Project SimpleRouter

Because routers can be open source and secure

By Thor Grotle 2023

Webpage: www.keroit.dk

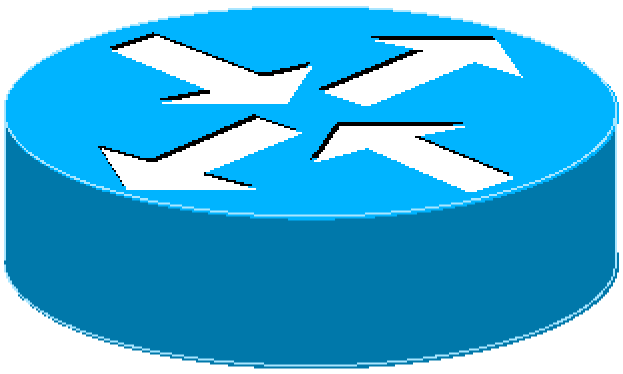


Table of Contents

Purpose

Most consumer routers suck! They are either limited in features, unstable/need reboot often, expensive or otherwise locked down.
Welcome to project SimpleRouter: The router that is based on open source, freely and configurable and is update-able. This will require knowledge on how to use command line.

Features

• -> Serving Internet from wifi/ethernet/3G/4G/5G to wifi/ethernet
• -> Always updatable, as this is a linux router stack, based on open source tools.
• -> DHCP, DNS & NTP server
• -> Default IPv4 but can be set to IPv6
• -> Optional: Cockpit Web interface
• -> Optional: Wireguard VPN - Server / Client

Additional services will be running in lxc/docker containers so if breached, they will be limited to that service.

This is a mini server that works in relative confined resources with CPU power/ram, but can be scaled to a fulltime beefy Xeon/Epyc server if needed.

Target

Me! And others interested in a small and powerful configurable core router with full control

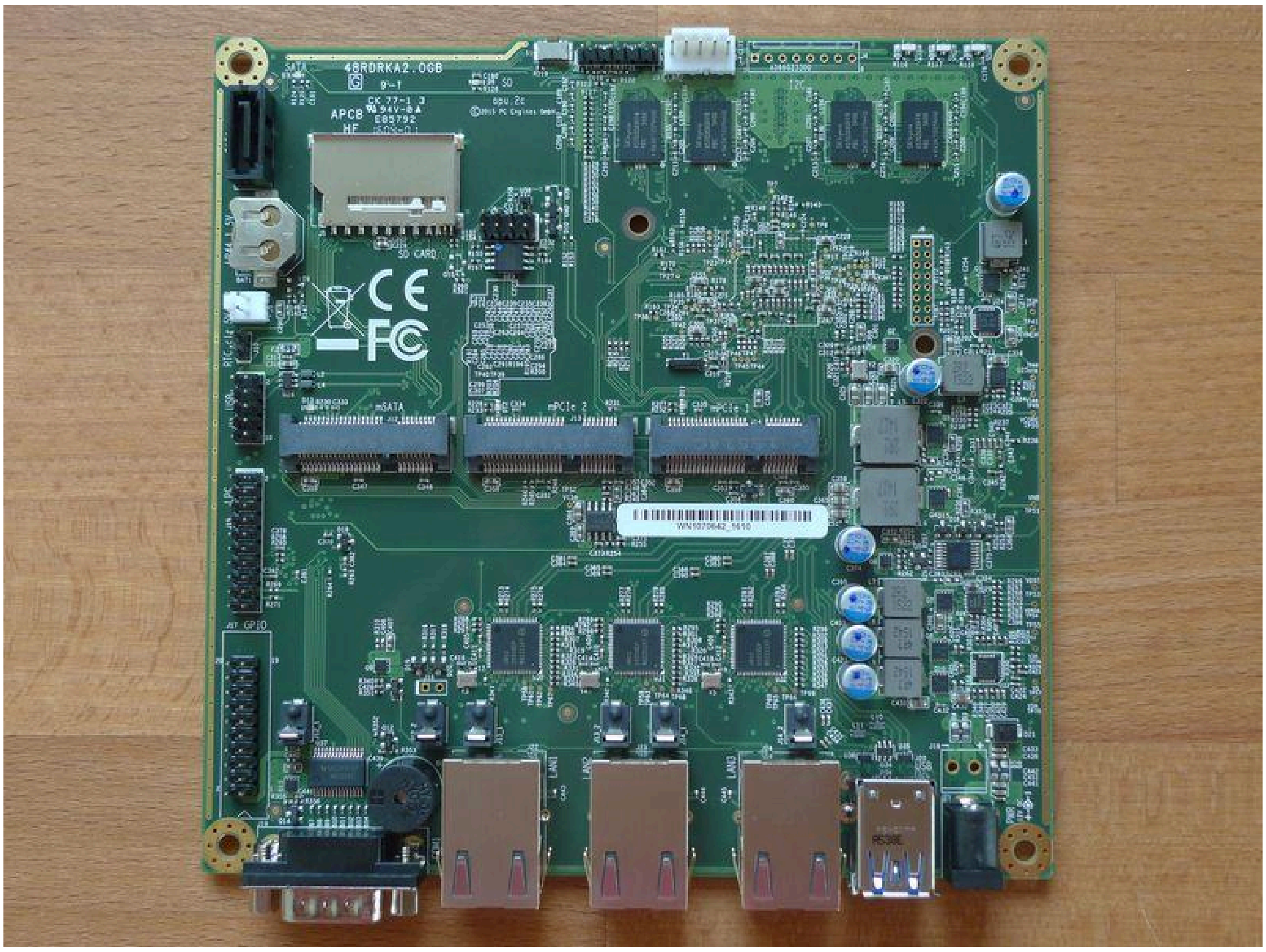
Who

Thor Miller Grotle
thor@keroit.dk
https://www.keroit.dk

Hardware/Technologies:

PC Engine APU boards

Aluminium Chassis - Customized
Mainboard PCEngines APU4B4 - Quad Core AMD Jaguar 1GHz x86/x64 - 4GB ddr3 RAM - 4xi211AT NIC
Mini PCI express WL900 wireless AC
Mini PCI express Msata 128 GB SSD - Salvage hardware
Mini PCI express 3COM 3G Modem - Salvage hardware
RS232 to USB serial link
3/6 Antennas
Power/Reset buttons



Software/Technologies:

List of technology

Serial Link / RS232 and Screen terminal emulation
Ubuntu Server 22.04
Coreboot
Netplan
Firewalld
Kea DNS Server / ISC-dhcp-server
BIND DNS Server
CronyD - Timeserver
Cockpit - Webinterface
OpenSSH - Remote access via command command line
Wireguard - Fast, simple and secure VPN server/client

Basics

The magic of the router is the Linux kernel and with routing and port control is handled by the firewalld and nftables.

