WireGuard VPN Setup Guide

- Originally written by JohnB in 2024-09
- This is an attempt to document the setup process and the lessons from that process, both for setting up a personal phone connection to the Mesh as well as setting up an Omni in Queens that connects back to the Mesh via VPN
- A note on permissions: You will need someone with write access to the IPRanges Google Spreadsheet, where the Mesh keeps track of all the different in-use IP addresses/networks/subnets. You will also need root access to the SN1 and/or SN3 VPN instances in order to edit files on them to support your connection

SN1 "Road Warrior" Remote Access for Phone

- WireGuard app 1.0.20231018 running on Android 14
- Usage: This type of setup allows for remote access of NYC Mesh IP addresses for remote debugging, Mesh access while on a rooftop independent of that rooftop to avoid bridge filter or other issues, and so on. This is a sort of hub-and-spoke setup where one WireGuard tunnel on the server will have many peers in the field
- Example configuration values
    - **NOTE: These are demo values and should be replaced when following the guide**
    - IPRanges spreadsheet reservation `10.70.73.0/24` has enough remaining room for many more setups
    - WireGuard phone tunnel IP `10.70.73.69` used here for the phone/laptop for demonstration purposes. Note that it is part of the subnet above
    - WireGuard server SSH IP `10.42.43.42`
    - WireGuard server public IP/Endpoint `199.167.59.4` with port `51822`
    - WireGuard server tunnel private key `[redacted]` and public key `04crAKqAju+ZlEXCdZGAa4OyhDe1k2CHIlshr2KoYAQ=`
    - WireGuard phone tunnel private key `CIHyP1xYRh3zl7bE6XYsXXFhrf8CXjn4mlIkEdfLAE0=` and public key `6jzT2XA7IX2E8htpWP9qJIGUzhjkFESFqvFowNVj5Xw=`

Phone Setup

- Install the WireGuard app
- Create a new tunnel from scratch
- Add an Interface Name of something similar to MESHSN1
- Generate a Private key, for example `CIHyP1xYRh3zl7bE6XYsXXFhrf8CXjn4mlIkEdfLAE0=`. This will also generate a corresponding Public key `6jzT2XA7IX2E8htpWP9qJIGUzhjkFESFqvFowNVj5Xw=` which should be noted down for the server setup later
- Add an Interface Address, such as `10.70.73.69/32`. The specific IP will be different from device to device, but the /32 should always be present so the phone always uses the exact address listed
- Add an Interface DNS server of `10.10.10.10`, which is the Mesh DNS server
- Add a Peer at the bottom to show more entry fields
- Add the WireGuard server tunnel public key to the Peer Public key field, for example `04crAKqAju+ZlEXCdZGAa4OyhDe1k2CHIlshr2KoYAQ=`
- Add the WireGuard server public IP/endpoint with port to the Endpoint field, for example `199.167.59.4:51822`
- Add `0.0.0.0/0` to the Allowed IPs field to route all traffic on the device through the WireGuard tunnel
    - Allowed IPs means the "range of IP addresses to be routed to the SN1 WireGuard Tunnel"
    - To only send Mesh traffic through the tunnel, instead enter `10.0.0.0/8,199.168.59.0/24,199.170.132.0/24,23.158.16.0/24`. These are the IP ranges commonly considered "Mesh" IPs, and are deployed as part of the default Mesh device configuration https://configgen.nycmesh.net/
- Push the Save button, and then enable the new configuration
- When the configuration is functional, open it and look for the "Latest handshake" at the bottom, as well as "Transfer" statistics showing the transmitted and received bytes
- The configuration can also be added as a file. Here's an example of how the file might look:

```
[Interface]
Address = 10.70.73.69/32
DNS = 10.10.10.10
PrivateKey = CIHyP1xYRh3zl7bE6XYsXXFhrf8CXjn4mlIkEdfLAE0=

[Peer]
AllowedIPs = 0.0.0.0/0
Endpoint = 199.167.59.4:51822
PublicKey = 04crAKqAju+ZlEXCdZGAa4OyhDe1k2CHIlshr2KoYAQ=
```

