120, RFC 6122 veröffentlicht. Das Protokoll basiert auf XML und ermöglicht den Austausch von Daten [49]. XMMP unterstützt mit seinen Erweiterungen einen Multi-User Chat, also Konferenzen mit mehreren Benutzern, Dateiübertragungen, Versenden von digitalen Signaturen, uvm. Jeder XMMP-Server kann ganz einfach Nachrichten mit anderen Servern austauschen. Dies bedeutet, dass auch Verbindungen zu anderen Anbietern kein Problem darstellen. Schickt ein Nutzer beispielsweise an den eigenen XMMP-Server eine Nachricht, kann diese dort vom Server an einen anderen Server weitergeleitet werden und dort dann an den Empfänger zugestellt werden.

Um ein XMMP Netzwerk bereitzustellen benötigt man mindestens einen XMMP Server. Dieser kann im Intranet als alleinige Kommunikationsstelle verwendet werden oder er stellt über das Internet eine Verbindung zu weiteren XMMP-Servern her. Die weiteren Server werden auch XMPP-Federation genannt. Die Nutzer innerhalb eines XMMP-Netzwerks, werden mittels des „Jabber Identifier“ (JID) adressiert und identifiziert.

Die Form hierbei sieht folgendermaßen aus: Bob@example.com Hier wird der Nutzername und der Server angegeben. Es ist unter anderem auch möglich sich mehrfach anzumelden. Dies ist durch das Ressourcen Konzept möglich.

1.2.2.1 Peer to Peer

Mit Jingle ist XMMP in der Lage Peer to Peer Sitzungen einzuleiten. Zuerst wurde XMMP erweitert, sodass Google Talk mit XMMP Voice over IP (VoIP) Anrufe tätigen konnte. 2005 wurde anschließend die Erweiterung „Jingle Signalling“ hinzugefügt, mit welcher XMMP auch Peer to Peer unterstützt [49].

1.2.2.2 Multi User Chat (MUC)

Es handelt sich hierbei um eine Spezifikation, die eine Rollenzuordnung der einzelnen Teilnehmer innerhalb des Chats, den Passwortschutz eines Chatraums, die Unsichtbarkeit eines Chatraums und vieles mehr unterstützt. Die Spezifikation ist zudem abwärtskompatibel zur früheren Spezifikation „Groupchat“. Die Konferenzräume werden durch den Jabber Identifier repräsentiert. Der Multi User Chat (MUC) ist aus Sicht des Anwenders ähnlich dem bekannten „Internet Relay Chat“. Zudem ist es möglich mit einem XMMP Netzwerk eine Kommunikation zu anderen Chat Netzwerken, wie z.B. Yahoo Messenger, Windows Live Messenger, Internal Relay Chat und vielen weiteren zu haben [49].

1.2.3 SIP

Das Session Initiation Protcol (SIP) wurde von der IETF entwickelt. Das Protokoll dient dem Aufbau, der Steuerung und dem Abbau einer Kommunikationssitzung zwischen mehreren Teilnehmern [10]. Das Protokoll wird vor allem bei der IP-Telefonie verwendet. SIP lehnt sich an das Hypertext Transfer Protocol an und ist somit deutlich besser für IP-Netze geeignet als H.323. Im Gegensatz zu H.323 ist es im Endkundennetz auch deutlich besser umzusetzen, da dort Firewalls und Router verwendet werden und somit eine Übersetzung von Netzwerkadressen stattfindet. Auch wenn SIP an http anlehnt, ist es jedoch nicht dazu kompatibel. Mit SIP ist es einfach möglich neue Erweiterungen hinzuzufügen ohne,

dass schon vorhandene Geräte diese Erweiterungen verstehen müssen. Zudem ist es mit SIP nicht nur möglich Telefonie zu verwalten, sondern auch Sitzungen beliebiger Art. Meistens kommt SIP im Bereich der Video- und Audioübertragung zum Einsatz. Vereinzelt verwenden auch Online Spielen SIP zur Verwaltung. Um die Details der Video- und Audioübertragung auszuhandeln wird meistens das Session Description Protocol (SDP) verwendet. Der Ablauf sieht so aus, dass die beiden Teilnehmer sich gegenseitig mitteilen, welche Audio- und Videoübertragungen sie beherrschen, welches Protokoll verwendet werden soll und welche Netzwerkadresse zum Senden und Empfangen verwendet werden soll [8]. Die Medienübertragung (Video und Audio) findet nun anschließend über das Real-Time Transport Protocol (RTP) statt. Möchte ein Hersteller für eine spezialisierte Anwendung SIP verwenden kann er eine eigene Medienaushandlung erstellen, falls er nicht auf ein vorhandenes Protokoll zurückgreifen kann [8].
Die Adressen der Teilnehmer werden im URI-Format dargestellt [8]:

- Unverschlüsselt SIP: sip:user@domain
- Verschlüsselt SIP: sips:user@domain
- Telefon: tel:nummer

1.2.3.1 Sicherheit

Die Datenströme und die Sitzung selbst können unabhängig voneinander verschlüsselt werden. Es wird das Protokoll Transport Layer Security (TLS) verwendet und somit wird die Sitzung über Session Initiation Protcol Secure (SIPS) und die Medien über SRTP verschlüsselt. Beide Datenströme werden gleichzeitig verschlüsselt. Der Austausch der symmetrischen Schlüssel des Medienstromse rfolgt über SDP. Am Anfang werden die symmetrischen Schlüssel per TLS ausgetauscht, allerdings greifen hier noch Mechanismen der SSL-Zertifikate. Dies bedeutet, dass die symmetrischen Schlüssel durch asymmetrische Schlüssel der SSL-Zertifikate wiederum verschlüsselt sind [60].

1.2.4 H.323

Bei H.323 handelt es sich um ein Protokoll für paketbasierte Multimedia Dienste. Auch VoIP oder Videokommunikation überträgt das Protokoll H.323. H.323 wurde das erste mal 1996 von der ITU eingeführt und ist somit das älteste Protokoll für VoIP. Anschließend wurde es immer weiterentwickelt, sodass die ersten VoIP Telekommunikationsanlagen diesen Standard verwendeten. Es sind bereits Gateway und Gatekeeper Funktionen in H.323 definiert. Ein Gatekeeper ist für die Regulierung des Breitbandmanagements und die Übersetzung von symbolischen Adressen in IP-Adressen verantwortlich. Das Gateway hingegen erlaubt es eine Verbindung zu anderen Sprachnetzen herzustellen [55].

1.2.5 ZRTP

Hierbei handelt es sich um ein Protokoll zur Verschlüsselung der Kommunikation zwischen zwei Endgeräten, welche VoIP über RTP nutzen. Bei ZRTP wird das Secure Real-Time Transport Protocol zur Verschlüsselung und der Diffie Hellmann Schlüsseltausch verwendet. Die Besonderheit bei ZRTP ist, dass weder die SIP-Signalisierung noch sonst eine Art von Server als Grundlage benötigt werden. Bei jeder Sitzung erstellt ZRTP neue Diffie-Hellmann Schlüssel, die nach der Sitzung ablaufen. Der Schlüsseltausch erfolgt über den RTP Datenstrom. ZRTP kann mit den verschiedensten Signalisierungsprotokollen verwendet werden. Hierzu zählen beispielsweise H.323, Jingle, SIP und verteilte Hashtabellen [67].

ZRTP kann auch mit jeder Art von Telefonnetzwerken betrieben werden. Beispielsweise funktioniert es ohne Probleme mit UMTS, ISDN, GSM, usw.

Da bei der ersten Sitzung, in welcher der Schlüssel ausgetauscht wird ausgeschlossen werden soll das es keinen Angreifer in der ersten Sitzung gibt, wird ein „Short Authentication String“ (SAS), welcher Nonce genannt wird, verwendet [68].

1.2.6 T.120

Mit dem Protokoll T.120 werden Datenanwendungen innerhalb der Videokonferenzen realisiert. Hier werden neue Richtlinien festgehalten, welche den Verbindungsaufbau und Verbindungsabbau, die Zusammenarbeit mit den MCUs, das Verwenden von interaktiven Whiteboards, die Flusskontrolle, den Dateitransfer und das Sharing von Anwendungen detailliert beschreiben [59].

1.2.7 H.239

H.239 kommt dann zum Einsatz, wenn bei einer Videokonferenz zu dem eigentlichen Videokanal noch ein zusätzlicher Kanal, beispielsweise für eine Präsentation, benötigt wird. Dies funktioniert nur, wenn H.320 und H.245 als Basissysteme verwendet werden [17].

1.3 Codecs

1.3.1 H.264

Bei H.264 handelt es sich um einen Standard, welcher zur Videokompression verwendet wird. Auch bekannt ist das Verfahren unter dem Namen MPEG-4 AVC. Ziel von H.264 ist es die Datenrate bei gleicher Qualität, um die Hälfte bei der Übertragung zu reduzieren. Im Vergleich zu MPEG-2 werden nur noch halb so viele Bits für die Übertragung benötigt. Die Technik wird meist bei der Übertragung von Videos über das Internet verendet [5]. Entwickelt wurde das Protokoll von Cisco Systems, Microsoft und dem Frauenhofer-Institut für Nachrichtentechnik.

1.3.2 H.265

Dieser Standard, auch bekannt als MPEG-H Teil 2 oder High Efficiency Video Coding (HEVC), dient der Kompression von Bildern und Videos. Es ist der Nachfolger des ebenso bekannten Standard H.264. H.265 ist der Konkurrenzstandard zu VP9 und AV1.

Entwickler sind hierbei der ISO/IEC Movie Picture Experts Group (MPEG) und die ITU-T Video Coding Experts Group (VCEG). Ziel ist es bei H.265 im Vergleich zum Vorgänger eine doppelt so starke Kompression bei gleichbleibender Qualität zu erhalten. Des Weiteren ist der Standard in der Lage von QVGA (320 x 240 Pixel) bis zu Ultra High Definition Television 2 (UHD2) und 8K (8192 x 4320 Pixel) zu skalieren [53].

Angewendet wird der Standard bei ultra-hochauflösenden Fernsehprogrammen, Blu-Rays mit 4K Auflösung, Camcordern oder Streaming Angeboten. Auch bei normalem HD, wie z.B. DVB-T2 kommt H.265 zum Einsatz. Auch manche Videokonferenzsysteme verwenden diesen Codec. Anforderungen sind bei dem Standard die Verringerung der Komplexität um 50 Prozent und die 25 Prozent größere Bitratenreduktion bei gleichbleibender Qualität im Vergleich zu H.264. H.265 arbeitet hauptsächlich effektiver, da das Bild in Blockgrößen von 4x4 bis 64x64 und nicht mehr in 16x16 eingeteilt wird. Besonders kommt diese Möglichkeit bei großen Bildern zur Erscheinung [43].

Abbildung 1.5: H.265 vs HEVC [48]

Bei Videos liegt der Vorteil hier bei den nun vorhandenen vier Transform-Blocks (vorher nur zwei Transform-Blocks), welche für die Bildbereiche genutzt werden. Diese werden benutzt, wenn sich die aufeinander folgenden Frames nicht verändern, sondern nur ihre Position ändern. Dies kommt beispielsweise beim Schwenken der Kamera vor.

1.3.3 Satin Codec

Microsoft möchte einen neuen Satin Codec einführen. Dies ist ein von Microsoft eigenentwickelter Codec, welcher in MS Teams eingesetzt werden soll. Der neue Codec ist in der Lage, selbst bei sehr schlechter Internetverbindung ein klares Breitbandaudiosignal zu übertragen. Es werden lediglich 7kb/s benötigt. Besonders bei mobilen Endgeräten bringt der Codec einen gewaltigen Fortschritt, da hier nicht immer hohe Bandbreiten gegeben sind bzw. mn nur eine begrenzte Anzahl an Datenvolumen zur Verfügung hat [48].

1.3.4 OPUS

Opus ist ein Audiocodec. Dieser Codec ist optimal für den Einsatz bei Musikübertragungen und interaktiven Gesprächen über das Internet. Es werden Audioqualitäten unterstützt, welche eine geringe Sampelrate von 8kHz haben bis hin zu Audioqualitäten mit bis zu 48kHz. Es werden außerdem bis zu 255 Channel unterstützt (multistream frames). Die Framegröße beträgt bei Opus 2,5ms bis 60ms. Die Bitraten selber liegen in einem Bereich von 6 kb/s bis 510 kb/s [56].

1.3.5 SILK

Bei SILK handelt es sich um einen weiteren Audiocodec, der von Skype Limited entwickelt wurde. Skype Limited gehört mittlerweile zu Microsoft. SILK ist hat einen Open Source Code, sodass ihn jeder kostenlos verwenden kann, der damit keinen Profit machen möchte. Soll SILK kommerziell verwendet werden, so soll eine Lizenz erworben werden. Der Code ist in C geschrieben, sodass der Codec sehr schnell arbeiten kann [**The H-Online_2010**]. SILK hat zudem ein sehr geringes Delay mit ungefähr 25ms. Außerdem ist in der folgenden Grafik zu sehen, dass selbst bei einer hohen Sampling Rate die Bit Rate sehr gering bleibt, sodass auch bei schwächeren Internetverbindungen eine gute Audioqualität erreicht werden kann. Auch die Anforderungen von SILK an den Prozessor sind für heutige Verhältnisse sehr gering [58].