Network is set up into 2 bridge interfaces: internet0, lan0
VLAN active: 76 Guest, 77 IoT, 78 NoNet
Firewalld is setup into 4 zones: public, home & IoT, and public
Kea DHCP: assigns IP addresses
Bind9: Acts as local DNS server
Optional: HostAPD: Manages the Wifi hotspot and is set to be in the home zone
Optional: Cockpit: Act as management web interface for this setup.

You add the network interfaces you want to act as Internet input into the bridge: internet0
You add the network interfaces you want as local lan to the bridge: lan0

You can add port/service rules to each of the 3 zones, either by editing the config files, or you can use the command: firewall-cmd.

OS-Choice

Current: Ubuntu 22.04 LTS (End Of Life date: 2027, extended support via ESM: 2032)

Pro: Quick to install, newer software, easy to use. Short install processed. Will be updatable to newer version

Con: Some pre-installed services that need removal.

Deployment Method

Currently - > Manually install and config
Future -> Ansible deployment
Future -> SD/USB Image

Base - Ubuntu Install

Ubuntu 22.04
Download the ubuntu server iso
Write that to USB key
Connect Console cable and listen at speed 115200 (**sudo screen /dev/ttyusb0 115200n8**)
    To exit screen press: CTRL+"a" followed by "d"
Press F10 and boot from USB
When grub menu appears press "E" on Install option, scroll down to initrd and add the following

```
console=ttyS0,115200n8
```

This will add serial enabled on boot. It will automatically be added to installation, so when rebooting after install you can still access with serial cable

As SSH Daemon is activated during the Ubuntu installation process, it is highly recommended to add Trusted SSH-keys from your github repo and allow those to access your main user.

After install, run **ip addr** to see interfaces like NIC and wlan + MAC addresses
Update system

```
sudo apt update && apt upgrade -y
```

Set timezone

Remember to set correct timezone:

```
sudo timedatectl set-timezone Europe/Copenhagen
```

Packages to install

```
sudo apt install bridge-utils wpasupplicant iw firewalld nano isc-dhcp-server bind9 screenfetch flashrom
hostapd mtools  bwm-ng cockpit
```

Packages to remove

```
sudo apt purge network-manager  ufw
```

Bios update/Flash bios

Packages:

```
sudo apt install flashrom mtools
```

Check downloaded rom checksum from: https://pcengines.github.io/

```
md5sum apu4_v4.8.0.4.rom
cat apu4_v4.8.0.4.rom.md5
```

If checksum is the consistent, go ahead and install new rom

Take a backup of system rom

```
flashrom -p internal -r apu.old.rom
```

Write new rom to bios

```
flashrom -p internal -w apu4_v4.8.0.4.rom
```

Verify that rom version is correctly installed

```
flashrom -p internal -v apu4_v4.8.0.4.rom
```

Base Network Setup

There are multiple ways of network setup, netplan, ifupdown, NetworkManager
In Ubuntu Server Netplan is the default, but you may chose otherwise.
If you want to control network and interface via Cockpit, follow the guide for NetworkManager.

Vlan

If you plan on using VLANS, vlan package needs to be installed, and the module needs to be activated.
Install package with APT

```
sudo apt install vlan
```

Follow the steps to enable kernel module here [LINK]

Network Setup (Netplan)

In the newer Ubuntu server environment Netplan is used, a sample netplan can be seen below for enabling NICS and assigning NIC to bridges, then adding VLAN on top of the bridges.
Copy in the config file and activate the new network
Once config is install apply with command:

```
sudo netplan –apply
```

/etc/netplan/00-installer-config.yaml

```
 version: 2
 ethernets:
   enp3s0:
     dhcp4: no
   enx00249b5100e0:
     dhcp4: true
 bridges:
   br0:
     interfaces: [enp3s0]
     dhcp4: no
 vlans:
   vlan25:
     id: 25
     link: br0
     addresses: [10.0.25.1/24]
   vlan26:
     id: 26
     link: br0
     addresses: [10.0.26.1/24]
```

Network Setup (ifupdown)

Ubuntu + Debian

```
sudo mkdir -p /etc/network
```

If running Ubuntu, we want to strip the system of netplan and network-manager as this will interfere with this setup.

```
sudo apt install ifupdown
sudo apt purge netplan.io network-manager ufw
sudo rm -rf /etc/netplan/*.yml
```

The /etc/network/interfaces file defines the physical network devices

All interface, except the wireless network should be declared in the interface file, like this:

```
allow-hotplug enp1s0
auto enp1s0
iface enp1s0 inet manual
```

You add the interface to the bridge like this:

```
auto internet0
iface internet0 inet dhcp
  bridge_ports enp1s0
  up /usr/sbin/brctl stp internet0 on
```

Here the first network port: enp1s0 is defined as my internet source.

```
auto internet0
iface internet0 inet dhcp
  bridge_ports enx928c43bdb8be
  up /usr/sbin/brctl stp internet0 on
```

Here my connected iPhone is defined as my internet source, the system recognized the phone as interface enx928c43bdb8be

/etc/network/interfaces

```
## template for adding to interfaces to bridge
# This matches my PC APU4B4


allow-hotplug enp1s0
auto enp1s0
iface enp1s0 inet manual

allow-hotplug enp2s0
auto enp2s0
iface enp2s0 inet manual

allow-hotplug enp3s0
auto enp3s0
```

```
iface enp3s0 inet manual

allow-hotplug enp4s0
auto enp4s0
iface enp4s0 inet manual

# My iPhone
#allow-hotplug enx928c43bdb8be
#auto enx928c43bdb8be
#iface enx928c43bdb8be inet manual

auto internet0
iface internet0 inet dhcp
   bridge_ports enp1s0
#   bridge_ports enx928c43bdb8be
   up /usr/sbin/brctl stp internet0 on

auto lan0
iface lan0 inet static
   address 10.0.75.1/24
   gateway 10.0.75.1
   bridge_ports enp2s0 enp3s0 enp4s0
   up /usr/sbin/brctl stp lan0 on
```