SN1 Server Setup

- SSH to the SN1 VPN1 WireGuard server with `ssh support@10.42.43.42` from inside the Mesh. Run `su` to gain root access
- Edit the `/etc/wireguard/wg2.conf` to add new lines to the bottom containing the new WireGuard Peer. Pick a new IP that has not already been used in the file while still being in the range of the subnet
  - Multiple WireGuard files and interfaces exist, but this `wg2` tunnel is specifically for remote access and is the only place that needs editing
  - The PublicKey line is the phone tunnel public key
  - The AllowedIPs line is to list the specific IPs that are able to pass traffic to and from WireGuard. In other words, Allowed IPs means "the range of accepted peer interface IP addresses for this particular peer." In this case a /32 and a specific address are used to ensure each remote access can only connect from a single IP. Otherwise, if a /24 or /0 were used, it's possible for multiple peers to assume the first IP in the range and then have address conflicts.

```
# JohnB phone
[Peer]
PublicKey = 6jzT2XA7IX2E8htpWP9qJIGUzhjkFESFqvFowNVj5Xw=
AllowedIPs = 10.70.73.69/32
```

- Disable and re-enable the network interface this WireGuard tunnel is connected to with `ifdown --force wg2` and then `ifup wg2`
  - This works because a configuration exists at `/etc/network/interfaces` for the `wg2` interface to perform actions before, during, and after taking the interface online or offline
  - This also works because the `wg2.conf` file specifies a ListenPort of 51822, and that port is open in the firewall using `iptables`. The rules can be viewed with `iptables -S`
- That's it! By adding a few lines and cycling the interface, the WireGuard server interface is ready to accept the phone's incoming connection
- The command `wg show wg2 latest-handshakes` can be run to see the timestamp of each peer's last connection. If a peer hasn't connected, a timestamp of `0` will be shown, otherwise a working connection should have a timestamp such as `1727705817`
  - `wg show wg2 transfer` will show the traffic amounts for each peer
  - `wg show wg2 allowed-ips` will show all the peer IP addresses

Alternate pfSense Setup

- This section is for connecting a home pfSense router to the Mesh for general Mesh IP access from any device behind it.
- Install the WireGuard package from System/Package Manager

- Go to VPN/WireGuard/Settings and change Interface Group Membership to "Only Unassigned Tunnels"
- Go to VPN/WireGuard/Tunnels and create a new tunnel. Generate a new keypair and note down the Public key for later. Let's assume in this case that it's the same as the phone setup above, with public key `6jzT2XA7IX2E8htpWP9qJIGUzhjkFESFqvFowNVj5Xw=`. Save the tunnel
- Go to VPN/WireGuard/Peers and create a new peer. Set its tunnel to the tunnel that was just created. Uncheck the Dynamic Endpoint. Set the Endpoint to the SN1 public IP `199.167.59.4` with port `51822`. Set the Keepalive to 25 seconds (TBD whether this is necessary). Set the Public Key to the SN1 WireGuard server tunnel public key `04crAKqAju+ZlEXCdZGAa4OyhDe1k2CHIlshr2KoYAQ=`. Set the allowed IPs to the NYC Mesh Internal IPs, `10.0.0.0/8`, `199.168.59.0/24`, `199.170.132.0/24`, and `23.158.16.0/24`
- Go to Interfaces/Assignments and pick the new WireGuard tunnel in the Available Network Ports and then hit "Add"
- Let the page reload and then click on the new OPT4 interface that was created for the WireGuard tunnel. Change its name to something descriptive such as MESHSN1, set its IPV4 to be Static, set the MTU to something lower than 1500 (1200 works, maybe 1344/1366 works), and then set the IPV4 address to the phone tunnel IP, `10.70.73.68/24`. Note the /24 is important here so the interface is aware of the entire subnet. In this case, this will not assign a random IP inside the /24. Make sure the block networks is turned off, and then save.
- Now go to System/Routing/Gateways and add a new Gateway. Pick the MESHSN1 interface created before, set the Gateway to the IP of the SN1 tunnel, in this case `10.70.73.1`. Save the Gateway
- Now go to System/Routing/Static Routes and add new Static Routes for each of the NYC Mesh Internal IPs, `10.0.0.0/8`, `199.168.59.0/24`, `199.170.132.0/24`, and `23.158.16.0/24`, with the Gateway set to the MESHSN1 gateway created earlier
- Now go to Firewall/Rules and pick the tab of the name of the Interface created earlier, in this case MESHSN1. Do not go to the WireGuard tab, that is irrelevant here. Once at the MESHSN1 Rules tab, add a new rule that Passes IPv4 traffic of Any protocol from Any source to Any destination, on interface MESHSN1. This will pass VPN traffic from WireGuard peers out to other interfaces
- At this point, the WireGuard tunnel should be online and able to pass traffic from any device connected through pfSense.
    - The status of the VPN link can be checked in Status/WireGuard and expanding the tunnel to see the latest handshake, the RX, and the TX
    - Go to Diagnostics/Routes to see the routes assigned to the new WireGuard tunnel
    - Go to Status/Gateways to see whether the MESHSN1 gateway is Online and what the round trip latency is to the endpoint