	Sampling Rate (kHz)	Bit Rate (kbps)	CPU (MHz on x86 core)
Narrowband for PSTN gateways and low end devices	8	6 - 20	12 - 30
Mediumband for devices with limited wideband capability	12	7 - 25	16 - 40
Wideband for all-IP platforms	16	8 - 30	20 - 50
Super-Wideband, a new standard in speech quality	24	12 - 40	30 - 80
Key Advantage	Optimize clarity under hardware and network constraints	Adjust to degraded network conditions in real time	Match complexity to CPU resources in real time

Abbildung 1.6: SILK bandwith, bit rate and complexity [58]

1.3.6 VP8 und VP9

Eine verlustbehaftete Kompression mit Videodaten ist mit den Formaten VP8 und VP9 möglich. Für diese Formate fallen keine Lizenzgebühren an. Sie bauen auf der TrueMotion Reihe auf. VP8 wurde ursprünglich von On2 Technologies entwickelt und anschließend von Google aufgekauft. VP8 soll als Konkurrenzprodukt zu H.264 verwendet werden. Im Test zeigt, sich allerdings, dass VP8 langsamer als H.264/AVC ist. Die Qualität von VP8 ist zudem bei höheren Bitraten geringer als bei seinem Konkurrenten. Somit benötigt VP8 mehr Speicherplatz als H.264 damit die gleiche Qualität erzielt werden kann [61].

Bei VP8 handelt es sich um blockbasiertes Transformationsverfahren. Der Nachfolger von VP8 nennt sich VP9. Die Neue Konkurrenz ist somit H.265/HEVC. Zusammen mit dem Audiocodec Opus findet man VP9 auch in WebM Dateien [38]. Zusätzlich wird auch bei der Übertragung von YouTube Videos der Codec VP9 verwendet [18]. Ein großer Unterschied zu seinem Vorgänger ist, dass VP9 fast in der Lage ist verlustfrei zu komprimieren. Es ist zusätzlich für UHD Videos verwendbar. Im direkten Vergleich benötigt VP9 doppelt so viel Bitrate um die gleiche Qualität wie HEVC zu liefern. Der algorithmische Vergleich zu H.264 zeigt, dass VP9 bis zu 50 Prozent besserer Bitrateneffizienz hat und dies zehn bis zwanzig Mal schneller kodiert. Zudem unterstützt VP9 8K Videos mit einer Auflösung von 7680 x 4320 Pixel und einer Bildwiederholungsrate von 120 Bildern [1]. Auch im Vergleich zu H.265 ist VP9 deutlich schneller, wie man der folgenden Grafik entnehmen kann:

Encoding Time - New Clip (1:36)	4K	1920	1280
x265	188:21	52:18	35:12
VP9	75:08	33:57	10:07

Abbildung 1.7: Time Consumption of Encoding [39]

1.4 Meeting Programme

Es gibt verschiedene Varianten von Meeting Tools, die für Audiokonferenzen verwendet werden können. Hierzu gehören beispielsweise Cisco WebEx, Microsoft Teams, Skype, Google Hangouts, Discord, Teamspeak, Jitsi und BigBlueButton.

1.4.1 Jitsi Meet

Bei Jisti Meet handelt es sich um ein Open Source System, bei dem SIP und XMMP für Telefonie und Sofortnachrichten verwendet werden. Bei Medienübertragungen kommen folgende Protokolle zum Einsatz:

- SILK
- G.722
- Opus
- H.264
- H.263
- Einführung von VP8
- Speex

Jitsi Meet basiert auf WebRTC. Auf Desktop Systemen kann man Jitsi ohne Probleme über den Webbrowser verwenden. Bei der mobilen Verwendung hat man die Möglichkeit eine App zu verwenden, welche hierfür entwickelt wurde. Zusätzlich gibt es auch Apps für Windows, Linux und MacOS [50].

Als zentrale Stelle wird bei Jtisi eine Serverinstanz verwenden, in welcher auch die einzelnen Räume geöffnet werden können. Die Besonderheit hierbei ist, dass jeder sich einen Server erstellen kann, welche als zentrale Instanz dient. Man ist also nicht auf Anbieter angewiesen, welche einem einen Jitsi Server anbieten. Für die Übertragung von Medien stellt Jitsi zusätzlich eine Peer to Peer Verbindung zur Verfügung, welche über Interactive Connectivity Establishment und Universal Plug and Play realisiert wird. Für die NAT Konfiguration bringt Jitsi zum Routen der RTP Pakete JingleNodes und Trun mit. Die Funktionen von Jitsi sind vielseitig. Hierzu gehören z.B. der Passwortschutz für Räume, ein Chat, das Bearbeiten von Dokumenten, die Telefoneinwahl mittels SIP, die Möglichkeit den Konferenzraum zu Plattformen wie YouTube zu streamen, ein Melden per Handzeichen, die Konferenz aufzunehmen, eine Statistik der Sprecherzeit und das Desktop Sharing. Zudem ist für die Verwendung von Jitsi keine Registrierung nötig [62].

1.4.2 BigBlueButton

Bei BigBlueButton kommt eine Open Source Implementierung vom Adobe Media Server zum Einsatz. Diese dient zur Unterstützung der Zusammenarbeit in Echtzeit. Die Clients basierten hierbei auf Adobe Flash, was aber nur noch aufgrund der Abwärtskompatibilität angeboten wird. Mittlerweile wird zur Übertragung von Video und Audio HTML5 und WebRTC verwendet. Da diese in fast allen Browsern standardmäßig implementiert sind, erfordert die Verwendung von BigBlueButton keine zusätzlichen Plugins. Die redis Datenbank wird bei dem System zur Verwaltung der Schlüssel-Werte Daten, welche die Nutzer widerspiegeln, verwendet [51]. Auch hier hat man die Möglichkeit einen eigenen Server zu betreiben, sodass man von keinem Anbieter abhängig ist Abgesehen von der reinen Übertragung von Video und Audio bietet BigBlueButton die Möglichkeit zu zoomen, Desktop Sharing, einen öffentlichen und privaten Chat, eine Rollenverteilung von Moderatoren und Betrachtern, die Präsentation von PDFs und MS Office Dokumenten, eine Whiteboard Funktion und Zeichnen [52]. BigBlueButton unterstützt zudem eine Deep LMS Integration. Dies bedeutet, dass es in Systeme wie Moodle, Sakai oder weitere integriert und innerhalb des Systems verwendet werden kann [54].

1.4.3 Microsoft Teams

Microsoft Teams ist ein Audio- und Videokonferenzsystem, das hauptsächlich für den Einsatz in Unternehmen und Bildungseinrichtungen konzipiert ist. Es ist der Nachfolger von Skype for Business beziehungsweise Lync. Mit Teams setzt Microsoft auf eine Client-Server Architektur, die vollkommen über die Cloud genutzt wird.

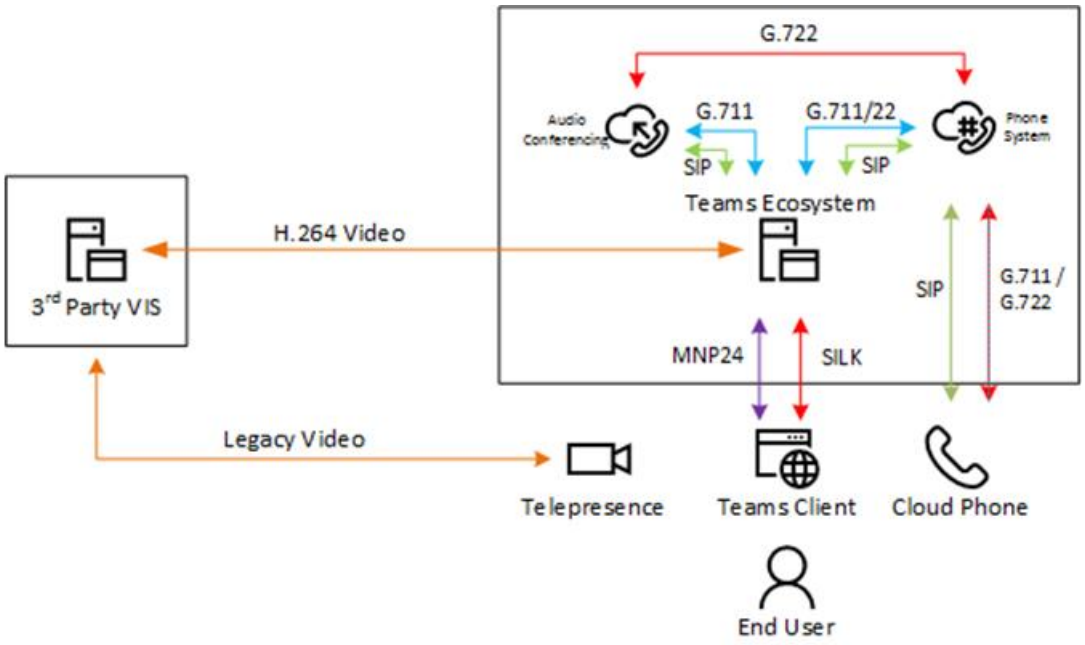


Abbildung 1.8: Architektur von MS Teams [9]

Teams verwendet dabei die gängigen Protokolle zum Umsetzen der Audio- und Videokonferenzen. Der zentrale Punkt bei Teams liegt dabei auf der Zusammenarbeit, die durch das Produkt erleichtert werden soll. Besonders die Integration in Microsoft 365 (ehemals Office 365), ist hier hervorzuheben. Dadurch bietet Teams eine direkte Dateiverwaltung über OneDrive und integrierte, gemeinsame Bearbeitung von Dokumenten über Word, Excel oder PowerPoint. Darüber hinaus stellt es auch Funktionalitäten zum Bildschirm teilen und der Fernsteuerung von Geräten zur Verfügung. Mit Teams können bis zu 10.000 Personen an einer Videokonferenz teilnehmen. Zusätzlich bietet es eine integrierte Funktion zum Aufzeichnen von Konferenzen sowie eine Liveübersetzung in Text [46] [47].

1.4.4 Google Hangouts

Bei Google Hangouts handelt es sich nicht um eine klassische Audio- und Videokonferenz Software. Hangouts ist als Teil des Social Networks Google+ gestartet und löste hier die Produkte Google Talk und den Google+ Messenger ab. Hangouts ist somit ein Messenger, der auch Audio- und Videokonferenzfunktionalitäten mitbringt.

Die Zielgruppe hier ist jedoch eher der private Bereich. Auch Hangouts nutzt eine Client-Server Architektur. Zum einen um die Möglichkeiten der Google Infrastruktur auszunutzen, zum anderen aber auch um eine Synchronisation zwischen den Geräten zu ermöglichen. Zum Realisieren der Audio- und Videokonferenzen für bis zu 10 (25 bei Business oder Education Lizenz) Teilnehmer verwendet Hangouts WebRTC [14] [13] [41].

1.5 Qualitätssicherung

In der Welt der Audio- und Videokonferenzsystemen gibt mehrere Techniken die im Hintergrund laufen und dafür sorgen, dass die Qualität einer Konferenz massiv verbessert wird. Hierbei liegt das Gefühl einer echten Konversation im Vordergrund.
Es wird versucht störende Nebengeräusche zu eliminieren und übertragene Daten wie Bild und Ton möglichst synchron wiederzugeben. Dadurch soll erreicht werden, dass der Nutzer keinen, oder möglichst wenig Unterschied zu einer echten Unterhalten feststellen kann.

Die folgenden Unterkapitel fokussieren sich dabei auf drei der wichtigsten Systeme. Die sogenannte Echokompensation, die Rauschunterdrückung und die Lippensynchronisation.

1.5.1 Echokompensation

Nehmen mehrere Personen an einer Audio- oder Videokonferenz teil, kann es vorkommen, dass die eigene Stimme durch die Lautsprecher im Laptop oder Besprechungszimmer von einem Echo überlagert wird. Selbst wenn es nur ein paar Sekunden lang auftritt, ist es extrem störend und kann die Gespräche unterbrechen oder ganz zum Erliegen bringen, bis das Problem behoben ist.
Da sowohl Audio- als auch Videokonferenzen jedoch eine immer größere Rolle einnehmen, ist ein grundlegendes Verständnis der Echokompensation wichtig, um die Probleme zu vermeiden oder in den Griff zu bekommen.

Zur Vermeidung des Nachklingens in einer Konferenzschaltung ist eine Acoustic Echo Cancellation (AEC) an beiden Endpunkten der Verbindung erforderlich. Wenn ein Gesprächsteilnehmer in sein Mikrofon spricht, läuft das Signal über eine Verbindung, ein VoIP-System oder das Internet, was eine Verzögerung des Signals zur Folge hat. Das Signal wird danach über die Lautsprecher im Raum der Gegenseite wiedergegeben und von den dort befindlichen Mikrofonen aufgenommen. Das verzögerte Signal wird dann auf die Seite der sprechenden Person zurück übertragen, wo es als Echo der eigenen Stimme zu hören ist. Wie bei einer Rückkopplung ist dies eine Situation, die jede Besprechung aus dem Tritt bringt [63].