Network Setup (Network Manager)

Start by installing NetworkManager:
```
sudo apt install network-manager
```

If you have opted to user Cockpit Web interface install this package:
```
sudo apt install cockpit-networkmanager
```

You will find config files etc/NetworkManager/system-connections

You can configure via gui: **Cockpit** or terminal command tool **nmtui**
You can also configure via terminal command: **nmcli**

Examples of nmcli

Create connection:
```
nmcli con add type ethernet ifname enp5s0 con-name "Wired connection 1"
```

Connection status and bringing interface up or down:
```
nmcli con show
nmcli con down 'Wired connection 1'
nmcli con up 'Wired connection 1''
```

Setting dhcp on connection:
```
nmcli con modify 'Wired connection 1'  ifname enp5s0 ipv4.method auto
```

Setting static IP on connection:

```
nmcli con modify 'Wired connection 1' ifname enp6s0 ipv4.method manual ipv4.addresses
10.0.75.15/24 gw4 10.0.75.1  ipv4.dns 8.8.8.8 connection.autoconnect-priority 10
```

Services

Firewall D: Firewall and its magic

This is where the internet routing happens (technically nftables)

Enable system wide +vlan routing kernel features and disable IPv6

```
sudo echo "net.ipv4.ip_forward = 1" >> /etc/sysctl.conf
sudo echo “net.ipv6.conf.all.disable_ipv6 = 1”  >> /etc/sysctl.conf
sudo echo “net.ipv6.conf.default.disable_ipv6 = 1”   >> /etc/sysctl.conf
sudo echo "8021q" >> /etc/modules # This will enable VLAN
```

Install Firewalld

```
sudo apt install firewalld
sudo systemctl enable firewalld
sudo systemctl start firewalld
```

Copy in the configuration files:

/etc/firewalld/policies/internet-access.xml
/etc/firewalld/firewalld.conf
/etc/firewalld/zones/public.xml
/etc/firewalld/zones/home.xml
/etc/firewalld/direct.xml (old version)

Reload Firewalld configuration

```
sudo firewall-cmd reload
```

Add service / port to zone

```
sudo firewall-cmd --zone=home --add-service=https
sudo firewall-cmd --zone=home --add-port=tcp:80
```

Forward port to IP behind firewall

```
sudo firewall-cmd --zone=public --add-forward-port=port=22:proto=tcp:toport=3753
:toaddr=10.0.75.240
```

For setting to be persistent add the --permanent:

```
sudo firewall-cmd --zone=home --add-service=https --permanent
```

To remove a service from the firewall:

```
sudo firewall-cmd --zone=home --remove-service=https --permanent
```

Add interface to zone:

```
sudo firewall-cmd --zone=public --change-interface=internet0
```

List rules for zone:

```
sudo firewall-cmd --zone=public --list-all
```

/etc/firewalld/firewalld.conf

```
DefaultZone=public
CleanupOnExit=yes
CleanupModulesOnExit=no
Lockdown=no
IPv6_rpfilter=no
IndividualCalls=no
LogDenied=off
FirewallBackend=nftables
FlushAllOnReload=yes
RFC3964_IPv4=yes
```

/etc/firewalld/zones/home.xml

```
<?xml version="1.0" encoding="utf-8"?>
<zone>
  <short>Home</short>
  <description>For use in home areas. You mostly trust the other computers on networks to not harm your computer. Only selected incoming connections are accepted.</description>
  <service name="ssh"/>
  <service name="mdns"/>
  <service name="dns"/>
  <service name="cockpit"/>
  <service name="http"/>
  <service name="tftp"/>
  <service name="dhcp"/>
  <service name="ntp"/>
  <service name="wireguard"/>
  <interface name="lan0"/>
  <forward/>
</zone>
```

/etc/firewalld/zones/work.xml

```
<?xml version="1.0" encoding="utf-8"?>
<zone>
  <short>Work</short>
  <description>For use in work areas. You mostly trust the other computers on networks to not harm your computer. Only selected incoming connections are accepted.</description>
  <service name="dhcp"/>
  <interface name="lan0.76"/>
  <forward/>
</zone>
```

/etc/firewalld/zones/iot.xml

```
<?xml version="1.0" encoding="utf-8"?>
<zone>
  <short>IoT</short>
  <description>Network for IoT units</description>
  <service name="dhcp"/>
  <interface name="lan0.77"/>
  <forward/>
</zone>
```

/etc/firewalld/zones/public.xml

```
<?xml version="1.0" encoding="utf-8"?>
<zone>
  <short>Public</short>
  <description>For use in public areas. You do not trust the other computers on networks to not
harm your computer. Only selected incoming connections are accepted.</description>
  <service name="wireguard"/>
  <masquerade/>
  <forward-port port="32400" protocol="tcp" to-port="32400" to-addr="10.0.75.62"/>
  <forward-port port="32400" protocol="udp" to-port="32400" to-addr="10.0.75.62"/>
  <forward-port port="32410" protocol="tcp" to-port="32410" to-addr="10.0.75.62"/>
  <forward-port port="32412" protocol="tcp" to-port="32412" to-addr="10.0.75.62"/>
  <forward-port port="32412" protocol="udp" to-port="32412" to-addr="10.0.75.62"/>
  <forward-port port="32414" protocol="udp" to-port="32414" to-addr="10.0.75.62"/>
  <forward-port port="32410" protocol="udp" to-port="32410" to-addr="10.0.75.62"/>
  <forward-port port="1900" protocol="udp" to-port="1900" to-addr="10.0.75.62"/>
  <forward-port port="32469" protocol="tcp" to-port="32469" to-addr="10.0.75.62"/>
  <forward-port port="3005" protocol="tcp" to-port="3005" to-addr="10.0.75.62"/>
  <forward-port port="8324" protocol="tcp" to-port="8324" to-addr="10.0.75.62"/>
  <interface name="enp1s0"/>
  <interface name="internet0"/>
```

/etc/firewalld/policies/internet-access.xml

```
<?xml version="1.0" encoding="utf-8"?>
<policy target="ACCEPT">
  <rule>
          <tcp-mss-clamp value="pmtu"/>
  </rule>
  <ingress-zone name="work"/>
  <ingress-zone name="home"/>
  <egress-zone name="public"/>
</policy>
</zone>
```

