Installation

- Docker installtion
- Client einrichten

Docker installtion

Beschreibung:

RustDesk installation via Docker

Vorrausetzung

docker-compose

```
apt install docker-compose
```

Ports die in der Firewall / per NAT zur Verfügung gestellt werden müssen

TCP (**21115, 21116, 21117, 21118, 21119**)
UDP (**21116**)

In der Community OSS version gibt es keinen Webclient

zu empfehlen is es sich eine Subdomain hilfe.example.com als Beispiel anzulegen, anstatt sich die IP zu merken.

Installtion

Zuert legen wir uns ein Verzeichnis an z.b

```
mkdir /root/rustdesk/data
cd /root/rustdesk
```

Darin legen wir uns eine .env Datei an.

```
nano /root/rustdesk/.env
```

Inhalt

```
DATA_PATH=/root/rustdesk/data
```

Nun die Composer File

```
nano /root/rustdata/docker-compose.yml
```

Inhalt

```
version: '3'
        services:
          hbbs:
            container_name: hbbs
            image: rustdesk/rustdesk-server:latest
            environment:
              - DATA_PATH=${DATA_PATH}
            command: hbbs
            volumes:
              - ${DATA_PATH}:/root
            network_mode: "host"
            depends_on:
              - hbbr
            restart: unless-stopped
          hbbr:
            container_name: hbbr
            image: rustdesk/rustdesk-server:latest
            command: hbbr
            volumes:
              - ${DATA_PATH}:/root
            network_mode: "host"
            restart: unless-stopped
```

mit docker-compose up starten wir das ganze

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

API KEY anzeigen, dieser wird später im Client benötigt

```
cat /root/rustdesk/daten/id_ed25519.pub
```

Diesen Schlüssel brauchen wir später, aber auch nur wenn wir den Pro Server installiert haben, sonst ist der überflüssig.
Am besten in einem Paswort Safe speichern

Ansible Script das die oberen Schritte ausführt

Die host.ini

Inhalt:

```
192.168.178.122 ansible_user=root rustdesk_data_path=/root/rustdesk/daten
```

Nun das Playbook

rustdesk.yml

Inhalt:

```
---
- name: Deploy RustDesk via Docker Compose
  hosts: all
  become: yes
  vars:
    rustdesk_data_path: "{{ hostvars[inventory_hostname].rustdesk_data_path }}"

  tasks:
    - name: Create RustDesk data directory
      file:
        path: "{{ rustdesk_data_path }}"
        state: directory

    - name: Create .env file for RustDesk
      copy:
        dest: "{{ rustdesk_data_path }}/.env"
        content: |
          DATA_PATH={{ rustdesk_data_path }}

    - name: Create Docker Compose file for RustDesk
      copy:
        dest: "{{ rustdesk_data_path }}/docker-compose.yml"
        content: |
          version: '3'
          services:
```

```
    hbbs:
      container_name: hbbs
      image: rustdesk/rustdesk-server:latest
      environment:
        - DATA_PATH=${DATA_PATH}
      command: hbbs
      volumes:
        - ${DATA_PATH}:/root
      network_mode: "host"
      depends_on:
        - hbbr
      restart: unless-stopped
    hbbr:
      container_name: hbbr
      image: rustdesk/rustdesk-server:latest
      command: hbbr
      volumes:
        - ${DATA_PATH}:/root
      network_mode: "host"
      restart: unless-stopped

- name: Install Docker and Docker Compose
  apt:
    name:
      - docker
      - docker-compose
    state: latest
  when: ansible_os_family == "Debian"

- name: Start and enable RustDesk Docker container
  docker_compose:
    project_src: "{{ rustdesk_data_path }}"
    state: present
    restarted: yes

- name: Wait for 60 seconds for services to start
  pause:
    seconds: 60

- name: hole Inhalt von {{ rustdesk_data_path }}/id_ed25519.pub
```

```
    command: cat {{ rustdesk_data_path }}/id_ed25519.pub
    register: file_content

  - name: zeige API KEY
    debug:
      msg: "{{ file_content.stdout }}"
```

nachdem die host.ini angepasst wurde kann mit

```
ansible-playbook -i host.ini rustdesk.yml
```

rustdesk ausgerollt werden.
Am ende des Scriptes wird der API KEy ausgeben.

