Jitsi Meet Videokonferenzen

von **Carsten Rieger** | Aktualisiert

Videokonferenzen werden zunehmend wichtiger im beruflichen aber auch im privaten Alltag. Der Bedarf an stabilen und sicheren Videokonfernezsystemen wächst täglich. Häufig wird zu proprietäre Lösungen wie z.B. Teams, Skype, WhatsApp, FaceTime und anderen Tools gegriffen, ohne dabei die Aspekte des Datenschutzes und der Privatsphäre bewertet zu haben. Eine meines Erachtens nach bessere Alternative stellt **Jitsi Meet** dar – eine quelloffene Software, mit der auch mehrere Teilnehmer gleichzeitg an Telefon- und Videokonferenzen teilnehmen können. Und das – nach Abarbeitung dieses Guides – auch ohne Google & Co!

Zu den Voraussetzungen (aus der Praxis!):

- Standalone-Server (aufgrund von benötigten Bandbreiten, Performance und Sicherheit)

bspw. vServer von netcup GmbH oder vServer von Hetzner

- native 64Bit Infrastruktur (**kein** ARM64 wie bsp. Raspberry PI oder Odroid C2)
- Ubuntu 20.04.x LTS oder Debian 10 Buster
- separate Domäne (bspw. **konferenz.dedyn.io**)
- coturn Server 4.5.+ (Installationsanleitung)

Für den Einsatz am PC oder Notebook empfehle ich sowohl den Chromium Browser, als auch Mozilla Firefox 75+ oder den Jitsi Desktop Client. Für Smartphones und Tablets finden Sie die Jitsi-App in den jeweiligen App-Stores. Weitere Herstellerinformationen finden Sie hier: Jitsi

Inhaltsverzeichnis

Aktualisierungen: 07. August 2020 // zwei Beispiel-Konfigurationen am Ende ergänzt

1. Vorbereitungen und Installation des nginx Webserver

Wechseln Sie zuerst in den privilegierten Benutzermodus

```
sudo -s
```

und installieren dann die folgenden Softwarepakete, die als Grundlage des Serverbetriebs notwendig sind:

```
apt install gnupg2 git lsb-release ssl-cert ca-certificates apt-transport-https tree locate software-properties-common dirmngr screen htop nano net-tools zip unzip curl ffmpeg ghostscript libfile-fcntllock-perl curl socat certbot -y
```

Fügen Sie die zukünftige Domäne der Hostdatei hinzu:

```
nano /etc/hosts

127.0.0.1 localhost konferenz.dedyn.io
```

HINWEIS!

konferenz.dedyn.io steht als Synonym für ihre Domäne.

Es gilt für den gesamten Guide, dass dieses Synonym an den rot markierten Stellen durch Ihre Domäne ersetzt werden muss!

Fügen Sie dem System die sogenannten Software-Repositories (Softwarequellen) hinzu, um die aktuellen Releases der jeweiligen Pakete installieren zu können.

Nur auf einem Ubuntu-System auszuführen:

```
apt-add-repository universe

echo "deb [arch=amd64] http://nginx.org/packages/mainline/ubuntu $(lsb_release -cs) nginx" | tee /etc/apt/sources.list.d/nginx.list
```

Nur auf einem Debian-System auszuführen:

```
echo "deb [arch=amd64] http://nginx.org/packages/mainline/debian $(lsb_release -cs) nginx" | tee /etc/apt/sources.list.d/nginx.list
```

Ab hier wieder auf beiden Systemen (Ubuntu und Debian) auszuführen:

```
echo "deb https://download.jitsi.org stable/" | tee /etc/apt/sources.list.d/jitsi.list
```

Wir ergänzen die notwendigen Schlüssel (nginx, jitsi) und aktualisieren das System.

```
curl -fsSL https://nginx.org/keys/nginx_signing.key | sudo apt-key add -

wget -qO -  https://download.jitsi.org/jitsi-key.gpg.key | sudo apt-key add -

apt update && apt upgrade -y
```

Um sicherzustellen, dass keine nginx und Apache2 Relikte früherer Installationen den Betrieb des Webserver stören, entfernen wir diese:

```
apt remove --purge nginx nginx-extras nginx-common nginx-full apache2 apache2-* -y --allow-change-held-packages
```

Nun sind die Vorbereitungen zur Installation des Webservers abgeschlossen und wir können die Software mit dem nachfolgenden Befehl installieren:

```
apt install nginx -y
```

Um den Webserver nginx als Dienst automatisch zu starten führen wir diesen Befehl aus:

```
systemctl enable nginx.service
```

einrichten. Mit Blick auf die späteren Anpassungen wird die Standardkonfiguration gesichert und eine neue Konfigurationsdatei geöffnet:

```
mv /etc/nginx/nginx.conf /etc/nginx/nginx.conf.bak && touch /etc/nginx/nginx.conf

nano /etc/nginx/nginx.conf
```

Kopieren Sie den gesamten nachfolgenden Inhalt in die Datei:

```
user www-data;

worker_processes auto;

pid /var/run/nginx.pid;

events {

worker_connections 1024;

multi_accept on; use epoll;
```

```
}
http {
server_names_hash_bucket_size 64;
access_log /var/log/nginx/access.log;
error_log /var/log/nginx/error.log warn;
set_real_ip_from 127.0.0.1;
real_ip_header X-Forwarded-For;
real_ip_recursive on;
include /etc/nginx/mime.types;
default_type application/octet-stream;
sendfile on;
send_timeout 3600;
tcp_nopush on;
tcp_nodelay on;
open_file_cache max=500 inactive=10m;
open_file_cache_errors on;
keepalive_timeout 65;
reset_timedout_connection on;
server_tokens off;
resolver 127.0.0.53 valid=15s;
resolver_timeout 5s;
include /etc/nginx/conf.d/*.conf;
}
```