SN3 OSPF MikroTik Access

- MikroTik RouterOS 7.16 running on an OmniTIK 5 ac
- Usage: This type of setup is reserved for expanding the Mesh, as new nodes can connect to the WireGuard node and advertise OSPF routes that are connectable from the rest of the Mesh. This sort of setup requires a separate WireGuard tunnel and network interface on the server for each new WireGuard peer connection.
- Example configuration values
  - **NOTE: These are demo values and should be replaced when following the guide**
  - Mesh Network Number `666`
  - OmniTIK 5 ac port 1 connected to Verizon Fios router. Fios Router gateway/internal IP is `10.1.102.1`, the subnet is `10.1.102.0/24` and Omni DHCP IP is `10.1.102.100`
  - IPRanges spreadsheet reservation is the `10.70.251.120/30` subnet/network. Note that a /31 will not work with Mikrotik devices. Note that the range must be unique across the Mesh
  - WireGuard SN3 tunnel IP is `10.70.251.121`. Note that it is part of the subnet above
  - WireGuard Omni tunnel IP is `10.70.251.122`. Note that it is part of the subnet above
  - WireGuard SN3 server SSH IP `199.170.132.4`
  - WireGuard SN3 server public IP/Endpoint `199.170.132.4` with port `51841`
  - WireGuard SN3 server tunnel private key `UOh8LbR8N0LFSQR57J8F1PN0v9xr+kFT/nS6zkQK4Go=` and public key `quKdVkH8qRNaJ/JwgQ8bIob5OwKtxSf9BrdN3Kgx5UE=`
  - WireGuard Omni tunnel private key `sAUBqTCSAAfoqna9F9lqRbc7t8W4xFFOfsoKQjbbbGo=` and public key `JFVGiXzPEQWBZTTJsYbgLl17zYPFrzgr5d+ZtAHV0kg=`

MikroTik Setup

- First generate and download a config from https://configgen.nycmesh.net/?version=testing&device=Omnitik5AC while entering the network number, in this case `666`
  - Note that the template is `omni-only-ros7.rsc.tmpl` which in other words is a template prepared for RouterOS 7 OmniTIK 5 ac or OmniTIK 5 PoE ac devices
  - Note that the config can be applied to other devices, but the specific interfaces and beeps may need to be modified depending on the target device's hardware
- Update to the latest RouterOS 7 Stable release, which as of writing is 7.16. This can be done in System/Packages and "Check for Updates" and then "Download&Install"
- Apply the config to the device. This may be most easily done by performing the following:
  - Upload the config by going to Files and then Upload and picking the `omni-only-ros7.rsc` file downloaded earlier
  - Go to System/Reset Configuration, check the "Keep users" and "No Default Configuration" options, and then picking the `omni-only-ros7.rsc` file in the "Run after Reset" box
  - Hit the "Reset Configuration" button and then wait five minutes for the Kernkraft 400 song to play

