Teletherapie mit dem Jitsi-Videokonferenzsystem

Server-Installation und Konfiguration

Jörg Mayer v1.2.2 / 25. April 2020

Inhaltsverzeichnis

1 Jitsi

Jitsi ist ein leistungsfähiges und sicheres Open-Source-Videokonferenzsystem. Die wichtigsten Funktionen:

- Videokonferenz (Live-Video und -Audio) plus parallele Text-Nachrichten.
- Bildschirmfreigabe: Statt des Live-Videobildes von der Webcam können alle Teilnehmer*innen den anderen Teilnehmer*innen ihren Bildschirm-Inhalt anzeigen (ganzer Bildschirm / ein Anwendungsfenster / ein Browser-Tab).
- Teilnehmer*innen entscheiden, ob sie die jeweils sprechende Person im Vollbild sehen (automatische Umschaltung zwischen sprechenden Personen) oder ob sie in einer Kachelansicht alle teilnehmenden Personen gleichzeitig sehen.
- Sowohl Kamera als auch Mikrophon können zeitweise stumm geschaltet werden.
- Das System ist einfach zu bedienen und kann sowohl mit der Maus als auch mit Tastaturkürzeln gesteuert werden; an Touchscreens (neuere Laptops, Tablets) ist auch Touch-Steuerung möglich.
- Auf Desktop- und Notebook-Computern muss keine zusätzliche Software installiert werden, um an einer Videokonferenz teilzunehmen, ein aktueller Browser genügt. Für Tablets (iPad und Android-Tablets) steht die Jitsi-App zur Verfügung.

Wichtige Eigenschaften bezüglich Sicherheit und Datenschutz:

- Bei zwei Teilnehmer*innen werden sämtliche Inhalte Ende-zu-Ende verschlüsselt übertragen. Bei drei und mehr Teilnehmer*innen sind die Daten auf dem Weg vom und zum Server verschlüsselt (Transportverschlüsselung), auf dem Server selbst liegen sie jedoch für die Dauer einer Videositzung unverschlüsselt vor.
- Sämtliche Inhalte einer Sitzung werden automatisch gelöscht, nachdem die Sitzung beendet ist, d.h. nachdem die letzte teilnehmende Person die Sitzung verlassen hat. Bei Bedarf kann bei einer eigenen Jitsi-Instanz eine Recording-Funktion aktiviert werden, so dass Sitzungen aufgenommen werden können.
- Von Hause aus ist Jitsi praktisch anonym nutzbar, d.h. weder um eine neue Sitzung zu starten, noch um einer bestehenden Sitzung beizutreten ist ein Benutzerkonto erforderlich. Abgesehen von der IP-Adresse (siehe Abschnitt 2.1) erfährt der Service-Betreiber nichts über die Teilnehmer*innen.
- Wer eine Jitsi-Instanz auf dem eigenen Server betreibt, kann die Zugangsmöglichkeiten einschränken; Beispiele:
  - Für das Starten einer neuen Sitzung kann ein Benutzername und ein Passwort verlangt werden.
  - Für das Beitreten zu einer Sitzung kann bei Bedarf ein Passwort oder sogar Benutzername plus Passwort verlangt werden.

2 Der eigene Server

2.1 Warum ein eigener Server?

Jitsi betreibt selbst einige Server, um unkompliziert, anonym und werbefrei eine Videokonferenz zu starten (Jitsi-Meet); diese Server stehen vermutlich in den USA. Auch andere Organisationen weltweit betreiben öffentliche Jitsi-Server, hier eine Liste. Die Kommunikation mittels Jitsi ist bei einer Konferenz mit zwei Teilneher*innen Ende-zu-Ende verschlüsselt, d.h. auf dem Weg von einem Kommunikationspartner zum anderen sind alle (Video-, Audio- und Text-) Daten verschlüsselt unterwegs und somit vor Lauschern sicher. Wo die Server stehen und wer die Server betreibt, über die der verschlüsselte Datenverkehr läuft, scheint daher mit Blick auf Datensicherheit und Datenschutz in diesem Fall weniger wichtig.

Bei mehr als zwei Teilnehmer*innen (z.B. Gruppentherapie) findet jedoch nach derzeitigem Kenntnisstand keine Ende-zu-Ende-Verschlüsselung statt. In diesem Fall ist nur der Transportweg verschlüsselt, auf dem Server selbst liegen die Daten jeoch – zumindet für die Dauer einer Videokonferenz – unverschlüsselt vor.

Außerdem fallen in jedem Fall neben den verschlüsselten Inhalten auch unverschlüsselte Metadaten an. Das Problem ist z.B. von WhatsApp bekannt: Die Nachrichteninhalte sind zwar Ende-zu-Ende verschlüsselt, der Betreiber (Facebook) erhält aber trotzdem viele wertvolle Daten, eben die unverschlüsselten Metadaten (wer hat wann wie lange mit wem kommuniziert usw.). Da Jitsi vollständig anonym genutzt werden kann, beschränken sich die anfallenden Metadaten auf die IP-Adressen der Konferenzteilnehmer. Das ist im Sinne der Datensparsamkeit löblich, dennoch ist zu beachten, dass IP-Adressen nach geltender deutscher Rechtsprechung zu den personenbezogenen Daten gehören.

Schließlich gilt für jeden Service, der über fremde Server läuft – ob kommerziell oder nicht –, dass man dem Betreiber vertrauen muss, dass der Service wirklich so datensicher implementiert und konfiguriert ist wie versprochen. Einem seriösen, der DSGVO verpflichteten Anbieter, mit dem eine Vereinbarung zur Auftragsverarbeitung geschlossen wurde, kann dieses Vertrauen entgegengebracht werden, einem öffentlichen Server nicht ohne weiteres.

Fazit: Um die Kontrolle über alle anfallenden Daten zu behalten, auch die unverschlüsselten Metadaten, muss Jitsi auf dem eigenen Server betrieben werden. Nur so wird die Privatsphäre aller Teilnehmer*innen optimal geschützt und der Videokonferenz-Service kann auf diese Weise sicher und DSGVO-konform betrieben werden.

Außerdem ist es nur bei einer selbst betriebenen Jitsi-Instanz möglich, sinnvolle Zugangsbeschränkungen zu konfigurieren, so dass z.B. nur die Therapeutin eine neue Sitzung anlegen und starten kann.

2.2 Kaufen oder Mieten

Ein eigener Server kann ein Computer sein, der in den eigenen Praxisräumen hinter verschlossenen Türen steht, aber z.B. auch ein angemieteter Root-Server bei einem vertrauenswürdigen Anbieter. Beide Varianten haben Vor- und Nachteile, die jede*r für sich abwägen muss.

Ein wichtiger Punkt, der einen Computer in den eigenen Räumen u.U. disqualifiziert, ist die Upload-Bandbreite. Internetanschlüsse für Privatkunden aber auch für viele kleinere Geschäftskunden sind in Deutschland häufig asymmetrisch ausgelegt, d.h. einer guten Download-Bandbreite steht eine bisweilen dürftige Upload-Bandbreite gegenüber. Der eigene Videokonferenz-Server,

der sämtliche Kommunikation zentral verwaltet, empfängt aber nicht nur Daten von den Teilnehmer*innen (Download), sondern verteilt diese Daten auch an die Teilnehmer*innen weiter (Upload). Eine ungenügende Upload-Bandbreite kann daher zu einem Flaschenhals werden und zu unbefriedigender Audio- und Videoqualität führen oder gar zu Verbindungsabbrüchen.