/etc/firewalld/policies/iot-access.xml

```
<?xml version="1.0" encoding="utf-8"?>
<policy target="ACCEPT">
  <ingress-zone name="home"/>
  <egress-zone name=”iot"/>
</policy>
</zone>
```

Bind9 DNS server

Bind9 is the primary DNS server of choice. It will allow DDNS updates from ISC-DHCP-Server, which will make the network hosts be pingable by local domain and host names.
Later on, I will add a DNS blocklist for ad-blocking

File permissions In /etc/bind/

```
-rw-r--r-- 1 root root 2403 May 17  2022 bind.keys
-rw-r--r-- 1 root root  237 Aug 25  2020 db.0
-rw-r--r-- 1 root root  271 Aug 25  2020 db.127
-rw-r--r-- 1 root root  237 Aug 25  2020 db.255
-rw-r--r-- 1 root root  353 Aug 25  2020 db.empty
-rw-r--r-- 1 bind bind 1582 Jun 22 04:33 db.lan
-rw-r--r-- 1 root root  270 Aug 28  2022 db.local
-rw-r--r-- 1 root bind  510 Dec  4  2022 named.conf
-rw-r--r-- 1 root bind  585 Mar 31 21:40 named.conf.default-zones
-rw-r--r-- 1 root bind  585 Mar 22 18:05 named.conf.local
-rw-r--r-- 1 root bind  864 Mar 22 17:58 named.conf.options
-rw-r----- 1 bind bind  100 Jul 14  2022 rndc.key
-rw-r--r-- 1 root bind  404 Mar 21 20:58 secret.lan
drw-r--r-- 2 root root   20 Aug 28  2022 zones
-rw-r--r-- 1 root root 1317 Aug 25  2020 zones.rfc1918
```

/etc/bind/named.conf

```
#acl internals {127.0.0.0/8; 10.0.75.0/24; };
include "/etc/bind/named.conf.options";
include "/etc/bind/named.conf.local";
include "/etc/bind/named.conf.default-zones";
```

/etc/bind/named.conf.options

```
options {
        directory "/var/cache/bind";
         forwarders {
        8.8.8.8;
         };
        dnssec-validation auto;
        listen-on-v6 { none; };
};
```

/etc/bind/named.conf.local

```
//include "/etc/bind/zones.rfc1918";
include "/etc/bind/rndc.key";
zone "keroit.lan" {
allow-query { 127.0.0.0/24; 10.0.75.0/24; 10.0.80.0/24; 10.0.76.0/24; };
type master;
file "/etc/bind/db.lan";
allow-update { key rndc-key; 127.0.0.1; };
};
```

/etc/bind/db.lan

```
$TTL 2d          ; default TTL for zone
$ORIGIN keroit.lan. ; base domain-name
; Start of Authority RR defining the key characteristics of the zone (domain)
@        IN       SOA   router.keroit.lan. root.keroit.lan. (
                         2003080800 ; serial number
                         12h      ; refresh
                         15m      ; update retry
                         3w       ; expiry
                         2h       ; minimum
                         )
; name server RR for the domain
         IN       NS     router.keroit.lan.
router            A      10.0.75.1
darkserver        A      10.0.75.149
```

Kea DHCP Server

Here will be documentation on how to setup Kea DHCP

Link: https://www.isc.org/kea/

ISC-DHCP-Server (OLD)

ISC DHCP as of the end of 2022 - Will now investigate and replaced with Kea DHCP

File permissions: /etc/dhcp

```
drwxr-x--- 2 root dhcpd    6 Jul 14  2022 ddns-keys
-rw-r--r-- 1 root root  1426 Aug  9  2021 debug
-rw-r--r-- 1 root root  1753 Dec  4  2022 dhclient.conf
drwxr-xr-x 2 root root    41 Feb 28 06:31 dhclient-enter-hooks.d
drwxr-xr-x 2 root root   116 Feb 28 06:31 dhclient-exit-hooks.d
-rw-r--r-- 1 root root  3331 Jun 21  2022 dhcpd6.conf
-rw-r--r-- 1 root root  2485 Mar 22 18:07 dhcpd.conf
-rw-r--r-- 1 root root  2764 Jan 16 08:56 dhcpd.conf.save
-rw------- 1 root root   100 Jul 14  2022 isc-dhcp-server-rndc.key
```

Packages

Ubuntu:

```
sudo apt install isc-dhcp-server
```

Copy the configuration file:
/etc/dhcp/dhcpd.conf

Start and enable the service:

```
sudo systemctl start isc-dhcp-server
sudo systemctl enable isc-dhcp-server
```

Fedora

```
sudo apt install dhcp-server
```

Copy the configuration file:

```
/etc/dhcp/dhcpd.conf
```

Start and enable the service:

```
sudo systemctl start dhcpd
sudo systemctl enable dhcpd
```

Show Current DHCP lease list

```
dhcp-lease-list --parsable
```

/etc/dhcp/dhcpd.conf

```
authoritative;
default-lease-time 1440;
max-lease-time 10080;
one-lease-per-client true;
allow booting;


# Needed for bind9 DNS
ddns-updates on;
ddns-update-style interim;
ddns-domainname "home.itso.dk";
update-static-leases on;
use-host-decl-names on;
option domain-name "home.itso.dk";


include "/etc/dhcp/isc-dhcp-server-rndc.key";
zone home.itso.dk. {
        primary 127.0.0.1;
        key rndc-key;
}

# SUBNET CONF - LAN
subnet 10.0.75.0 netmask 255.255.255.0{
  range 10.0.75.11 10.0.75.200;
  option domain-name-servers 10.0.75.1;
  option domain-name "home.itso.dk";
  option routers 10.0.75.1;
  option subnet-mask 255.255.255.0;
  filename "pxelinux.0";
  next-server 10.0.75.1;
#  option bootfile-name "pxelinux/pxelinux.0";
}



 # Guest vlan 76
subnet 10.0.76.0 netmask 255.255.255.0{
  range 10.0.76.11 10.0.76.200;
  option domain-name-servers 8.8.8.8;
  option routers 10.0.76.1;
  option subnet-mask 255.255.255.0;
}

 # IoT vlan 77
subnet 10.0.77.0 netmask 255.255.255.0{
  range 10.0.77.11 10.0.77.200;
  option domain-name-servers 8.8.8.8;
  option routers 10.0.77.1;
  option subnet-mask 255.255.255.0;
}

# Defined HOSTS

#host darkwulf2004 {
#  hardware ethernet e0:d5:5e:6f:62:eb;
#  fixed-address 10.0.75.10;
#}

#host ikeahub {
#  hardware ethernet 8c:45:00:69:e4:29;
#  fixed-address 10.0.75.20;
#}

#host Tadohub {
#  hardware ethernet ec:e5:12:15:93:9a;
#  fixed-address 10.0.75.21;
#}

#host HP2700 {
#  hardware ethernet 14:cb:19:51:17:b3;
#  fixed-address 10.0.75.50;
#}


#host thor-iPhone {
#  hardware ethernet b4:1b:b0:eb:b3:24;
#  fixed-address 10.0.75.9;
#}

#host NANCY-iPhoneSE2020 {
#  hardware ethernet a6:23:03:40:0c:b0;
#  fixed-address 10.0.75.11;
#}

#host DarkPlex {
#  hardware ethernet b2:5a:da:79:37:29;
#  fixed-address 10.0.75.62;
#}

#host Smokedetector {
#  hardware ethernet a4:e5:7c:ad:7b:73;
#  fixed-address 10.0.75.59;
#}
```