- Begin making modifications for the WireGuard setup:
- Remove port1/ether1 from the Mesh bridge so it can be used to get Internet access and DHCP from the Fios router. Go to Bridge/Ports and push the "Remove" button next to `ether1`
- Add a DHCP client to ether1 so it can get Internet access. Go to IP/DHCP Client, hit "New" and set the Interface to `ether1` and then make sure the "Add Default Route" is set to No. Hit OK and the DHCP-assigned IP should now show here, as well as in the IP/Addresses page
  - Note: The default route is what allows the Omni to get raw internet access through the Fios router, but it will conflict with the OSPF default route to be added later and cause the WireGuard connection to flap
- Add a WireGuard Tunnel.
  - Name it `nycmesh-sn3-wg-vpn`
  - Set the MTU to `1344`. The MTU is an open question of optimization. The default RouterOS provides is 1420, but Mesh configs seem to often use 1366 or 1344. The smaller the number, the less chance there is of packet fragmentation, but the greater chance of slower speeds because of less data per packet
  - Hit OK.
  - Reopen the newly created tunnel and note down its Public Key, for example `JFVGiXzPEQWBZTTJsYbgLl17zYPFrzgr5d+ZtAHV0kg=`
- Add an IP to the WireGuard Tunnel interface
  - Go to IP/Addresses and create a New Address
  - Set the Address to the IP reserved for the Omni end of the tunnel, in this case `10.70.251.122/30`.
  - Set the Network to the beginning of the IP range set aside for this Tunnel, in this case `10.70.251.120`
  - Set the Interface to `nycmesh-sn3-wg-vpn`
- Add a WireGuard Peer to the Tunnel. Go to the WireGuard/Peers tab and hit "New"
  - Set the interface to `nycmesh-sn3-wg-vpn`
  - Set the public key to the WireGuard server tunnel public key, such as `quKdVkH8qRNaJ/JwgQ8bIob5OwKtxSf9BrdN3Kgx5UE=`
  - Set the endpoint to the SN3 public IP `199.170.132.4`
  - Set the Endpoint Port to the chosen port, such as `51841`
  - Set the Allowed Address to `0.0.0.0/0` which will route all traffic through this WireGuard Peer
  - Set the Persistent Keepalive to 25 seconds. This is an open question of whether it is necessary or not. Presumably it's there because it was needed, as the WireGuard default is to have no Persistent Keepalive
  - Hit OK to save the Peer
- Now configure OSPF Interface Templates. Go to Routing/OSPF and the Interface Templates tab, and hit "New"
  - Set the Interface to `nycmesh-sn3-wg-vpn`
  - Set the Network to the subnet reserved for this tunnel, in this case `10.70.251.120/30`
  - Set the Network Type to `ptmp`
  - Set the Cost to the chosen value for this link, in this case `100`
  - Set the Priority to 1

    - Set the Auth ID to 1
    - Hit OK to save
- Now configure an OSPF Static Neighbor. Go to Routing/OSPF and the Static Neighbor tab, hit "New" and then fill out the Address, which should be a combination of the SN3 tunnel IP and the Omni WireGuard interface name, in this case `10.70.251.121%nycmesh-sn3-wg-vpn`
- Now configure a Static Route that tells the Omni to route traffic destined for the SN3 public IP to the Fios gateway, not through the WireGuard tunnel.
    - Go to IP/Routes and hit "New"
    - Set the Destination Address to the SN3 public IP plus a /32, in this case `199.170.132.4/32`
    - Set the Gateway to the Fios router's gateway/internal IP, in this case `10.1.102.1`
    - Add a Distance and set it to 1, representing an OSPF cost of 1
    - Note: If this route were not in place: The tunnel link would come online, it would advertise the 0.0.0.0/0 default route, all traffic would be routed through the tunnel INCLUDING the WireGuard-wrapped traffic, and then the tunnel link would drop because it's trying to send its traffic inside itself
    - Note: If this gateway IP changes, say when the DHCP client on ether1 gets a new lease, this Static Route will have to be updated manually. Until that point, the WireGuard link will function for traffic inside the Mesh but no public Internet traffic
- Now configure a Route Filter that prevents the Fios LAN subnet from being advertised via OSPF. Go to Routing/Filters and hit "New"
    - Set the Chain to `ospf-out`
    - Set the Rule to the Fios router's LAN subnet, in this case `if (dst in 10.1.102.0/24) {reject;}`
    - Hit "OK"
    - Then click and drag to re-order this rule to sit above the `ospf-out accept` rule
- At this point, assuming the SN3 side is set up, the tunnel should be functional. Some of the tests that can be performed:
    - Go to the WireGuard Tunnel `nycmesh-sn3-wg-vpn` and scroll down to its Traffic section to see if traffic is flowing in both directions
    - Go to the WireGuard Peer and scroll down to see the Last Handshake, as well as the Rx and Tx traffic counts
    - Ping the SN3 tunnel's IP address `10.70.251.121` or Google's DNS IP `8.8.8.8` or another Mesh node such as `10.69.5.84` with Tools/Ping
    - Try to resolve DNS by running `put [resolve google.com]` in the Terminal
    - Observe the Log to ensure there aren't WireGuard flaps causing OSPF to cycle between down, init, and full (also visible in Routing/OSPF/Neighbors)
    - Plug a device into one of ether2 thru ether5, or connect to the Community Wifi SSID, and see if it gets internet