Mit einem gehosteten Root-Server umgeht man dieses Problem. Diese Server stehen in Rechenzentren, die symmetrisch und mit hoher Bandbreite an das Internet angeschlossen sind. Weitere Vorteile: Die Rechenzentren sind klimatisiert, verfügen über eine Notstromversorgung und die Hardware wird rund um die Uhr überwacht. Auch lässt sich die Leistung eines gemieteten Servers (CPUs, Arbeitsspeicher, Datenspeicher) in aller Regel kurzfristig und einfach an veränderte Bedürfnisse anpassen. Bei eigener Hardware ist das aufwändiger oder nur durch einen Neukauf zu realisieren. Wenn in der Praxis kein (geeigneter) Computer zur Verfügung steht, ist man mit einem Miet-Server auch viel schneller am Start: Von der Online-Bestellung bis zur Bereitstellung vergehen meist nur Minuten; einen Server zu kaufen dauert Tage bis Wochen…

Eigene Erfahrung

Wir betreiben unser Jitsi-Videokonferenzsystem derzeit auf einem eher leistungsschwachen, dafür aber günstigen gemieteten Server: Ein virtualisierter Server mit 2 nicht fest zugewiesenen 'virtuellen' CPU-Kernen, 4 GB RAM und 40 GB SSD (Kosten: 5,30- €/Monat). Bei bisher maximal 4 gleichzeitigen Teilnehmer*innen und auch bei zwei Konferenzen gleichzeitig funktioniert das sehr gut und scheint noch einige Luft nach oben zu haben hinsichtlich CPU und RAM-Auslastung.

Sollte der Bedarf steigen (mehr gleichzeitige Konferenzen/Teilnehmer*innen), wäre ein Upgrade (mehr CPU-Kerne, mehr RAM) gegen Aufpreis einfach und ohne Neuinstallation möglich, sozusagen im laufenden Betrieb – auch das ein großer Vorteil gegenüber eigener Hardware.

Mit dem Anbieter muss natürlich eine **Vereinbarung zur Auftragsverarbeitung** vorliegen bzw. neu geschlossen werden.

Die folgende Anleitung geht von einem vergleichbaren System aus, also einem angemieteten Server, der ausschließlich für das Jitsi-Videokonferenzsystem zuständig ist. Vieles lässt sich jedoch auch auf andere Konstellationen übertragen.

2.3 Betriebssystem

Jitsi kann entweder auf herkömmliche Weise direkt in ein (Linux-) System installiert werden oder als Docker-Container betrieben werden.

Herkömmliche Installation: Die Jitsi-Installationsanleitung (und auch unsere Anleitung in Abschnitt 5) beschreibt die notwendigen Schritte für ein Debian- oder Ubuntu-System. Versierte Linux-Nutzer können versuchen, das System auch unter anderen Betriebssystemen zu installieren, ansonsten lautet die klare Empfehlung, den eigenen Jitsi-Server mit Debian (oder Ubuntu) auszustatten.

Betrieb in Docker-Containern: Läuft eine Anwendung in einem Container, fungiert das eigentliche Server-Betriebssystem nur als Wirtsystem und ist von untergeordneter Bedeutung. Die verfügbare Anleitung für ein containerisiertes Jitsi-System nutzt als Container-Lösung `docker` und `docker-compose`, so dass im Prinzip jedes Docker-kompatible System als Wirt geeignet ist. (Achtung: Aktuelle Systeme aus dem Redhat-Umfeld (RHEL, CentOS, Fedora)

sind auf eine andere Container-Lösung umgestiegen (`podman`) und haben (noch) keinen vollwertigen Ersatz für `docker-compose`.) Da sich Debian als Server-Betriebssystem über viele Jahre bewährt hat und nach wie vor auf Docker setzt, ist auch hier Debian eine gute Wahl.

Fazit: Wenn andere Anforderungen nicht dagegen sprechen, ist Debian die beste Wahl für den gemieteten Server; aktuell ist Debian 10 (Codename *Buster*). Eine Debian-Minimalinstallation reicht aus. Insbesondere kann bei einem gemieteten Server auf graphische Desktop-Software verzichtet werden, weil der Server ohnehin über eine Text-Konsole administriert wird. Die meisten Server-Hoster bieten verschiedene Betriebssysteme an, die sozusagen auf Knopfdruck installiert werden können. Alternativ gibt es in der Regel die Möglichkeit, eigene CDROM-Abbilder (z.B. ein heruntergeladenes Debian-Abbild) zum Hoster hochzuladen und auf dem Miet-Server zu installieren.

2.4 Installation oder Container?

Unsere Erfahrung reicht nicht aus, um hier eine klare Empfehlung zu geben, und auch die 'Profis' sind sich – soweit wir das übersehen – nicht einig, ob bzw. wann eine herkömmliche Installation einer Container-Lösung vorzuziehen ist oder umgekehrt.

Auf einem Server, der außer für Jitsi auch noch für andere Dinge zuständig ist, ist die Containerlösung u.U. vorzuziehen. Das gilt um so mehr, wenn `docker` bereits auf dem Server installiert und aktiviert ist; in diesem Fall ist Jitsi tatsächlich sehr schnell einsatzbereit. Auf einem frisch aufgesetzten, dezidierten Server erscheint uns dagegen eine herkömmliche Installation sinnvoller und übersichtlicher zu konfigurieren.

Eigene Erfahrung

Wir haben beide Varianten ausprobiert und uns zunächst für die Containerlösung entschieden – ganz einfach, weil sie sofort gut funktioniert hat.

Mit der herkömmlichen Installation anhand der offiziellen Jitsi-Installationsanleitung haben wir dagegen keinen vollständig funktionsfähigen Betrieb erreicht. Ehrlicherweise muss aber dazu gesagt werden, dass die ersten Versuche unter einem gewissen Zeitdruck stattgefunden haben, da wir schnell mit der Teletherapie loslegen wollten.

Nach etlichen Versuchen ist uns mittlerweile auch die herkömmliche Installation gelungen (s. Anleitung in Abschnitt 5) und wir sind auf diese Variante umgestiegen, u.a. weil die Konfiguration für uns einfacher und transparenter ist.

3 Domain und Zugang

3.1 Domain

Bei einem gemieteten Server erhält man in der Regel eine feste IPv4-Adresse, um mit dem Server zu kommunizieren. Für administrative Aufgaben und Testzwecke reicht das aus, für den produktiven Einsatz ist aber eine aussagekräftige URL sicher besser geeignet, z.B.:

```
https://tele.praxis.de
```

anstatt eine IP-Adresse wie:

```
https://192.168.94.127
```

Wenn man bereits eine Domain besitzt (z.B. `praxis.de`) und die Möglichkeit besteht, Sub-Domains zu konfigurieren (z.B. `tele.praxis.de`), kann man einfach die neue Sub-Domain anlegen und auf die IPv4-Adresse des Servers verweisen. Das funktioniert auch über Anbieter hinweg: Wenn Sie z.B. bei Anbieter A Ihre Website hosten und Ihre Domains verwalten, der Server aber von Anbieter B bereit gestellt wird, können Sie bei Anbieter A eine Weiterleitung der Sub-Domain zu einer IP-Adresse im Netzwerk von Anbieter B einrichten.