Nun legen wir einen Standard-vHost an, um später die SSL-Zertifikate beantragen zu können.

```
nano /etc/nginx/conf.d/http.conf
server {
        server_name konferenz.dedyn.io;
        listen 80 default_server;
        listen [::]:80 default_server;
        charset utf-8;
        root /var/www;
        location ^~ /.well-known/acme-challenge {
            default_type text/plain;
            root /var/www/letsencrypt;
        }
        location / {
                return 301 https://$host$request_uri;
        }
 }
```

Speichern Sie die Datei und schließen Sie diese, um im Anschluß daran den Webserver neu zu starten:

```
service nginx restart
```

Vorbereitend für die Jitsi-Installation legen wir drei Ordner an:

```
mkdir -p /etc/nginx/sites-available
mkdir -p /etc/nginx/sites-enabled
mkdir -p /etc/nginx/modules-enabled
```

Wir installieren die Firewall UFW, aktivieren diese und öffnen die für Jitsi notwendigen Ports:
https://github.com/jitsi/jitsi-meet/blob/master/doc/quick-install.md#advanced-configuration

```
apt install ufw -y
```

Bitte passen Sie bei geänderten Ports die nachfolgenden Befehlszeilen an:

SSH:

```
ufw allow 22/tcp
```

HTTP(s):

```
ufw allow 80/tcp
ufw allow 443/tcp
```

JITSI:

```
ufw allow 5000/udp
ufw allow 10000/udp
ufw logging medium && ufw default deny incoming && ufw enable && service ufw restart
```

Wir beantragen nun die SSL-Zertifikate mittels des acme:

```
adduser acmeuser
usermod -a -G www-data acmeuser
```

Wechseln Sie in die Shell des neuen Benutzers (acmeuser) um die Zertifikatssoftware zu installieren und verlassen diese Shell danach wieder:

```
su - acmeuser
curl https://get.acme.sh | sh
exit
```

Legen Sie drei Verzeichnisse mit den entsprechenden Berechtigungen an, um u.a. die neuen Zertifikate requestieren und darin speichern zu können:

```
mkdir -p /var/www/letsencrypt/.well-known/acme-challenge /etc/letsencrypt/rsa-certs /etc/letsencrypt/ecc-certs
chmod -R 775 /var/www/letsencrypt /etc/letsencrypt && chown -R www-data:www-data /var/www/ /etc/letsencrypt
```

Wechseln Sie erneut in die Shell des neuen Benutzers

```
su - acmeuser
```

und requestieren (beantragen) die SSL-Zertifikate. Ersetzen Sie dabei **konferenz.dedyn.io** mit Ihrer Domain:

```
acme.sh --issue -d konferenz.dedyn.io --keylength 4096 -w /var/www/letsencrypt --key-file /etc/letsencrypt/rsa-certs/privkey.pem --ca-file /etc/letsencrypt/rsa-certs/chain.pem --cert-file /etc/letsencrypt/rsa-certs/cert.pem --fullchain-file /etc/letsencrypt/rsa-certs/fullchain.pem
```

```
acme.sh --issue -d konferenz.dedyn.io --keylength ec-384 -w /var/www/letsencrypt --key-file /etc/letsencrypt/ecc-certs/privkey.pem --ca-file /etc/letsencrypt/ecc-certs/chain.pem --cert-file /etc/letsencrypt/ecc-certs/cert.pem --fullchain-file /etc/letsencrypt/ecc-certs/fullchain.pem
```

Verlassen Sie die Shell des neuen Benutzers

```
exit
```

Sie finden die Zertifikate unter folgendem absoluten Pfad, den Sie im Rahmen der nachfolgenden Installation angeben müssen:

Schlüssel (ssl key file):

/etc/letsencrypt/rsa-certs/privkey.pem

Zertifikat (ssl certificate file):

/etc/letsencrypt/rsa-certs/fullchain.pem

Wir entfernen die Standardkonfiguration von nginx und starten den Webserver neu.

```
cd /etc/nginx/conf.d

mv default.conf default.conf.disabled && touch default.conf

service nginx restart

cd /
```

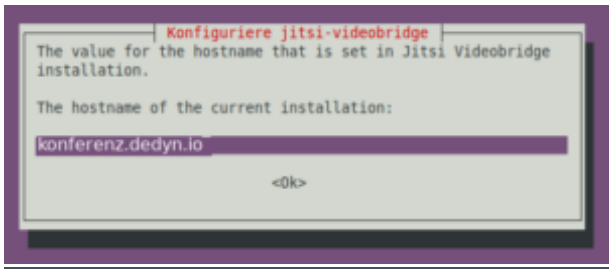
2. Starten wir nun die Installation von Jitsi:

Zur Installation wird der nachfolgende Befehl verwendet:

```
apt install -y --no-install-recommends jitsi-meet
```

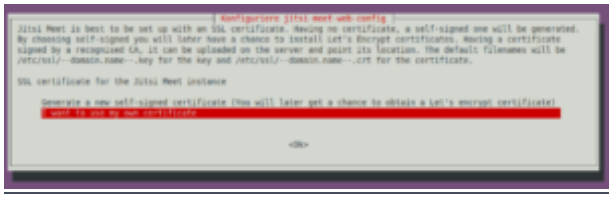
Durch die Verwendung der Aufrufparameter (**–no-install-recommends**) wird das Überschreiben bestehender Serverkonfigurationen bspw. des coturn-Servers verhindert.