/etc/default/isc-dhcp-server

```
# Defaults for isc-dhcp-server (sourced by /etc/init.d/isc-dhcp-server)

# Path to dhcpd's config file (default: /etc/dhcp/dhcpd.conf).
DHCPDv4_CONF=/etc/dhcp/dhcpd.conf
#DHCPDv6_CONF=/etc/dhcp/dhcpd6.conf

# Path to dhcpd's PID file (default: /var/run/dhcpd.pid).
#DHCPDv4_PID=/var/run/dhcpd.pid
#DHCPDv6_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".
INTERFACESv4="lan0 lan0.76"
#INTERFACESv6=""
```

Cockpit: Web Interface for router

Package:

```
sudo apt install cockpit
sudo systemctl start cockpit
sudo systemctl enable cockpit
```

Access by localhost using webbrowser https://10.0.75.1:9090

Copy in the configuration files:

/etc/cockpit/cockpit.conf

/etc/cockpit/cockpit.conf

```
[WebService]
Origins = https://10.0.75.1:9090
```

ChronyD Time Server

```
sudo apt install chrony
firewall-cmd --zone=home --add-service=ntp --permanent && firewall-cmd --zone=home --add-service=ntp
nano /etc/chrony.conf # Add content from below
systemctl enable --now chronyd
```

/etc/chrony.conf

```
pool 8.8.8.8 iburst
allow 0.0.0.0/0
sourcedir /run/chrony-dhcp
driftfile /var/lib/chrony/drift
makestep 1.0 3
rtcsync
keyfile /etc/chrony.keys
ntsdumpdir /var/lib/chrony
leapsectz right/UTC
logdir /var/log/chrony
```

Wireguard VPN

For a quick and dirty selfhosted VPN, you can use Wireguard. In this example we bind this to the **lan0** Interface, meaning we give Wireguard clients access to our internal network. If wanted, change the wireguard output to **internet0** interface instead

Wireguard Setup LINK

https://www.digitalocean.com/community/tutorials/how-to-set-up-wireguard-on-ubuntu-20-04

Wireguard install

sudo apt install wireguard qrencode
sudo firewall-cmd –zone=public –add-port=51820/udp && sudo firewall-cmd –zone=public –add-port=51820/udp –permanent

NB: Make sure to have IP forwarding enable: LINK (See FirewallD section)

Create privatekey & public key

wg genkey | sudo tee /etc/wireguard/private.key && sudo chmod go= /etc/wireguard/private.key
sudo cat /etc/wireguard/private.key | wg pubkey | sudo tee /etc/wireguard/public.key

Show QR Code from client config:

```
qrencode -t ansiutf8 wg-client.conf
qrencode -t ansiutf8 < wg-client.conf
```

Let us save the QR code as a PNG file:

```
qrencode -t png -o user-qr.png -r wg-client.conf
```

/etc/wireguard/wg0.conf (Example config Host)

```
[Interface]
Address = 10.0.80.1/24
ListenPort = 51820
PrivateKey = “Privat key”
PostUp = iptables -A FORWARD -i %i -j ACCEPT; iptables -t nat -A POSTROUTING -o lan0 -j MASQUERADE
PostDown = iptables -D FORWARD -i %i -j ACCEPT; iptables -t nat -D POSTROUTING -o lan0 -j MASQUERADE

[Peer]
PublicKey = wYNhKTv+UcwWbIvvPNOEsswal07BAYM4EX+diog5kwE=
AllowedIPs = 10.0.80.2/32

[Peer]
PublicKey = +SgKWLR4paP9iJ86pTj5Xd2Pq6FIL+0vB+BzwshekUY=
AllowedIPs = 10.0.80.3/32

[Peer]
PublicKey = ye32WNdIPUPmsax6TOg7hjMxmI85A5PfYMVIxrQwwC8=
AllowedIPs = 10.0.80.4/32

[Peer]
PublicKey = QhIjeyv1P8HKJE23rnBk88WWMZZJ9E+bIz4h+hHvTQ0=
AllowedIPs = 10.0.80.5/32

[Peer]
PublicKey = 68zCAxMKRi0XJsu1SyniZ5tRhQwJFzGQwbvYIpONBAE=
AllowedIPs = 10.0.80.6/32

[Peer]
PublicKey = nqWafI12QRSt71RJRfv0WK9qINUGDCrII1ktCsPz0gk=
AllowedIPs = 10.0.80.7/32

[Peer]
PublicKey = Dj/kBGRxQ8fpLqwmVk3NdZBzoIVEB+P5Vp2+8zBkvn0=
AllowedIPs = 10.0.80.8/32

[Peer]
PublicKey = +2LDNWkHyoKJ/SU0/+TEEvfGiRXwUCVIEphdfUp/jiY=
AllowedIPs = 10.0.80.9/32

[Peer]
PublicKey = R9Ug4wUr4raSwkwDBhIzj9i+PwmABJCQVKu/Yrnf+wI=
AllowedIPs = 10.0.80.10/32

[Peer]
PublicKey = AtMteaDI9a8ka+bD8DPGFJIvcd/uV0C5LGIiVn50PQo=
AllowedIPs = 10.0.80.11/32

[Peer]
PublicKey = HI8uumhfCQ3dX9XA2Ew658uR41TmJOdP5TPq1bIudmE=
AllowedIPs = 10.0.80.13/32
```

/etc/wireguard/client.conf (on client device)

```
[Interface]

Address = 10.0.80.4/24
ListenPort = 51820
PrivateKey = "private key client"
```

```
DNS = 10.0.80.1, keroit.lan

[Peer]
PublicKey = 2JCh2EkruOuRn4HijM1ndPr+CV6kESqXtatpxrzo9A4=
AllowedIPs = 0.0.0.0/0, ::/0
Endpoint = MYIP:51820
```

Securing SSH

On Ubuntu Server, SSH service is enabled by default. Consider if you really want remote access over SSH once router has been setup.