Verfügen Sie nicht über eine eigene Domain, können Sie entweder bei Ihrem Server-Hoster eine beantragen (eine de-Domain kostet etwa 5,– €/Jahr) oder Sie nutzen einen Service für dynamisches DNS, haben dann aber nicht die volle Kontrolle über den Domainnamen. Für einen Server in den eigenen Räumen hinter einem Internetanschluss ohne feste IP-Adresse ist dynamisches DNS ebenfalls die Methode der Wahl, um den Server von außen erreichbar zu machen. Besitzer*innen einer Fritzbox können hierfür den MyFritz-Dienst nutzen.

3.2 ssh-Zugang

Ein gemieteter Server wird in aller Regel per Text-Konsole verwaltet, d.h. alle Arbeiten (im wesentlichen Software-Installation und das Bearbeiten von Konfigurationsdateien) werden ohne den gewohnten Komfort einer graphischen Benutzeroberfläche erledigt. Die Verbindung zwischen dem eigenen Rechner und dem Server wird über das verschlüsselte ssh-Protokoll hergestellt. Dafür wird eine IP-Adresse, ein Benutzername und ein Passwort benötigt (nach dem Erstkontakt kann auch ein Zugang mit Public Key statt Passwort eingerichtet werden; s. Abschnitt 4.2).

Wenn der Server mit fertig installiertem Betriebssystem bereit gestellt wird, erhalten Sie die Angaben vom Hoster (nach dem ersten Login nicht vergessen, mit dem Befehl `passwd` das Passwort zu ändern). Bei einem selbst installierten Betriebssystem benötigen Sie vom Hoster nur die IP-Adresse, da Sie im Laufe der Installation selbst ein Benutzerkonto einrichten.

Liegen alle Angaben vor, öffnen Sie auf ihrem eigenen Rechner eine Text-Konsole:

- Windows: In aktuellen Versionen von Windows 10 genügt vermutlich die Eingabeaufforderung oder PowerShell; ansonsten scheint Putty eine bewährte ssh-Anwendung zu sein; ich habe weder mit dem einen noch mit dem anderen Erfahrung.
- macOS: Öffnen Sie die Terminal-Applikation.
- Linux: Öffnen Sie eines der zahlreichen Terminals.

In der Text-Konsole tippen Sie dann einen Befehl nach dem folgenden Muster ein (alle Befehle werden durch Drücken der ENTER-Taste ausgeführt):

`ssh <Name>@<Ihre-IP-Adresse>`

z.B.:

`ssh root@192.168.94.127`

Bestätigen Sie die folgende Abfrage zum Schlüsselaustausch mit `yes` und geben Sie auf Nachfrage Ihr Passwort ein – ab jetzt arbeiten Sie auf dem Server. Um die ssh-Sitzung zu beenden, tippen Sie `exit`. Die hier vorgestellte Anmeldemethode als Root mit Passwort ist anfangs notwendig, sollte aber besser früher als später deaktiviert und durch ein sichereres Verfahren ersetzt werden (s. Abschnitt 4.2).

4 Vor der Installation

4.1 Vorbemerkungen

Diese Anleitung geht davon aus,

- dass Jitsi auf einem frisch aufgesetzten Server mit Debian-10-Minimalsystem installiert wird,
- dass dieser Server ausschließlich für Jitsi zuständig sein soll,
- dass Sie über einen Server-Zugang mit administrativen Rechten verfügen (root-Zugang oder sudo-Rechte),
- dass Sie eine (Sub-) Domain konfiguriert haben, die auf diesen Server verweist.

Die Anleitung nutzt als Beispiel die Domain `praxis.de` und die Subdomain `tele`. Die Basis-URL des installierten Systems würde demnach am Ende `https://tele.praxis.de` lauten. Entsprechende Angaben in der Installationsanleitung müssen Sie natürlich an ihre (Sub-) Domain anpassen. In den folgenden Listings ist die Beispiel-Domain farbig markiert: `tele.praxis.de`; in den Befehlen müssen Sie diesen Teil jeweils durch Ihre Domain ersetzen, in den Konfigurationsdateien steht nach der Installation meist schon Ihre richtige Domain anstatt meiner Besipiel-Domain.

Befehle, die Sie nach der Verbindung via ssh im Server-Terminal eingeben müssen, sind in solchen Kästen gelistet:

```
# befehl
```

Das #-Zeichen steht für den sogenannten Prompt, der bei Ihnen sicher anders aussieht (eher so: `root@servername:~#`) und der natürlich nicht zu kopieren ist. Wenn Sie nicht als root sondern als normaler Benutzer mit sudo-Rechten eingeloggt sind (`ihrname@servername:~$`), müssen Sie den meisten Befehlen ein `sudo` voranstellen.

Sehr lange Befehle werden aus Darstellungsgründen in mehrere Zeilen umgebrochen; das \-Zeichen signalisiert der Terminal-Shell, dass der Befehl in der nächsten Zeile fortgesetzt wird.

Um Copy & Paste zu erleichtern, verzichte ich in den Folgezeilen auf den Prompt; die Shell wird voraussichtlich ein >-Zeichen einfügen, was in Ordnung ist.

```
# befehl argument1 argument2 \
   argument3 argument4 \
   argument5 argument6
# neuerbefehl
```

Ausschnitte aus Konfigurationsdateien, in denen Sie etwas editieren müssen, sehen so aus:

```
...
Lorem ipsum dolor sit amet
Consetetur sadipscing elitr
...
```

Wichtige Zeilen, in denen etwas geändert werden soll, sind fett gedruckt, die Auslassungspunkte deuten an, dass es sich um einen Ausschnitt handelt und davor und dahinter weiterer Text stehen kann.

4.2 Erstkontakt

Nachdem Sie sich zum ersten Mal mit Ihrem neuen Server verbunden haben, stellen Sie sicher, dass das System auf dem neuesten Stand ist:

```
# apt update && apt -y upgrade
```

Aus Sicherheitsgründen sind folgende Vorarbeiten unbedingt empfehlenswert:

- **Umstellung der ssh-Anmeldung auf Public-Key-Authentication** (Anleitungen gibt es zahlreich im Internet, Stichwort ”ssh anmeldung ohne passwort”)
- **Root-Zugang per ssh verbieten**, so dass man sich nur noch als ’normaler’ Nutzer verbinden kann. Für administrative Aufgaben nutzt man `sudo` oder wechselt in das Root-Konto.
- **Installation einer einfachen Firewall** und Konfiguration für ssh sowie Jitsi (die Frage, die nach dem Befehl `ufw enable` gestellt wird, kann mit `y` beantwortet werden):

  ```
  # apt -y install ufw
  # ufw enable
  # ufw allow in ssh
  # ufw allow in 80/tcp
  # ufw allow in 443/tcp
  # ufw allow in 10000:20000/udp
  # ufw status verbose
  ```

Evtl. reicht die Freigabe des udp-Ports 10000 aus, um Jitsi (ohne die *jigasi*-Komponente) zu betreiben; in diesem Fall würde die vorletzte Zeile **`ufw allow in 10000/udp`** lauten; die Informationen, die ich dazu gefunden habe, sind jedoch widersprüchlich. Experimentierfreudige können es versuchen …

- **Schutz der ssh-Anmeldung** vor Brute-Force-Angriffen:

```
# apt -y install fail2ban
```

Aus Komfortgründen ziehe ich den Texteditor `micro` den vorinstallierten Alternativen `vi` oder `nano` vor; das ist natürlich optional:

```
# apt -y install curl
# cd /usr/local/bin
# curl https://getmic.ro | bash
```

Wenn Sie einen anderen Editor nutzen möchten, ersetzten Sie in den folgenden Listings `micro` durch den Befehl für Ihren Editor.