Tragen Sie in den aufkommenden Dialogfenstern Ihre Domäne ein:



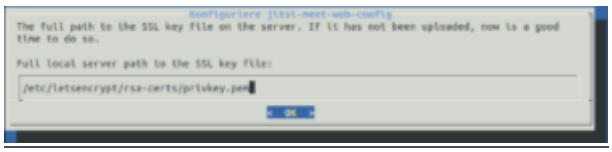
konferenz.dedyn.io mit Ihrer Domäne

Wählen Sie im nächsten Dialogfenster „I want to use my own certificate“ aus:



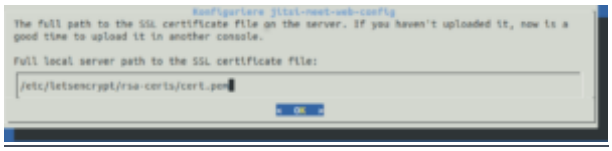
In den darauf folgenden Dialogen tragen Sie den SSL Key

```
/etc/letsencrypt/rsa-certs/privkey.pem
```



und das Zertifikat

```
/etc/letsencrypt/rsa-certs/fullchain.pem
```



ein und bestätigen jeweils mit <OK>.

Nach wenigen Minuten ist die Installation ohne weitere Interaktionen abgeschlossen.

3. Wir fahren nun mit der Konfiguration fort:

Zuerst erhöhen wir die Anzahl maximaler Prozesse und geöffneter Dateien in der Systemdatei /etc/systemd/system.conf:

```
nano /etc/systemd/system.conf
```

Suchen Sie die nachfolgenden Werte und entfernen Sie das Raute-Zeichen (#) am Anfang der Zeile und tragen den Wert 65000 hinter das Gleich-Zeichen (=) ein:

```
DefaultLimitNOFILE=65000
DefaultLimitNPROC=65000
DefaultTasksMax=65000
```

Nach den abgeschlossenen Änderungen müssen diese mit

```
systemctl daemon-reload
```

einmalig übernommen werden. Näheres dazu finden Sie hier. Fahren wir nun fort und entfernen bzw. verschieben die Jitsi-vHostdatei aus dem Webserververzeichnis in das conf.d-Verzeichnis.

```
rm -f /etc/nginx/sites-enabled/konferenz.dedyn.io.conf
mv /etc/nginx/sites-available/konferenz.dedyn.io.conf /etc/nginx/conf.d/
cd /etc/nginx/conf.d/
mv http.conf http.conf.disabled
```

Ändern Sie die Datei /etc/nginx/conf.d/**konferenz.dedyn.io**.conf ab: Server-Block 80 und Server-Block 443.:

```
nano /etc/nginx/conf.d/konferenz.dedyn.io.conf
server {
        listen 80 default_server;
        listen [::]:80 default_server;
        server_name konferenz.dedyn.io;
        location ^~ /.well-known/acme-challenge/ {
                default_type "text/plain";
                root /usr/share/jitsi-meet;
        }
        location = /.well-known/acme-challenge/ {
```

```
        return 404;
    }
    location / {
        return 301 https://$host$request_uri;
    }
}
server {
    listen 443 ssl http2;
    listen [::]:443 ssl http2;
    server_name konferenz.dedyn.io;

    ssl_protocols TLSv1.2 TLSv1.3;
[...]
location = /config.js {
    alias /etc/jitsi/meet/konferenz.dedyn.io-config.js;
}
[...]
```