Ein so genannter Echo Canceller hört auf der Seite des Zuhörers sowohl den von den Mikrofonen im Raum aufgenommenen Schall, als auch das Signal, das über die Lautsprecher im Raum wiedergegeben wird. Erkennt er, dass das ausgehende Signal mit dem ankommenden identisch ist, wird Ersteres elektronisch ausgelöscht, sodass es nicht wieder zurück an die Lautsprecher der sprechenden Person gesandt wird. Dadurch verschwindet das Echo [63].

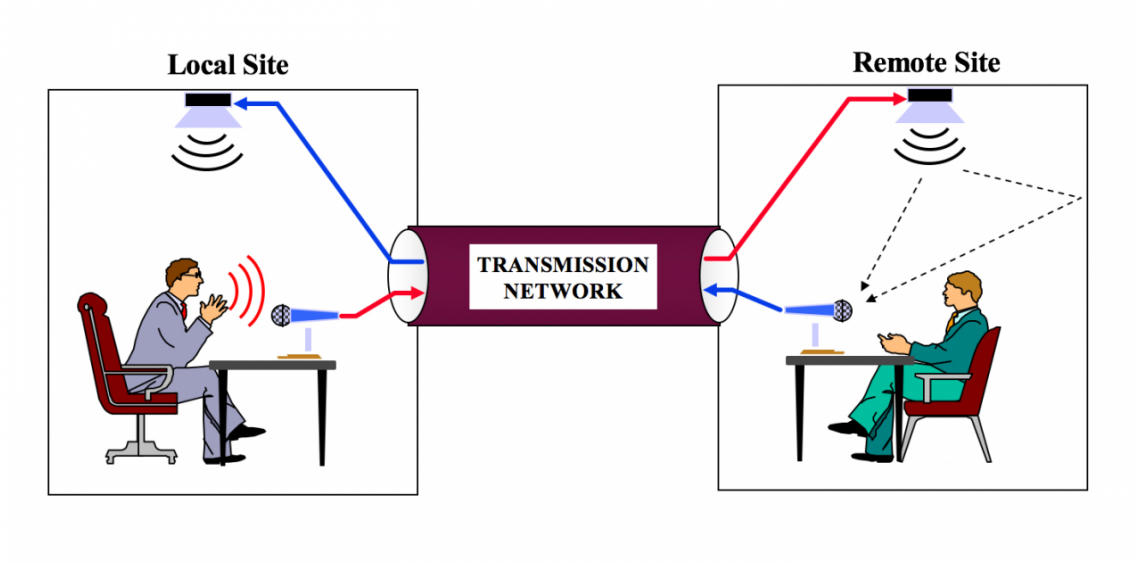


Abbildung 1.9: Übertragungsweg eines Echos [24]

Obwohl die AEC auf der Seite des Zuhörers dem Sprecher zugutekommt, muss für ein Vollduplex-Gespräch an beiden Endpunkten der Verbindung ein Echo Canceller vorhanden sein, sodass alle Gesprächspartner ein Echo-freies Signal empfangen. Wenn man ein Echo der eigenen Stimme hört, so liegt der Fehler tatsächlich bei der Gegenseite, da deren System keine funktionierende Echokompensation umfasst. Sie lässt es gewissermaßen zu, dass die eigene Stimme wieder zu einem zurückgeworfen wird.

1.5.1.1 WebRTC

WebRTCs Echo Controller arbeitet mit einer Sampling Rate von 8k bzw. 16k. Es gibt jedoch spezielle Prozesse in Chromium, um den Echo Controller mit dem iSAC codec, d.h. 32k bzw. mit dem Opus Codec d.h. 48k arbeiten zu lassen. [21]

Es existiert jedoch noch heute das Problem, dass mobile Geräte wie Smartphones nicht mit komplexen Implementierungen von AEC klarkommen. Aktuell kommen noch ca. 40% aller Android Geräte nicht mit AEC klar. Daher wurde Acoustic Echo Cancellation for MOBILE (AECM) entwickelt, welches jedoch massive Performance Probleme liefert. AECM liefert schlechte Soundqualität und verursacht ein sehr lautes Hintergrundrauschen.
Zudem existieren noch immer Probleme wie [21]:

- AECM kommt nicht damit klar, wenn zwei oder mehr Personen gleichzeitig reden
- AECM ist auf 8k/16k Sampling Rate begrenzt

1.5.1.2 BigBlueButton und Jitsi

In BigBlueButton und Jitsi wird die integrierte AEC verwendet. In den meisten Fällen sollte man daher kein Echo von Remote-Benutzern hören.
Es wird jedoch empfohlen dass, jeder Remote-Nutzer ein Headset mit Mikrofon verwendet. Dadurch wird das beste Audio-Erlebnis in einer Sitzung sichergestellt [4].

Die integrierte AEC basiert auf der Implementierung von WebRTC [4].

1.5.2 Rauschunterdrückung

Im Gegensatz zu den bisherigen Systemen sind Discord und MS Teams noch einen Schritt weitergegangen. Neben dem Standard AEC haben beide jeweils eine Artificial Intelligence (AI)-Basierte Rauschunterdrückung eingeführt.

1.5.2.1 Discord/Krisp

Discord hat die Corona-Krise und die damit hergehenden Veränderungen im Verhalten der Menschen als Grund genommen eine Partnerschaft mit Krisp.ai einzugehen. Die Menschen arbeiten aufgrund der Krise von zu Hause aus, wo nicht immer die Ruhe herrscht die in einem Besprechungsraum herrscht.

Die neue Technik erkennt und entfernt Hintergrundgeräusche, die um den Redner herum passieren, so dass die Stimme deutlich zu hören ist. Die AI geht soweit, dass sich sogar Staubsauger im Hintergrund herausfiltern lassen [22].

Krisp ist eine auf machine learning basierende Rauschfiltersoftware. Trainiert wurde die AI bisher mit [22]:

- 20.000 unterschiedlichen Geräuschen
- 50.000 verschiedenen Rednern
- 2.500 Audio-Stunden

Auf dieser Grundlage wurde ein neuronales Netzwerk namens krispNet DNN entwickelt.

Krisp fügt eine zusätzliche Schicht zwischen dem physischen Mikrofon/Lautsprecher und den Konferenzanwendungen hinzu, die keine Geräusche durchlässt. Dabei wird die gesamte Audioverarbeitung lokal durchgeführt. D.h. es werden keinerlei Daten an Drittanbieter weitergeleitet [22].

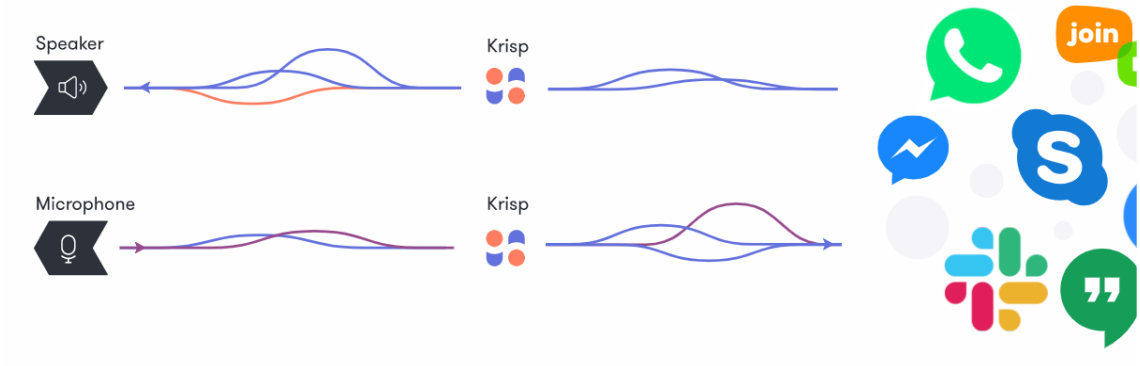


Abbildung 1.10: Visualisierung der Filterschicht von Krisp [22]

1.5.2.2 MS Teams

Das besondere bei MS Teams und der neuen Geräuschunterdrückung ist, dass nicht nur stationäre Geräusche unterdrückt werden, sondern auch nicht stationäre Geräusche. Hierzu führt Microsoft eine neue AI ein [40].

Normalerweise gehen die meisten Systeme hin und untersuchen in den Sprechpausen, welche Geräusche kontinuierlich im Hintergrund vorhanden sind. Hierzu zählen beispielsweise Ventilatoren, Klimaanlagen Staubsauger oder ähnliches. Bei der nicht stationären Geräuschunterdrückung ist das System in der Lage kurzzeitig auftretende, wie z.B. das Rascheln mit einer Tüte, das Zufallen einer Tür und Weiteres zu filtern [40].

Was durch die AI nicht herausgefiltert wird ist beispielsweise Lachen, verschiedene Musikinstrumente oder singen. Das Problem ist, dass Microsoft Stimmen anderer Personen nur schwer bis nicht herausfiltern kann, da sie sich in den gleichen Frequenzbereichen befinden [40].

Microsoft kann nicht einfach den Klang menschlicher Stimmen isolieren, weil auf denselben Frequenzen auch andere Geräusche auftreten. In einem Spektrogramm eines Sprachsignals erscheinen unerwünschte Geräusche in den Lücken zwischen der Sprache und überlagern sich mit der Sprache. Es ist daher nahezu unmöglich, den Lärm herauszufiltern - wenn sich Sprache und Lärm überlappen, kann man die beiden nicht unterscheiden. Stattdessen muss vorher ein neuronales Netz darauf trainiert werden, wie Lärm und Sprache aussieht [40].

Während man bei Microsoft noch auf den Release ihrer AI wartet, kann man bei MS Teams auch die Krisp.ai nutzen um ein besseres Konferenzerlebnis zu erhalten.

1.5.2.3 Open Broadcaster Software

Die meisten Streamer nutzen bei ihren Livestreams auf Twitch oder Youtube die beliebte Software Open Broadcaster Software (OBS). Im OBS Studio kann der Ton, mit Hilfe von Audiofiltern, deutlich verbessert werden. Dabei können OBS eigene Filter, wie z.B. Rauschunterdrückung, Expander oder Noise Gate verwendet werden [44].

Die Noise Suppression ist, obwohl er auch leichte Hintergrundgeräusche entfernen kann, für das Herausfiltern des Hintergrundrauschens (auch als weißes Rauschen oder White Noise bezeichnet) verantwortlich.
Der so genannte Expander kann die Umgebungsgeräusche wie PC-Lüfter, Tastaturanschläge, Mausklicken, Atemgeräusche und Schmatzlaute entfernen.
Das Noice Gate „schaltet“ das Mikrofon ab, wenn man nicht redest. Der Signalpegel des Mikrofons wird dabei erst weitergeleitet, wenn das Eingangsignal (Man beginnt zu Sprechen) einen, von einem bestimmten, Wert überschreitet.
Und sobald ein bestimmter Wert unterschritten wird (Man hat aufgehört zu sprechen) wird das Eingangssignal wieder blockiert [44].

1.5.2.4 Ein Ausblick auf die Zukunft

Mit Nvidia RTX Voice bietet der Grafikspezialist eine KI-gestützte Geräuschunterdrückung für die Ein- und Ausgabe jeglicher Audio-Hardware an.
Das Besondere an Nvidia RTX Voice ist, dass die Software zur intelligenten und Selbstlernenden Geräuschunterdrückung die spezialisierten Tensor-AI-Einheiten der gleichnamigen RTX-Grafikprozessoren nutzen.
Dabei verbessert RTX Voice nicht nur die ausgehende Sprachqualität übers Mikrofon, sondern kann auch eingehende Gespräche von störenden Nebengeräuschen befreien [6].

Aktuell bietet Nvidia noch die Beta-Version von RTX Voice an. Dabei werden schon folgende Systeme unterstützt [6]:

- OBS Studio
- Streamlabs
- XSplit Broadcaster
- XSplit Gamecaster
- Twitch Studio
- Discord
- Google Chrome
- Battle.net Chat
- WebEx
- Skype
- Zoom
- Slack
- MS Teams
- Steam Chat

1.5.3 Lippen Synchronizität

Ein weiteres wichtiges System in Videokonferenzen ist die Audio-zu-Video-Synchronisation, oder auch lip sync genannt. Digitale oder analoge Audio-Video Streams erhalten in der Regel eine Art von Synchronisationsmechanismus. Entweder in Form von verschachtelten Video- und Audiodaten oder durch explizite timestamps von Daten.

1.5.3.1 WebRTC

Die Lippensynchronisation is ein Prozess, der auf der Empfängerseite des WebRTC stattfindet, wo die Audio- und Videospuren synchronisiert werden [11].