Jetzt wird es ernst. Wenn Sie Jitsi herkömmlich installieren möchten, geht es mit Abschnitt 5 weiter, wenn Sie die Container-Lösung bevorzugen, springen Sie zu Abschnitt 6.

5 Jitsi: Herkömmliche Installation und Konfiguration

5.1 Installation

Zuerst wird der Hostname des Servers und die Datei `/etc/hosts` geändert (nicht vergessen: statt `tele` und `tele.praxis.de` setzen Sie bitte Ihre (Sub-) Domain ein):

```
# hostnamectl set-hostname tele
# sed -i 's/^127.0.1.1.*$/127.0.1.1 tele.praxis.de tele/g' /etc/hosts
```

Als nächstes wird der Webserver `nginx` installiert. (Laut offizieller Anleitung ist das nicht nötig, allerdings ist es mir nur mit `nginx` gelungen, erfolgreich Lets-Encrypt-Zertifikate zu beziehen.)

```
# apt -y install nginx
# systemctl start nginx.service
# systemctl enable nginx.service
```

Da Jitsi (noch) nicht in den offiziellen Paketquellen vertreten ist, muss eine neue Quelle angelegt werden. So kann Jitsi in Zukunft im Rahmen eines gewöhnlichen System-Updates aktuell gehalten werden.

```
# apt -y install gnupg
# wget https://download.jitsi.org/jitsi-key.gpg.key
# apt-key add jitsi-key.gpg.key
# sh -c "echo 'deb https://download.jitsi.org stable/' > \
   /etc/apt/sources.list.d/jitsi-stable.list"
# apt update
# apt -y install jitsi-meet
```

Der letzte Befehl startet die eigentliche Installation. Im Laufe der Installation wird nach der Domain gefragt; hier geben Sie Ihre Domain nach dem Muster `tele.praxis.de` an (ohne `https://`). Als nächstes wird nach der Art des Zertifikats gefragt; hier wählen Sie die erste Option: `Generate a new self-signed certificate...`

Wenn die Installation abgeschlossen ist (das kann eine Weile dauern), aktivieren Sie noch den Let's Encrypt-Mechanismus, wodurch der Jitsi-Server ein kostenloses, von allen Browsern akzeptiertes TLS-Zertifikat von der Zertifizierungsstelle Let's Encrypt erhält, so dass die Transportverschlüsselung mittels `https` zuverlässig funktioniert:

```
# /usr/share/jitsi-meet/scripts/install-letsencrypt-cert.sh
```

Das Skript fragt zu Beginn nach einer Email-Adresse; hier geben Sie eine gültige Adresse an, um keine Nachrichten von Let's Encrypt zu verpassen – die kommen selten und wenn, sind sie normalerweise wichtig.

Das war's! Es spricht nichts dagegen, Ihr eigenes Jitsi-System einem ersten Test zu unterziehen, bevor wir uns an die weitere Konfiguration machen. Beim Aufruf Ihrer URL im Browser (z.B. `https://tele.praxis.de`) sollte die Jitsi-Startseite erscheinen und Sie können sofort Ihre erste selbst-gehostete Videokonferenz starten.

5.2 Konfiguration

5.2.1 Authentifizierung

Wenn der Start einer neuen Sitzung durch eine Benutzer-Authentifizierung geschützt werden soll, sind die folgenden Schritte notwendig. Danach muss man zum Start einer Sitzung Benutzername und Passwort eingeben, nicht jedoch um einer bereits gestarteten Sitzung beizutreten. Wenn auf die Konfiguration einer Authentifizierung verzichtet wird, kann jede beliebige Person, die Ihren Server findet, darüber Videokonferenzen abhalten.

Zur Einrichtung müssen zunächst drei Konfigurationsdateien bearbeitet werden. (Es ist ratsam, vor der Bearbeitung wichtiger Dateien eine Sicherungskopie der Originaldatei anzulegen.) Dann muss mindestens ein Benutzerkonto erstellt werden und zum Abschluss werden alle Server-Komponenten neu gestartet.

1. Der XMPP-Server *prosody*

   Die folgenden Befehle wechseln in das *prosody*-Verzeichnis, legen eine Sicherungskopie der Konfigurationsdatei an und öffnen die Datei dann zur Bearbeitung (anstatt `micro` funktioniert auch `nano` oder `vi`):

   ```
   # cd /etc/prosody/conf.avail/
   # cp tele.praxis.de.cfg.lua tele.praxis.de.cfg.lua.BKP
   # micro tele.praxis.de.cfg.lua
   ```

   Im Abschnitt `VirtualHost "tele.praxis.de"` (hier steht nach der obigen Installation schon Ihre korrekte Domain!) ändern Sie die Option `authentication` (Zeilen, die mit '--' beginnen, sind in `lua`-Dateien Kommentare).

   Aus:

   ```
   ...
   VirtualHost "tele.praxis.de"
       -- enabled = false -- Remove this line to enable this host
       authentication = "anonymous"
   ...
   ```

   wird:

   ```
   ...
   VirtualHost "tele.praxis.de"
       -- enabled = false -- Remove this line to enable this host
       authentication = "internal_hashed"
   ...
   ```

   Dann fügen Sie nach dem Abschnitt `VirtualHost "tele.praxis.de"` einen neuen VirtualHost-Abschnitt hinzu (achten Sie darauf, dass die Optionen gegenüber der 'Überschrift' eingerückt sind); hier müssen Sie die Beispiel-Domain durch Ihre eigene ersetzen:

   ```
   ...
       c2s_require_encryption = false

   -- neuer VirtualHost-Abschnitt:
   VirtualHost "guest.tele.praxis.de"
       authentication = "anonymous"
       c2s_require_encryption = false

   Component "conference.tele.praxis.de" "muc"
   ...
   ```

   Wenn alle Änderungen vorgenommen wurden, kann die Konfiguration mit `STRG-S` (nano: `STRG-O`) gespeichert werden. Der micro-Editor wird mit `STRG-Q` (nano: `STRG-X`) beendet.

2. Die Jitsi-Komponente *jicofo*

   Die folgenden Befehle wechseln in das *jicofo*-Verzeichnis, legen eine Sicherungskopie der Konfigurationsdatei an und öffnen die Datei dann zur Bearbeitung:

```
# cd /etc/jitsi/jicofo/
# cp sip-communicator.properties sip-communicator.properties.BKP
# micro sip-communicator.properties
```

   Fügen Sie am Ende der Datei die folgende Zeile hinzu:

```
...
org.jitsi.jicofo.auth.URL=XMPP:tele.praxis.de
```

3. Die Jitsi-Komponente *meet*

   Die folgenden Befehle wechseln in das *meet*-Verzeichnis, legen eine Sicherungskopie der Konfigurationsdatei an und öffnen die Datei dann zur Bearbeitung:

```
# cd /etc/jitsi/meet/
# cp tele.praxis.de-config.js tele.praxis.de-config.js.BKP
# micro tele.praxis.de-config.js
```

   Im Abschnitt `hosts:` aktivieren Sie die Option `anonymousdomain` und passen den Wert an Ihre Domain an (Zeilen, die mit '//' beginnen, sind in `js`-Dateien Kommentare).