In Anlehnung an meine Nextcloud Installationsanleitung habe ich noch weitere, **optionale** Anpassungen durchgeführt, so dass diese Konfigurationsdatei bei mir wie folgt aussieht:

```
server {
    listen 80 default_server;
    listen [::]:80 default_server;
    server_name konferenz.dedyn.io;
    location ^~ /.well-known/acme-challenge/ {
        default_type "text/plain";
        root /usr/share/jitsi-meet;
    }
    location = /.well-known/acme-challenge/ {
        return 404;
    }
    location / {
        return 301 https://$host$request_uri;
    }
}
server {
    listen 443 ssl http2 default_server;
    listen [::]:443 ssl http2 default_server;
    server_name konferenz.dedyn.io;
    ssl_protocols TLSv1.2 TLSv1.3;
```

```
        ssl_prefer_server_ciphers on;

        #ssl_ciphers
"EECDH+ECDSA+AESGCM:EECDH+aRSA+AESGCM:EECDH+ECDSA+SHA256:EECDH+aRSA+SHA256:EECDH+ECDSA+SHA384:EEC
DH+ECDSA+SHA256:EECDH+aRSA+SHA384:EDH+aRSA+AESGCM:EDH+aRSA+SHA256:EDH+aRSA:EECDH:!aNULL:!eNULL:!MED
IUM:!LOW:!3DES:!MD5:!EXP:!PSK:!SRP:!DSS:!RC4:!SEED";

        ssl_ciphers "TLS-CHACHA20-POLY1305-SHA256:TLS-AES-256-GCM-SHA384:ECDHE-ECDSA-CHACHA20-
POLY1305:ECDHE-RSA-CHACHA20-POLY1305:ECDHE-RSA-AES256-GCM-SHA512:DHE-RSA-AES256-GCM-SHA512:ECDHE-RSA-
AES256-GCM-SHA384";

        ssl_ecdh_curve X448:secp521r1:secp384r1:prime256v1;

        ssl_session_timeout 1d;

        ssl_session_cache shared:SSL:50m;

        ssl_session_tickets off;

        ssl_stapling on;

        ssl_stapling_verify on;

        add_header Strict-Transport-Security "max-age=31536000";

        ssl_certificate /etc/letsencrypt/rsa-certs/fullchain.pem;

        ssl_certificate_key /etc/letsencrypt/rsa-certs/privkey.pem;

        ssl_certificate /etc/letsencrypt/ecc-certs/fullchain.pem;

        ssl_certificate_key /etc/letsencrypt/ecc-certs/privkey.pem;

        ssl_trusted_certificate /etc/letsencrypt/ecc-certs/chain.pem;

        ssl_dhparam /etc/ssl/certs/dhparam.pem;

        root /usr/share/jitsi-meet;

        # ssi on with javascript for multidomain variables in config.js

        ssi on;

        ssi_types application/x-javascript application/javascript;

        index index.html index.htm;

        error_page 404 /static/404.html;

        gzip on;

        gzip_types text/plain text/css application/javascript application/json;

        gzip_vary on;

        location = /config.js {

                alias /etc/jitsi/meet/konferenz.dedyn.io-config.js;

        }

        location = /external_api.js {

                alias /usr/share/jitsi-meet/libs/external_api.min.js;

        }

        #ensure all static content can always be found first

        location ~ ^/(libs|css|static|images|fonts|lang|sounds|connection_optimization|.well-known)/(.*)$

        {

                add_header 'Access-Control-Allow-Origin' '*';

                alias /usr/share/jitsi-meet/$1/$2;
```

```
}
# BOSH
location = /http-bind {
        proxy_pass      http://localhost:5280/http-bind;
        proxy_set_header X-Forwarded-For $remote_addr;
        proxy_set_header Host $http_host;
}

# xmpp websockets
location = /xmpp-websocket {
        proxy_pass http://127.0.0.1:5280/xmpp-websocket?prefix=$prefix&$args;
        proxy_http_version 1.1;
        proxy_set_header Upgrade $http_upgrade;
        proxy_set_header Connection "upgrade";
        proxy_set_header Host $http_host;
        tcp_nodelay on;
}
location ~ ^/([^/?&:'"]+)$ {
        try_files $uri @root_path;
}
location @root_path {
        rewrite ^/(.*)$ / break;
}
location ~ ^/([^/?&:'"]+)/config.js$
{
      set $subdomain "$1.";
      set $subdir "$1/";
      alias /etc/jitsi/meet/konferenz.dedyn.io-config.js;
}
#Anything that didn't match above, and isn't a real file, assume it's a room name and redirect to /
location ~ ^/([^/?&:'"]+)/(.*)$ {
        set $subdomain "$1.";
        set $subdir "$1/";
        rewrite ^/([^/?&:'"]+)/(.*)$ /$2;
}
# BOSH for subdomains
location ~ ^/([^/?&:'"]+)/http-bind {
        set $subdomain "$1.";
        set $subdir "$1/";
```

```
                set $prefix "$1";


                rewrite ^/(.*)$ /http-bind;
        }
        # websockets for subdomains
        location ~ ^/([^/?&:'"]+)/xmpp-websocket {
                set $subdomain "$1.";
                set $subdir "$1/";
                set $prefix "$1";
                rewrite ^/(.*)$ /xmpp-websocket;
        }
}
```

Die ganze Beispieldatei (/etc/nginx/conf.d/**konferenz.dedyn.io**.conf) können Sie hier herunterladen. Passen Sie nun die Stun-Server an:

```
nano /etc/jitsi/meet/konferenz.dedyn.io-config.js
```

Ersetzen Sie den bestehenden stunServer Block mit den Google-Servereinträgen bspw. durch den nachfolgenden Block:

```
[...]
stunServers: [
        { urls: 'stun.1und1.de:3478' },
        { urls: 'stun.t-online.de:3478' },
        { urls: 'stun.nextcloud.com:443' },
        { urls: 'stun.sipgate.net:3478' }
],
[...]
useStunTurn: true,
[...]
p2p: { useStunTurn: true,
[...]
```

Betreiben Sie einen eigenen coturn/stun-Server, bspw. wie hier beschrieben, so tragen Sie Ihren cotrun-Server und -Port ein:

```
[...]
stunServers: [
        { urls: 'stun:konferenz.dedyn.io:5349' }
],
[...]
useStunTurn: true,
[...]
p2p: { useStunTurn: true,
```

```
[...]
```

Bei einem eigenen coturn-/stun-Server muss zudem auch noch diese Datei angepasst werden:

```
nano /etc/prosody/conf.d/konferenz.dedyn.io.cfg.lua
```

Ersetzen Sie den Block:

```
turncredentials_secret = "Das-Secret-aus-ihrer-coturn-Konfiguration";

turncredentials = {
{ type = "stun", host = "konferenz.dedyn.io", port = "5349" },
{ type = "turn", host = "konferenz.dedyn.io", port = "5349", transport = "udp" },
{ type = "turns", host = "konferenz.dedyn.io", port = "5349", transport = "tcp" }
};
```

