Welcome LUG Frankfurt!

This is a local mini-server for today’s session.

Wi-Fi access

Start Page

Scan **after** connecting to Wi-Fi

SSID	`lugfrankfurt`
Password	`workshop@lugf`
DNS	`*.lug.lan → this server / 192.168.200.1`
Startpage	`http://lug.lan → this page`

Please use **http** in this internal network.

Services

lug.lan/materials
Materials for today's session ↗

files.lug.lan
Data exchange (files)
Login: **lugfrankfurt / pw4lugfrankfurt** ↗

git.lug.lan/explore/repos
Gitea (repos, issues, wiki) ↗

chat.lug.lan/p/lugfrankfurt
Etherpad (live notes) ↗

lug.lan/games/mines.html
Minesweeper (have fun!) ↗

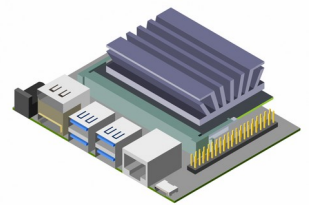
ai.lug.lan
llama.cpp (local AI demo) ↗

The problem:
Workshop chaos

- No or expensive internet
- Blocked ports
- "Please download materials beforehand" (half didn't…)
- Cloud provider outage
- "The cloud is just somebody else's computer"

The idea:
Own the infrastructure

- Independence
- Reliability
- Consistency
- Privacy & Security
- Portability & Simplicity



DHCP & DNS

Access Point

Reverse Proxy & Webserver

Containerized Services

- Fileserver
- Wiki
- Git
- Collaborative Pad
- Docker registry
- Python repository
- Games
- AI Server
- ...

Required: AP-capable Wifi card.

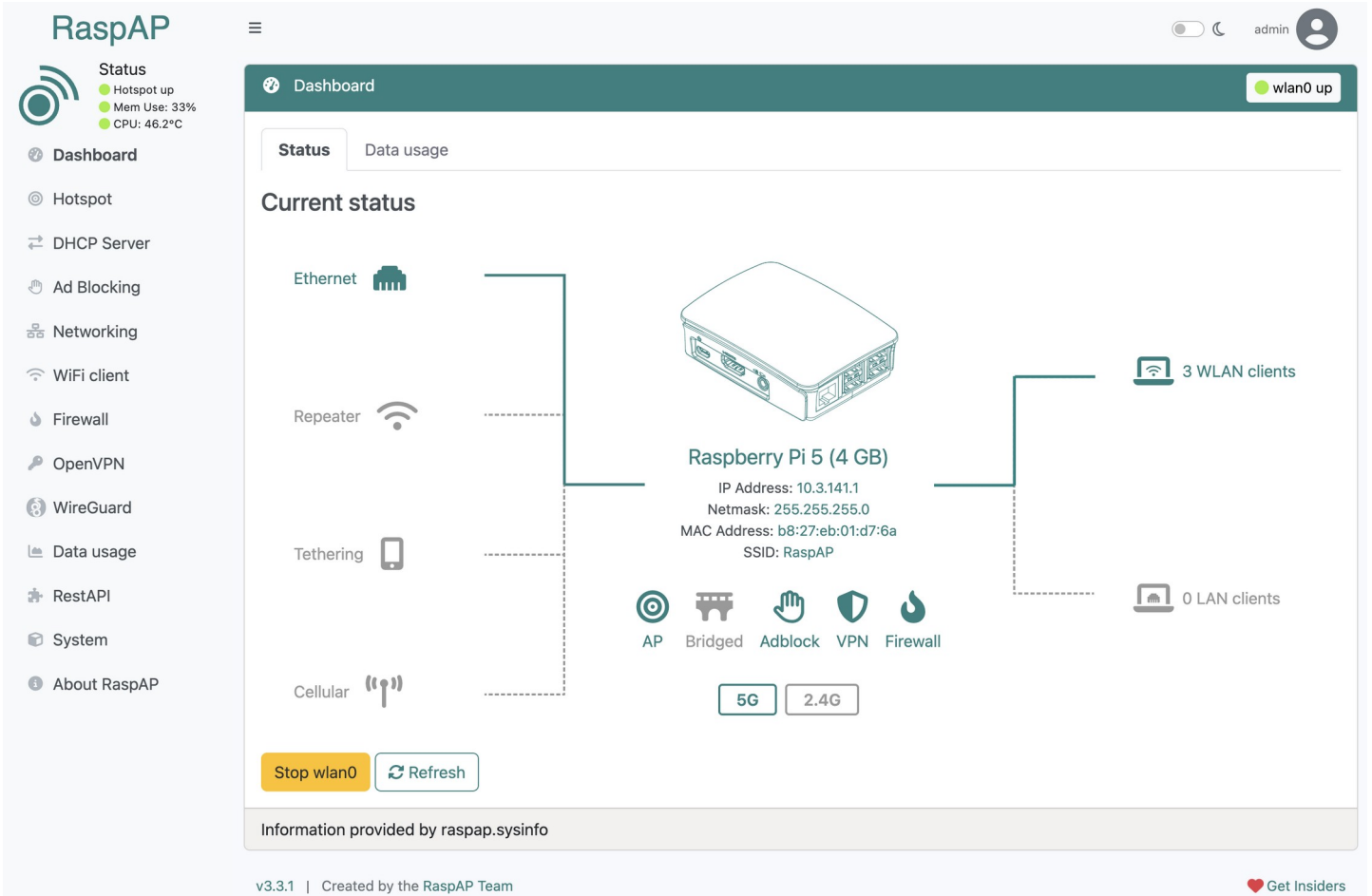
Test with:

```
$> iw list
```


```
Supported interface modes:
 * IBSS
 * managed
 * AP
 * AP/VLAN
 * monitor
```

```
Capabilities: 0x11ef
RX LDPC
HT20/HT40
SM Power Save disabled
RX HT20 SGI
RX HT40 SGI
TX STBC
RX STBC 1-stream
```

