

FreiFunk Münster

Hier soll alles rund um das Thema Freifunk gesammelt werden bis es eine geeignetere Stelle gibt.

Organisation

Wir nutzen den IP Block des (ehemaligen) [Piraten Funk Münster](#) und haben uns zusätzlich einen IPv6 Block im [Freifunk Wiki](#) gesichert. Als Kontaktadresse gibt es <mailto:freifunkmuenster@warpzone.ms> die bisher auf Void und [Sandzwerg](#) weiterleitet.

IP-Bereich

IPv4: 10.43.0.0/16 IPv6: fd68:e2ea:a53::/48

Reservierter IP-Bereich

Dieser Bereich ist für feste IPs, Infrastruktur, Dienste usw. reserviert.

IPv4: 10.43.0.0/21 IPv6: fd68:e2ea:a53::/50

IP-Bereich Gateway 1 (VOID-DEV)

IPv4-Adresse: 10.43.0.2 IPv6-Adresse: fd68:e2ea:a53::2/48

IPv4: 10.43.8.0/21 IPv6: fd68:e2ea:a53:4000::/50

IP Bereich Gateway 2 (Warpzone)

IPv4-Adresse: 10.43.0.3 IPv6-Adresse: fd68:e2ea:a53::3/48

IPv4: 10.43.16.0/21 IPv6: fd68:e2ea:a53:8000::/50

IP Bereich Gateway 3 (Fusselkater)

IPv4-Adresse: 10.43.0.4 IPv6-Adresse: fd68:e2ea:a53::4/48

IPv4: 10.43.24.0/21 IPv6: fd68:e2ea:a53:C000::/50

Freifunk am Hawerkamp

Der Hawerkamp soll mit freien WLAN, am besten auf Basis von Freifunk ausgestattet werden. Es gibt eine Spende von 25 Cisco Access Points vom Typ AIR-AP-1131AG-E-K9

Hardware

Hersteller	Modell	WLAN	ext.Antenne?	PoE	openwrt
Cisco	AIR-AP-1131AG-E-K9	a/b/g	nein	ja	nein
TP-Link	WR841ND	b/g/n 2.4Ghz	ja	nein	ja
Ubiquity	Nanostation M5 Loco	5 GHz	nein	ja	nein

Firmware

Die Firmware wird vermutlich auf [Gluon](#) vom Freifunk Lübeck basieren. Ein Fork der [Lübecker Einstellungen](#) unter eigenem Namen ist auf Github als [site-ffms](#) zu finden. Außerdem gibt es ein allgemeines Münsteraner Freifunk git unter [gitolite@warpzone.ms:freifunk.git](#) zu finden.

To-Do

- Community bei Freifunk.net als Subdomain/Wordpress Instanz anlegen lassen
- [API File](#) erstellen ([Anleitung](#))
- Gateway VM auf Warpzone Server einrichten (server.warpzone.ms -p 2223)
 - Fastd
 - VPN (openvpn)
 - DNS/DHCP (dnsmasq)
 - DHCP (isc-dhcpd)
 - ravid (dhcp ipv6)
 - alfred (teil von batman / Userspace / <https://github.com/tcatm/alfred>)
 - alfred-json (<https://github.com/tcatm/alfred-json>)
 - batman-adv (meshing) [Vorsicht: zu neuen Kernel nutzen die neue batman Version mit neuem kompatibility level -> nicht kompatibel]
 - FFMap-Backend (Übersichtskarte <https://github.com/ffnord/ffmap-backend>)
 - FFMAP (Übersichtskarte <https://github.com/ffnord/ffmap-d3>)
 - Webserver nötig falls nicht auf bestehendem System
 - DOKU(!)
- VPN: Bei Lübecker / Berlinern nach Modalitäten für VPN anfragen: Alternativ eigenen Zugang, mit kosten verbunden (testweise genutzt: <http://mullvad.net/>)
- Firmware - eigene VM als Zentrales Buildsystem(server.warpzone.ms:2224) + Anleitung zum selber bauen
 - Key zum signieren der Firmware: Zentral über Buildserver (alternativ: dezierte entwickler bauen selber)
- Schlüsselverwaltung der Router - Anmeldung per Email (Später: Kontaktformular?)/ Verwalten im git / fastd checkt automatisch aus & konfiguriert
- Verwaltung von DNS-Einträgen für .ffms

Gateway

Anleitung zum Gateway einrichten, Vorlage ist die Dokumentation des FreiFunk [Lübeck](#)