Achten Sie darauf, dass sowohl das coturn-**Secret**, als auch der coturn-**Port** übereinstimmen und auch in der Firewall freigegeben sind. Eine Liste weiterer, öffentlicher „NICHT-„Google-Server finden Sie hier: STUN-Server. Mit den folgenden Anpassungen optimieren wir Jitsi noch, bevor wir es letzmalig neu starten:

```
nano /etc/jitsi/meet/konferenz.dedyn.io-config.js
[...]
defaultLanguage: 'de',
// Standardsprache auf Deutsch setzen
[...]
disableThirdPartyRequests: true,
// Externe Verbindungen bspw. gravatar.com werden verhindert
[...]
channelLastN: 4,
// Es werden nur die letzten 4 aktiven Sprecher gestreamt
[...]
enableLayerSuspension: true,
// Reduzierung der CPU u. Bandbreite - Übertragung nur des Streams, die angesehen werden
[...]
startAudioOnly: true,
// Video ist beim Start des Meetings deaktiviert, kann bzw. muss manuell aktiviert werden
[...]
disableAudioLevels: true,
// Deaktivierung der blauen Audio-Dots beim Speaker
[...]
nano /etc/jitsi/videobridge/logging.properties
.level=WARNING
# Logging reduzieren
```

Nun ist Jitsi sowohl für Firefox 75+ als auch Chromium vorbereitet. Starten Sie den Webserver nginx und Jitsi letztmalig neu:

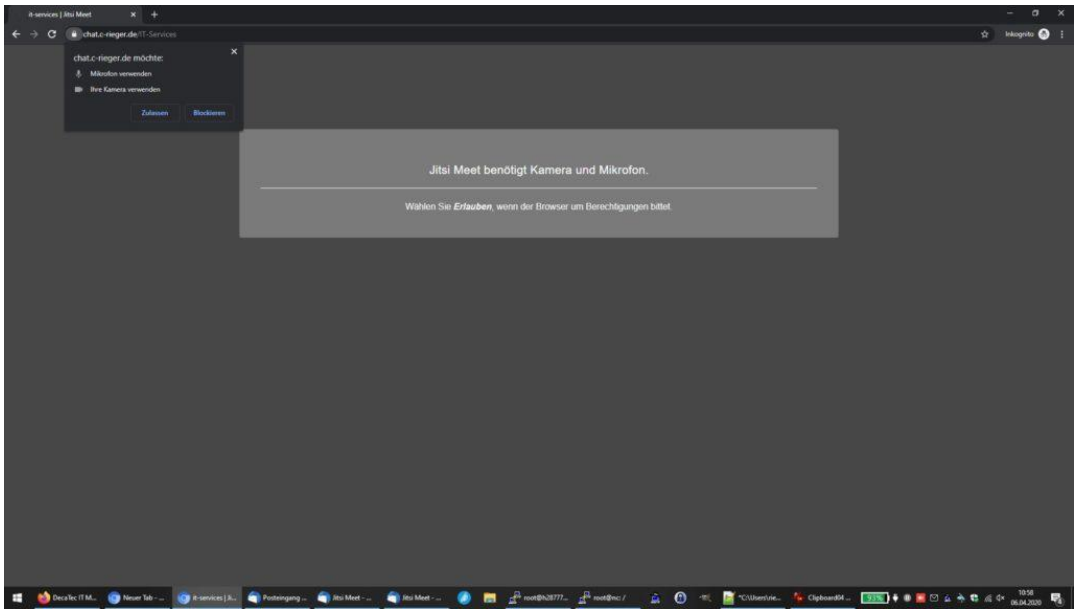
```
service nginx restart && service prosody restart && service jicofo restart && service jitsi-videobridge2 restart
```

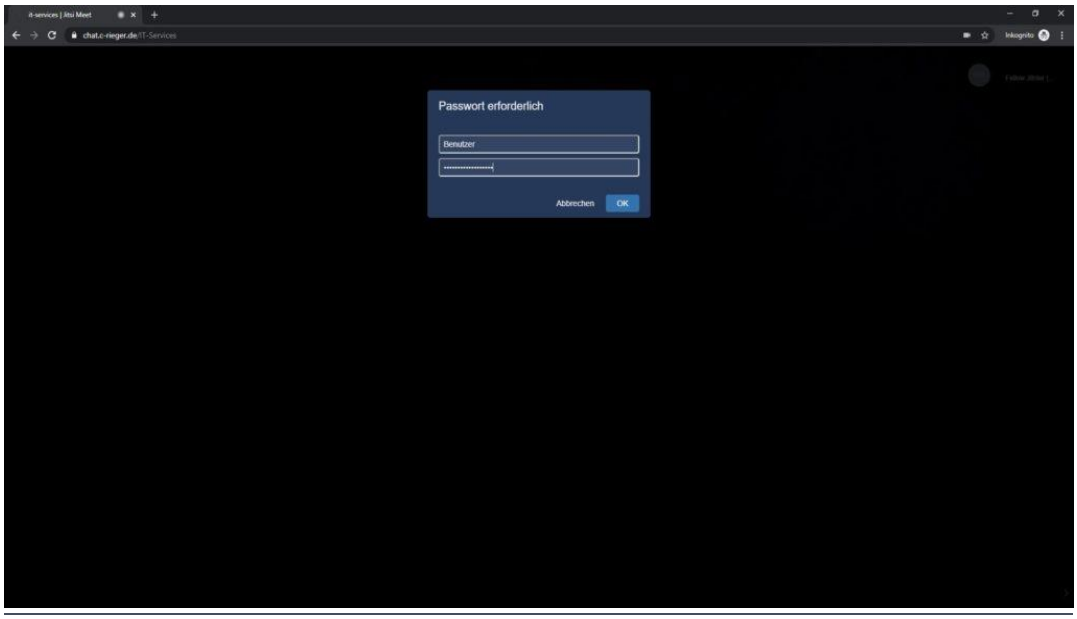
Um Jitsi weiter abzusichern können Sie diesem Link folgen:
https://github.com/jitsi/jicofo/blob/master/README.md#secure-domain