Below is an example of the full config pulled from the Omni, scrubbed of PII. Note that this config is specific to NN666, and cannot be copypasta'd to another device without modification of IP ranges, addresses, keys, etc.

```
# 2024-09-30 14:27:44 by RouterOS 7.17beta2
# software id = 12345
#
# model = RBOmniTikG-5HacD
# serial number = 12345
/interface bridge add fast-forward=no name=mesh protocol-mode=none
/interface bridge add fast-forward=no name=wds protocol-mode=none
/interface ethernet set [ find default-name=ether1 ] comment="JohnB home Fios uplink for WireGuard"
/interface wireguard add comment="SN3 WireGuard" listen-port=13231 mtu=1344 name=nycmesh-sn3-wg-vpn
/interface wireless security-profiles set [ find default=yes ] supplicant-identity=MikroTik
/interface wireless security-profiles add authentication-types=wpa-psk,wpa2-psk management-protection=allowed mode=dynamic-keys name=nycmeshnet supplicant-identity=nycmesh
/interface wireless set [ find default-name=wlan1 ] antenna-gain=0 band=5ghz-a/n/ac channel-width=20/40/80mhz-Ceee default-forwarding=no disabled=no frequency=5180 installation=any mode=ap-bridge radio-name=nycmesh-666-omni rx-chains=0,1 security-profile=nycmeshnet ssid=nycmesh-666-omni tx-chains=0,1 wireless-protocol=802.11 wps-mode=disabled
/interface wireless add disabled=no mac-address=7A:9A:18:51:A3:A4 master-interface=wlan1 name=wlan2 ssid="-NYC Mesh Community WiFi-" wps-mode=disabled
/interface wireless add disabled=no mac-address=7A:9A:18:51:A3:A5 master-interface=wlan1 name=wlan3 security-profile=nycmeshnet ssid=nycmesh-wds wds-default-bridge=wds wds-mode=dynamic-mesh wps-mode=disabled
/interface wireless add comment="uses nycmesh-xxxx-omni via mesh bridge" mac-address=7A:9A:18:51:A3:A6 master-interface=wlan1 mode=station-bridge name=wlan4 security-profile=nycmeshnet ssid=nycmesh-xxxx-omni wds-default-bridge=mesh
/interface wireless manual-tx-power-table set wlan4 comment="uses nycmesh-xxxx-omni via mesh bridge"
/interface wireless nstreme set *C comment="uses nycmesh-xxxx-omni via mesh bridge"
/ip pool add name=local ranges=10.96.166.134-10.96.166.185
/ip dhcp-server add address-pool=local interface=mesh name=localdhcp
/routing ospf instance add disabled=no in-filter-chain=ospf-in name=default originate-default=never out-filter-chain=ospf-out redistribute=connected router-id=10.69.6.66
/routing ospf area add disabled=no instance=default name=backbone
/interface bridge filter add action=drop chain=forward in-bridge=mesh
/interface bridge filter add action=drop chain=forward in-bridge=wds
/interface bridge filter add action=drop chain=forward in-interface=wlan2
/interface bridge port add bridge=mesh interface=ether2
/interface bridge port add bridge=mesh interface=ether3
```

```
/interface bridge port add bridge=mesh interface=ether4
/interface bridge port add bridge=mesh interface=ether5
/interface bridge port add bridge=mesh interface=wlan1
/interface bridge port add bridge=mesh interface=wlan2
/interface bridge port add bridge=mesh interface=wlan4
/interface bridge port add bridge=wds interface=wlan3
/interface bridge port add bridge=wds interface=dynamic internal-path-cost=100 path-cost=100
/interface bridge settings set use-ip-firewall=yes
/interface wireguard peers add allowed-address=0.0.0.0/0 endpoint-address=199.170.132.4 endpoint-port=51841 interface=nycmesh-sn3-wg-vpn name=peer1 persistent-keepalive=25s public-key="quKdVkH8qRNaJ/JwgQ8bIob5OwKtxSf9BrdN3Kgx5UE="
/interface wireless connect-list add allow-signal-out-of-range=3s interface=wlan3 security-profile=nycmeshnet signal-range=-65..120
/interface wireless connect-list add connect=no interface=wlan3 security-profile=nycmeshnet signal-range=-120..-65
/ip address add address=10.96.166.129/26 interface=mesh network=10.96.166.128
/ip address add address=10.69.6.66/16 interface=mesh network=10.69.0.0