Diesen Schlüssel brauchen wir später, aber auch nur wenn wir den Pro Server installiert haben, sonst ist der überflüssig.
Am besten in einem Paswort Safe speichern.

Client einrichten

Beschreibung:

Nun da der Server läuft müssen wir den Client einrichten.

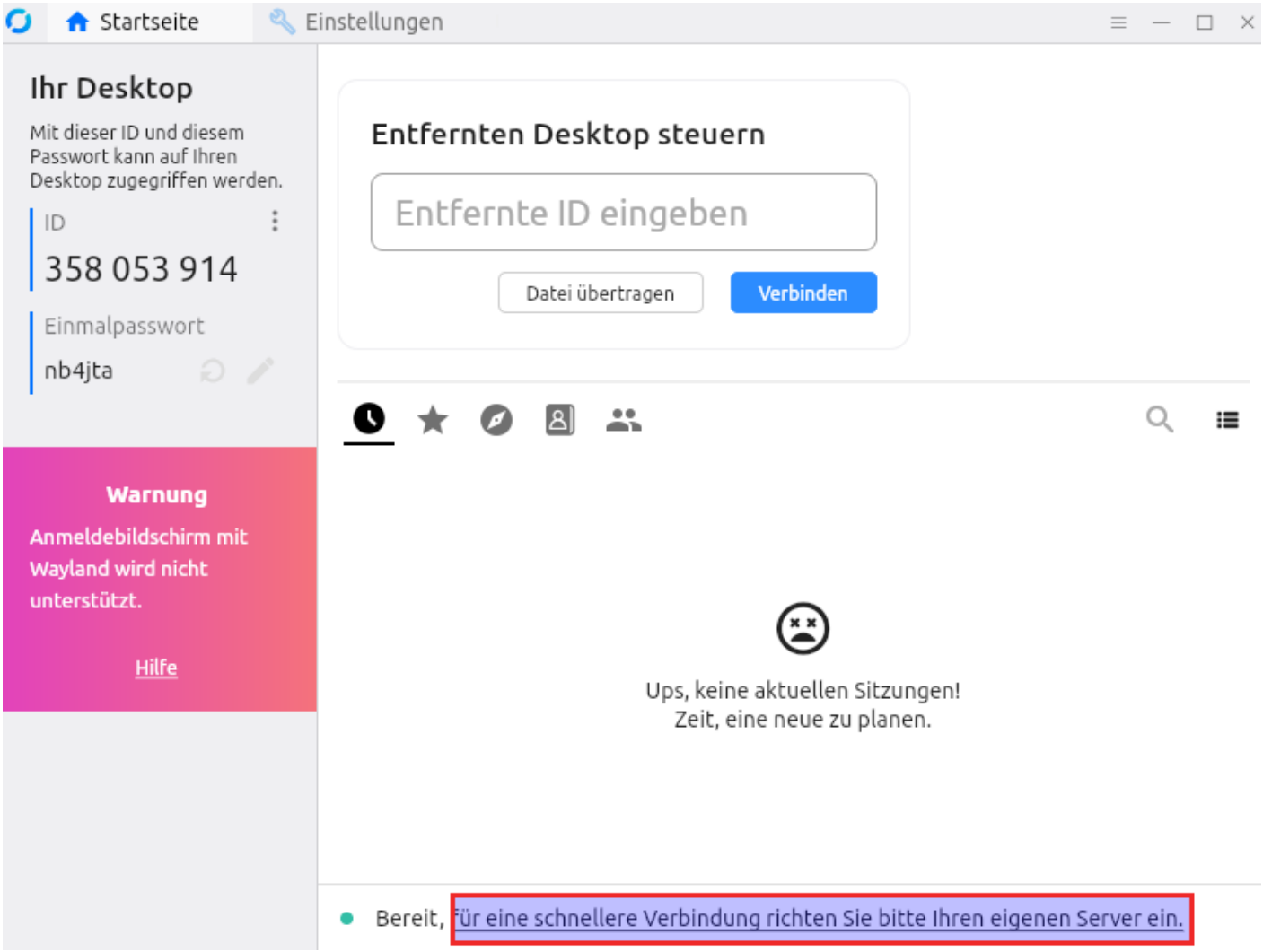
Installation

Dazu laden wir den den entsprechenden Clint für unser Betriebssystem von

