Netzwerk-Voraussetzungen

INHALTSVERZEICHNIS

- Web-Verbindung
- Signalserver
- WebRTC-Client Netzwerkverbindungen
- Grundsätzliches zu den Verbindungsarten
- STUN/TURN und Firewalls
- STUN/TURN-Ports
- Quellen
- Fußnoten

Web-Verbindungen

- Patientus ist eine Web-Anwendung und benötigt eine HTTPS-Verbindung zum Web-Server https://patientus.de/ oder https://security.patientus.de/
- Benutzer können vom Web-Client HTTP-Verbindungen zum Web-Server aufbauen - diese werden immer nach HTTPS umgeleitet

Signalserver

- Unser Applikation baut eine sichere HTTPS-Verbindung (HTTPS) zu https://xmpp-production.patientus.de auf
- Wir benutzen xmpp aus zwei Gründen:
  1. um den Online-Zustand der Kommunikationspartner gegenseitig bekannt zu machen und
  2. um Verbindungsdaten für die Peer-to-Peer-Verbindung auszutauschen (Signaling). Bei den Verbindungsdaten handelt es sich um ICE-Verbindungs-Kandidaten (ICE: Interactive Connectivity Establishment).
- Multimediadaten (Audio, Video) werden hier nicht übertragen, es wird nur ausgehandelt, wie diese übertragen werden können, **Peer-To-Peer**-Übertragung wird dabei bevorzugt.

WebRTC-Client Netzwerkverbindungen

- **WebRTC**-Client initiiert Multimedia-Verbindungen mit Verbindungsvorschlägen im Web-Browser
- Die Verbindungsvorschläge werden von beiden Kommunikationspartnern, die sich in unterschiedlichen Netzwerk-Infrastruktur-Architekturen befinden können, der Reihe nach versucht und ausgehandelt
- **Peer-to-Peer** wird dabei bevorzugt und mit Hilfe von **STUN**[1] vermittelt
- **NAT** ist dabei ein übliches Problem, das über den TURN-Server aufgelöst wird
- **UDP** hole punching funktioniert oft nur mit einfachen **NAT**-Routern, ist aber dann Voraussetzung für eine funktionierende **Peer-to-Peer-Verbindung**. In einigen Fällen unterdrücken Firewalls dieses Vorgehen
- Um **UDP**-Verbindungsversuche für Audio/Video erfolgreich durchführen zu können, wird SRTP via **UDP** verwendet (**Peer-to-Peer**) wird ermöglicht). RTP verwendet „zufällige" Portnummern > 1024
- **TURN**[2] versucht die Sockets (IP-Adresse und Port) bekannt zu machen, kann Clients auch Sockets für Relay-Modus bereitstellen, ermöglicht es aber auch alle benötigten Verbindungen über die bestehende ausgehende **TURN**-Verbindung „zurückzureichen", es arbeitet dann als Relay für beide Kommunikationspartner
- Der Relay-Modus kann aus Qualitäts- und Datenschutzgründen die schlechtere Wahl sein, es sollte deswegen das Ermöglichen von **Peer-to-Peer** via **UDP** hole punching im Zusammenhang mit der jeweiligen eigenen Netzwerk-Infrastruktur ggf. geprüft werden.
- Es müssen nicht unbedingt alle u.g. Verbindungen geöffnet sein, wenn **UDP** hole punching am eigenen Router funktioniert, ist z.Z. ausgehend transparentes **UDP** und/oder **TCP** zu turn.patientus.de die einzige weitere Voraussetzung. In Zukunft werden wir auch die **TURN**-Standard-Ports verwenden, deswegen sollten auch diese ausgehend mindestens zu turn.patientus.de geöffnet werden.
- Wenn **UDP** hole punching oder **STUN**-Application-Level-Gateway an mindestens einem der beteiligten Router nicht funktioniert, ist funktionierende ein und ausgehende **UDP**-Verbindungsmöglichkeit zu turn.patientus.de via **UDP** und dem Portbereich 32355-65535 empfelenswert, dieses ermöglicht dann gute Video- und Audioqualität, eine **Peer-To-Peer**-Verbindung ist es dann leider nicht mehr
- Wenn auch dieses vom eigenen Router/Firewall-Produkt nicht ermöglicht wird, bleibt allein **UDP** zu den **STUN**-Standard-Ports oder als letzten Versuch Port 443 zu **TURN**. Dabei arbeitet **TURN** im Relay-Modus und transportiert auch die gekapselten Video- und Audiodaten über bestehende Verbindungen gekapselt zurück, was qualitativ die schlechteste Lösung sein kann.