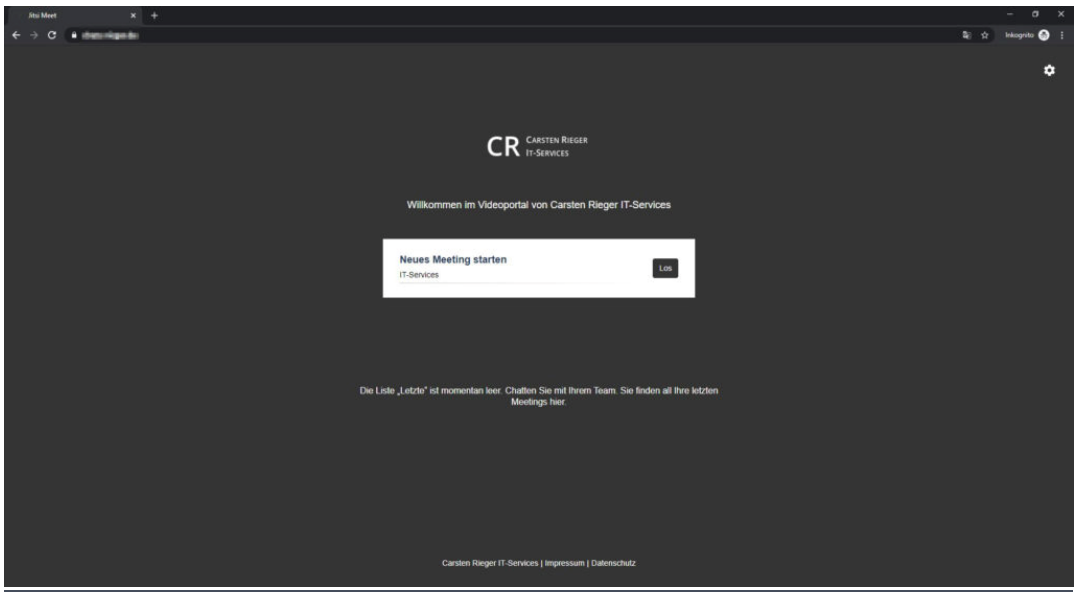
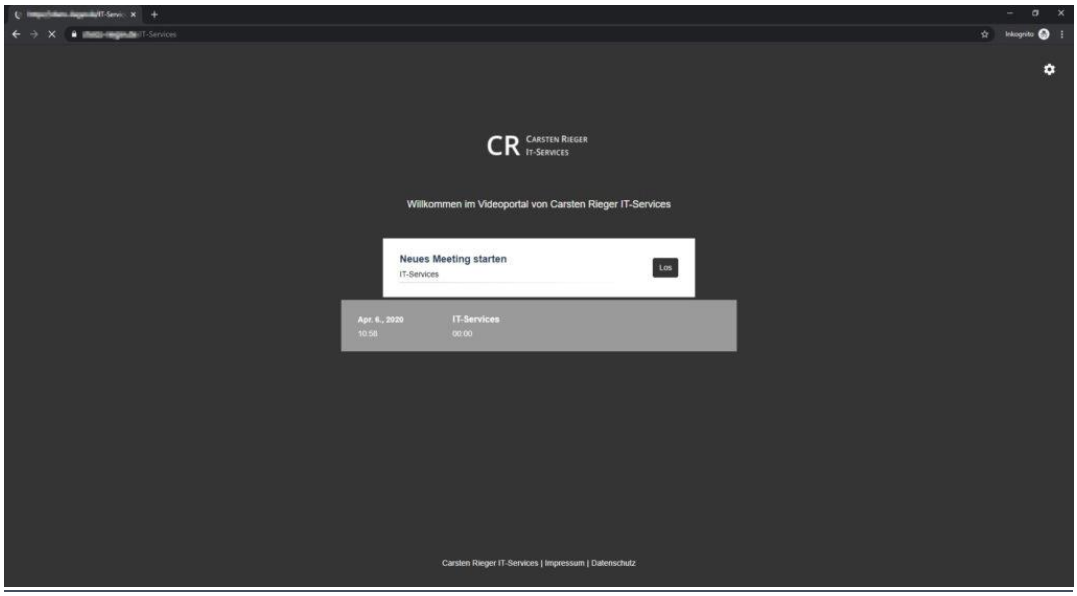
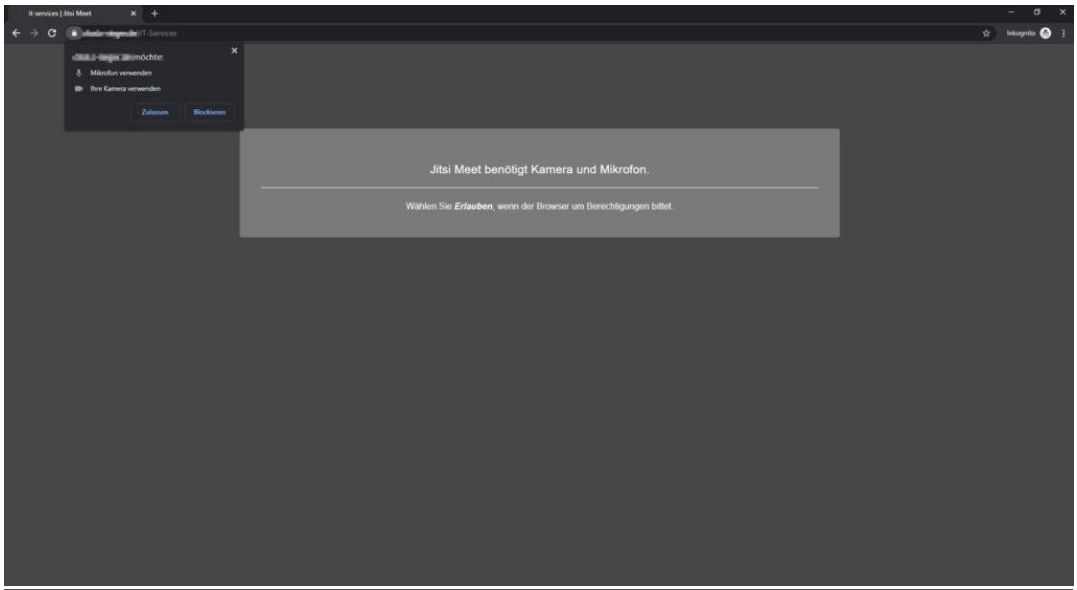
Betreiben Sie den Jitsi-Server hinter einem NAT, so finden Sie hier weitere, **notwendige Massnahmen**:
https://github.com/jitsi/jitsi-meet/blob/master/doc/quick-install.md#advanced-configuration