/ip address add address=10.68.6.66/16 interface=wds network=10.68.0.0
/ip address add address=10.70.251.122/30 comment="SN3 WireGuard P2P" interface=nycmesh-sn3-wg-vpn network=10.70.251.120
/ip dhcp-client add add-default-route=no comment="PoE and VLAN to pfSense OMNI VLAN" interface=ether1
/ip dhcp-server network add address=10.96.166.128/26 dns-server=10.10.10.10,10.96.166.129 gateway=10.96.166.129 netmask=26
/ip dns set allow-remote-requests=yes servers=10.10.10.10,1.1.1.1
/ip firewall address-list add address=10.0.0.0/8 list=meshaddr
/ip firewall address-list add address=199.167.59.0/24 list=meshaddr
/ip firewall address-list add address=199.170.132.0/24 list=meshaddr
/ip firewall address-list add address=23.158.16.0/24 list=meshaddr
/ip firewall filter add action=accept chain=input protocol=icmp
/ip firewall filter add action=accept chain=input dst-port=53 protocol=udp
/ip firewall filter add action=accept chain=input connection-state=established,related
/ip firewall filter add action=drop chain=input in-bridge-port=wlan2
/ip firewall filter add action=drop chain=input src-address-list=!meshaddr
/ip firewall service-port set ftp disabled=yes
/ip firewall service-port set tftp disabled=yes
/ip firewall service-port set h323 disabled=yes
/ip firewall service-port set sip disabled=yes
/ip firewall service-port set pptp disabled=yes
```

```
/ip hotspot profile set [ find default=yes ] html-directory=hotspot
/ip route add comment="SN3 VPN Static Route via JohnB Fios" disabled=no distance=1 dst-address=199.170.132.4/32 gateway=10.1.102.1 routing-table=main suppress-hw-offload=no
/routing bfd configuration add disabled=no interfaces=all
/routing filter rule add chain=ospf-in disabled=no rule="set distance 205;\
    \nset bgp-communities 65000:110;\
    \naccept;"
/routing filter rule add chain=ospf-out disabled=no rule="if (dst in 10.1.102.0/24) {reject;}"
/routing filter rule add chain=ospf-out disabled=no rule="if (dst == 10.69.0.0/16) {reject;}"
/routing filter rule add chain=ospf-out disabled=no rule=accept
/routing ospf interface-template add area=backbone comment="mesh bridge" cost=10 disabled=no interfaces=mesh networks=10.69.0.0/16 priority=1 type=ptmp-broadcast use-bfd=yes
/routing ospf interface-template add area=backbone comment="wds bridge" cost=100 disabled=no interfaces=wds networks=10.68.0.0/16 priority=1 type=ptmp-broadcast use-bfd=yes
/routing ospf interface-template add area=backbone auth-id=1 comment=nycmesh-sn3-wg-vpn cost=100 disabled=no interfaces=nycmesh-sn3-wg-vpn networks=10.70.251.120/30 priority=1 type=ptmp
/routing ospf static-neighbor add address=10.70.251.121%nycmesh-sn3-wg-vpn area=backbone disabled=no
/snmp set enabled=yes
/system clock set time-zone-autodetect=no time-zone-name=America/New_York
/system identity set name=nycmesh-666-omni
/system note set note="NN:666 #16144 JohnB. ROS7 configured with OSPF WireGuard. Port1 uplink via Fios" show-at-cli-login=yes
/system ntp client servers add address=10.10.10.123
/system scheduler add disabled=yes interval=15s name=exstart_check on-event=exstart_repair policy=ftp,reboot,read,write,policy,test,password,sniff,sensitive,romon start-date=2024-01-15 start-time=21:11:00
/system script add dont-require-permissions=no name=exstart_repair owner=admin policy=ftp,reboot,read,write,policy,test,password,sniff,sensitive,romon source=":local badNeighbor [/routing/ospf/neighbor/get [:pick [/routing/ospf/neighbor/find state=\"ExStart\"] 0 1] address]\
    \n\
    \n:log error \"Bad Neighbor Found: \$badNeighbor\"\
    \n\
    \n/routing/filter/rule/add chain=ospf-in place-before=0 rule=\"if (dst == \$badNeighbor/32) { reject }\"\
    \n\
    \n:delay 5000ms\
    \n\
    \n/routing/filter/rule/remove [find chain=ospf-in rule=\"if (dst == \$badNeighbor/32) {
reject }\"]"
/tool graphing interface add
/tool graphing queue add
/tool graphing resource add
```