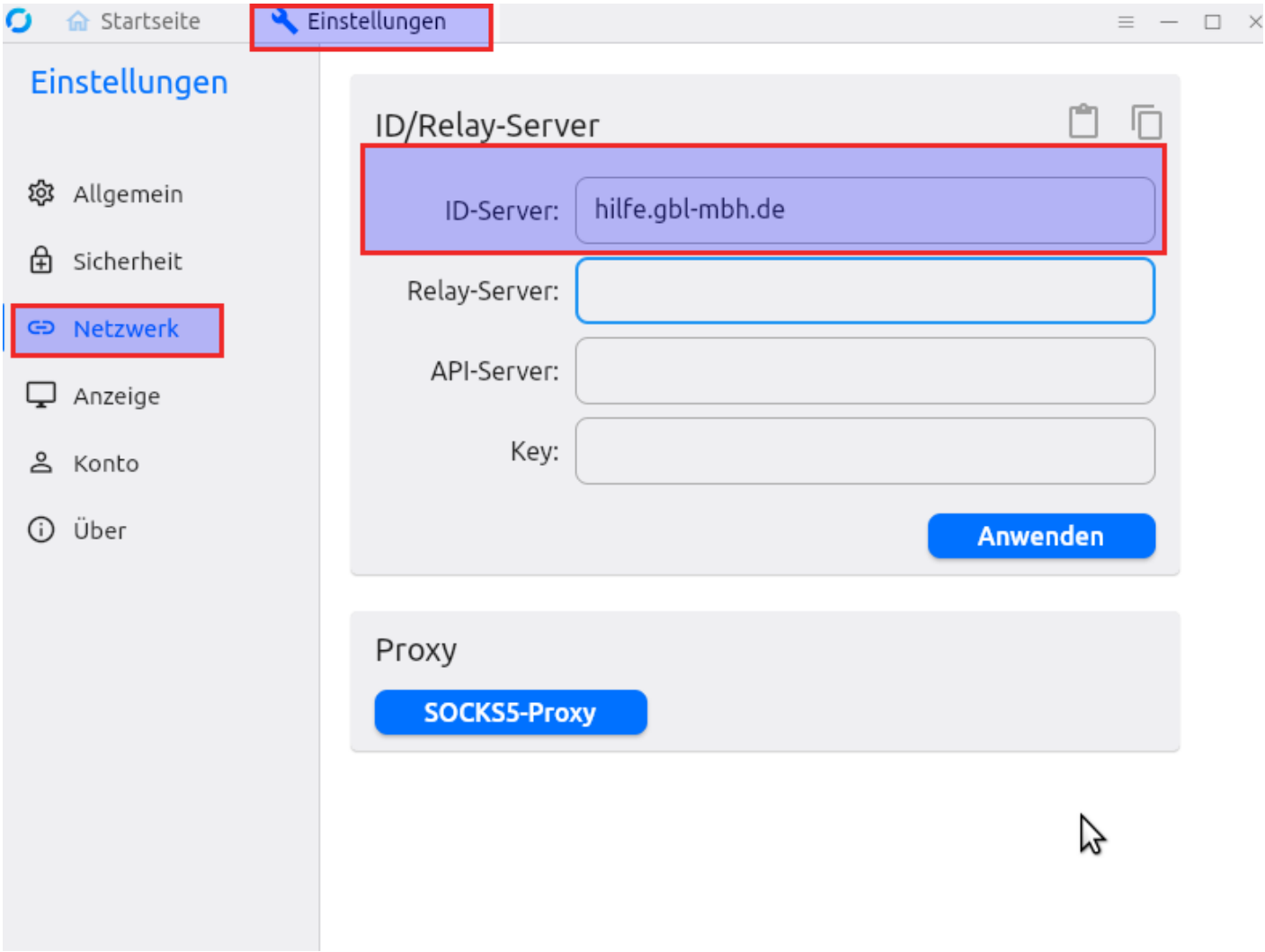
https://rustdesk.com/docs/en/client/

Konfiguration

Nach dem ersten Start wäre der Client mit Public servern schon startklar, allerdings sind die verbindungen langsam und gehen extern. Ob der Client Public server benutzt sieht man an dem text für eine schnellere Verbindung, richten sie ihren eigenen Server ein.

