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)
  - ravd (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 neue Kernel nutzen die neue batman version mit neuerem kompability 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 DNSeinträgen 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 -
```

- Dann brauchen wir noch die Paketquellen aus unserem OBS

```
echo 'deb
http://download.opensuse.org/repositories/home:/fusselkater:/ffms/Debian_7.0
/ /' >> /etc/apt/sources.list.d/ffms.list
```

```
wget
http://download.opensuse.org/repositories/home:fusselkater:ffms/Debian_7.0/R
elease.key
apt-key add - < Release.key
```

- Dann einmal apt-get update
- **Notwendige Pakete:**
  - bridge-utils (Verwaltung der Netzwerkbrücken)
  - batctl (B.A.T.M.A.N. Verwaltungstools)
  - openvpn (VPN zu MULLVAD)
  - haveged (Entropie)
  - fastd (VPN zu den Nodes)
  - radvd (IPv6 Router Advertisements)
  - isc-dhcp-server (DHCP)
  - bind9 (DNS)
  - git
  - alfred-stable
  - batman-adv-dkms
- **batman-adv 2013.04 installieren**

Nachdem batman-adv-dkms installiert ist, schmeißt dkms einen Fehler, da in den aktuellen Kernel-Versionen eine neuere batman-adv Version ist, als über dkms gebaut wurde. Es ist nötig, die Installation des 2013.04er Modules zu erzwingen. Dazu macht man folgendes:

```
dkms remove batman-adv/2013.4.0 --all
dkms --force install batman-adv/2013.4.0
```

Danach einmal rebooten. Nun sollte batman-adv Version 2013.4 geladen sein. Verifizieren kann man dies mit:

```
cat /sys/module/batman_adv/version
```

- **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 modul`batman-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 Kompabilitätsversion 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

    ```
    # Netwerkbrücke 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 Inteface 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
        post-up start-stop-daemon -b --start --exec
/usr/sbin/alfred -- -i br0 -b bat0;
        post-up start-stop-daemon -b --start --exec
/usr/sbin/batadv-vis -- -i bat0 -s;
```

- 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.
- Sobald das Interface hoch kommt, starten wir alfred und batadv-vis
- TODO: Konfiguration vom 2. Gateway übernehmen, da diese Lösung nicht mit neuren Debian Versionen kompatibel ist
- Dazu erzeugen wir die `/etc/iptables.up.rules` 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). Zusätzlich werden Ausgehende Pakete, die eigentlich über eth0 gehen würde, und UDP/TCP Port 53 (DNS) haben, auch über Table 42 geschickt, um bind zu zwingen, über das VPN zu gehen.

- 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.up.rules` 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
```

- Nun erzeugen wir ein Shell-Script, das beim initialisieren eines Interfaces die IpTables-Regeln läd

- Datei: /etc/network/if-pre-up.d/iptables

```
#!/bin/sh
/sbin/iptables-restore < /etc/iptables.up.rules
```

- Und machen sie ausführbar:

```
chmod +x /etc/network/if-pre-up.d/iptables
```

- Iptables laden mit iptables-restore < /etc/iptables.up.rules
- **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.

- tun in die /etc/modules eintragen und dann modprobe tun
- 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 akzeptiet wird
#on verify "true";

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

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.
- Die backbones werden included, um die Verbindung zwischen den Servern herzustellen
- **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 50;

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

    RDNSS fd68:e2ea:a53::z {
      AdvRDNSSLifetime 100;
    };

    DNSSL ffms
    {
      AdvDNSSLLifetime 100;
    };

};
```

- 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`

dhcpdv6

- Damit auch via dhcpdv6 Adressen verteilt werden (Was vorallem für Windows-Hosts wichtig ist, da diese den RDNSS-Parameter von radvd nicht kennen, werden folgende Dateien eingerichtet:

/etc/init.d/isc-dhcp6-server

```
#!/bin/sh
#
#

### BEGIN INIT INFO
# Provides:          isc-dhcp6-server
# Required-Start:    $remote_fs $network $syslog
# Required-Stop:     $remote_fs $network $syslog
# Should-Start:      $local_fs slapd $named
# Should-Stop:       $local_fs slapd
# Default-Start:     2 3 4 5
# Default-Stop:      0 1 6
# Short-Description: DHCP6 server
```

```
# Description:       Dynamic Host Configuration Protocol Server V6
### END INIT INFO

PATH=/sbin:/bin:/usr/sbin:/usr/bin

test -f /usr/sbin/dhcpd || exit 0

DHCPD_DEFAULT="${DHCPD_DEFAULT:-/etc/default/isc-dhcp6-server}"

# It is not safe to start if we don't have a default configuration...
if [ ! -f "$DHCPD_DEFAULT" ]; then
        echo "$DHCPD_DEFAULT does not exist! - Aborting..."
        if [ "$DHCPD_DEFAULT" = "/etc/default/isc-dhcp6-server" ]; then
                echo "Run 'dpkg-reconfigure isc-dhcp-server' to fix the
problem."
        fi
        exit 0
fi

. /lib/lsb/init-functions

# Read init script configuration
[ -f "$DHCPD_DEFAULT" ] && . "$DHCPD_DEFAULT"

NAME=dhcpd
DESC="ISC DHCP6 server"
# fallback to default config file
DHCPD_CONF=${DHCPD_CONF:-/etc/dhcp/dhcpd6.conf}
# try to read pid file name from config file, with fallback to
/var/run/dhcpd.pid
if [ -z "$DHCPD_PID" ]; then
        DHCPD_PID=$(sed -n -e 's/^[ \t]*pid-file-name[ \t]*"(.*)"[
\t]*;.*$/\1/p' < "$DHCPD_CONF" 2>/dev/null | head -n 1)
fi
DHCPD_PID="${DHCPD_PID:-/var/run/dhcpd6.pid}"

test_config()
{
        if ! /usr/sbin/dhcpd -6 -t $OPTIONS -q -cf "$DHCPD_CONF" > /dev/null
2>&1; then
                echo "dhcpd self-test failed. Please fix $DHCPD_CONF."
                echo "The error was: "
                /usr/sbin/dhcpd -6 -t $OPTIONS -cf "$DHCPD_CONF"
                exit 1
        fi
}

# single arg is -v for messages, -q for none
check_status()
{
    if [ ! -r "$DHCPD_PID" ]; then
```

```
        test "$1" != -v || echo "$NAME is not running."
        return 3
    fi
    if read pid < "$DHCPD_PID" && ps -p "$pid" > /dev/null 2>&1; then
        test "$1" != -v || echo "$NAME is running."
        return 0
    else
        test "$1" != -v || echo "$NAME is not running but $DHCPD_PID
exists."
        return 1
    fi
}

case "$1" in
        start)
                test_config
                log_daemon_msg "Starting $DESC" "$NAME"
                start-stop-daemon --start --quiet --pidfile "$DHCPD_PID" \
                        --exec /usr/sbin/dhcpd -- \
                        -6 -q $OPTIONS -cf "$DHCPD_CONF" -pf "$DHCPD_PID"
$INTERFACES
                sleep 2

                if check_status -q; then
                        log_end_msg 0
                else
                        log_failure_msg "check syslog for diagnostics."
                        log_end_msg 1
                        exit 1
                fi
                ;;
        stop)
                log_daemon_msg "Stopping $DESC" "$NAME"
                start-stop-daemon --stop --quiet --pidfile "$DHCPD_PID"
                log_end_msg $?
                rm -f "$DHCPD_PID"
                ;;
        restart | force-reload)
                test_config
                $0 stop
                sleep 2
                $0 start
                if [ "$?" != "0" ]; then
                        exit 1
                fi
                ;;
        status)
                echo -n "Status of $DESC: "
                check_status -v
                exit "$?"
                ;;
```

```
        *)
                echo "Usage: $0 {start|stop|restart|force-reload|status}"
                exit 1
esac

exit 0
```

- Danach chmod +x /etc/init.d/isc-dhcp6-server
- Und update-rc.d isc-dhcp6-server defaults

/etc/default/isc-dhcp6-server

```
# Defaults for isc-dhcp-server initscript
# sourced by /etc/init.d/isc-dhcp-server
# installed at /etc/default/isc-dhcp-server by the maintainer scripts

#
# This is a POSIX shell fragment
#

# Path to dhcpd's config file (default: /etc/dhcp/dhcpd6.conf).
#DHCPD_CONF=/etc/dhcp/dhcpd6.conf

# Path to dhcpd's PID file (default: /var/run/dhcpd6.pid).
#DHCPD_PID=/var/run/dhcpd6.pid

# Additional options to start dhcpd with.
#       Don't use options -cf or -pf here; use DHCPD_CONF/ DHCPD_PID instead
#OPTIONS=""

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

- Danach touch /var/lib/dhcp/dhcpd6.leases

/etc/dhcp/dhcpd6.conf

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

authoritative;

log-facility local7;

subnet6 fd68:e2ea:a53::/48 {
    range6 fd68:e2ea:a53:zzzz::1 fd68:e2ea:a53:zzzz:ffff:ffff:ffff:ffff;
    option dhcp6.name-servers fd68:e2ea:a53::4;
    option dhcp6.domain-search "ffms";
}
```

- Nun service isc-dhcp6-server start

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.ffms";
};
```

- 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` ausgeführtwerden.
  - 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 https://www.warpzone.ms/freifunk/ erreichbar. 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

WAN-Mesh auf Knoten aktualisieren

Um Wan-Mesh auf einem Knoten zu verwenden müssen einmalig die folgenden UCI-Einträge über SSH gesetzt werden. Anschließend kann das WAN-Mesh über den Expoertenmodus (Schnittstelle) ein und aus geschaltet werden.

```
uci set network.mesh_wan=interface
uci set network.mesh_wan.ifname=br-wan
uci set network.mesh_wan.proto=batadv
uci set network.mesh_wan.mesh=bat0
uci set network.mesh_wan.auto=1
uci commit
```