88.99.67.133	**www.patientus.de**	Webserver mit Informationen
136.243.51.150	**security.patientus.de**	Webserver für Kommunikationssystem
138.201.53.22	**xmpp-production.patientus.de**	Signalserver
136.243.51.149	**turn.patientus.de**	Vermittlungsserver für **STUN** und **TURN**

- **PEER** steht hier stellvertretend für die IP-Adresse des jeweiligen Verbindungspartners. Da sich diese dynamisch ändern kann, sollte der Internet-Router oder das Firewall-Produkt die eingehenden Verbindungen im Kontext vorheriger ausgehender **UDP**-Verbindungen (**UDP hole punching**) oder im Kontext des Inhaltes der **STUN**/TURN-Kommunikation öffnen (**Application-Level-Gateway**). Ein pauschales Öffnen aller denkbaren **UDP** Portbereiche der **SRTP**-Verbindungen gegenüber dem gesamten Internet ist aus Sicherheitsgründen i.d.R. nicht empfehlenswert. Das Öffnen von **ICMP** ist i.d.R. ein geringes Risiko, eröffnet einem potentiellen Angreifer aber einige Netzwerk-Informationen, die dieser aber oft auch anders erlangen könnte.

Empfohlene Firewall-Konfiguration

Ziel	Richtung	Protokoll	Port(s)	Anwendungsprotokoll
www.patientus.de	A	TCP	80	HTTP-Redirects nach HTTPS
www.patientus.de	A	TCP	443	HTTPS
security.patientus.de	A	TCP	80	HTTP-Redirects nach HTTPS
security.patientus.de	A	TCP	443	HTTPS
xmpp-production.patientus.de	A	TCP	443	HTTPS
turn.patientus.de	A	TCP/UDP	443	STUNS/TURNS
PEER	EA	UDP	1024-65535	SRTP (nur dynamisch öffnen via UDP hole punching oder Application-Level-Gateway)
turn.patientus.de	EA	UDP	32355-65535	SRTP
PEER	EA	ICMP		Fehlerrückmeldungen
xmpp-production.patientus.de	EA	ICMP		Fehlerrückmeldungen
turn.patientus.de	EA	ICMP		Fehlerrückmeldungen

E = **E**ingehend zum Client
A = **A**usgehend vom Client

Grundsätzliches zu den Verbindungstabellen

- wenn o.g. SRTP-UDP-Verbindungen fehlschlagen schlechtere Multimedia-Übertragung und schlechterer Datenschutz, da kein Peer-to-Peer

Ziel	Richtung	Protokoll	Port(s)	Anwendungsprotokoll
www.patientus.de	A	TCP	80	HTTP-Redirects nach HTTPS
www.patientus.de	A	TCP	443	HTTPS
security.patientus.de	A	TCP	80	HTTP-Redirects nach HTTPS
security.patientus.de	A	TCP	443	HTTPS
xmpp-production.patientus.de	A	TCP	443	HTTPS
turn.patientus.de	A	TCP/UDP	443	STUN/STUNS/TURNS

E = **E**ingehend zum Client
A = **A**usgehend vom Client

STUN/TURN und Firewalls

- **WebRTC** und **TURN** wurden unter strenger Berücksichtigung von Sicherheit und Datenschutz entwickelt. So verwendet **WebRTC** ausschließlich verschlüsselte Video- und Audio-Datenströme (**SRTP**) und **TURN** verwendet Integritätsschutz, Authorisierung kritischer Funktionen und optionale Verschlüsselung. Ein Application-Level-Gateway einer „Firewall" kann mit unverschlüsselten **STUN/TURN** besser interagieren, da es dann im Kontext Ports für die Video- und Audio-Datenströme dynamisch öffnen und so auch **Peer-to-Peer**-Verbindungen ermöglichen könnte. Die mit **TURN** übertragenen Steuer-Parameter sind durch Authorisierung und Integritätsschutz gegen Manipulation geschützt.
- Traversal Using Relays around NAT (TURN): Relay Extensions to Session Traversal Utilities for NAT (STUN) 17.2. Firewall Considerations http://tools.ietf.org/html/rfc5766#section-17.2

STUN/TURN[3]-Ports

- Wir verwenden für **STUN/TURN** den Port 443, um über einfache Router/NATGateways/"Firewalls" die Videosprechstunde überhaupt zu ermöglichen.
- Komplexere Firewalls mit Application-Level haben möglicherweise Probleme mit **UDP(DTLS)/443** bei **STUN/TURN**. Der Grund liegt möglicherweise darin, dass STUN/TURN keine Standardprotokolle für den Port 443 sind (lt. IANA).

Quellen

- https://de.wikipedia.org/wiki/WebRTC
- https://en.wikipedia.org/wiki/Traversal_Using_Relays_around_NAT
- RFC5766 Traversal Using Relays around **NAT (TURN)**: Relay Extensions to Session Traversal Utilities for **NAT (STUN)**
- **ICE, Kandidaten, STUN und TURN** http://www.msxfaq.de/lync/technik/ice.htm

Fußnoten

1. Session Traversal Utilities for NAT (STUN) http://tools.ietf.org/html/rfc5389
2. Traversal Using Relays around NAT (TURN): Relay Extensions to Session Traversal Utilities for NAT (STUN) http://tools.ietf.org/html/rfc5766
3. https://tools.ietf.org/html/rfc5766#section-4