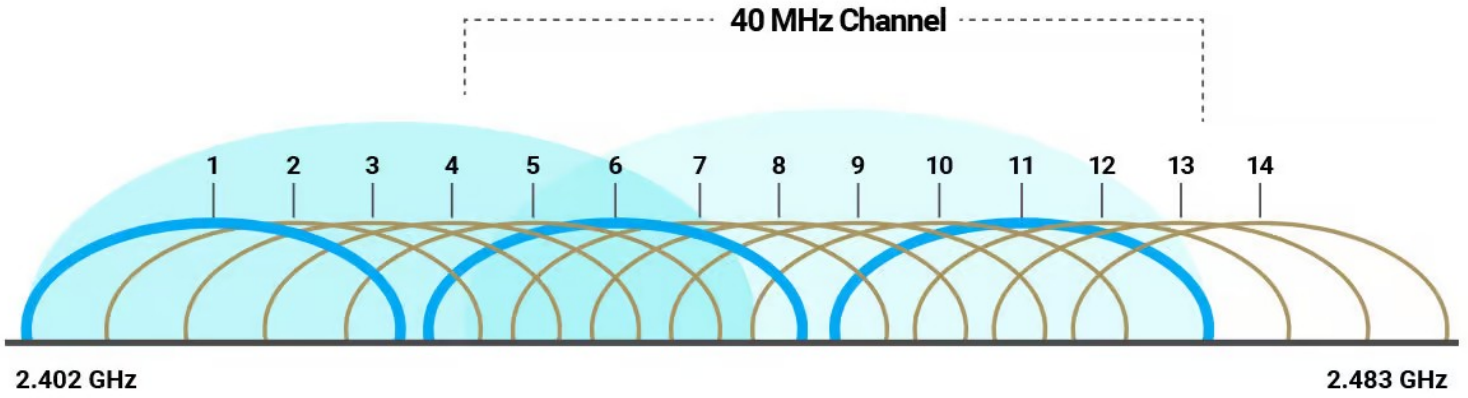
Component	**Package**	**Alternatives**
Access Point	`hostapd`	`NetworkManager (nmcli), RaspAP`
DHCP & DNS	`dnsmasq`	
Webserver & Reverse Proxy	`caddy`	`apache, nginx`
Services	`docker / docker-compose`	`podman`, distro-provided package, build from source