- Debian Jessie (Kernel: 3.10.x neuere Kernel Version hat neueres Batman Support level - nicht kompatibel)
- **Zusätzliche Paketquellen**
- deb <http://repo.universe-factory.net/debian/> sid main

```
gpg --keyserver pgpkeys.mit.edu --recv-key 16EF3F64CB201D9C
gpg -a --export 16EF3F64CB201D9C | apt-key add -
```

- **Notwendige Pakete:**

- bridge-utils (Verwaltung der Netzwerkbrücken)
- batctl (B.A.T.M.A.N. Verwaltungstools)
- iptables-persistent (Initialisierung der Firewall beim Booten)
- openvpn (VPN zu MULLVAD)
- haveged (Entropie)
- fastd (VPN zu den Nodes)
- radvd (IPv6 Router Advertisements)
- isc-dhcp-server (DHCP)
- bind9 (DNS)
- git

- **IPv6 Forwarding aktivieren**

- Konfigurationsdatei /etc/sysctl.d/forwarding.conf

```
# IPv4 Forwarding
net.ipv4.ip_forward=1

# IPv6 Forwarding
net.ipv6.conf.all.forwarding = 1
```

- Anschließend Reboot des Servers

- **Batman Einrichten**

- Bei der /etc/modules das modulbatman-adv hinzufügen
- Neustarten oder Modul händisch per modprobe batman-adv laden
- Überprüfen ob module geladen wurde: es existiert der pfad /sys/module/batman_adv/ und dort gibt die Datei version die Kompatibilitätsversion von Batman an

- **Netzwerk anpassen**

- Eine Netzwerkbrücke als Schnittstelle zwischen dem Mesh auf der einen Seite und dem VPN nach XYZ als exist auf der anderen Seite dazu die /etc/network/interfaces anpassen
- Erstellen eines Bridge Interfaces das eine IP im FF-IP Block hat

```
# Netzwerkbrücke für Freifunk
# - Hier läuft der Traffic von den einzelnen Routern und dem
externen VPN zusammen
# - Unter der hier konfigurierten IP ist der Server selber im
Freifunk Netz erreichbar
# - bridge_ports none sorgt dafür, dass die brücke auch ohne
```

```
Interface erstellt wird
auto br0

iface br0 inet static
    address 10.43.0.3
    netmask 255.255.0.0
    bridge_ports none

iface br0 inet6 static
    address fd68:e2ea:a53::3
    netmask 48
```

- Batman Interface hinzufügen und an Bridge Interface binden

```
# Batman Interface
# - Erstellt das virtuelle Interface für das Batman-Modul und
bindet dieses an die Netzwerkbrücke
# - Die unten angelegte Routing-Tabelle wird später für das
Routing innerhalb von Freifunk (Router/VPN) verwendet

allow-hotplug bat0

iface bat0 inet6 manual
    pre-up modprobe batman-adv
    post-up ip link set dev bat0 up
    post-up brctl addif br0 bat0
    post-up batctl it 10000
    post-up ip rule add from all fwmark 0x1 table 42
```

- Config anwenden indem das Netzwerk per service networking restart neustartet wird
- Table 42 die wir im Bridge Interface definiert haben muss noch mit Regeln gefüllt werden.
- TODO: Konfiguration vom 2. Gateway übernehmen, da diese Lösung nicht mit neuen Debian Versionen kompatibel ist
- Dazu bearbeiten wir die /etc/iptables/rules.v4 und fügen

```
*filter # in wie weit ist das notwendig?
:INPUT ACCEPT [0:0]
:FORWARD ACCEPT [0:0]
:OUTPUT ACCEPT [0:0]
COMMIT
# Regeln zum markieren eingehender Pakete
*mangle
:PREROUTING ACCEPT [0:0]
:INPUT ACCEPT [0:0]
:FORWARD ACCEPT [0:0]
:OUTPUT ACCEPT [0:0]
:POSTROUTING ACCEPT [0:0]
-A PREROUTING -i br0 -j MARK --set-xmark 0x1/0xffffffff
-A OUTPUT -o eth0 -p udp --dport 53 -j MARK --set-xmark
0x1/0xffffffff
-A OUTPUT -o eth0 -p tcp --dport 53 -j MARK --set-xmark
```

```
0x1/0xffffffff
COMMIT
```

ein um alle Pakete die über die Bridge reinkommen mit dem 0x1 Flag zu markieren damit sie an Table 42 geschickt werden(d.h. nicht! die default route).