   Aus:

```
...
    hosts: {
        // XMPP domain.
        domain: 'tele.praxis.de',

        // When using authentication, domain for guest users.
        // anonymousdomain: 'guest.example.com',
...
```

   wird:

```
...
    hosts: {
        // XMPP domain.
        domain: 'tele.praxis.de',

        // When using authentication, domain for guest users.
        anonymousdomain: 'guest.tele.praxis.de',
...
```

4. Benutzerkonto

   Mit dem folgenden Befehl legen Sie ein Benutzerkonto an. Statt *benutzername* wählen Sie einen passenden Namen, statt *passwort* ein sicheres Passwort (bitte notieren; das Passwort wird verschlüsselt gespeichert und kann nicht rekonstruiert werden). Nach dem gleichen Muster können bei Bedarf (auch später noch) weitere Benutzerkonten angelegt werden:

   ```
   # prosodyctl register benutzername tele.praxis.de passwort
   ```

5. Zum Abschluss werden alle Server-Komponenten neu gestartet, um die Änderungen zu aktivieren:

   ```
   # systemctl restart nginx.service prosody.service \
      jicofo.service jitsi-videobridge2.service
   ```

5.2.2 Standardsprache und STUN-Server

Die Standardsprache der Oberfläche ist auf Englisch voreingestellt. Um auf Deutsch zu wechseln, muss die *meet*-Konfiguration bearbeitet werden (die Originaldatei haben wir bereits im letzten Abschnitt gesichert):

```
# cd /etc/jitsi/meet/
# micro tele.praxis.de-config.js
```

Suchen Sie die beiden folgenden Zeilen:

```
...
    // Default language for the user interface.
    // defaultLanguage: 'en',
...
```

und aktivieren/ändern Sie die Option `defaultLanguage`:

```
...
    // Default language for the user interface.
    defaultLanguage: 'de',
...
```

In der gleichen Datei werden die STUN-Server konfiguriert. (Mithilfe eines STUN-Servers können Computer hinter einer Firewall eine direkte Peer-to-Peer Kommunikation mit einem anderen

Computer außerhalb des lokalen Netzwerks aufbauen.) Vorkonfiguriert sind hier Google-Server[1]. Wer Google nicht unnötigerweise mit Daten füttern und den Service DSGVO-konform betreiben möchte, sollte hier andere, am besten in Europa gehostete Server eintragen. Hier gibt es eine Liste mit STUN-Servern. (Der Hinweis auf die Google-STUN-Server stammt vom sehr lesenswerten Blog von Mike Kuketz.)

Suchen Sie den folgenden Abschnitt:

```
...
        // The STUN servers that will be used in the peer to peer conn...
        stunServers: [

            //  urls: 'stun:tele.praxis.de:443' ,
            { urls: 'stun:stun.l.google.com:19302' },
            { urls: 'stun:stun1.l.google.com:19302' },
            { urls: 'stun:stun2.l.google.com:19302' }
        ],
...
```

und tragen Sie neue STUN-Server ein. Die alten Angaben können Sie löschen oder – wie hier – auskommentieren, um sie schnell wieder aktivieren zu können, falls die neuen Server nicht korrekt arbeiten:

```
...
        // The STUN servers that will be used in the peer to peer conn...
        stunServers: [

            //  urls: 'stun:tele.praxis.de:443' ,
            { urls: 'stun:stun.1und1.de:3478' },
            { urls: 'stun:stun.t-online.de:3478' },
            { urls: 'stun:stun.hosteurope.de:3478' },
            { urls: 'stun:stun.gmx.de:3478' }
            //{ urls: 'stun:stun.l.google.com:19302' },
            //{ urls: 'stun:stun1.l.google.com:19302' },
            //{ urls: 'stun:stun2.l.google.com:19302' }
        ],
...
```

Bevor Sie die Datei speichern und schließen, können Sie auch noch nach dieser Option suchen:

```
    // disableThirdPartyRequests: false,
```

und für mehr Privatsphäre aktivieren:

```
    disableThirdPartyRequests: true,
```

[1] Jitsi hat reagiert: In der neuesten Version (getestet am 4.4.2020) ist ein Jitsi-STUN-Server vorkonfiguriert, die Google-Server tauchen nicht mehr auf. Es spricht aber nicht dagegen, stattdessen oder zusätzlich andere STUN-Server hinzuzufügen, so wie hier beschrieben.

5.2.3 Nginx: SSL-Protokoll-Versionen beschränken

In der Standard-Konfiguration erlaubt der Webserver Nginx die veralteten/unsicheren TLS-Protokolle `TLSv1` und `TLSv1.1`. Um die verschlüsselte `https`-Verbindung robuster zu machen, sollten diese beiden Protokolle deaktiviert werden; stattdessen kann man die neuste TLS-Version `TLSv1.3` zusätzlich zu `TLSv1.2` erlauben. Die folgenden Befehle wechseln in das zuständige Nginx-Verzeichnis, legen eine Sicherungskopie der Konfigurationsdatei an und öffnen die Datei dann zur Bearbeitung (Ihre Konfigurationsdatei ist selbstverständlich nach Ihrer Domain benannt):

```
# cd /etc/nginx/sites-available
# cp tele.praxis.de.conf tele.praxis.de.conf.BKP
# micro tele.praxis.de.conf
```

Im zweiten `server`-Block finden Sie die Option für die erlaubten Protokolle:

```
...
    ssl_protocols TLSv1 TLSv1.1 TLSv1.2;
...
```

Entfernen Sie `TLSv1` und `TLSv1.1` und fügen Sie `TLSv1.3` hinzu:

```
...
    ssl_protocols TLSv1.2 TLSv1.3;
...
```

Nach Änderungen an der Nginx-Konfiguration muss der Web-Server neu gestartet werden, um die Änderungen zu aktivieren:

```
# systemctl restart nginx.service
```

Weitere Konfigurationsvorschläge finden Sie im Jitsi-Wiki. Die regelmäßig aktualisierten Empfehlungen zu einer sicheren SSL/TLS-Konfiguration, veröffentlicht von den SSL Labs, liefern Hintergrundinformationen.

5.2.4 Die Kür

Achtung: Alle Anpassungen, die in diesem Abschnitt vorgestellt werden, können bei einem Jitsi-Update verloren gehen! Jitsi wird aktiv entwickelt, d.h. es werden recht häufig neue Versionen veröffentlicht, die im Rahmen einer normalen System-Aktualisierung via

```
# apt update && apt -y upgrade
```

(welche unbedingt regelmäßig durchgeführt werden sollte) automatisch eingespielt werden. Diese Versions-Updates überschreiben ohne Nachfrage die Dateien, die im folgenden bearbeitet werden. Wenn Sie also die ein oder andere hier vorgeschlagene Anpassung vornehmen, erstellen Sie unbedingt eine Kopie der entsprechenden Datei an einem sicheren Ort, so dass Sie Ihre Änderungen nach einem Jitsi-Update wieder einpflegen können. 'Einpflegen' bedeutet in den meisten Fällen, die gewünschten Anpassungen in den *neuen*, durch das Update eingespielten Dateien erneut vorzunehmen, wobei die Sicherungskopien als Vorlage dienen (ein graphischer Diff-Editor ist hierbei hilfreich); auf keinen Fall sollte man die neuen Dateien einfach blind durch die Sicherungskopien ersetzen, da dadurch ggf. wichtige Änderungen der Jitsi-Enwickler überschrieben werden.