Wenn Medien in WebRTC erfasst werden, versieht der Urheber der Medien die Rohdaten mit einem Zeitstempel, die dann kodiert und übertragen werden. Während dieses Prozesses werden Audio und Video getrennt behandelt, da sie verschiedene Encoder mit unterschiedlichem Verhalten und Strategien durchlaufen. Die Absicht des Absenders ist es, die Medien so schnell wie möglich zu senden, ohne die Performance zu beeinträchtigen [11].

Die empfangende Seite sammelt die Medienpakete, leitet sie durch einen Jitter Buffer und verzögert die Video- oder Audiodaten, um die Lippensynchronisation zu erreichen. Dies geschieht durch den Abgleich der Zeitstempel in den verschiedenen Medienspuren. Wie lange eine Datei im Buffer zurückgehalten wird, hängt davon wie groß der Buffer ist [11].

Er arbeitet nach dem FIFO-Prinzip (First In First Out). Die Datenpakete, die zuerst eintreffen werden zuerst wieder ausgegeben. Es kommt zu keiner Veränderung der Paketreihenfolge durch den Buffer.
Je größer der Jitterbuffer ist, desto größere Laufzeitunterschiede kann er ausgleichen. Aufgrund der Zwischenspeicherung der Daten im Buffer erhöht sich jedoch deren Gesamtlaufzeit [25].

Bei VoIP ist ein Jitter Buffer ein Bereich für gemeinsam genutzte Daten. In diesem Puffer lassen sich Sprachpakete sammeln, speichern und an den Sprachprozessor in Intervallen mit gleichen Abständen weiterleiten. Wegen Netzwerkstau, Timing Drift oder Änderungen bei der Route können Abweichungen bei der Ankunft der Pakete auftreten. Dieser Umstand wird als Jitter bezeichnet.

Dabei unterscheidet man zwischen statischen und dynamischen Jitter-Buffern. Ein statischer Jitter Buffer ist Hardware-basiert und wird vom Hersteller konfiguriert. Ein dynamischer Jitter Buffer basiert auf Software und lässt sich vom Netzwerk-Administrator konfigurieren. Somit kann er auf Änderungen bei den Verzögerungen im Netzwerk reagieren [25].

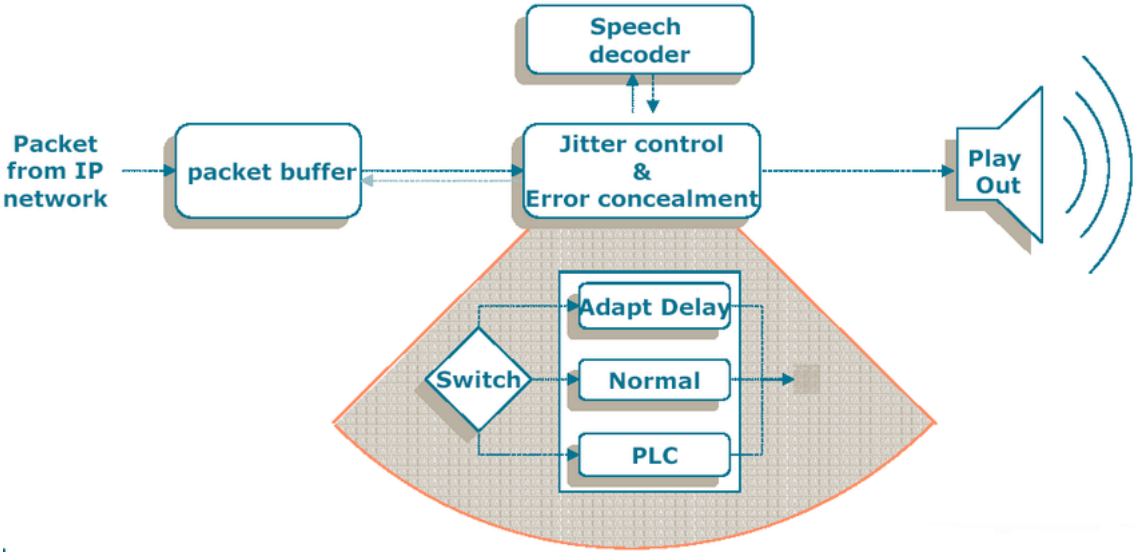


Abbildung 1.11: Visualisierung eines Jitter-Buffer Moduls [12]

1.5.3.2 Ein Ausblick auf die Zukunft

Durch den Einsatz von künstlicher Intelligenz haben Informatiker des Max Planck Instituts für Informatik, Saarbrücken, ein Softwarepaket entwickelt, das den Gesichtsausdruck von Schauspielern an eine synchronisierte Filmversion anpassen kann.
Die Software kann jedoch auch zur Korrektur von Blick- und Kopfhaltung in Videokonferenzen verwendet werden [19].
Das Projekt mit dem Namen Deep Video Portraits [19] kann im Gegensatz zu früheren Methoden, die sich nur auf die Bewerbungen des Gesichtsinneren konzentrieren, auch das gesamte Gesicht einschließlich Augen, Augenbrauen und Kopfposition in Videos animieren. Es kann sogar einen plausiblen statischen Videohintergrund synthetisieren, wenn der Kopf bewegt wird.

Laut Hyeongwoo Kim vom Max-Planck-Institut für Informatik arbeitet man zur Zeit mit modellbasierten 3D-Gesichtsaufnahmen, um die detaillierten Bewegungen der Augenbrauen, des Mundes, der Nase und der Kopfposition des Synchronsprechers in einem Video aufzuzeichnen. Das System überträgt diese Bewegungen auf den Ziel-Akteur im Film, um die Lippen und Gesichtsbewegungen exakt mit dem neuen Ton zu synchronisieren.
Die Forschung befindet sich derzeit im Proof-of-Concept-Stadium und muss noch in die Praxis umgesetzt werden, doch die Forscher gehen davon aus, dass dieser Ansatz Teile der visuellen Unterhaltungsindustrie massiv verändern könnte [19].

Diese Technologie kann ebenfalls dazu verwendet werden um Ton und Bild in Videokonferenzen zu synchronisieren [19].

2 Video Streaming

2.1 On-Demand Video Streaming

Das On-Demand Video Streaming beschreibt den Abruf einer bereits existierenden Quelle und lässt sich dahingehend grundlegend zu Echtzeit Kommunikationssystemen abgrenzen, die eine Quelle in Echtzeit übertragen.
Ziel ist es, den Konsumenten eine möglichst gute Bild- und Tonqualität auf Abruf zur Verfügung zu stellen und gleichzeitig ein unterbrechungsfreies Abspielen zu gewährleisten. Hierbei liegt das Zufriedenheitsgefühl des Konsumenten im Vordergrund, welches durch diverse Techniken maximiert wird.

Mit mehreren Millionen Konsumenten weltweit, die zeitgleich digitale Inhalte streamen, stellen sich hohe Anforderungen an das bereitstellende System welches innerhalb weniger Sekunden eine hohe Qualität des Inhaltes bereitstellen muss. Erschwert wird dies durch verschiedene Faktoren, angefangen mit Video-Codecs des Quellmaterials bis hin zur Internet Anbindung des individuellen Konsumenten. Auf diese Problematik und Lösungsansätze wird näher in Unterabschnitt 2.1.4 eingegangen.

Die folgenden Unterkapiteln fokussieren sich auf den Anbieter Netflix, der mit vielen Open-Source Komponenten und Veröffentlichungen einen detaillierten Einblick in die Plattform bietet, als Anbieter wie Amazon Prime oder Sky.

2.1.1 Netflix Architektur

Neben dem Streamen von Inhalten müssen Plattformen auch die Authentifizierung und Autorisierung von Konsumenten abdecken sowie weitere Funktionalitäten wie eine Such-Funktion oder Vorschlage basierend auf dem aktuellen Nutzerverhalten mit einfließen lassen.

Bei Netflix kommen hierbei eine Vielzahl von Microservices zum Einsatz, die anders als eine klassische 3-Tier Architektur (Webserver, Appserver, Datenbank) verteilt agieren und ein komplexes Kommunikationsnetz bilden. Das Konzept von Microservices basiert auf der Unix Philosophy *do one thing, and do it right* und unterteilt Geschäftsprozesse in viele eigenständige Bereiche, die autark aktualisiert werden können. Im direkten Vergleich verschiebt sich die Komplexität von der Anwendungsebene in die Kommunikationsebene [31]. Welche Services im Detail von Netflix zur Verfügung gestellt werden ist nicht öffentlich, mehrere Vorträge und Einblicke in Blog-Beiträgen lassen jedoch darauf schließen, dass sich die Zahl auf mehr als 100 aggregiert [31][35].

Die Bereitstellung der Inhalte ist vergleichsweise simpel und wird erst im letzten Schritt komplex:

1. Das Quell-Material von Produzenten wird in der originalen Qualität gespeichert
2. Das Quell-Material wird in verschiedene Formate und Bitraten Transkodiert
3. Transkodierte Inhalte werden über ein Content Delivery Network (CDN) an die Endgeräte ausgeliefert

Der zweite und dritte Schritt stellen hierbei die Komplexität dar. Unterschiedliche Geräte, als Kontrastbeispiel ein älteres Smartphone und ein 4k-fähiger Fernseher, benötigen unterschiedliche Formate und Bitraten, welche schlussendlich auch von der Internet Anbindung abhängig sind. Auf die genaue Funktionsweise wird in den folgenden Unterkapiteln eingegangen.

2.1.2 Netflix Infrastruktur und Content Delivery Networks

Nachdem Netflix in den letzten Jahren rasant an Popularität gewann, wurde die Entscheidung getroffen, keine eigenen Rechenzentren mehr zu betreiben und stattdessen Dienste in die Amazon Web Services Public Cloud zu migrieren. Eine Analyse aus dem Jahr 2012 ergab zudem, dass Netflix auf drei große CDN Anbieter (Level 3, Limelight, Akamai) setzte um Inhalte auszuliefern [2]. Mit weiterem Wachstum von Netflix stieg der globale Anteil an Netzwerkverkehr der auf Netflix Streaming Dienste zurückzuführen war stark an (bereits 2012 zu Spitzenzeiten > 25% [2]), was zu einer engeren Zusammenarbeit mit Internet Service Provider (ISP) führte.

Die steigenden Anzahl der Nutzer und verfügbarer Inhalte spiegelte sich auch in dem Bedürfnis, eine größere Kontrolle über die Caching-Struktur zu erhalten.

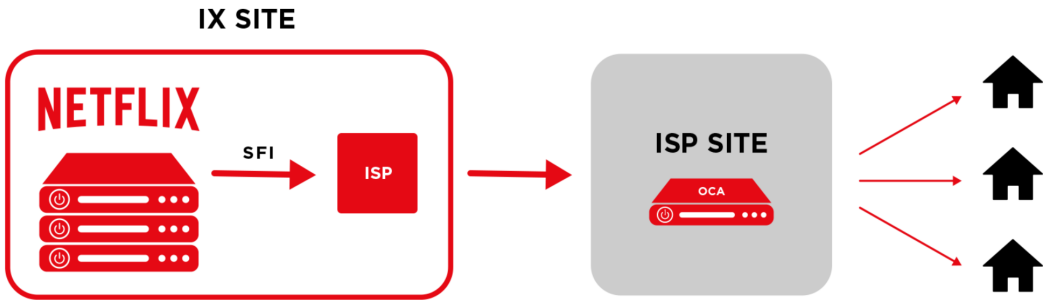


Abbildung 2.1: Internet Exchange Point (IXP) basierter Netzwerkaufbau für Open Connect [34]

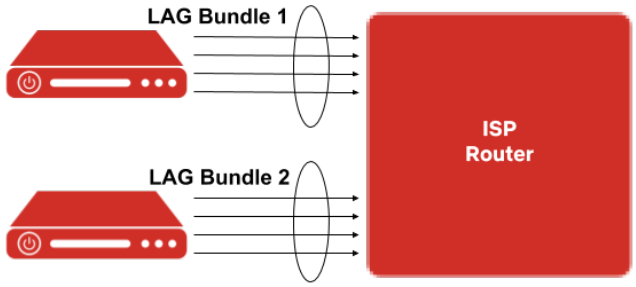


Abbildung 2.2: Private Network Interconnect (PNI) basierter Netzwerkaufbau für Open Connect [33]

Beide Punkte führten zu dem Start der Open Connect Initiative, einem Netflix eigenen CDN. Dieses basiert auf mehreren Open Connect Appliance (OCA), die einen Cluster bilden und über einen Private Network Interconnect (PNI) (Abbildung 2.2) oder eine Settlement-free Interconnection (SFI) an einem IXP (Abbildung 2.1). Eine OCA ist speziell angepasste Hardware, die auf Open-Source Software aufbaut. Verschiedene optimierten Instanz-Typen erlauben einen effektiven Datendurchsatz zwischen 8 und 36 Gbps und stellen bis zu 288TB an lokalem Speicher bereit (Stand: 23.06.2020)[32]. Als Betriebssystem kommt FreeBSD zum Einsatz. Die Internet-Konnektivität (Dual Stack) wird von dem Routing Daemon BIRD sichergestellt und der Cache mit Nginx ausgeliefert [32].