- Nun wird alles von der bridge an den VPN Tunnel(der später eingerichtet wird) per nat weiter geleitetet dafür fügt man ebenfalls in der /etc/iptables/rules.v4 folgendes ein:

```
# Route an VPN per nat.
*nat
:PREROUTING ACCEPT [0:0]
:INPUT ACCEPT [0:0]
:OUTPUT ACCEPT [0:0]
:POSTROUTING ACCEPT [0:0]
-A POSTROUTING -o tun0 -j MASQUERADE
COMMIT
```

- Iptables Neustarten über service iptables-persistent restart

- **VPN Einrichten**

- Unser VPN geht aktuell nach Schweden (Anbieter: <https://mullvad.net/en/>) dieser stellt passende openVPN Konfigurationsdateien zur Verfügung (siehe [hier](#) nach vorherigem einloggen)
- Die Konfigurationsdateien (ca.crt / crl.pem / mullvad.crt / mullvad.key / mullvad_linux.conf) werden nach /etc/openvpn/ kopiert und die datei mullvad_linux.conf wird ergänzt. Ganz am Ende wird

```
#custom
route-noexec
up /etc/openvpn/mullvad_up.sh
```

angefügt. Das route-noexec sorgt dafür das openvpn keine routen setzt.

- Nun wird das im vorherigen Punkt erwähnte Skript mullvad_up.sh mit folgendem Inhalt angelegt:

```
#!/bin/sh
ip route replace 0.0.0.0/1 via $5 table 42
ip route replace 128.0.0.0/1 via $5 table 42
exit 0
```

Das Skript liegt in der Routing Tabelle 42 fest das aller Verkehr der durch die Tabelle 42 geroutet wird an die IP des VPN Gateways(\$5) geroutet und setzt batman in den Server Mode. Das heißt das batman sich als Gateway versteht und das im Batman Mesh bekannt gibt.

- Openvpn wird per service openvpn start gestartet

- **Fastd Einrichten**

- Konfigurationsverzeichnis erstellen

```
mkdir -p /etc/fastd/vpn/peers
```

- Schlüssel für den Server erzeugen. Der Schlüssel wird in diesem Schritt nur erzeugt und auf der Kommandozeile ausgegeben. Secret und Public müssen in die Fastd-Konfiguration des Servers bzw. in die Router-Firmware übernommen werden.

```
fastd --generate-key
```

- Konfigurationsdatei /etc/fastd/vpn/fastd.conf erstellen:

```
bind 0.0.0.0:14242 interface "eth0";
interface "mesh-vpn";
user "nobody";
mode tap;
method "salsa2012+gmac";
mtu 1426; # 1426 - ipv4 header - fastd header
secret "SERVER-SECRET-KEY";

log to syslog level debug;
#folgende Zeile sorgt dafuer das jeder Peer akzeptiert wird
#on verify "true";

include peers from "/var/gateway-ffms/nodes/";

on up "
    ip link set dev $INTERFACE address de:ad:be:ef:43:0X
    ip link set dev $INTERFACE up
    ifup bat0
    batctl if add $INTERFACE
    batctl gw server
";
```

- Zum Testen kann fastd mit fastd -c /etc/fastd/vpn/fastd.conf in der Kommandozeile gestartet werden.
- Anschließend fastd mit service start fastd starten.
- **IPv6 Router Advertisements Einrichten**
- Jeder Gateway-Server erhält ein eigenes IPv6 Prefix für Router Anouncements
- Achtung: radvd kann maximal /64 Netze vergeben
- Der Eintrag RDNSS muss die IP-Adresse des Gateway enthalten
- Konfigurationsdatei /etc/radvd.conf

```
interface br0
{
    AdvSendAdvert on;
    IgnoreIfMissing on;
    MaxRtrAdvInterval 200;

    prefix fd68:e2ea:a53:zzzz::/64
    {
    };

    RDNSS fd68:e2ea:a53::z {
    };
```

```
};
```

- Start des Dienstes mit `service radvd restart`
- **DHCP Server Einrichten**
- Jeder Gateway-Server erhält einen Teil des IP Bereiches um Adressen zu vergeben
- Die Optionen Router und Domain-Name-Servers enthalten jeweils die IP des Servers
- Konfigurationsdatei `/etc/dhcp/dhcpd.conf`

```
default-lease-time 600;
max-lease-time 3600;

authoritative;

log-facility local7;

subnet 10.43.0.0 netmask 255.255.0.0 {
    range 10.43.zz.1 10.43.zz.254;

    option routers 10.43.0.x;
    option domain-name-servers 10.43.0.x;
}
```