Sämtliche Dateien, die das Aussehen der Jitsi-Oberfläche steuern, finden sich im Verzeichnis `/usr/share/jitsi-meet` bzw. in dessen Unterverzeichnissen. Wir beginnen mit einer Konfigurationsdatei, die zahlreiche Einstellungen bereitstellt, um die Oberfläche weiter anzupassen:

```
# cd /usr/share/jitsi-meet/
# cp interface_config.js interface_config.js.BKP
# micro interface_config.js
```

Sie können relativ gefahrlos verschiedene Einstellung ändern und den Effekt sofort – manchmal auch mit kurzer Verzögerung – im Browser überprüfen (Seite neu laden). Hier ein paar Beispiele:

`TOOLBAR_ALWAYS_VISIBLE: true,`	Verhindert das Ausblenden der Bedienelemente (Stummschalten, Auflegen usw.).
`SHOW_JITSI_WATERMARK: false,`	Entfernt das Jitsi-Logo von der Startseite.
`APP_NAME: 'Praxisname',` `NATIVE_APP_NAME: 'Praxisname',`	Der Praxisname erscheint in Bookmarks und an einigen anderen Stellen.
`GENERATE_ROOMNAMES_ON_WELCOME_PAGE: false,`	Stellt die wenig brauchbaren Vorschläge für Sitzungsnamen ab.
`DISPLAY_WELCOME_PAGE_CONTENT: true,`	Zeigt selbst erstellte Inhalte auf der Startseite an (s.u.).

Falls Sie die Oberfläche kaputt konfigurieren, stellen Sie einfach die Originaldatei wieder her:

```
# cd /usr/share/jitsi-meet/
# mv interface_config.js.BKP interface_config.js
```

Sollten Sie für Ihren Web-Auftritt ein Favicon in Form einer Datei `favicon.ico` vorliegen haben. können Sie damit das Jitsi-Favicon ersetzen. Übertragen Sie die Datei `favicon.ico` von ihrem Rechner auf den Server (im Terminal mit `scp` oder mit einem graphischen scp-Tool) und kopieren Sie die Datei dann in das korrekte Verzeichnis:

```
# cp favicon.ico /usr/share/jitsi-meet/images/
```

Wer noch weitergehende Anpassungen wünscht, die nicht über die Einstellungen in den Konfigurationsdateien zu bekommen sind, muss an's Eingemachte. Um z.B. die sehr kleine Schrift im

Text-Chat größer zu machen, muss man einen Wert in der Style-Datei ändern (zumindest habe ich bisher keine andere Möglichkeit gefunden):

```
# cd /usr/share/jitsi-meet/css/
# cp all.css all.css.BKP
# micro all.css
```

Suchen Sie mit STRG-F nach `chatconversation` und ändern Sie den Wert der css-Eigenschaft `font-size`, z.B. auf 14pt.

Auf diese Weise lassen sich alle möglichen Aspekte der Oberfläche an die eigenen Wünsche anpassen. Voraussetzung ist ein bisschen Detektivsinn und zumindet Grundlagenwissen in HTML und CSS.

Wenn Sie im Footer der Startseite selbst erstellte Inhalte (z.B. einen Link zu Ihrer Website, zu einem Impressum und/oder zu einer Datenschutzerklärung) anzeigen wollen, bearbeiten Sie die folgende Datei:

```
# cd /usr/share/jitsi-meet/static/
# cp welcomePageAdditionalContent.html welcomePageAdditionalContent.html.BKP
# micro welcomePageAdditionalContent.html
```

Fügen Sie ihre Inhalte zwischen die *template*-Tags ein, z.B.:

```
<template id = "welcome-page-additional-content-template">
  <style type="text/css">
    #footer {
      margin-bottom: 2rem;
      font-size: 14pt;
      color: #fff;
    }
  </style>

  <div id="footer">
    <center>
      <a href="https://www.praxisname.de" target="_blank">Praxisname</a>
    </center>
  </div>
</template>
```

Damit die Inhalte angezeigt werden, muss in der Datei `interface_config.js` die Option `DISPLAY_WELCOME_PAGE_CONTENT` den Wert `true` erhalten (s.o.). Falls der Footer wie im Beispiel einen Link enthält, funktioniert dieser nur nach einer weiteren Änderung in der css-Datei:

```
# cd /usr/share/jitsi-meet/css/
# micro all.css
```

Suchen Sie mit `STRG-F` nach `welcome-watermark` und entfernen Sie die css-Eigenschaft `height: 100%`.

Um andere Texte auf der Startseite (oder an anderer Stelle) anzupassen, kann die Datei bearbeitet werden, die die deutschen Übersetzungen für die eingebauten (englischsprachigen) Text-Komponenten enthält:

```
# cd /usr/share/jitsi-meet/lang/
# cp main-de.json main-de.json.BKP
# micro main-de.json
```

Ersetzen Sie die vorgegebenen Texte durch eigene Kreationen, z.B. den Titel und den Beschreibungstext auf der Startseite. Aus:

```
...
   "appDescription": "Auf geht's! Starten Sie eine Videokonferenz...",
...
   "title": "Sichere, mit umfassenden Funktionen ausgestattete..."
...
```

wird z.B.:

```
...
   "appDescription": "Willkommen bei unserem Teletherapie-Angebot.",
...
   "title": "Logopädische Praxis Mustermann"
...
```

Das funktioniert aber nur, wenn Sie die Standardsprache auf Deutsch umgestellt haben (s. Abschnitt 5.2.2) und Ihre Nutzer keine andere Sprache einstellen…

Zum Abschluss noch einmal der Hinweis: Alle in diesem Abschnitt vorgeschlagenen Anpassungen müssen gesichert und nach einem Jitsi-Update ggf. rekonstruiert werden.

6 Jitsi im Container

Bei einem frisch aufgesetzten Server mit Debian-Minimalinstallation ist die Container-Verwaltung `docker` und das Hilfsprogramm `docker-compose` noch nicht installiert. Das kann mit

```
# docker -v
# docker-compose -v
```

überprüft werden. Wird eine Fehlermeldung ausgegeben (*command not found* o.ä.), steht eine Installation an, wird eine Versionsnummer ausgegeben (*Docker version...*), haben Sie Glück gehabt und können mit Abschnitt 6.2 fortfahren.

6.1 Docker installieren

Die Docker-Installation muss mit administrativen Rechten erfolgen. Wenn Sie nicht mit dem Root-Konto auf dem Server eingeloggt sind, stellen Sie sicher, dass Ihr Nutzerkonto sudo-Rechte hat und stellen Sie den folgenden Befehlen ein `sudo` voran.

```
# apt update
# apt -y install curl gnupg2 git ca-certificates
# apt -y install apt-transport-https software-properties-common
# curl -fsSL https://download.docker.com/linux/debian/gpg |\
   apt-key add -
# add-apt-repository \
   "deb [arch=amd64] https://download.docker.com/linux/debian \
   $(lsb_release -cs) stable"
# apt update
# apt install docker-ce docker-ce-cli containerd.io
```

Zur Installation von `docker-compose` (Version 1.25.4) genügen diese zwei Befehle:

```
# curl -L "https://github.com/docker/compose/releases/download\
   /1.25.4/docker-compose-$(uname -s)-$(uname -m)" \
   -o /usr/local/bin/docker-compose
# chmod +x /usr/local/bin/docker-compose
```

Hat alles geklappt, sollten diese beiden Befehle eine Versionsnummer ausgeben:

```
# docker -v
# docker-compose -v
```

6.2 Jitsi konfigurieren und starten

Auch hier gilt: Wenn Sie nicht mit dem Root-Konto auf dem Server eingeloggt sind, stellen Sie sicher, dass Ihr Nutzerkonto sudo-Rechte hat und stellen Sie den folgenden Befehlen ein `sudo` voran.