SN3 Server Setup

- SSH to the SN3 VPN1 WireGuard server with `ssh support@199.170.132.4` from inside the Mesh. Run `su` to gain root access
- First identify an open port that can be used. In this case, port `51841` was selected after using `iptables -S` to print existing ports in use
- Create a new network interface based on the Network Number `666` named `wg666` by adding the following to the bottom of the `/etc/network/interfaces` file
    - Consider making a backup of the file before editing, perhaps using the date in the filename: `cp /etc/network/interfaces /etc/network/interfaces.bak.$(date '+%Y%m%d')`
    - Lines beginning with a `#` are comments and can be placed anywhere
    - Address should be the WireGuard SN3 tunnel IP is `10.70.251.121`
    - netmask should be the correct value for a /30, which is `255.255.255.252`
    - MTU should be set to a value that matches the MikroTik device, in this case 1344
    - The other lines help configure WireGuard's connection to this network interface

```
# johnb 10.70.251.120/30 IPRanges
auto wg666
iface wg666 inet static
        address 10.70.251.121/30
        netmask 255.255.255.252
        mtu 1344
        pre-up ip link add $IFACE type wireguard
        pre-up wg setconf $IFACE /etc/wireguard/$IFACE.conf
        post-down ip link del $IFACE
```

- Create a new WireGuard tunnel by making a new file `/etc/wireguard/wg666.conf` based on the Network Number `666`
    - Set the PrivateKey to the WireGuard SN3 server tunnel private key `UOh8LbR8N0LFSQR57J8F1PN0v9xr+kFT/nS6zkQK4Go=`
    - Add the corresponding WireGuard SN3 server tunnel public key as a comment for easy access `quKdVkH8qRNaJ/JwgQ8bIob5OwKtxSf9BrdN3Kgx5UE=`
    - Add the ListenPort `51841` that was selected earlier

    - Add the PublicKey of the MikroTik Omni device noted down earlier `JFVGiXzPEQWBZTTJsYbgLl17zYPFrzgr5d+ZtAHV0kg=`
    - Add the AllowedIPs of `0.0.0.0/0` to ensure any IP can connect through this tunnel. This allows OSPF neighbors on the other end of the tunnel to still pass traffic, not just the directly-connected Omni with IP `10.70.251.122`
    - Set the Persistent Keepalive to 25 seconds. This is an open question of whether it is necessary or not. Presumably it's there because it was needed, as the WireGuard default is to have no Persistent Keepalive.

```
[Interface]
PrivateKey = UOh8LbR8N0LFSQR57J8F1PN0v9xr+kFT/nS6zkQK4Go=
# PublicKey = quKdVkH8qRNaJ/JwgQ8bIob5OwKtxSf9BrdN3Kgx5UE=
ListenPort = 51841

# Node 666 johnb
[Peer]
PublicKey = JFVGiXzPEQWBZTTJsYbgLl17zYPFrzgr5d+ZtAHV0kg=
AllowedIPs = 0.0.0.0/0, ::/0
PersistentKeepalive = 25
```

- Then edit the Bird routing software configuration file `/etc/bird/bird.conf` to configure the new OSPF neighbor at the other end of the WireGuard tunnel. Ensure the structure of the file is maintained with regard to curly braces, indentation, etc.
    - Consider making a backup of the file before editing, perhaps using the date in the filename: `cp /etc/bird/bird.conf /etc/bird/bird.conf.bak.$(date '+%Y%m%d')`
    - Set the interface to the same name given to the `/etc/network/interfaces` interface that was created, `wg666`
    - Set the Cost to the chosen value for this link, in this case `100`
    - Set the TX length to match the MTU, in this case `1344`
    - Set the neighbors to the WireGuard Omni tunnel IP `10.70.251.122`

```
                interface "wg666" {
□□                #johnb
                        cost 100;
                        tx length 1344;
                        type ptmp;
                        neighbors {
                                10.70.251.122;
                        };
                };
```