Wie Abbildung 2.3 zu entnehmen ist, werden die sogenannten Edge Caches über eine Kontrolleinheit in Amazon Web Services (AWS) gesteuert. Das genaue Caching-Verhalten wird im folgenden Unterkapitel beschrieben.

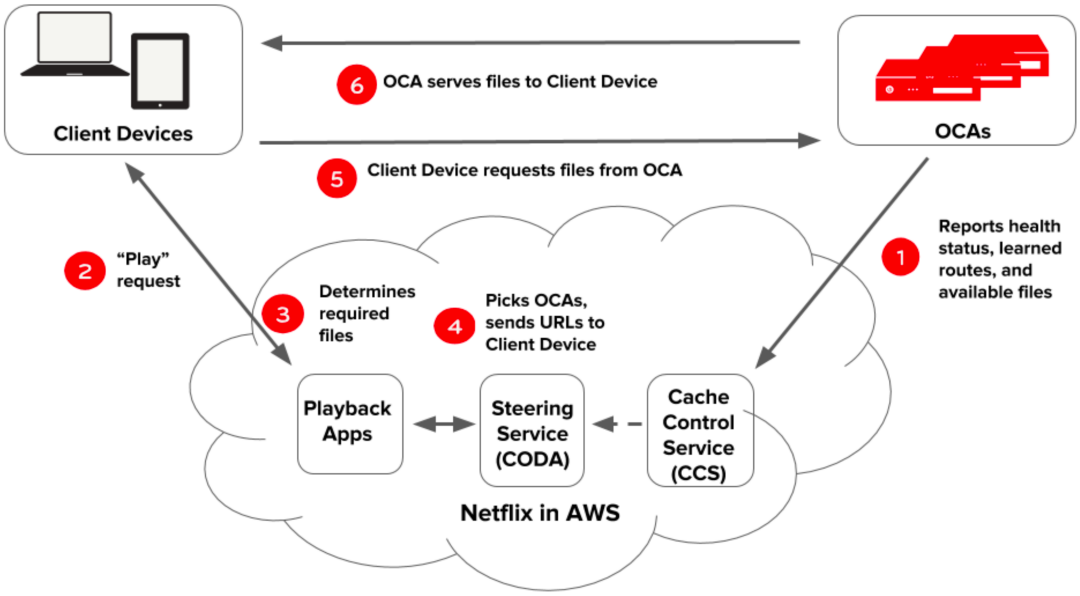


Abbildung 2.3: Netflix Open Connect Architecture [34]

2.1.3 Dynamic Adaptive Streaming Over HTTP (DASH)

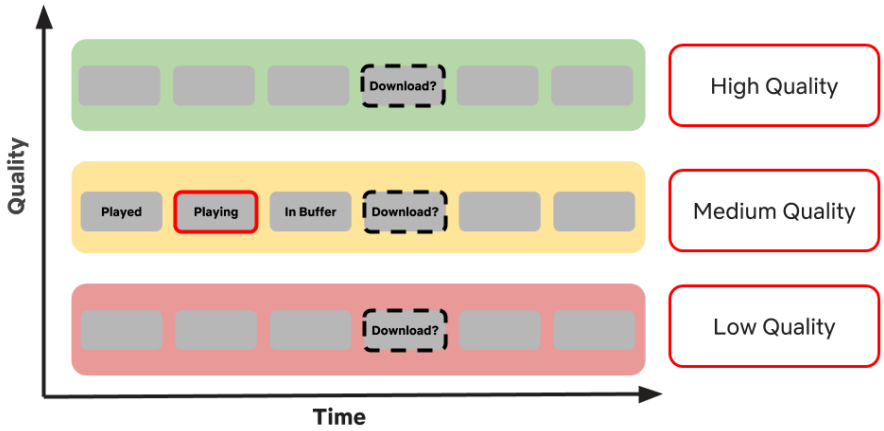


Abbildung 2.4: DASH - abrufen des nächsten Chunk [36]

Netflix nutzt Dynamic Adaptive Streaming Over HTTP (DASH) als Streaming Protokoll. Hierbei werden die in viele Formate und Bitraten kodierten Inhalte in kleine Blöcke von wenigen Sekunden Länge unterteilt. Der Netflix Video Player (plattformunabhängig), im Folgenden als Client bezeichnet, fragt einen Block nach dem anderen über das HTTP Protokoll ab [2]. Bei Download wird die Übertragungsgeschwindigkeit ermittelt um die optimale Bitrate des nächsten Block zu ermitteln (Abbildung 2.4). Eine einfache Technik hierbei ist, die höchstmögliche Bitrate für den nächsten Block auszuwählen, sodass der Download vor Ende des derzeit abgespielten Blocks abgeschlossen ist. Um ein Puffern des Videos zu verhindern, werden meist n Segmente im Voraus heruntergeladen, sodass immer das n+1 Element angefragt wird, welches nicht direkt im Anschluss abgespielt wird. Mit dieser Pufferverwaltung ist es möglich, kurzzeitige Netzwerkaussetzer beispielsweise, im Mobilfunk, vor dem Konsumenten zu verbergen.

2.1.4 Optimierungen in DASH

Optimierungen in Protokollen

Da DASH auf das HTTP Protokoll aufbaut, kann es von diversen Optimierungen im selbigen profitieren. Mit HTTP/2 wurde die Latenz als auch der Protokoll-Overhead reduziert. Insbesondere HTTP/3 mit UDP als Transport-Protokoll bringt Verbesserungen für Übertragungen im Mobilfunk, wo HTTP/2 mit persistenten TCP Verbindungen keine Vorteile bieten kann. Vergleiche von Cloudflare zeigen, dass sich insbesondere die Latenz reduziert (Abbildung 2.5). Mit Weiterentwicklungen in TLS wurde neben der Sicherheit auch die Geschwindigkeit erhöht. Mit TLS 1.3 wurde die Round-Trip-Time für einen vollen Handshake auf eins reduziert, was sich auch in der Playback Latenz widerspiegelt (Abbildung 2.6) [30].

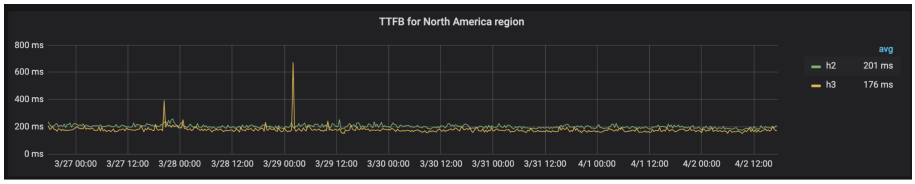


Abbildung 2.5: Vergleich der TTFB zwischen HTTP/2 und HTTP/3 [7]

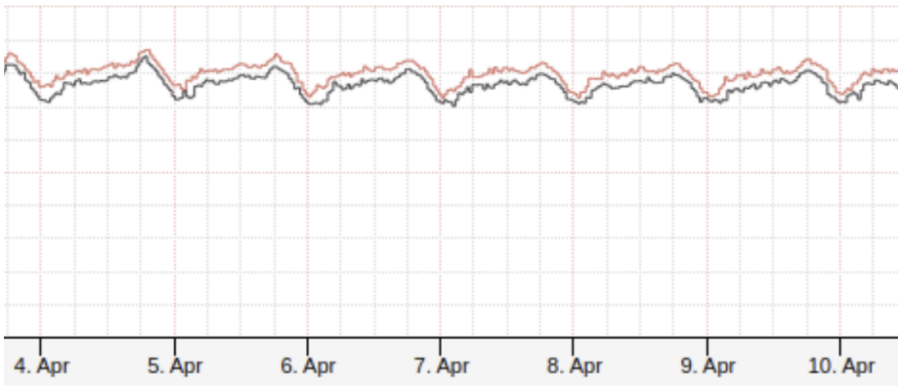


Abbildung 2.6: Auswirkungen auf die Abspiel-Verzögerung zwischen TLS1.2 und TLS1.3 [30]

Optimierungen mit neuen Technologien

Neben den inkrementellen Verbesserungen der unterliegenden Protokolle können Optimierungen auch in der Wahl der angefragten Daten liegen. Netflix nutzt hierbei Metriken des Konsumenten, beispielsweise welches Endgerät er benutzt, um den bestmöglichen, von dem Gerät möglichst in Hardware unterstützten Codec zu wählen.

Insbesondere bei mobilen Endgeräten nutzt Netflix Machine-Learning, um basierend auf der Bandbreitenmessungen der letzten 15 Minuten (Abbildung 2.7) die Bandbreite der nächsten 15 Minuten vorherzusagen.

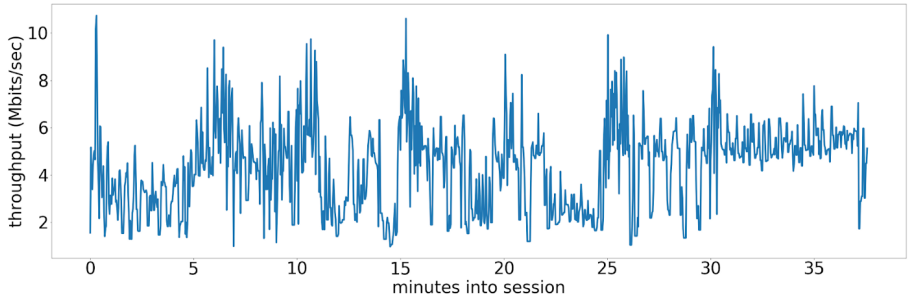


Abbildung 2.7: Bandbreiten-Aufzeichnung eines Endgerätes bei Netflix [36]

Das Netflix-eigene CDN erlaubt ebenfalls, angefragte Daten geographisch möglichst nah an den Konsumenten zu bewegen. Die Popularität von Serien und Filmen sowie das regionale Verhalten der Nutzer wird zudem dazu verwendet, um auf den CDN möglichst relevante Daten bereits über Nacht zu synchronisieren, um Ladezeiten zu verringern [36].

2.2 Live Video Streaming

Live Video Streaming beschreibt die Übertragung eines Videosignals nahezu in Echtzeit. Dieses Videosignal stammt im Vergleich zu On-Demand Streaming nicht von einer bereits existierenden Quelle, sondern wird gerade erzeugt im Sinne einer Kamera- oder Bildschirmaufnahme. Des weiteren steht bei Livestreaming keine unbegrenzte Zeit vor der Wiedergabe bereit, sodass ein Stream zum weiteren Transport schnellstmöglich passend verarbeitet werden muss.

2.2.1 Real-Time Messaging Protocol

Das Real Time Messaging Protocol (RTMP)-Protokoll ist ein von Adobe entwickeltes proprietäres Streamingprotokoll, welches ursprünglich für die Verwendung mit Flashplayern gedacht war. Adobe legte die Spezifikationen 2009 offen.

RTMP ist in der Lage Video, Audio und auch generell Daten zu übertragen. Zur Übertragung werden folgende Codecs unterstützt[42]:

- Audio
  - AAC
  - AAC-LC
  - HE-AAC+ v1 & v2
  - MP3
  - Speex
  - Opus
  - Vorbis
- Video
  - H.264
  - VP8
  - VP6
  - Sorenson Spark
  - Screen Video v1 & v2

RTMP arbeitet basierend auf TCP und stellt somit eine verlustfreie Übertragung sicher. Es gibt jedoch weiter Varianten:

RTMPS über TLS/SSL
RTMPE verschlüsselt
RTMPT verkapselt in HTTP zur Überbrückung von Firewalls
RTMFP über UDP statt TCP, ermöglicht Peer-to-Peer

RTMP baut zu Beginn eine persistente Verbindung auf. Dies vermeidet Overhead für neue Verbindungen, was die Latenz senkt. Zur Übertragung über TCP wird der Stream in Fragmente geteilt, die Größe hierfür wird dynamisch zwischen Server und Client ausgehandelt. Die Standard Fragmentgröße beträgt 128 Bytes für Video und 64 Bytes für Audio.

2.2.2 HTTP-Live Streaming (HLS)

Das HLS-Protokoll ist ein von Apple entwickeltes Protokoll, das es ermöglicht Livestreaming Video mit einem normalen Webserver zu übertragen. Bevor Video übertragen werden kann, muss der ankommenden Stream codiert und zerteilt werden. Hierzu wird serverseitig ein Media Encoder, ein Stream Segmenter und ein Distributor eingesetzt.