Der folgende Befehl erstellt auf dem Server das Verzeichnis `docker-jitsi-meet` und kopiert die benötigten Dateien in dieses Verzeichnis:

```
# git clone https://github.com/jitsi/docker-jitsi-meet
```

Als nächstes wechseln Sie in das neue Verzeichnis, erstellen aus der vorgegebenen Beispiel-Konfiguration (`env.example`) die später tatsächlich verwendete Konfiguration (`.env`) und öffnen diese zum Bearbeiten (statt `micro` geht auch `nano` oder `vi` etc.):

```
# cd docker-jitsi-meet
# cp env.example .env
# micro .env
```

Die Konfigurationsmöglichkeiten sind sehr umfangreich und zu jeder Option gibt es eine kurze Erklärung in der Datei. Zeilen, die mit "#"beginnen, sind Kommentare und werden bei der Konfiguration nicht berücksichtigt. Das betrifft natürlich die Erklärungen, aber auch manche Optionen sind durch ein vorangestelltes "#"deaktiviert. Durch Entfernen des Kommentarzeichens können diese Optionen aktiviert werden. Die Optionen selbst folgen dem einfachen Muster *NAME_DER_OPTION=Wert*.

Die folgenden Änderungen halten wir für sinnvoll (Erläuterungen zu den Änderungen folgen unten):

Vorher	**Nachher**
Damit beim Aufruf der Jitsi-URL später keine Port-Angabe notwendig ist, sollten die Standard-Ports eingestellt werden (80 für http und 443 für https):[1]	
HTTP_PORT=8000 HTTPS_PORT=8443	HTTP_PORT=80 HTTPS_PORT=443
Der Automatische Bezug eines TLS-Zertifikats ist in der Vorgabe nicht aktiv; dies sollte man unbedingt aktivieren:[2]	
#ENABLE_LETSENCRYPT=1 #LETSENCRYPT_DOMAIN=meet.example.com #LETSENCRYPT_EMAIL=alice@atlanta.net	ENABLE_LETSENCRYPT=1 LETSENCRYPT_DOMAIN=tele.praxis.de LETSENCRYPT_EMAIL=info@praxis.de
Um das Starten einer neuen Sitzung mit Benutzername und Passwort zu schützen und zugleich anderen Personen zu erlauben, ohne Authentifizierung teilzunehmen:[3]	
#ENABLE_AUTH=1 #ENABLE_GUESTS=1 #AUTH_TYPE=internal	ENABLE_AUTH=1 ENABLE_GUESTS=1 AUTH_TYPE=internal
Um einen Aufruf ohne Transportverschlüsselung per http zu verhindern, können mit dieser Option alle Aufrufe automatisch auf https umgeleitet werden; das kann aktiviert werden, weil wir oben die Voraussetzungen für https geschaffen haben:	
#ENABLE_HTTP_REDIRECT=1	ENABLE_HTTP_REDIRECT=1
Die Aufnahmefunktion kann mit dieser Option aktiviert (=1) oder deaktiviert (=0) werden:	
#ENABLE_RECORDING=1	ENABLE_RECORDING=0

Wenn alle Änderungen vorgenommen wurden, kann die Konfiguration mit STRG-S (nano: STRG-O) gespeichert werden. Der micro-Editor wird mit STRG-Q (nano: STRG-X) beendet.

[1]Die Port-Änderungen führen dazu, dass der Docker-Container und damit das Jitsi-System auf dem Server über die Standarports erreichbar ist. Das ist dann sinnvoll, wenn der Server nur für Jitsi zuständig ist und keine anderen Webinhalte ausliefert. Ohne diese Änderung würde die Adresse des Servers so lauten:

`https://tele.praxis.de:8443`

anstatt so:

`https://tele.praxis.de`

(Wird der Server jedoch auch für andere Dinge benutzt, muss der Administrator u.U. einen sog. Proxy-Server vorschalten, der den ankommenden http-Verkehr regelt; in diesem Fall sollte man die Vorgabewerte nicht oder nur nach Rücksprache mit dem Administrator ändern.)

[2]Gültige TLS-Zertifikate sind notwendig, damit das Jitsi-System per https, also mit Transportverschlüsselung aufgerufen werden kann. Das ist unerlässlich, weil moderne Browser einer Seite, die ohne Transportverschlüsselung, also nur per http aufgerufen wird, die Freigabe von Kamera und Mikrophon verweigern – d.h. ohne vertrauenswürdige Zertifikate und ohne https keine Videokonferenz! Der Jitsis Docker-Container hat einen Mechanismus eingebaut, der kostenlose TLS-Zertifikate von der Zertifizierungsstelle Let's Encrypt bezieht. Mit ENABLE_LETSENCRYPT=1 wird dieser Mechanismus aktiviert. Bei der Domain- und Email-Option für Let's Encrypt müssen natürlich gültige Werte angegeben werden, `tele.praxis.de` und `info@praxis.de` dienen im Beispiel oben nur als Platzhalter.

[3]Die Authentifizierungseinstellungen können nach Bedarf angepasst werden. Die vorgeschlagene Konfiguration erlaubt das Starten einer neuen Video-Sitzung nur nach Authentifizierung, also nach Eingabe eines Benutzernamens und eines Passwortes (ENABLE_AUTH=1). Ist eine Sitzung gestartet, können weitere Personen jedoch ohne Authentifizierung teilnehmen (ENABLE_GUESTS=1), wobei die Möglichkeit, eine neu gestartete Sitzung mit einem Passwort zu schützen, bestehen belibt. Die dritte Option legt die Art der Authentifizierung fest, dazu später mehr.

Ist die Konfiguration erledigt, wird das Jitsi-System mit dem folgenden Befehl gestartet (dieser und die folgenden Befehle müssen im Verzeichnis `docker-jitsi-meet` abgesetzt werden):

```
# docker-compose up -d
```

Beim ersten Aufruf lädt der Befehl zunächst vier Image-Dateien als Basis für die Container, die das Jitsi-System beherbergen werden; deshalb dauert der erste Start einige Minuten länger als spätere Starts. Dann werden die Container erstellt und miteinander verknüpft, so dass alle Jitsi-Komponenten nahtlos zusammenarbeiten können. Hat alles geklappt, sollte das System jetzt unter der zuvor eingerichteten Domain erreichbar sein, in diesem Beispiel also unter `https://tele.praxis.de`.

Wenn Sie dem obigen Konfigurationsvorschlag gefolgt sind, kann allerdings noch keine Videokonferenz gestartet werden, da für den Start einer Konferenz eine Authentifizierung gefordert wird, wir aber noch kein Benutzerkonto angelegt haben. Um das zu ändern muss in der zuständigen Komponente, dem Prosody-Server, mindestens ein Benutzerkonto angelegt werden. Mit den folgenden Befehlen wird zuerst eine Eingabeaufforderung innerhalb des Prosody-Containers gestartet (der veränderte Prompt signalisiert, dass wir uns jetzt innerhalb eines Containers

befinden). Der zweite Befehl erzeugt ein Benutzerkonto (die Platzhalter <benutzername> und <passwort> durch die gewünschte Name/Passwort-Kombination ersetzen); sollen mehrere Konten angelegt werden, muss dieser Befehl entsprechend mehrfach ausgeführt werden; mit dem dritten Befehl wird der Container wieder verlassen:

```
# docker-compose exec prosody /bin/bash
> prosodyctl --config /config/prosody.cfg.lua \
   register <benutzername> meet.jitsi <passwort>
> exit
```

Es kann sein, dass das Benutzerkonto verloren geht, wenn die Container heruntergefahren werden (noch nicht getestet). In diesem Fall muss das Benutzerkonto bei jedem Container-Neustart erneut angelegt werden (von Hand wie hier oder – für versierte Nutzer – per Dockerfile).

