whatsapp web for more than 4 linked devices

Whatsapp has this annoying limitation of a maximum of 4 linked devices, which i constantly exceed. So i keep linking my phone and browser several times a week which is extremely annoying.

Unfortunately I haven't found a way to actually link more than 4 devices with whatsapp, but this method described here will make it possible to share one linked browser session across multiple pc's so you don't exceed your 4 browser limit.

In a nutshell, we run a docker container on a server somewhere in the internet. in the docker container we have a chromium (the community version of google's chrome) browser running in full screen mode showing the whatsapp web app page linked to our phone. To access that browser, we will use a web-based remote-desktop solution based on novnc - kasm. Luckily, the people at linuxserver.io have already taken care of most of the work involved in this by providing their awesome “Webtop” docker containers. Actually, they even have a “chromium” container which is based on their webtop container and serves a remote chromium browser. So there is actually very little work for us to do!

Another nice thing about this solution is, that the webtop based container's remote desktop webpage can be installed as a progressive webapp, just like the whatsapp web app, so with a few tricks, we can even keep this functionality and keep the original user experience almost completely.

Limitations

One thing we're going to loose is the whatsapp icon with the number of conversations with unread messages attached to it.. on the orignal whatsapp web app the favicon is updated with a number showing you how many chats you have with unread messages, that's not going to survive through our remote desktop session. Notifications on the other hand will work, so you will still get pop-ups for new messages like you did with the original web app.

the other inconvenience is going to be attaching files. you first have to upload files to the remote browser container and then attach them to the whatsapp message. the same is true for drag and drop, What did work on the other hand was copy/pasting images, so i could still take a screenshot and then simply paste it into a chat.

Security

A word of caution here: whatsapp obviously spends a lot of time thinking about and implementing security into their product. the Limit of a maximum of 4 device is certainly related to that effort. It is advisable, to follow some basic best practices to keep your whatsapp under your own control, but i'm not going to include all this in here, as this page is purely about getting the remote browser up and running and making it feel like the real whatsapp app. Most probably, using a reverse proxy in front of your remote desktop container is going to be the simplest and best solution to keep your personal whatsapp web secured good enough. you could use something like Authentik or Authelia as a reverse proxy with authentication built in to implement a HTTPS protected web-app with MFA if you so desire.

For your remote browser to be installable and work as progressive web-app you will have to either have valid, signed SSL certificates or use plain HTTP which is not recommended anywhere further than on localhost. You can run the remote browser with self signed certificates but the installed PWA might not work correctly (in my case the PWA only showed the ssl certificate warning and as soon as i clicked to continue it opened the actuall app in a browser window and no longer in its own window).

the setup

the setup shown here is done using docker-compose, of course you can also adopt that to other methods to run docker containers.

we start with our docker-compose.yml file:

docker-compose.yml

```
---
version: "2.1"
services:
  whatsapp:
    image: lscr.io/linuxserver/chromium:latest
    security_opt:
      - seccomp:unconfined #optional
    environment:
      - PUID=1000
      - PGID=1000
      - TZ=Europe/Zurich
      - SUBFOLDER=/ #optional
      - TITLE=Shared Whatsapp #optional
      - CUSTOM_USER=whatsappuser
      - PASSWORD=REPLACEmeWithYourOwnSecretPassword!
      - DISABLE_IPV6=true
      - NO_DECOR=true
      - CHROME_CLI=--kiosk https://web.whatsapp.com
    volumes:
      - ./data:/config
# get the favicon: wget https://web.whatsapp.com/favicon.ico
      - ./conf/favicon.ico:/kclient/public/favicon.ico
# get the manifest: wget https://web.whatsapp.com/data/manifest.json
      - ./conf/manifest.json:/kclient/public/manifest.json
    ports:
      - 3000:3000
      - 3001:3001
    shm_size: "1gb"
    restart: unless-stopped
```

IMPORTANT: obviously you should adjust your username and password!

next we have to create the `conf` directory and get the `favicon.ico` and `manifest.json` from the original whatsapp web-app for that PWA look and feel. This is not really necessary, if you don't mind having your PWA run with the standard chromium icon. so it's purely cosmetic, but if you decide on skipping this, you need to remove the two volume mounts from the `docker-compose.yml` above

```
mkdir conf
cd conf
wget https://web.whatsapp.com/favicon.ico
wget https://web.whatsapp.com/data/manifest.json
cd ..
chown -R 1000.1000 conf
```

now start your docker-compose app using `docker compose up -d` and then open a browser on https://localhost:3001 or http://localhost:3000 and login with your username and password. The first time you will have to link your device, but once this is done, you won't see this screen anymore for a long time :)

Other use cases

This solution can also be used for all other kinds of web-services and is not restricted to whatsapp. simply adjust the url in the `docker-compose.yml` file and you did the same thing for a different webpage :)