RaspAP
Status
Hotspot up
Mem Use: 33%
CPU: 46.2°C
Dashboard
Hotspot
DHCP Server
Ad Blocking
Networking
WiFi client
Firewall
OpenVPN
WireGuard
Data usage
RestAPI
System
About RaspAP
admin
Dashboard
wlan0 up
Status
Data usage
Current status
Ethernet
Repeater
Tethering
Cellular
Raspberry Pi 5 (4 GB)
IP Address: 10.3.141.1
Netmask: 255.255.255.0
MAC Address: b8:27:eb:01:d7:6a
SSID: RaspAP
AP
Bridged
Adblock
VPN
Firewall
5G
2.4G
3 WLAN clients
0 LAN clients
Stop wlan0
Refresh
Information provided by raspap.sysinfo
v3.3.1 | Created by the RaspAP Team
Get Insiders

	Layer	**Sublayer**	
4	Transport		TCP/UDP
3	Network		IP
2	Data Link	Logical link control	802.2 LLC (LAN)
		Media access control	CSMA/CA Collision Avoidance
1	Physical		**802.11 WIFI** **2.4 / 5** / 6 GHz, IR FHSS (Frequency Hopping), **DSSS** (Direct Sequence Spread)

	IEEE Standard	Streams	Channel Widths MHz	Data Rate MBit/s per Stream & Channel Width	2.4 GHz	5 GHz	6 GHz	Year
	802.11	1	20	2	X			1999
	802.11b	1	20	11	X			1999
	802.11a/g	1	20	54	X	X		2003
WIFI 4	802.11n	1-4	20-80	75	X	X		2009
WIFI 5	802.11ac	1-8	20-160	108		X		2014
WIFI 6	802.11ax	1-8	20-160	144	X	X	(X)	2021
WIFI 7	802.11be	1-8	20-160	155	X	X	X	2024
WIFI 8	802.11bn							2029