If you device to have SSH server enabled, make sure to only allow SSH from either management, or home/lan FirewallD zone
Another suggestion is to only allow user with SSH-Key to access system
If you imported your Github SSH keys, these should be installed as the default user.

/etc/ssh/sshd_config

```
PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/games
Include /etc/ssh/sshd_config.d/*.conf
ListenAddress 10.0.75.1
#HostKey /etc/ssh/ssh_host_rsa_key
#HostKey /etc/ssh/ssh_host_ecdsa_key
#HostKey /etc/ssh/ssh_host_ed25519_key
PermitRootLogin prohibit-password
PubkeyAuthentication yes
#HostbasedAuthentication no
PasswordAuthentication no
PermitEmptyPasswords no
KbdInteractiveAuthentication no
UsePAM yes

AllowAgentForwarding yes
AllowTcpForwarding yes
X11Forwarding no
X11UseLocalhost no
PrintMotd ywa
PrintLastLog yes
TCPKeepAlive yes
#Banner none
AcceptEnv LANG LC_*
# override default of no subsystems
Subsystem sftp    /usr/lib/openssh/sftp-server
X11Forwarding no
#    AllowTcpForwarding no
PasswordAuthentication no
```

Unattended-Upgrades

Installing this will make sure to keep the system up to date.

/etc/apt/apt.conf.d/50unattended-upgrades

```
Unattended-Upgrade::Allowed-Origins {
	"${distro_id}:${distro_codename}";
	"${distro_id}:${distro_codename}-security";
	"${distro_id}ESMApps:${distro_codename}-apps-security";
	"${distro_id}ESM:${distro_codename}-infra-security";
	"${distro_id}:${distro_codename}-updates";
};

// Python regular expressions, matching packages to exclude from upgrading
Unattended-Upgrade::Package-Blacklist {
	// The following matches all packages starting with linux-
//  "linux-"
```

/etc/apt/apt.conf.d/10periodic

```
APT::Periodic::Update-Package-Lists "1";
APT::Periodic::Download-Upgradeable-Packages "0";
APT::Periodic::AutocleanInterval "0";
```

Hostapd: Creating WiFi Hotspot

The wireless network is controlled by the application hostapd together with wpasupplimentals. This can be started and be stopped as a service with the command

```
sudo systemctl start/stop wifi
sudo systemctl stop wifi
```

Packages:

```
sudo apt install hostapd wpasupplimentals
```

Copy install the configuration and service files

/etc/hostapd/hostapd.conf

/etc/systemctl/system/wifi.service

Aftter installing "the wifi.service" run the following to have that recognized

```
sudo systemctl daemon-reload
```

The wireless network is controlled by the application hostapd and can be started and be stopped as a service with the command

```
sudo systemctl start/stop wifi.service
sudo systemctl stop wifi.service
```

To enable or disable Hostapd on startup:

```
sudo systemctl enable wifi.service
sudo systemctl disable wifi.service
```

5GHz Wifi Channels

- UNII-1
  - 36, 40, 44, 48
- UNII-2a
  - 52, 56, 60, 64
- UNII-2c Extended
  - 100, 104, 108, 112, 116, 120, 124, 128, 132, 136, 140, 144
- UNII-3
  - 149, 153, 157, 161, 165

/etc/hostapd/wifi5G.conf

```
interface=wlp5s0
bridge=lan0
ssid=grotle4_5G
wpa_passphrase=de55erenkode
country_code=dk
ieee80211d=0
wpa=2
hw_mode=a
channel=36
driver=nl80211
wpa_pairwise=TKIP
rsn_pairwise=CCMP
ieee80211n=1
ieee80211ac=1
wmm_enabled=1
```

/etc/hostapd/wifi5G-2.conf

```
interface=wlp5s0_0
ssid=WEB2.0
bridge=lan0
hw_mode=a
channel=36
wpa=2
wpa_passphrase=brumbrum
wpa_key_mgmt=WPA-PSK
rsn_pairwise=CCMP

# Make sure channel is set to same as wifi5G.conf
```

/etc/hostapd/wifi5G-3.conf

```
interface=wlp5s0_1
ssid=WEB2.0
bridge=lan0
hw_mode=a
channel=36
wpa=2
wpa_passphrase=brumbrum
wpa_key_mgmt=WPA-PSK
rsn_pairwise=CCMP

# Make sure channel is set to same as wifi5G.conf
```

/etc/hostapd/wifi.conf

```
interface=wlx00179ab0bc78
bridge=lan0
driver=nl80211
ssid=grotle4
wpa=2
hw_mode=b
channel=1
wpa_passphrase=trunte
wpa_key_mgmt=WPA-PSK
rsn_pairwise=CCMP
```

/etc/systemd/system/wifi5G.service

```
Unit]
Description=Wireless AP Daemon
Documentation=http://www.itso.dk
Wants=basic.target
#After=basic.target
After=network.target

[Service]
Type=oneshot
RemainAfterExit=yes
#User=root
#ExecStart=/usr/bin/bash -c 'iw dev wlp5s0 interface add wlp5s0_0 type __ap; ip addr flush dev wlp5s0;
ifconfig wlp5s0 in>
ExecStart=/usr/bin/bash -c 'ip addr flush dev wlp5s0; ifconfig wlp5s0 inet 10.0.75.2 netmask 255.255.255.0
up; hostapd -B>
ExecStop=/usr/bin/bash -c 'pkill -f hostapd1.conf; ip addr flush dev wlp5s0'

[Install]
WantedBy=multi-user.target

# iw dev wlp5s0 interface add wlp5s0_0 type __ap
```