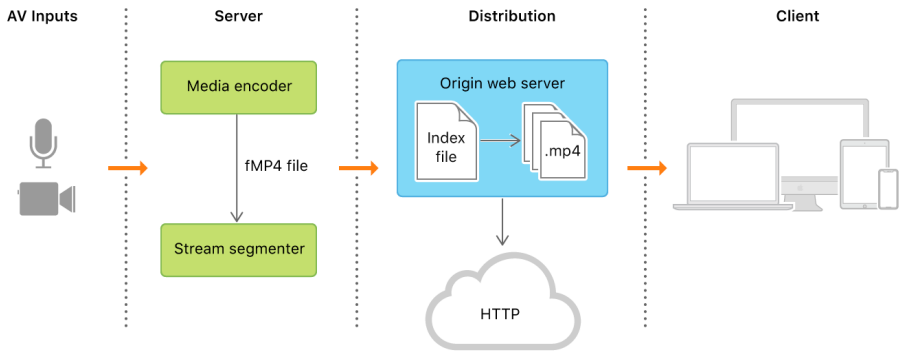


Abbildung 2.8: Komponenten HTTP Live Stream [15]

Komponenten für einen HLS Stream:

Media Encoder

Der Media Encoder codiert den Videostream als H.264 oder H.265(HEVC) und packt diesen als fragmented MP4 (fMP4) oder in einem MPEG2 Transport Stream. Audio kann direkt in MP4 embedded werden oder in einem separaten MPEG elementary stream transportiert werden. Es ist möglich, dass vom Encoder verschiedene Qualitätsstufen bereitgestellt werden.

Stream Segmenter

Der Stream Segmenter teilt den vom Media Encoder bereitgestellten Stream in Fragmente gleicher Größe. Parallel dazu wird ein Indexfile im Format .m3u8 erzeugt.

Diese Datei wird als Playlist vom Client heruntergeladen, um in der richtigen Reihenfolge auf die Transport-Stream-Segmente zuzugreifen.

Distributor

Der Distributor liefert den Stream an den Client aus. Ein Webserver stellt die Masterplaylist, welche die Infos zu verschiedenen Qualitäten enthält, sowie die Indexplaylisten der einzelnen Qualitätsstufen bereit.

Client

Der Client lädt per HTTP die Masterplaylist herunter. Anhand der dort enthaltenen Informationen wird eine passende Qualitätsstufe ausgewählt. Die Auswahl kann die Clientsoftware anhand von Parametern treffen, wie der zur Verfügung stehenden Bandbreite oder den verfügbaren Hardwaredecodern. Im Anschluss an die Qualitätsauswahl wird deine Indexdatei heruntergeladen. Die Indexdatei beinhaltet die Location der Mediendateien. Der Client beginnt die Mediendateien sequenziell herunterzuladen. Die einzelnen Videofragmente werden von dem Client aneinandergereiht und wiedergegeben. Wenn der Download eines Fragements fehlschlägt, ist in der Regeln genügend Zeit, den Download zu wiederholen, ohne das die Wiedergabe unterbrochen werden muss.

2.2.3 YouTube

YouTube als größte Video-on-Demand Plattform bietet als eine der wenigen Plattformen Livestreaming von Inhalten mit einer Auslösung von 2160p60 an. Hierzu sind jedoch hohe Bandbreiten am Aufnahmeort erforderlich.

YouTube stellt drei verschiedene Latenzen zur Übertragung eines Livestreams bereit. Die Latenzeinstellungen beeinflussen die Verfügbarkeit von Funktionen und höherer Auflösungen [23]:

normale Latenz

YouTube empfiehlt diese Stufe für Streams die keine Interaktion mit dem Publikum enthalten. Es stehen alle Qualitätsstufen und Funktionen zur Verfügung.

niedrige Latenz

Bei dieser Einstellung wird durch die geringere Latenz eine Interaktion mit Zuschauern möglich. Es wird kein Stream mehr mit einer Auflösung von 2160p unterstützt.

extrem niedrige Latenz

Bei dieser Stufe sollen Live-Interaktionen möglich werden, jedoch geht dies zu Lasten der Stabilität des Streams. Auflösungen größer als FullHD (1080p) werden nicht mehr unterstützt.

Zur Entgegennahme eines Livestreams stellt YouTube drei Protokolle zur Verfügung:

RTMP

Unter Verwendung von RTMP kann ein Stream mit H.264 Codecs entgegengenommen werden. Es stehen alle Latenz Optionen zur Verfügung.

HLS

HTTP Live Streaming (HLS) unterstützt die Übertragung von H.264 und H.265(HEVC) Inhalten. Die Unterstützung von Inhalten mit höherer Auflösung wie 2160p ist hier durch modernere Codes wie HEVC besser gewährleistet. Mit HLS wird kein ultra-low latency Stream unterstützt.

DASH

Mithilfe von DASH können Streams codiert in H.264 und VP9 übertragen werden. Der Support von VP9 ermöglicht einen besseren Support für hochauflösende Inhalte. Auch mit DASH wird der ultra-low latency Stream nicht unterstützt.

Zur Auslieferung eines Streams werden DASH und HLS verwendet. Jedoch wurde von Google in Verwendung mit dem hauseigenen Browser Chrome auch noch das Protokoll QUIC angeboten.

2.2.3.1 Quick UDP Internet Connections

Bei dem Quick UDP Internet Connections (QUIC) Protokoll handelt es sich um ein experimentelles Netzwerkprotokoll welches zu Beginn, von Google entwickelt wird und seit 2017 von der IETF zu einer Standardisierung ausgearbeitet wird. QUIC mit UDP wird in HTTP/3 zum Einsatz kommen und wird dort TCP als Transportprotokoll ablösen.

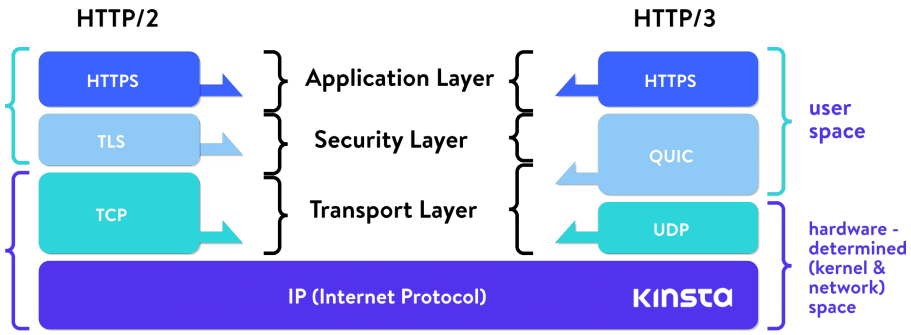


Abbildung 2.9: HTTP/2 vs HTTP/3 Layer [20]

QUIC verfolgt den Ansatz Probleme, die durch Verwendung von TCP und HTTP/2 Funktionen entstehen zu beheben. HTTP/2 kann eine TCP-Verbindungen für mehrere Streams verwenden, geht jedoch bei einem Stream ein Paket verloren, kommt die ganze Verbindung ins Stocken, bis TCP das Paket erneut gesendet hat. Ein Paket eines Streams blockiert in diesem Fall unabhängige andere Streams. Mit der Verwendung von QUIC mit UDP soll dieses Problem behoben werden. Dies führt auch bei bereits übertragenen Paketen zu einer Blockierung im Zielknoten. QUIC ermöglicht zuverlässige Übertragungen trotz der Abschaffung von TCP. Oberhalb der UDP-Schicht ist es möglich die Übertragung von Paketen erneut anzufordern.

In Regionen mit verlustreichen Verbindungen kann dies dazu führen, dass der Einsatz von HTTP/2 keinen echten Vorteil gegenüber HTTP/1 bildet, da fehlende Pakete den Kompletten Transfer beeinflussen.

Mit QUIC wird eine verschlüsselte Verbindung sichergestellt. Es gibt keine Klartextversion dieses Protokolls. Zur Verschlüsselung komm TLS 1.3 zum Einsatz. Im Vergleich zu älteren TLS Versionen trägt es durch die geringe round trip time zur Geschwindigkeit des Protokolls bei.

Ob QUIC direkt mit der Fertigstellung der Standardisierung zum Einsatz komm ist fraglich. Die Bereitstellung könnte auf Grund von fehlender Optimierung noch dauern. Durch den flächendeckenden Einsatz von TCP, ist der TCP-Stack in unter Linux besser optimiert als UDP. TCP- und TLS-Offloading für Hardware ist bereits vorhanden, fehlt bei UDP meistens und spezifisch für QUIC ist diese nicht existent.[45]

2.2.4 Twitch

Twitch ist aktuell eine der größten und beliebtesten Live-Streaming-Plattformen der Welt mit durchschnittlich 2 Millionen permanenten Zuschauern. Twitch bietet eine Plattform, die weit über nur Videostreaming hinausgeht. Neben einen Chat für die Zuschauer, wird von Twitch eine API bereitgestellt, die es erlaubt Extensions zu programmieren, mit denen direkt auf Aktionen der Zuschauer im Live-Stream reagiert werden kann.

2.2.4.1 Twichttranscoder

Bei Livestreaming spielt besonders das Thema Latenz eine Rolle. Nicht nur das es zu Verzögerungen durch den Transport kommt, außerdem müssen die Streams noch weiterverarbeitet werden. Das CDN von Twitch ist hierzu in 4 Komponenten geteilt:

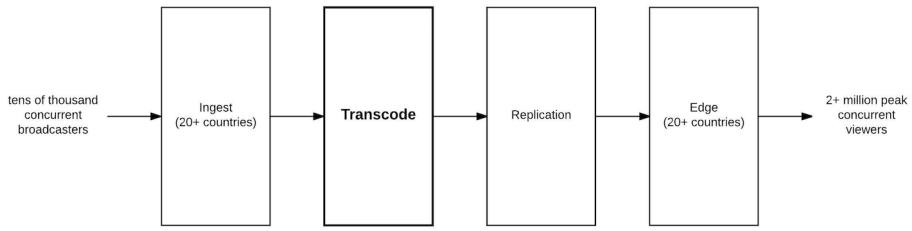


Abbildung 2.10: Twitch CDN [65]

Ingest Stream wird vom Broadcaster entgegen genommen
Transcode Stream wird verarbeitet zur weiteren Verteilung
Replication Stream wird vervielfältigt und an Edgeknoten versendet
Edge Stream wird zum Zuschauer transportiert

Dieses Kapitel wird sich hauptsächlich mit der Transcodekomponente beschäftigen.

Twitch erhält den Stream der Broadcaster per RTMP und gibt dieses anschließend an den Transcoder weiter. Der Transcoder transcoded und transmuxed den Stream um diesen als HLS Stream weiter verteilen zu können. Zur Verteilung sollen aus einen folgende Streamvarianten generiert werden:

- H.264/HLS 1080p60
- H.264/HLS 720p60
- H.264/HLS 720p30
- H.264/HLS 480p30
- H.264/HLS 360p30
- H.264/HLS 160p30

Zur Umwandlung des eingehenden Streams kann freie Software wie FFmpeg verwendet werden. Jedoch ergeben sich hiermit einige Nachteile, auf welche nachfolgend im Vergleich zum hauseigenen Twichtdecoder eingegangen wird.

Angenommen ein Stream codiert in H.264, gesendet in RTMP, mit einer Auflösung von 1080p60 soll in diese 4 HLS Varianten codiert werden:

- H.264/HLS 1080p60
- H.264/HLS 720p60
- H.264/HLS 720p30
- H.264/HLS 480p30

Zur Umsetzung mit FFMPG können alle vier Streams in je einer FFmpeg Instanz parallel verarbeitet werden. Da durch Encoding eine hohe CPU-Last verursacht wird, wird idealerweise ein Stream transmuxed statt transcodiert. Im Fall von Transmuxing werden Audio und Video nicht angefasst, es wird lediglich der Stream neu organisiert, da der Codec H.264 sowohl von RTMP als auch HLS unterstützt wird. Hierdurch kann Zeit und Rechenleistung eingespart werden.

Unter Verwendung von Transmuxing, können mit FFmpeg Probleme in der Segmentgröße auftauchen. Zu Beginn eines jeden Segments erwartet HLS einen IDR Frame. Ein IDR Frame gibt an, wann sich die darauf folgenden Bilder nicht mehr auf ein Bild davor beziehen können [37]. Unter Verwendung von Transcoding und Transmuxing sind die Segmente der verschiedenen Qualitätsstufen nicht mehr zueinander ausgerichtet. Dies geschieht dadurch, dass an dieser Stelle beim transmuxen die IDR Frames nicht verändert werden. Sie werden wie von der Streamquelle gesetzt weiterverwendet. Durch die verlorene Ausrichtung(2.11) der Segmente ist es einem HLS Player nicht mehr möglich zwischen den verschiedenen Qualitätsstufen zu wechseln.

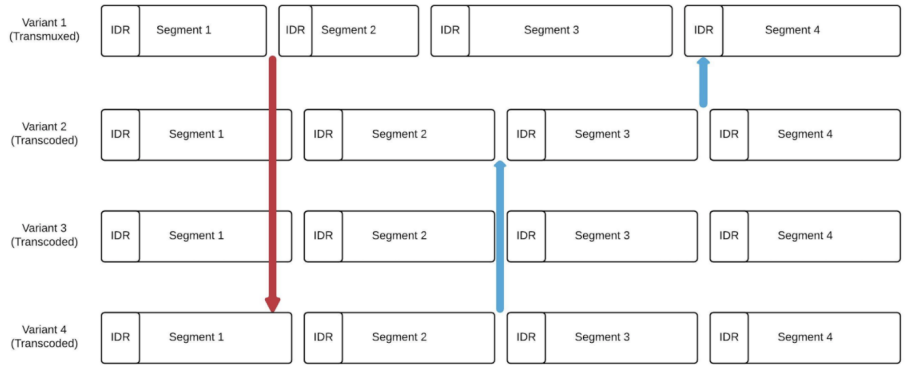


Abbildung 2.11: falsche Segmentausrichtung bei FFmpeg [65]

Twitch löst dieses Problem in ihrem eigenen Transcoder, in dem alle transcodierten Varianten sich in ihrer Segmentgröße an der Segmentgröße orientieren, die vom RTMP-Stream festgelegt wurde.