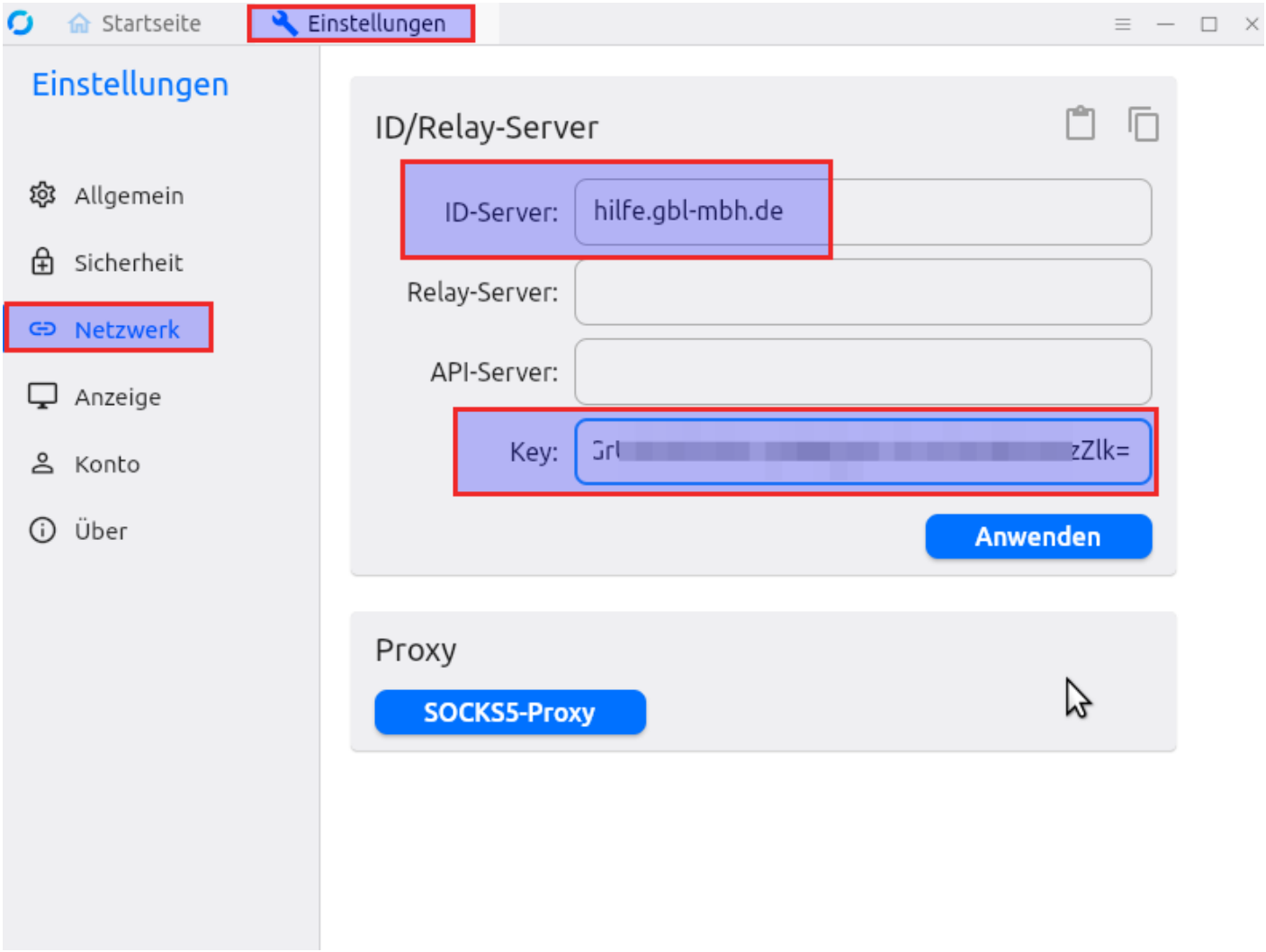


Wenn der Client gestartet ist dann unter Einstellungen Netzwerk gehen.
Dort bei ID Server die subdomain eintragen.
Die anderen Felder können bei der OSS Version leer gelassen werden.
beim Pro Server müssen die anderen Felder auch angebenen werden.
Dann auf Anwenden klicken