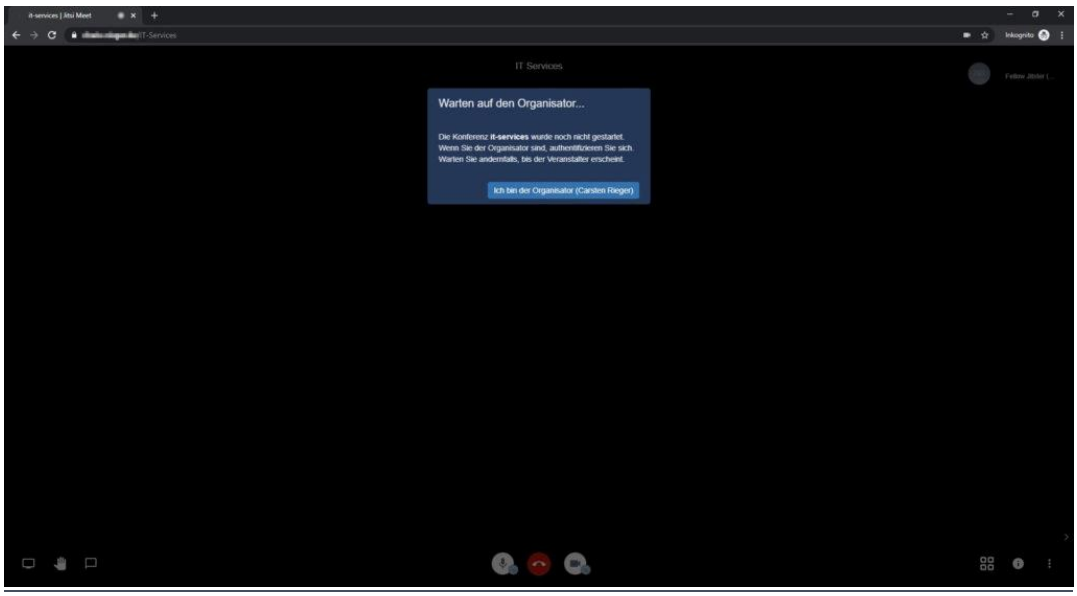
Anpassungen der Startseite, bspw. mit einem Datenschutzhinweis, können Sie u.a. nach dieser Anleitung pflegen: KUKETZ-Blog