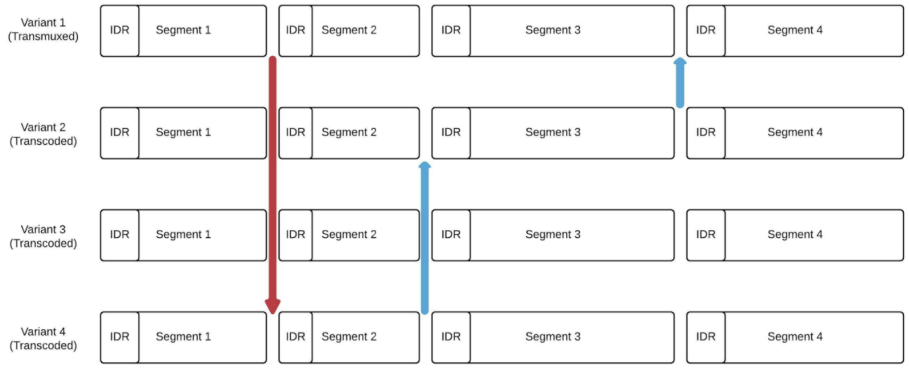


Abbildung 2.12: angepasste Segmentausrichtung bei Twitchencoder [65]

Diese Problematik lässt sich mit FFmpeg nur lösen, wenn die höchste Variante (1080p60) auch transcodiert wird. So würden alle Varianten eine einheitliche Segmentgröße haben. Ein weiteres Hindernis stellt die Unabhängigkeit der einzelnen Instanzen dar. Variante 2 bis 4 erhalten beim Transcodieren keine Informationen darüber, dass die Segmentgrößen voneinander abweichen.

Eine weitere wichtige Rolle beim Erzeugen der drei übrigen HLS Varianten spielt Redundanz. Beim Transcodieren eines Videos, in eine bestimmte Auflösung und Format, wird dieses zuerst decodiert, im Anschluss skaliert und wieder codiert. Wird jedoch die selbe Quelle verwendet um daraus mehrere Varianten zu transcodieren, so wird unter normalen Umständen das Video für jede Variante decodiert. Twitch vereint im Twitchdecoder diesen Schritt um die Performance und Effizienz zu steigern.

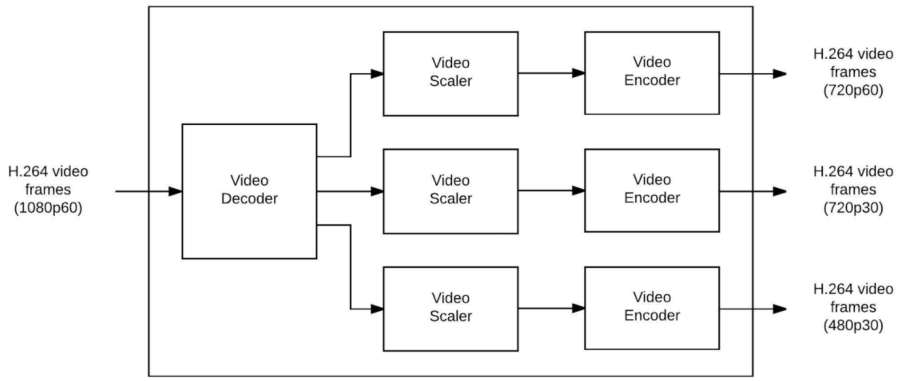


Abbildung 2.13: gemeinsamer Decoder für 3 Videostreams [65]

Es gilt jedoch noch eine weitere Redundanz zu entfernen. Varianten die die selbe Auflösung besitzen, jedoch eine unterschiedliche Bildrate haben, können sich einen Scaler teilen. Da die Skalierung ein sehr rechenintensiver Prozess ist, kann hiermit die Performance weiter verbessert werden.

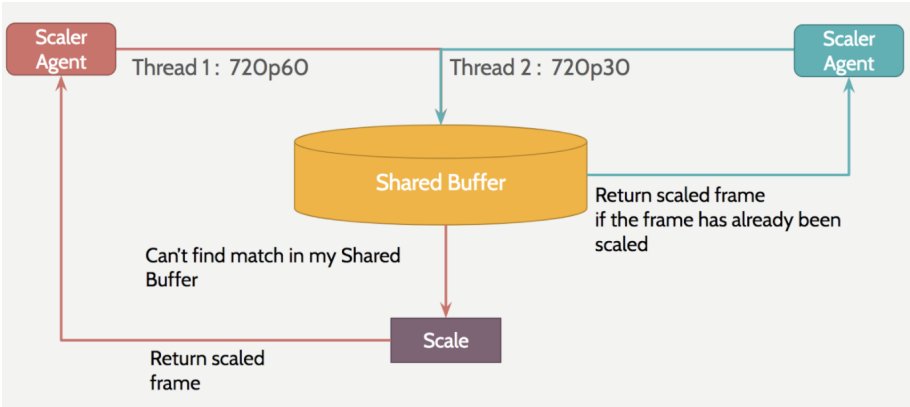
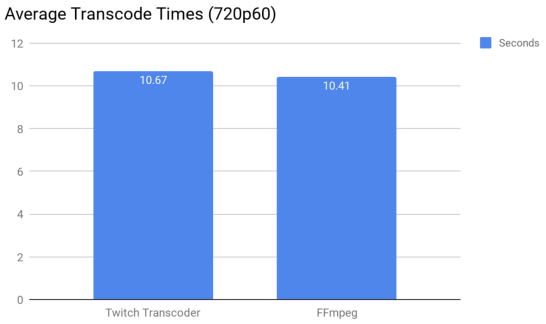


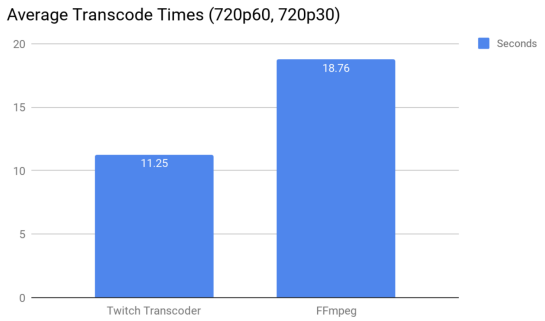
Abbildung 2.14: gemeinsamer Decoder für 3 Videostreams [65]

Nachdem Decoder und Scaler geteilt werden, werden zur effizienten Nutzung von modernen Prozessoren alle Varianten parallel verarbeitet.

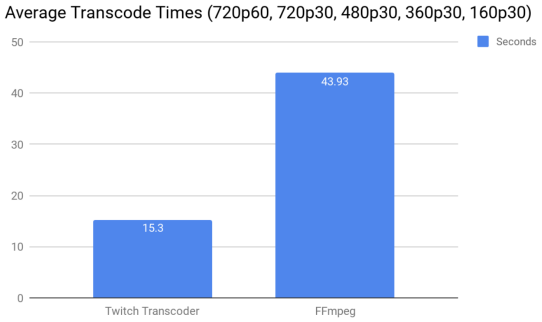
Um eine bessere Vorstellung der Unterschiede zu bekommen, lohnt es sich die transcode Zeiten von FFmpeg und dem TwitchTranscoder im Vergleich anzuschauen.



(a) Eine Ausgabevarinate



(b) Zwei Ausgabevarinaten



(c) Fünf Ausgabevarinaten

Abbildung 2.15: Trancode Zeit FFmpeg vs. TwitchTranscoder [66]

Bei der Ausgabe eines einzelnen Streams ist der Twichtencoder langsamer, was daran liegt, dass dieser normal viel mehr Aufgaben zusätzlich abdeckt. In diesem Fall erfüllt er jedoch keine Funktion. Die Darstellung in 2.15c entspricht den typischen genutzten Qualitätsstufen. Twitch gibt an im Vergleich zu FFmpeg, mit dem TwitchTranscoder bis zu 65% Ausführungszeit zu sparen [66].

3 Fazit

Systeme für Echtzeitkommunikation mit Video- und Audioübertragung von allen Teilnehmern, Livestreaming und Video-On-Demand Streaming scheinen sich auf den ersten Blick drastisch voneinander abzugrenzen. Dies spiegelt sich in den Kommunikationsstrukturen wieder, nicht aber in den unterliegenden Grundbausteinen wie Video- und Audiocodecs. Besonders Live-Streaming und Video-On-Demand Streaming baut auf ähnlichen Protokollen auf, beispielsweise die Segmentierung von Video- und Ton-Daten in kleine Blöcke, die individuell übertragen werden. Lediglich die Erzeugung und Aufbereitung derer unterscheiden sich.

Insgesamt profitieren alle Kommunikationstechnologien von dem technologischen Fortschritt in offenen Standards. Insbesondere Livestreaming und On-Demand Streaming können großen Nutzen von neuen HTTP-Standards und Weiterentwicklungen der Transport Layer Security.

Abkürzungsverzeichnis

CDN Content Delivery Network
OCA Open Connect Appliance
PNI Private Network Interconnect
SFI Settlement-free Interconnection
ISP Internet Service Provider
IXP Internet Exchange Point
DASH Dynamic Adaptive Streaming Over HTTP
AEC Acoustic Echo Cancellation
AECM Acoustic Echo Cancellation for MOBILE
AI Artificial Intelligence
VoIP Voice over IP
OBS Open Broadcaster Software
RTMP Real Time Messaging Protocol
HLS HTTP Live Streaming
QUIC Quick UDP Internet Connections
NAT Network Adress Translation
WebRTC Web Real Time Communication
DTLS Datagram Transport Layer Security
SRTP Secure Real Time Transport Protocol
STUN Simple Traversal of UDP through NAT
XMMP Extensible Messeging and Presence Protocol
IETF Internet Engineering Task Force
MUC Multi User Chat
SIP Session Initiation Protcol
SDP Session Description Protocol
RTP Real-Time Transport Protocol

TLS Transport Layer Security
SIPS Session Initiation Protcol Secure
HEVC High Efficiency Video Coding
MPEG Movie Picture Experts Group
AWS Amazon Web Services

Abbildungsverzeichnis

Literatur

[1] Jain Richardson Abharana Bhat. *How to stream better quality video: Part 3 - Ultra High Definition, 4K and next generation video codecs.* 5. Sep. 2014. URL: https://www.vcodex.com/news/how-to-stream-better-quality-video-part-3-ultra-high-definition-4k-and-next-generation-video-codecs/ (besucht am 17. 07. 2020).

[2] Vijay Kumar Adhikari u. a. „Unreeling netflix: Understanding and improving multi-CDN movie delivery". In: *2012 Proceedings IEEE INFOCOM*. IEEE, März 2012. DOI: 10.1109/infcom.2012.6195531.

[3] Henning Baset Salman A. und Schulzrinne. *An Analysis of the Skype Peer-to-Peer Internet Telephony Protocol.* 12. Sep. 2004. URL: https://arxiv.org/ftp/cs/papers/0412/0412017.pdf (besucht am 06. 08. 2020).

[4] BigBlueButton. *FAQ.* URL: https://docs.bigbluebutton.org/support/faq.html (besucht am 04. 07. 2020).

[5] Blackbox. *H.264 Videokompression.* URL: https://www.black-box.de/de-de/page/38306/Information/Technische-Ressourcen/black-box-erklaert/multimedia/H264-Videokompression (besucht am 15. 07. 2020).

[6] Gerardo Delgado Cabrera. *How Microsoft Teams will use AI to filter out typing, barking, and other noise from video calls.* 16. Apr. 2020. URL: https://www.nvidia.com/en-us/geforce/guides/nvidia-rtx-voice-setup-guide/ (besucht am 13. 07. 2020).

[7] Inc. Cloudflare. *Comparing HTTP/3 vs. HTTP/2 Performance. QCon 20.* 23. Juli 2020. URL: https://blog.cloudflare.com/http-3-vs-http-2/ (besucht am 23. 06. 2020).

[8] Dipl.- Ing. (FH) Stefan Luber Dipl.- Ing (FH) Andreas Donner. *Was ist SIP ?* 1. Aug. 2018. URL: https://www.ip-insider.de/was-ist-sip-a-663855/ (besucht am 14. 07. 2020).

[9] Djeek. *Microsoft Teams and the protocols it uses*. 28. Jan. 2018. URL: http://www.djeek.com/2018/01/microsoft-teams-and-the-protocols-it-uses-opus-and-mnp24/ (besucht am 06. 08. 2020).