Sofern als Sprache English eingestellt ist, erscheint nach dem Start einer neuen Videokonferenz die Meldung *Waiting for the host…* bzw. bei deutscher Spracheinstellung *Warten auf den Organisator…*. Klicken Sie auf *I am the host* bzw. *Ich bin der Organisator* und geben Sie die oben konfigurierten Benutzerdaten ein, um die Sitzung zu starten.

Im Prinzip ist jetzt alles bereit. Verteilen Sie den Link zu der neuen Sitzung an alle Teilnehmer*innen und legen Sie los mit der ersten Videokonferenz auf dem eigenen Server.

Um den Jitsi-Service zu beenden (z.B. für Wartungsarbeiten, ein Update o.ä.), werden die Container mit dem folgenden Befehl heruntergefahren:

```
# docker-compose down
```

6.3 Feineinstellungen

Die folgenden Einstellungen können im laufenden System vorgenommen werden. Änderungen sind sofort bzw. mit kurzer Verzögerung wirksam.

6.3.1 Sprache

Nach einer Neuinstallation ist Englisch als Standardsprache für die Benutzeroberfläche eingestellt. Das kann jede*r Teilnehmer*in individuell in den Einstellungen ändern, es ist aber auch möglich, die Oberfläche standardmäßig auf Deutsch umzustellen. Öffnen Sie hierzu mit dem folgende Befehl die entsprechende Konfigurationsdatei :

```
# micro ~/.jitsi-meet-cfg/web/config.js
```

Suchen Sie die Zeile

```
// defaultLanguage: 'en',
```

und ändern Sie die Zeile wie folgt:

```
defaultLanguage: 'de',
```

In der Datei lassen sich etliche weitere Optionen anpassen. Schauen Sie sich ruhig um, aber seinen Sie vorsichtig…

6.3.2 Oberfläche

Im gleichen Verzeichnis gibt es eine zweite Konfigurationsdatei, in der weitere Aspekte der Benutzeroberfläche angepasst werden können:

```
# micro ~/.jitsi-meet-cfg/web/interface_config.js
```

Einige Beispiele:

`TOOLBAR_ALWAYS_VISIBLE: true`	Verhindert das Ausblenden der Bedienelemente (Stummschalten, Auflegen usw.).
`SHOW_JITSI_WATERMARK: false`	Entfernt das Jitsi-Logo von der Startseite.
`APP_NAME: 'Praxisname'` `NATIVE_APP_NAME: 'Praxisname'`	Der Praxisname erscheint in Bookmarks und an einigen anderen Stellen.
`GENERATE_ROOMNAMES_ON_WELCOME_PAGE: false`	Stellt die wenig brauchbaren Vorschläge für Sitzungsnamen ab.

6.3.3 Innereien

Weitere Anpassungen können vorgenommen werden, indem Dateien innerhalb der laufenden Container verändert werden. Hier ist zu beachten, dass diese Änderungen verloren gehen, sobald die Container heruntergefahren werden. Wichtige Änderungen müssen also gut dokumentiert werden, damit sie einfach repliziert werden können, wenn die Conatiner neu gestartet werden. Versierte Nutzer können auch die Container-Konfiguration entsprechend anpassen (z.B. das `Dockerfile` im Verzeichnis `/docker-jitsi-meet/web`).

Die folgenden Anpassungen werden im web-Container vorgenommen, ermitteln Sie also zunächst dessen ID:

```
# docker container ls
```

Der Befehl listet die vier laufenden Jitsi-Container auf. Suchen Sie nach der Zeile, die in der zweiten Spalte `jitsi/web` enthält. Die ID finden Sie in der ersten Spalte:

```
CONTAINER ID     IMAGE            COMMAND     CREATED      ...
7b0b23e2e447     jitsi/jicofo     "/init"     9 days ago   ...
17ede23c6557     jitsi/jvb        "/init"     9 days ago   ...
6a3cbff5e60a     jitsi/web        "/init"     9 days ago   ...
4578fb6c3c2f     jitsi/prosody    "/init"     9 days ago   ...
```

Für die folgenden Beispiele nutze ich die ID **`6a3cbff5e60a`**; diese müssen Sie durch die von Ihnen ermittelte ID ersetzen.

Favicon: Laden Sie ihr Favicon zunächst auf den Server hoch, dann kopieren Sie die Datei mit dem folgenden Befehl in den web-Container:

```
# docker cp favicon.ico 6a3cbff5e60a:/usr/share/jitsi-meet/images/
```

Um eine Engabeaufforderung im laufenden web-Container zu starten, setzen Sie den folgenden Befehl im Verzeichnis `docker-jitsi-meet` ab:

```
# docker-compose exec web /bin/bash
```

Dann können z.B. die Textbausteine in der deutschen Übersetzung verändert werden (was man laut `readme.md` nicht tun sollte…):

```
> nano /usr/share/jitsi-meet/lang/main-de.json
```

Oder der Seitentitel und weitere Angaben im Header der Startseite:

```
> nano /usr/share/jitsi-meet/title.html
```

Oder die Schriftgröße im Text-Chat:

```
> nano /usr/share/jitsi-meet/css/all.css
```

(Suchen Sie nach `chatconversation` und ändern Sie den Wert der css-Eigenschaft `font-size`.)

Auf diese Weise kann fast alles individuell angepasst werden. Veränderte Dateien sollten aber unbedingt außerhalb des Containers gesichert werden, um sie nach einem Container-Neustart händisch oder automatisiert per `Dockerfile` einfach wiederherstellen zu können.

Container verlassen, ein Verzeichnis für Sicherungskopien erstellen und – als Beispiel – die css-Datei sichern:

```
> exit
# mkdir ~/jitsi-anpassungen
# cd ~/jitsi-anpassungen
# docker cp 6a3cbff5e60a:/usr/share/jitsi-meet/css/all.css .
```

7 Epilog

Im Idealfall haben Sie jetzt ein funktionstüchtiges Jitsi-Videokonferenzsystem auf dem eigenen Server am Laufen – Glückwunsch!

Der eigene Server will aber auch gepflegt und evtl. weiter konfiguriert und gehärtet werden. Wenn ich Zeit und Informationen finde, werde ich versuchen, diese Anleitung in dieser Hinsicht auszubauen. Oder Sie informieren sich selbst über empfehlenswerte Maßnahmen (und sagen mir gerne Bescheid!). Bis dahin sollte der Server zumindest regelmäßig (Sicherheits-) Updates erhalten. Dazu stellen Sie eine ssh-Verbindung her und führen auf dem Server die folgenden Befehle aus (Docker-Nutzer bauen einfach neue Container aus aktualisierten Image-Dateien):

```
# apt update && apt -y upgrade
```

Natürlich können Sie mir auch gerne Verbesserungsvorschläge oder anderes Feedback schicken:

joerg.mayer@sprachtherapie-intensiv.de

Die jeweils aktuellste Version dieses Dokuments finden Sie hier zum Download:

https://www.sprachtherapie-intensiv.de/Merkblaetter/Jitsi_Installation.pdf

— Danke an die freundlichen Menschen im Kuketz-Forum für Tipps und Hinweise! —