- Then configure the firewall to have the selected port `51841` forwarded, so the WireGuard handshake from the Omni can make it to the WireGuard instance running on the server

    - Consider making a backup of the file before editing, perhaps using the date in the filename: `iptables-save > ~/iptables.rules.bak.$(date '+%Y%m%d')`
    - Note that `iptables -S` can be used to print the current rules in the terminal
    - Copy the current firewall rules to a file for editing: `iptables-save > ~/iptables.editing`
    - Edit the file to add a new port-forward line with a similar format to the other lines. It may look similar to `-A INPUT -p udp -m state --state NEW -m udp --dport 51840 -j ACCEPT`. Note that order matters, so the new line needs to be added not at the bottom but alongside the other port-forward lines. It's essentially accepting incoming UDP traffic on port `51841`
    - Once the file is edited, apply it and actually make it active with `iptables-restore < ~/iptables.editing`
    - Save it permanently with `iptables-save > /etc/iptables.rules` to allow the rule to persist between reboots or service restarts. Without this, the new rule may no longer be present (and prevent WireGuard from working) hours or days down the line. This can be checked with `iptables -S`
    - The below example shows the single line alongside other pre-existing lines:

```
-A INPUT -p udp -m state --state NEW -m udp --dport 51840 -j ACCEPT
-A INPUT -p udp -m state --state NEW -m udp --dport 51841 -j ACCEPT
-A INPUT -p tcp -m state --state NEW -m tcp --dport 5201 -j ACCEPT
-A INPUT -p tcp -m state --state NEW -m tcp --dport 5001 -j ACCEPT
-A INPUT -p udp -m udp --dport 33434:33474 -j REJECT --reject-with icmp-port-unreachable
-A INPUT -p icmp -m icmp --icmp-type 8 -j ACCEPT
-A INPUT -j DROP
-A FORWARD -p tcp -m tcp --tcp-flags SYN,RST SYN -j TCPMSS --clamp-mss-to-pmtu
-A FORWARD -s 10.0.0.0/8 -j ACCEPT
-A FORWARD -j ACCEPT
COMMIT
```

- Then bring the new network interface online with `ifup wg666`
- Then reload Bird with the new configuration with `systemctl reload bird`
- At this point, the configuration can be validated and if the Omni side of the WireGuard tunnel is active, some stats can be seen
    - `ifconfig wg666` will show the newly created interface, its IP, traffic quantity
    - `wg show wg666` will show the WireGuard configuration with its listening port, public key, last handshake
    - `birdc show interfaces summary` will show the currently running bird configuration

Resources

- IPRanges spreadsheet, limited access https://docs.google.com/spreadsheets/d/1PKakX4vZyZ7iBO3VtwchJ4vwptBruuzNJIVy92cn2TY/edit?gid=0#gid=0
- WireGuard vanity address generator https://github.com/warner/wireguard-vanity-address
- A good read on WireGuard configuration values https://forum.netgate.com/topic/184254/setup-docs-incomplete-for-wireguard-confused-about-terms-having-a-challenging-time-setting-up-wireguard-read-here
- Helpful Slack thread on the setup process, thanks Daniel https://nycmesh.slack.com/archives/C01G52U577X/p1723939020701829?thread_ts=1723922015.029239&cid=C01G52U577X
- Background info on the wiki on the Mesh infrastructure that allows OSPF routing over WireGuard https://wiki.nycmesh.net/link/104
- Preexisting Wiki page for VPN setup https://wiki.nycmesh.net/link/103
- Piotr pfSense setup https://nycmesh.slack.com/archives/C679UKBUK/p1677281265233999?thread_ts=1677281002.296399&cid=C679UKBUK
- Netgate pfSense site to site WireGuard guide https://docs.netgate.com/pfsense/en/latest/recipes/wireguard-s2s.html
- More info on the usage of Keepalive https://www.wireguard.com/quickstart/#nat-and-firewall-traversal-persistence

Revision #7
Created 30 September 2024 18:32:44 by johnb
Updated 30 October 2024 19:12:13 by johnb