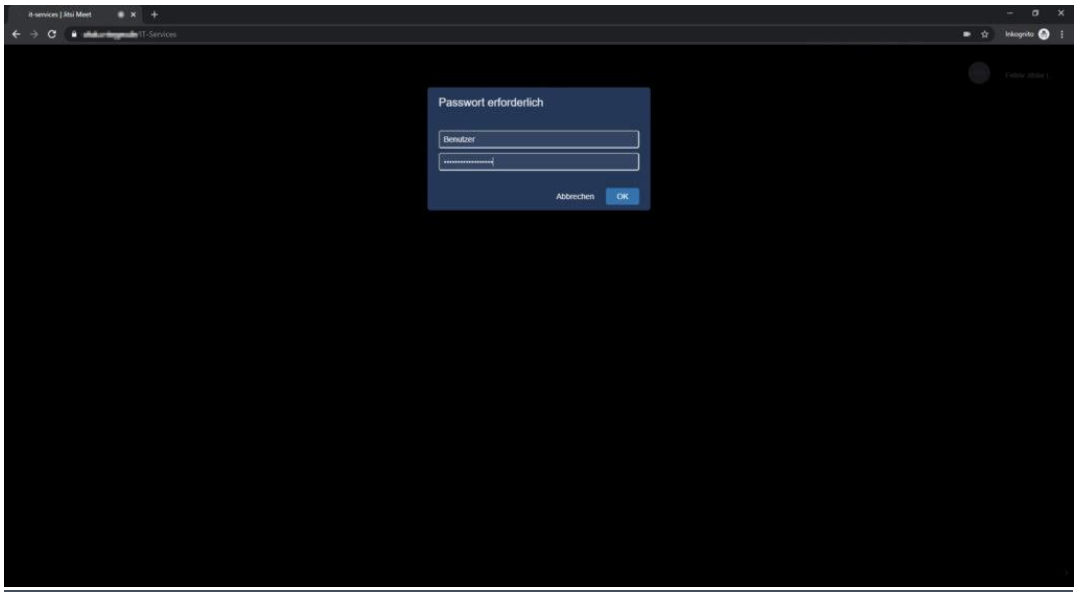
Ihr Konferenzserver erwartet Sie ab sofort unter https://**konferenz.dedyn.io**:

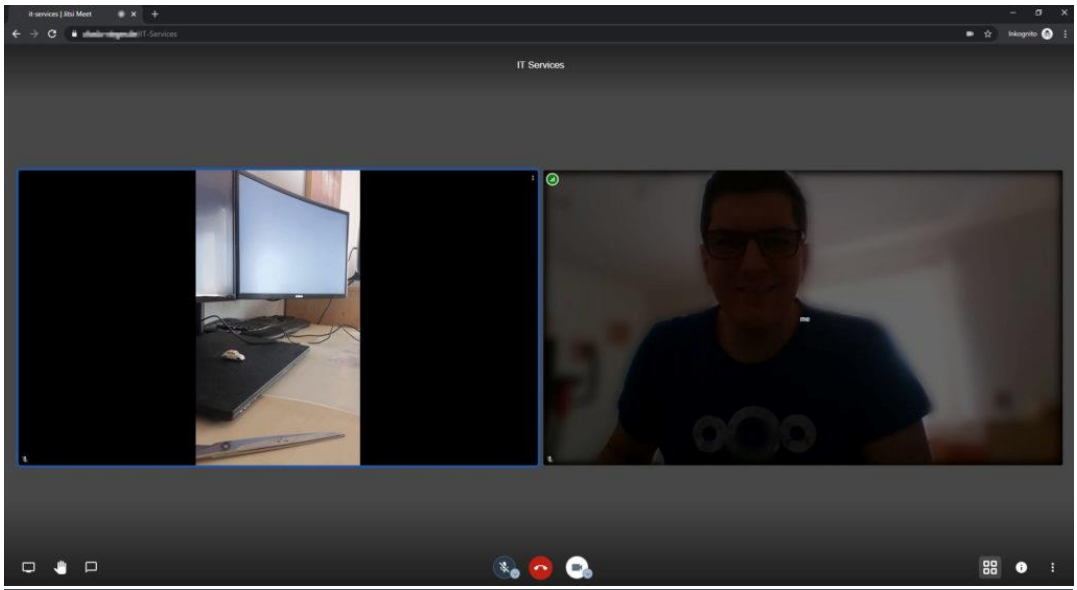
- 

- 

- 
Jitsi Meet
Inkognito
CR Carsten Rieger IT-Services
Willkommen im Videoportal von Carsten Rieger IT-Services
Neues Meeting starten
IT-Services
Los
Die Liste „Letzte" ist momentan leer. Chatten Sie mit Ihrem Team. Sie finden all Ihre letzten Meetings hier.
Carsten Rieger IT-Services | Impressum | Datenschutz
- 
Inkognito
CR Carsten Rieger IT-Services
Willkommen im Videoportal von Carsten Rieger IT-Services
Neues Meeting starten
IT-Services
Los
Apr. 6., 2020
10:58
IT-Services
00:00
Carsten Rieger IT-Services | Impressum | Datenschutz
- 
Inkognito
möchte:
Mikrofon verwenden
Ihre Kamera verwenden
Zulassen
Blockieren
Jitsi Meet benötigt Kamera und Mikrofon.
Wählen Sie Erlauben, wenn der Browser um Berechtigungen bittet.

- 

- 

- 

Um Jitsi im Hinblick auf ein bestehendes CI o.Ä. anzupassen, finden Sie nachfolgend zwei **Beispiel**anpassungen der Jitsi-Startseite:

/usr/share/jitsi-meet/static/welcomePageAdditionalContent.html

/usr/share/jitsi-meet/interface_config.js

Installieren Sie sich noch die Mobil-App und genießen Sie neue Konferenz- und Meetingerfahrungen!

Möchten Sie einen Blick in die Logfiles werfen, so finden Sie diese hier:

```
/var/log/jitsi/jvb.log

/var/log/jitsi/jicofo.log

/var/log/prosody/prosody.log
```

Die Installation ist abgeschlossen und Sie sind ein weiteres Stück unabhängiger von Skype, WhatsApp, Google und Co.