[10] J. Rosenberg H. Schulzrinne G. Gamarillo A. Johnston J. Peterson B. Sparks M. Handley E.Schooler. *Overview of SIP Functionality*. Juli 2002. URL: https://tools.ietf.org/html/rfc3261 (besucht am 30. 06. 2020).

[11] WebRTC Glossary. *Lip Synchronization*. 31. Dez. 2016. URL: https://webrtcglossary.com/lip-synchronization/#:~:text=Lip%20Synchronization%20is%20a%20process,and%20sent%20over%20the%20network. (besucht am 06. 07. 2020).

[12] ResearchGate GmbH. *Jitter Buffer Module-Interface Diagram Network Packet Loss Handler*. URL: https://www.researchgate.net/figure/Jitter-Buffer-Module-Interface-Diagram-Network-Packet-Loss-Handler-Upon-successful-call_fig5_46422138 (besucht am 06. 07. 2020).

[13] Google. *Gruppenunterhaltungen starten – Computer*. URL: https://support.google.com/hangouts/answer/3111943?co=GENIE.Platform%3DDesktop&hl=de (besucht am 06. 08. 2020).

[14] Google. *Hangouts*. URL: https://play.google.com/store/apps/details?id=com.google.android.talk (besucht am 06. 08. 2020).

[15] *HTTP Live Streaming*. URL: https://developer.apple.com/documentation/http_live_streaming.

[16] Jens Ihlenfeld. *Google will VP8 und Opus zur Pflicht machen*. 30. Juli 2012. URL: https://www.golem.de/news/webrtc-google-will-vp8-und-opus-zur-pflicht-machen-1207-93523.html (besucht am 14. 07. 2020).

[17] Jens Ihlenfeld. *H.239 Summary*. Okt. 2014. URL: https://www.itu.int/rec/T-REC-H.239-201410-I/en (besucht am 15. 07. 2020).

[18] Jens Ihlenfeld. *Youtube führt Kompression mit VP9 als Standard ein*. 7. Apr. 2015. URL: https://www.zdnet.de/88230897/youtube-fuehrt-kompression-mit-vp9-als-standard-ein/ (besucht am 17. 07. 2020).

[19] max planck institut informatik. *Dank künstlicher Intelligenz könnte schlechte Lippensynchronisierung der Vergangenheit angehören*. 2018. URL: https://www.mpi-inf.mpg.de/de/aktuelles/pressemitteilungen/2018/dank-kuenstlicher-intelligenz-koennte-schlechte-lippensynchronisierung-der-vergangenheit-angehoeren/ (besucht am 13. 07. 2020).

[20] Tonino Jankov. *Was ist HTTP/3? – Hintergrundinformationen über das schnelle neue UDP-basierte Protokoll.* URL: https://kinsta.com/de/blog/http3/.

[21] Muaz Khan. *Echo in WebRTC; Why?* URL: https://www.webrtc-experiment.com/pdf/Echo-in-WebRTC-Why.pdf (besucht am 04. 07. 2020).

[22] krisp. *World's Best Innovative Noise Cancellation Technology Powered by Deep Neural Network.* URL: https://krisp.ai/technology/ (besucht am 05. 07. 2020).

[23] *Latenz bei Livestreaming.* URL: https://support.google.com/youtube/answer/7444635?hl=de&ref_topic=9257892.

[24] Andrew Low. *Wie funktioniert die akustische Echokompensation bei einer Videokonferenz?* 11. Sep. 2017. URL: https://www.shure.com/de-DE/konferenzen-meetings/ignite/wie-funktioniert-die-akustische-echokompensation-bei-einer-videokonferenz (besucht am 04. 07. 2020).

[25] Dipl.-Ing. (FH) Stefan Luber. *Was ist ein Jitterbuffer?* 1. Aug. 2018. URL: https://www.ip-insider.de/was-ist-ein-jitterbuffer-a-654702/ (besucht am 06. 07. 2020).

[26] Stefan Luber. *Was ist das Client-Server-Modell?* 19. Mai 2020. URL: https://www.ip-insider.de/was-ist-das-client-server-modell-a-940627/ (besucht am 06. 08. 2020).

[27] Stefan Luber. *Was ist das Peer-to-Peer-Modell?* 15. Juli 2020. URL: https://www.ip-insider.de/was-ist-das-peer-to-peer-modell-a-943950/ (besucht am 06. 08. 2020).

[28] Stefan Luber. *Was ist Peer-to-Peer (P2P)?* 1. Aug. 2018. URL: https://www.ip-insider.de/was-ist-peer-to-peer-p2p-a-654713/ (besucht am 06. 08. 2020).

[29] Hans Jörg Maron. *Skype verabschiedet sich von seinen Peer-to-Peer-Wurzeln.* 21. Juli 2016. URL: https://www.inside-it.ch/de/post/skype-verabschiedet-sich-von-seinen-peer-to-peer-wurzeln-20160721 (besucht am 06. 08. 2020).

[30] Inc. Netflix. *How Netflix brings safer and faster streaming experiences to the living room on crowded networks using TLS 1.3. QCon 20.* 23. Juni 2020. URL: https://netflixtechblog.com/how-netflix-brings-safer-and-faster-streaming-experience-to-the-living-room-on-crowded-networks-78b8de7f758c (besucht am 23. 06. 2020).

[31] Inc. Netflix. *Mastering Chaos - A Netflix Guide to Microservices*. Hrsg. von Josh Evans. 23. Juni 2020. URL: https://www.infoq.com/presentations/netflix-chaos-microservices/ (besucht am 23. 06. 2020).

[32] Inc. Netflix. *Netflix | OCAs. QCon 20*. 23. Juni 2020. URL: https://openconnect.netflix.com/de_de/appliances/#the-hardware (besucht am 23. 06. 2020).

[33] Inc. Netflix. *Open Connect - Network Configuration. QCon 20*. 23. Juni 2020. URL: https://openconnect.zendesk.com/hc/en-us/articles/360035533071 (besucht am 23. 06. 2020).

[34] Inc. Netflix. *Open Connect Overview. QCon 20*. 23. Juni 2020. URL: https://openconnect.netflix.com/Open-Connect-Overview.pdf (besucht am 23. 06. 2020).

[35] Inc. Netflix. *Open Sourcing Zuul 2*. 23. Juni 2020. URL: https://netflixtechblog.com/open-sourcing-zuul-2-82ea476cb2b3 (besucht am 23. 06. 2020).

[36] Inc. Netflix. *Using Machine Learning to Improve Streaming Quality at Netflix. QCon 20*. 23. Juni 2020. URL: https://netflixtechblog.com/using-machine-learning-to-improve-streaming-quality-at-netflix-9651263ef09f (besucht am 23. 06. 2020).

[37] Jan Ozer. *Everything You Ever Wanted to Know About IDR Frames*. URL: https://streaminglearningcenter.com/articles/everything-you-ever-wanted-to-know-about-idr-frames-but-were-afraid-to-ask.html.

[38] Beverloo Peter. *VP9 and Opus, Background Position Offset and Ruby Positioning*. 18. Dez. 2012. URL: https://peter.sh/2012/12/vp9-and-opus-background-position-offset-and-ruby-positioning/ (besucht am 17. 07. 2020).

[39] Beverloo Peter. *VP9 vs. H.265 (HEVC) – Which Codec Is Better*. 12. März 2020. URL: https://www.videoproc.com/media-converter/vp9-vs-h265.htm (besucht am 10. 07. 2020).

[40] Emil Protalinski. *How Microsoft Teams will use AI to filter out typing, barking, and other noise from video calls*. 9. Apr. 2020. URL: https://venturebeat.com/2020/04/09/microsoft-teams-ai-machine-learning-real-time-noise-suppression-typing/ (besucht am 05. 07. 2020).

[41] Janko Roettgers. *The technology behind Google+ Hangouts*. 28. Jan. 2018. URL: https://gigaom.com/2011/06/30/google-hangouts-technology/ (besucht am 06. 08. 2020).

[42] Traci Ruether. *RTMP Streaming: The Real-Time Messaging Protocol Explained*. 05.08.2020. URL: https://www.wowza.com/blog/rtmp-streaming-real-time-messaging-protocol.

[43] Stephen Schankland. *HEVC, a new weapn in codec wars, to appear in September*. 22. Aug. 2012. URL: https://www.cnet.com/news/hevc-a-new-weapon-in-codec-wars-to-appear-in-september/ (besucht am 15. 07. 2020).

[44] Open Broadcaster Software. *Funktionen*. 20. Aug. 2020. URL: https://obsproject.com/de (besucht am 20. 08. 2020).

[45] Daniel Stenberg. *Kernel und CPU-Last*. URL: https://http3-explained.haxx.se/de/proc/proc-status%5C#kernel-und-cpu-last.

[46] Microsoft Teams. *Microsoft Teams*. URL: https://www.microsoft.com/de-de/microsoft-365/microsoft-teams/ (besucht am 06. 08. 2020).

[47] Microsoft Teams. *Onlinebesprechungen*. URL: https://www.microsoft.com/de-de/microsoft-365/microsoft-teams/online-meeting (besucht am 06. 08. 2020).

[48] Arbuthnor Tom. *New Satin codec coming to Microsoft Teams*. 21. März 2020. URL: https://tomtalks.blog/2020/03/new-satin-codec-coming-to-microsoft-teams (besucht am 15. 07. 2020).

[49] unknown. *An Overview of XMPP*. URL: https://xmpp.org/about/technology-overview.html (besucht am 14. 07. 2020).

[50] unknown. *Architecture*. 30. Juli 2020. URL: https://jitsi.github.io/handbook/docs/architecture (besucht am 03. 08. 2020).

[51] unknown. *Architecture*. URL: https://docs.bigbluebutton.org/2.2/architecture.html#Architecture_ (besucht am 03. 08. 2020).

[52] unknown. *BigBlueButton Features*. URL: https://bigbluebutton.org/teachers/ (besucht am 03. 08. 2020).

[53] unknown. *Comparison between HEVC and AVC*. URL: https://www.macxdvd.com/mac-dvd-video-converter-how-to/h265-vs-h264.htm (besucht am 29. 06. 2020).

[54] unknown. *Deep integration with your LMS*. URL: https://bigbluebutton.org/schools/ (besucht am 03. 08. 2020).

[55] unknown. *H.323*. 20. Apr. 2017. URL: https://www.itwissen.info/HDOT-323-H323.html (besucht am 13. 07. 2020).

[56] unknown. *Opus*. URL: https://opus-codec.org/ (besucht am 11. 07. 2020).

[57] unknown. *Router sind die Schwachstellen*. URL: https://www.golem.de/news/datenschutz-wie-sicher-ist-die-ip-telefonie-1502-112193-3.html (besucht am 10. 07. 2020).

[58] unknown. *SILK*. URL: https://web.archive.org/web/20111123141335/http://developer.skype.com/resources/SILKDataSheet.pdf (besucht am 11. 07. 2020).

[59] unknown. *T.120*. 20. Juli 2012. URL: https://www.itwissen.info/TDOT-120-T120.html (besucht am 30. 06. 2020).

[60] unknown. *VOIP und SIP Sicherheit*. URL: https://www.iant.de/sip-wiki/voip-und-sip-sicherheit (besucht am 15. 07. 2020).

[61] unknown. *VP8 (VP8 video codec)*. 1. Juni 2016. URL: https://www.itwissen.info/VP8-VP8-video-codec-VP8-Videocodec.html (besucht am 17. 07. 2020).

[62] unknown. *What are the features of Jitsi Meet*. URL: https://jitsi.org/user-faq/ (besucht am 03. 08. 2020).

[63] unknown. *Wie funktioniert die akustische Echokompensation bei einer Videokonferenz?* 13. Dez. 2017. URL: https://de.wikipedia.org/wiki/Echokompensation (besucht am 04. 07. 2020).

[64] Ilma Voigt. *Was ist WebRTC und wie deaktiviert man es ?* URL: https://nordvpn.com/de/blog/was-ist-webrtc/ (besucht am 14. 07. 2020).

[65] Shen Yueshi u. a. *Live Video Transmuxing/Transcoding: FFmpeg vs TwitchTranscoder, Part I*. URL: https://blog.twitch.tv/en/2017/10/10/live-video-transmuxing-transcoding-f-fmpeg-vs-twitch-transcoder-part-i-489c1c125f28/.

[66] Shen Yueshi u. a. *Live Video Transmuxing/Transcoding: FFmpeg vs TwitchTranscoder, Part II*. URL: https://blog.twitch.tv/en/2017/10/23/live-video-transmuxing-transcoding-f-fmpeg-vs-twitch-transcoder-part-ii-4973f475f8a3/.

[67] Phil Zimmermann. *Abstract*. 17. Juni 2010. URL: https://tools.ietf.org/html/draft-zimmermann-avt-zrtp-22 (besucht am 30. 06. 2020).

[68] Phil Zimmermann. *Introduction*. 17. Juni 2010. URL: https://tools.ietf.org/html/draft-zimmermann-avt-zrtp-22 (besucht am 30. 06. 2020).