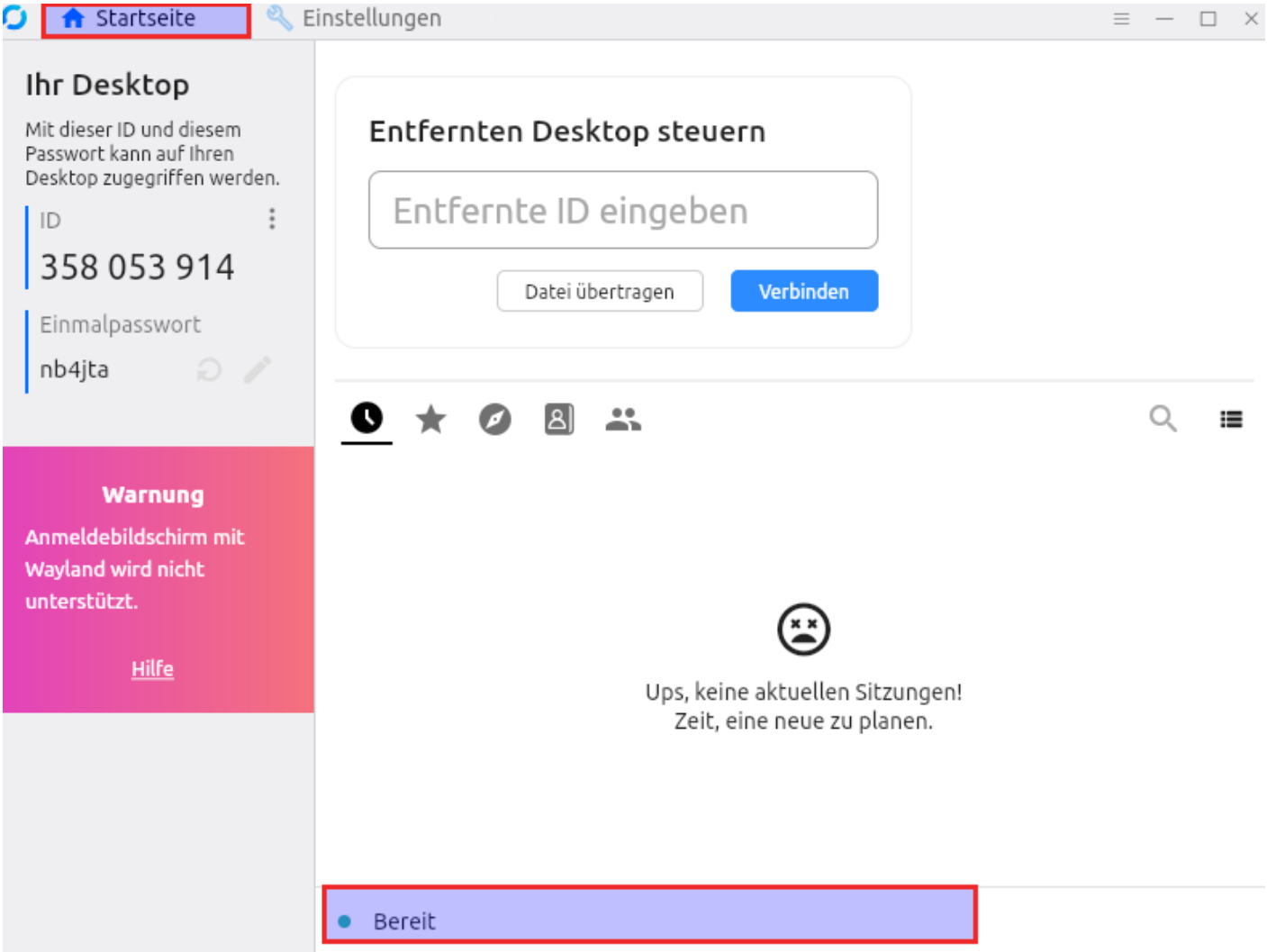


Einstellungen nur für Pro Server

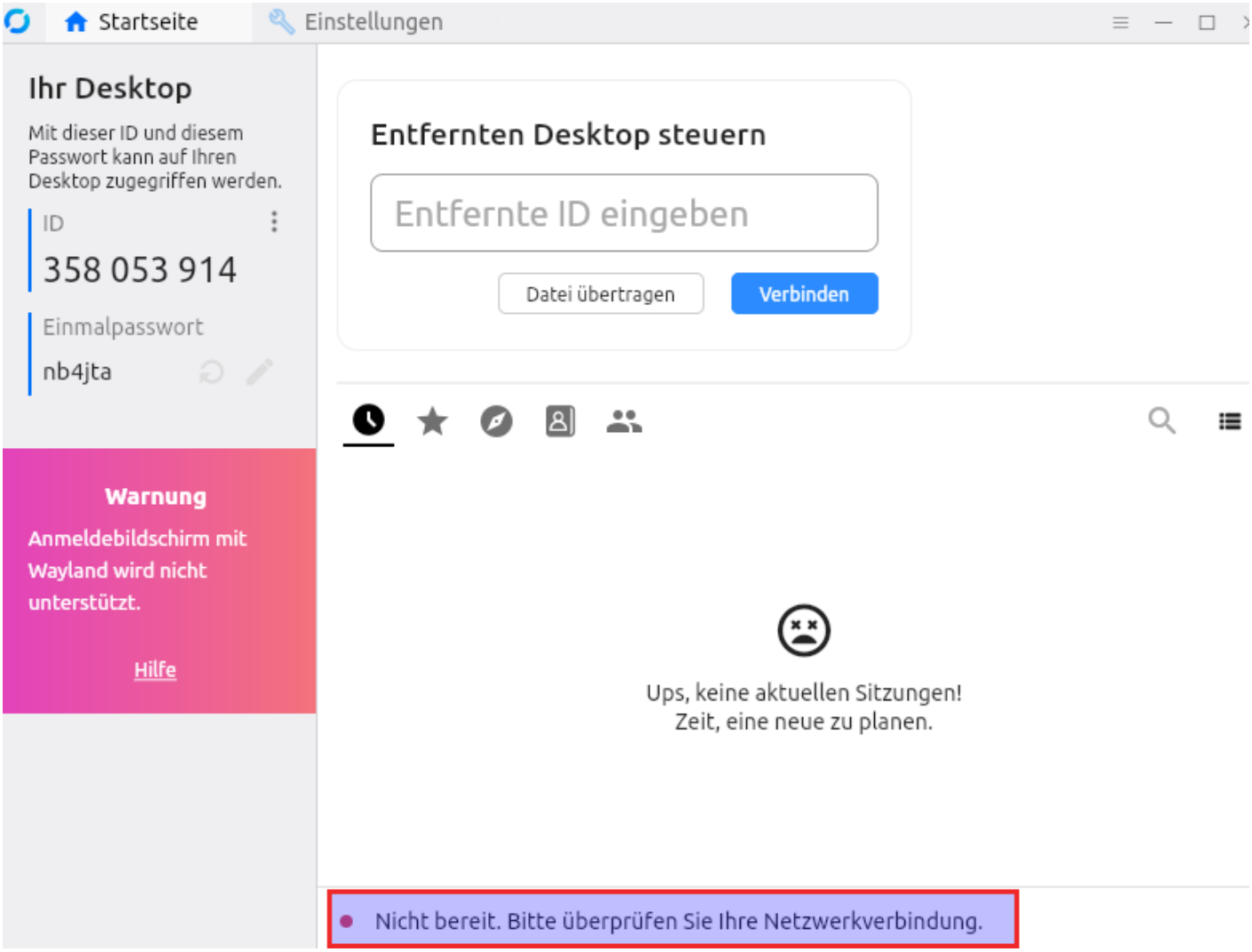
Da muss noch der Key mit angeben werden



Wenn man jetzt wieder auf Startseite klickt steht unten bereit, der Text ist weg.



hat man falsche domain angeben oder ist nicht erreichbar gibts diesen fehler.
Prüfen ob domain richtig geschrieben ob der Server über alle Ports erreichbar ist und der Server läuft


Startseite
Einstellungen
Ihr Desktop
Mit dieser ID und diesem Passwort kann auf Ihren Desktop zugegriffen werden.
ID
358 053 914
Einmalpasswort
nb4jta
Warnung
Anmeldebildschirm mit Wayland wird nicht unterstützt.
Hilfe
Entfernten Desktop steuern
Entfernte ID eingeben
Datei übertragen
Verbinden
Ups, keine aktuellen Sitzungen!
Zeit, eine neue zu planen.
Nicht bereit. Bitte überprüfen Sie Ihre Netzwerkverbindung.