Wifi: ISM Band, Channel Width and Interference

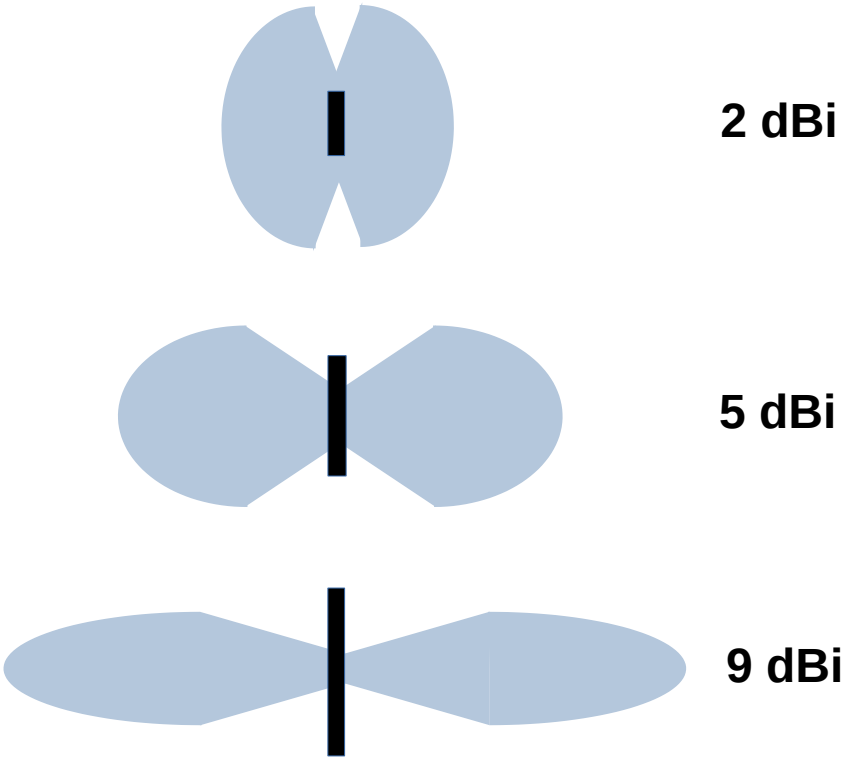


- **ISM Band** (Industrial, Scientific, Medical): **License-free**
- Originally reserved for industrial noise (RF Welding, Microwave Ovens, ...)
- Ironically now one of the **most crowded** frequency bands
- Only channels **1, 6, 11 on 20 MHz** don't interfere each other
- **40 MHz** channel width usually too much interference

Wifi: Antenna Gain dBi

dBi = "Radiation increase **in a direction** compared to an **isometric** radiator"

A longer Wifi antenna does not increase total radiation power, but redistributes it.



Wifi Access Point: Tips

- Use **5 GHz** band if possible, but **regulatory issues** (weather radar)
- **hostapd with Intel cards** usually only supports **WIFI 4** on 2.4 GHz
- Check WiFi space beforehand; use **lowest crowded channel**
- Use only channels **1, 6, or 11** in the 2.4 GHz band
- Higher throughput of **40 MHz** on 2.4 GHz mostly **too unstable**
- Choose a suitable **antenna size**
- 5 GHz has higher **absorption rate** than 2.4 GHz
- Always use security; at least **WPA2**
- **Never use WEP** security (trivially hackable)
- **Test setup** beforehand

QR-Code:	"**Q**uick **R**esponse Code"
Inventor:	1994 Densu / Toyota logistics Inspired by the Go boardgame
Standardized:	ISO/IEC 18004:2006

```
# Encoding
qrencode   --output=<filename>
           --level={LMQH}
           --type={PNG,SVG,ASCII,…}
           <string>

# Decoding
zbarimg <filename>
```

De-facto Standards by NTT Docomo Inc.:

Type	Encoding
URL	`https://<URL>`
eMail	`mailto:<email>`
Phone number	`tel:<phone number>`
Contact	`BEGIN:VCARD <contents> END:VCARD`
Calendar	`BEGIN:VEVENT <contents> END:VEVENT`
Coordinate	`geo:<LAT>,<LON>,<HEIGHT>`
Wifi Network	`WIFI:T:<AUTH TYPE>;S:<SSID>;P:<PASSWORD>;;`
2FA TOTP	`otpauth://totp/<SERVICE>:<USERNAME>?secret=<secret>`
App Store	`market://details?id=<play store url>`

https://github.com/zxing/zxing/wiki/Barcode-Contents

Component		Configuration file
Network interfaces		`/etc/network/interfaces`
Access Point (hostapd)		`/etc/hostapd/hostapd.conf`
DHCP / DNS (dnsmasq)		`/etc/dnsmasq.d/miniserver.conf`
Reverse Proxy (caddy)	Distro-based	`/etc/caddy/Caddyfile`
	Docker-based	`/srv/config/caddy/Caddyfile`
Services (docker)		`/srv/docker-compose.yaml`

Component	Monitoring
hostapd	`hostapd_cli`
dnsmasq	`/var/lib/misc/dnsmasq.leases`
docker	`docker {stats, logs}`
throughput	`iperf3`
system load	`top, htop, glances`
Wifi environment	`iw <dev> scan`

Command	Usage
`docker images`	Show local images
`docker pull <image>`	Download image from dockerhub
`docker ps`	Show containers
`docker logs <container>`	Show logs for a container
`docker compose up`	Start docker-compose setup
`docker compose down`	Stop & dismantle compose setup
`docker stats`	Display resource usage
`docker rmi <image>`	Remove image
`docker rm <container>`	Remove container