/etc/systemd/system/wifi.service

```
[Unit]
Description=Wireless AP Daemon
Documentation=http://www.itso.dk
Wants=basic.target
#After=basic.target
After=network.target

[Service]
Type=oneshot
RemainAfterExit=yes
#User=root
#ExecStart=/usr/bin/bash -c 'iw dev wlp5s0 interface add wlp5s0_0 type __ap; ip addr flush dev wlp5s0;
ifconfig wlp5s0 in>
ExecStart=/usr/bin/bash -c 'ip addr flush dev wlx00179ab0bc78; ifconfig wlx00179ab0bc78 inet 10.0.75.3
netmask 255.255.25>
ExecStop=/usr/bin/bash -c 'pkill -f hostapd3.conf; ip addr flush dev wlx00179ab0bc78'

[Install]
WantedBy=multi-user.target
```

/etc/systemd/system/wifi5G-2.service

```
[Unit]
Description=Wireless AP Daemon
Documentation=http://www.itso.dk
Wants=basic.target
#After=basic.target
After=wifi5G.service

[Service]
Type=oneshot
RemainAfterExit=yes
User=root
ExecStart=/usr/bin/bash -c 'iw dev wlp5s0 interface add wlp5s0_0 type __ap; sleep 5; ip addr flush dev
wlp5s0_0; ifconfig wlp5s0_0>
ExecStop=/usr/bin/bash -c 'pkill -f hostapd2.conf; ip addr flush dev wlp5s0_0; sleep 10; iw dev wlp5s0_0 del'

[Install]
WantedBy=multi-user.target
```

/etc/default/hostapd

```
# Defaults for hostapd initscript
#
# WARNING: The DAEMON_CONF setting has been deprecated and will be removed
#          in future package releases.
#
# See /usr/share/doc/hostapd/README.Debian for information about alternative
# methods of managing hostapd.
#
# Uncomment and set DAEMON_CONF to the absolute path of a hostapd configuration
# file and hostapd will be started during system boot. An example configuration
# file can be found at /usr/share/doc/hostapd/examples/hostapd.conf.gz
#
#DAEMON_CONF=""

# Additional daemon options to be appended to hostapd command:-
#       -d   show more debug messages (-dd for even more)
#       -K   include key data in debug messages
#       -t   include timestamps in some debug messages
#
# Note that -B (daemon mode) and -P (pidfile) options are automatically
# configured by the init.d script and must not be added to DAEMON_OPTS.
#
#DAEMON_OPTS=""
DAEMON_CONF="/etc/hostapd/hostapd.conf"
```

NTPD Timeservice (OLD)

Package:

```
sudo apt install ntp
sudo systemctl start ntpd
sudo systemctl enable ntpd
```

Copy in the configuration files:

/etc/ntp/ntpd.conf

/etc/ntp.conf (OLD)

```
# /etc/ntp.conf, configuration for ntpd; see ntp.conf(5) for help

driftfile /var/lib/ntp/ntp.drift

# Leap seconds definition provided by tzdata
leapfile /usr/share/zoneinfo/leap-seconds.list

# Enable this if you want statistics to be logged.
#statsdir /var/log/ntpstats/

statistics loopstats peerstats clockstats
filegen loopstats file loopstats type day enable
filegen peerstats file peerstats type day enable
filegen clockstats file clockstats type day enable

# Specify one or more NTP servers.
# Use servers from the NTP Pool Project. Approved by Ubuntu Technical Board
# on 2011-02-08 (LP: #104525). See http://www.pool.ntp.org/join.html for
# more information.
pool 0.ubuntu.pool.ntp.org iburst
pool 1.ubuntu.pool.ntp.org iburst
pool 2.ubuntu.pool.ntp.org iburst
pool 3.ubuntu.pool.ntp.org iburst

# Use Ubuntu's ntp server as a fallback.
pool ntp.ubuntu.com

# Access control configuration; see /usr/share/doc/ntp-doc/html/accopt.html for
# details.  The web page <http://support.ntp.org/bin/view/Support/AccessRestrictions>
# might also be helpful.
#
# Note that "restrict" applies to both servers and clients, so a configuration
# that might be intended to block requests from certain clients could also end
# up blocking replies from your own upstream servers.

# By default, exchange time with everybody, but don't allow configuration.
restrict -4 default kod notrap nomodify nopeer noquery limited
restrict -6 default kod notrap nomodify nopeer noquery limited

# Local users may interrogate the ntp server more closely.
restrict 127.0.0.1
restrict ::1

# Needed for adding pool entries
restrict source notrap nomodify noquery

# Clients from this (example!) subnet have unlimited access, but only if
# cryptographically authenticated.
#restrict 192.168.123.0 mask 255.255.255.0 notrust


# If you want to provide time to your local subnet, change the next line.
# (Again, the address is an example only.)
#broadcast 192.168.123.255

# If you want to listen to time broadcasts on your local subnet, de-comment the
# next lines.  Please do this only if you trust everybody on the network!
#disable auth
#broadcastclient

#Changes required to use pps synchronization as explained in documentation:
#http://www.ntp.org/ntpfaq/NTP-s-config-adv.htm#AEN3918

#server 127.127.8.1 mode 135 prefer    # Meinberg GPS167 with PPS
#fudge 127.127.8.1 time1 0.0042        # relative to PPS for my hardware

#server 127.127.22.1                   # ATOM(PPS)
#fudge 127.127.22.1 flag3 1            # enable PPS API
```

FirewallD - Configs (OLD)

(Old version) /etc/firewalld/direct.xml

```
<?xml version="1.0" encoding="utf-8"?>
<direct>
  <rule ipv="ipv4" table="nat" chain="POSTROUTING" priority="0">-o --zone=public -j MASQUERADE</rule>
  <rule ipv="ipv4" table="filter" chain="FORWARD" priority="0">-i --zone=home -o public -j ACCEPT</rule>
  <rule ipv="ipv4" table="filter" chain="FORWARD" priority="0">-i --zone=home -o --zone=public -j
ACCEPT</rule>
  <rule ipv="ipv4" table="filter" chain="FORWARD" priority="0">-i --zone=public -o --zone=home -m state
--state RELATED,ESTABLISHED -j ACCEPT</rule>
</direct>
```