- Zudem wird der DHCP Server noch auf das Bridge-Interface festgelegt. Hierzu wird in der Datei `/etc/default/isc-dhcp-server` die Ofolgende Option gesetzt:

```
# On what interfaces should the DHCP server (dhcpd) serve DHCP requests?
#           Separate multiple interfaces with spaces, e.g. "eth0 eth1".
INTERFACES="br0"
```

- Test des DHCP Servers mit `dhcpd -f -d`
- Anschließend DHCP Server starten mit `service isc-dhcp-server restart`

bind Einrichten

- Konfigurationsdatei `/etc/bind/named.conf.options`

```
options {
    directory "/var/cache/bind";

    // If there is a firewall between you and nameservers you want
    // to talk to, you may need to fix the firewall to allow multiple
    // ports to talk.  See http://www.kb.cert.org/vuls/id/800113

    // If your ISP provided one or more IP addresses for stable
    // nameservers, you probably want to use them as forwarders.
    // Uncomment the following block, and insert the addresses replacing
    // the all-0's placeholder.

    // forwarders {
    //     0.0.0.0;
```

```
// };

//=====
// If BIND logs error messages about the root key being expired,
// you will need to update your keys.  See
https://www.isc.org/bind-keys
//=====

dnssec-validation auto;
recursion yes;
allow-recursion { localnets; localhost; };

auth-nxdomain no;      # conform to RFC1035
listen-on-v6 { any; };

};

logging {
    category "default" { "debug"; };
    category "general" { "debug"; };
    category "database" { "debug"; };
    category "security" { "debug"; };
    category "config" { "debug"; };
    category "resolver" { "debug"; };
    category "xfer-in" { "debug"; };
    category "xfer-out" { "debug"; };
    category "notify" { "debug"; };
    category "client" { "debug"; };
    category "unmatched" { "debug"; };
    category "network" { "debug"; };
    category "update" { "debug"; };
    category "queries" { "debug"; };
    category "dispatch" { "debug"; };
    category "dnssec" { "debug"; };
    category "lame-servers" { "debug"; };
    channel "debug" {
        file "/tmp/nameddbg" versions 2 size 50m;
        print-time yes;
        print-category yes;
    };
};
```

- Konfigurationsdatei /etc/bind/named.conf.local

```
//  
// Do any local configuration here  
//  
  
// Consider adding the 1918 zones here, if they are not used in your  
// organization  
include "/etc/bind/zones.rfc1918";
```

```
zone "ffms" {
    type master;
    file "/var/gateway-ffms/dns/db.ffdms";
};
```

- Dienst starten mit `service bind9 restart`
- **Konfigurations-Git clonen**
- `cd /var`
- `git clone https://github.com/FreiFunkMuenster/gateway-ffms`
- **Gateway-Script**
 - Damit der Gateway seine Funktion aufnimmt und über batman als Gateway anerkannt wird (erst dann funktioniert Routing, DHCP, usw.) muss das Kommando `batctl gw server` ausgeführt werden.
 - Hierfür sollte idealerweise ein Gateway-Überwachungsscript erstellt werden.

NanoStation Loco M5

Die Original Firmware für die NanoStation Loco M5 findet sich bei [Ubiquiti](#) das Updaten funktioniert über TFTP wie im [Openwrt Wiki](#) beschrieben. Es Funktionierte nur mit Firmware Version „XM-v5.5.8.build20991.bin“ und nicht mit der Version 5.5.9. (Platform: airMAX ISP Solutions / Model: NanoStation M5)

API File

Das API File liegt im `/var/www/html/` des Buildservices `ssh root@warpzone.ms -p 2224` bzw `scp -P 2224 FreifunkMuenster-api.json root@warpzone.ms:/var/www/html/` dadurch wird es automatisch unter [erreichbar](https://www.warpzone.ms/freifunk/). Die Datei heißt `FreifunkMuenster-api.json` der gesamte Pfad zur API Datei lautet also <https://www.warpzone.ms/freifunk/FreifunkMuenster-api.json>

Knotenanzahl Automatisch Updaten lassen

TODO: <http://luebeck.freifunk.net/wiki/Netzwerk:Skripte>

From:
<http://wiki.warpzone.ms/> - **warpzone**

Permanent link:
<http://wiki.warpzone.ms/infrastruktur:freifunkmuenster?rev=1405462213>

Last update: **01.03.2017**