(Old version) /etc/firewalld/firewalld.conf

```
# firewalld config file

# default zone
# The default zone used if an empty zone string is used.
# Default: public
DefaultZone=public

# Clean up on exit
# If set to no or false the firewall configuration will not get cleaned up
# on exit or stop of firewalld
# Default: yes
CleanupOnExit=yes

# Lockdown
# If set to enabled, firewall changes with the D-Bus interface will be limited
# to applications that are listed in the lockdown whitelist.
# The lockdown whitelist file is lockdown-whitelist.xml
# Default: no
Lockdown=no

# IPv6_rpfilter
# Performs a reverse path filter test on a packet for IPv6. If a reply to the
# packet would be sent via the same interface that the packet arrived on, the
# packet will match and be accepted, otherwise dropped.
# The rp_filter for IPv4 is controlled using sysctl.
# Default: yes
IPv6_rpfilter=yes

# IndividualCalls
# Do not use combined -restore calls, but individual calls. This increases the
# time that is needed to apply changes and to start the daemon, but is good for
# debugging.
# Default: no
IndividualCalls=no

# LogDenied
# Add logging rules right before reject and drop rules in the INPUT, FORWARD
# and OUTPUT chains for the default rules and also final reject and drop rules
# in zones. Possible values are: all, unicast, broadcast, multicast and off.
# Default: off
LogDenied=off

# FirewallBackend
# Selects the firewall backend implementation.
# Choices are:
#       - nftables (default)
#       - iptables (iptables, ip6tables, ebtables and ipset)
FirewallBackend=nftables

# FlushAllOnReload
# Flush all runtime rules on a reload. In previous releases some runtime
# configuration was retained during a reload, namely; interface to zone
# internet.
# Defaults to "yes".
RFC3964_IPv4=yes

# AllowZoneDrifting
# Older versions of firewalld had undocumented behavior known as "zone
# drifting". This allowed packets to ingress multiple zones - this is a
# violation of zone based firewalls. However, some users rely on this behavior
# to have a "catch-all" zone, e.g. the default zone. You can enable this if you
# desire such behavior. It's disabled by default for security reasons.
# Note: If "yes" packets will only drift from source based zones to interface
# based zones (including the default zone). Packets never drift from interface
# based zones to other interfaces based zones (including the default zone).
# Possible values; "yes", "no". Defaults to "no".
AllowZoneDrifting=no
```

(Old version) /etc/firewalld/zones/public.xml

```
<?xml version="1.0" encoding="utf-8"?>
<zone>
  <short>Public</short>
  <description>For use in public areas. You do not trust the other computers on networks to not harm your
computer. Only selected incoming connections are accepted.</description>
 # <service name="ssh"/>
  <masquerade/>
  <interface name="internet0"/>
</zone>
```

(Old version) /etc/firewalld/zones/home.xml

```
<?xml version="1.0" encoding="utf-8"?>
<zone>
  <short>Home</short>
  <description>For use in home areas. You mostly trust the other computers on networks to not harm your
computer. Only selected incoming connections are accepted.</description>
  <service name="ssh"/>
  <service name="mdns"/>
  <service name="samba-client"/>
  <service name="dhcpv6-client"/>
  <service name="dhcpv4-client"/>

  <service name="dns"/>
  <service name="cockpit"/>
  <interface name="lan0"/>
</zone>
```

(Old version) /etc/firewalld/zones/guest.xml

```
<?xml version="1.0" encoding="utf-8"?>
<zone>
  <short>Guest</short>
  <description>For use in home areas. You mostly trust the other computers on networks to not harm your computer. Only selected incoming connections are accepted.</description>
  <service name="ssh"/>
  <service name="mdns"/>
  <service name="samba-client"/>
  <service name="dhcpv6-client"/>
  <service name="dns"/>
  <interface name="guest"/>
</zone>
```

ISC-DHCP-Server Config (OLD)

/etc/dhcp/dhcpd.conf (OLD)

```
#option domain-name "home.lan";
default-lease-time 600;
max-lease-time 7200;
ddns-update-style none;
subnet 10.0.75.0 netmask 255.255.255.0{
  range 10.0.75.11 10.0.75.200;
  option domain-name-servers 10.0.75.1;
  option routers 10.0.75.1;
  option subnet-mask 255.255.255.0;
}

subnet 10.0.76.0 netmask 255.255.255.0{
  range 10.0.76.11 10.0.76.200;
  option domain-name-servers 10.0.76.1;
  option routers 10.0.76.1;
  option subnet-mask 255.255.255.0;
}

host Tadohub {
  hardware ethernet ec:e5:12:15:93:9a;
  fixed-address 10.0.75.21;
}
```

Unbound DNS Server (OLD)

Installation

```
sudo apt install unbound
sudo systemctl enable unbound
sudo systemctl start unbound
```

/etc/unbound/unbound.conf.d/acl.conf (OLD)

```
server:
        interface: 10.0.75.1
        access-control: 10.0.75.0/24 allow
        port: 53
        do-ip4: yes
        do-ip6: no
        private-address: 10.0.75.0/24
forward-zone:
        name: .
        forward-addr: 8.8.8.8
        forward-addr: 8.8.4.4
```

Download my Hostfile:

```
wget http://itso.dk/files/other/hosts
```

Create adblock.conf from my Adblock hosts file:

```
wget https://itso.dk/files/other/hosts
echo “server:” > /etc/unbound/unbound.d/adblock.conf
cat hosts | grep '^127\.0\.0\.1' | awk '{print "local-zone: \""$2"\" deny"}' >>
/etc/unbound/unbound.d/adblock.conf
```

/etc/unbound/unbound.conf.d/adblock.conf (OLD)

```
server:
  local-zone: "ads.youtube.com” deny
  local-zone: “connect.facebook.net” deny
```

Troubleshooting

When with Bind9 and ISC-DHCP-Server and big changes, it can be a good idea bring down service and delete system state files:

```
/var/lib/dhcp
/etc/bind/*.jlp
```

If any of the services fail, remember in linux systems everything is logged.
When you start a service log at the console output, you can also get a status on each service:

```
sudo systemctl status bind9
```

Logs are stored in **/var/log**
Realtime logs can be access by Cockpit webinterface and the tab: LOGS
Realtime logs can be accessed by the commands:

```
sudo journalctl -f
```

Or

```
tail -F /var/logs/syslog
```

Links:

https://www.itso.dk
https://teklager.se/en/
http://www.pcengines.ch/apu4b4.htm