# Mission (im)possible

## Switch setup

Enable SSH on the switch:

ena

conf t

hostname switch-pieter

ip domain-name pieter\_netwerk

crypto key generate rsa

1048

ip ssh version 2

line vty 1 10

transport input ssh

login local

username admin password t

aaa new-model

python script to generate new config:

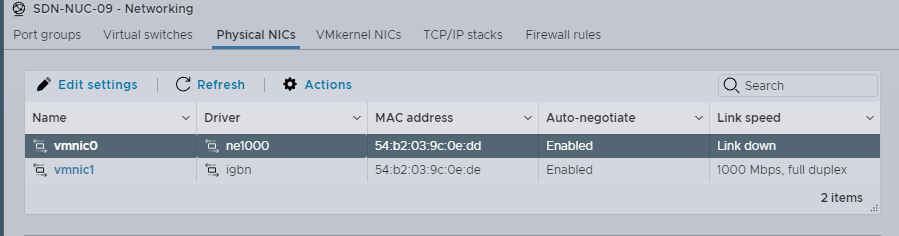
import ipaddress  
  
# Define the parent subnet (10.0.0.0/8)  
parent\_subnet = ipaddress.IPv4Network('10.0.0.0/8')  
  
# Calculate the subnet prefix length (/28)  
subnet\_prefix\_length = 28  
  
# Calculate the number of subnets to create  
num\_subnets = 200  
  
# Calculate the size of each subnet  
subnet\_size = 2 \*\* (32 - subnet\_prefix\_length)  
  
# Open a file for writing the configuration  
with open("switch\_config.txt", "w") as config\_file:  
 # Write the initial configuration  
 config\_file.write("configure terminal\n")  
  
  
 # Check if there are enough addresses in the parent subnet  
 if num\_subnets \* subnet\_size > parent\_subnet.num\_addresses:  
 print("Error: Not enough addresses in the parent subnet to create 200 /28 subnets.")  
 else:  
 subnets = list(parent\_subnet.subnets(new\_prefix=subnet\_prefix\_length))  
  
 for i, subnet in enumerate(subnets[:num\_subnets]):  
 vlan\_name = f"VLAN{i+2}"  
 vlan\_network = str(subnet.network\_address)  
 vlan\_ip = str(subnet.network\_address + 1)  
  
 config\_file.write(f"vlan {i + 2}\n")  
 config\_file.write(f"name {vlan\_name}\n")  
 config\_file.write(f"interface vlan {i+2}\n")  
 config\_file.write(f"ip address {vlan\_ip} 255.255.255.240\n")  
 config\_file.write("!\n")  
  
 config\_file.write("end\n")  
  
print("Configuration file generated: switch\_config.txt")

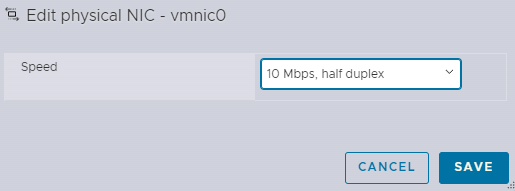
This generates a file called switch\_config.txt

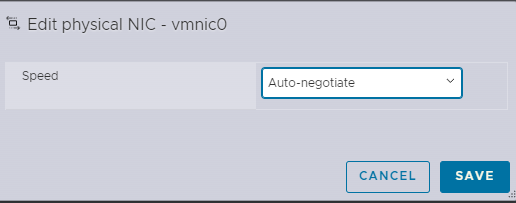
You can ssh into switch and copy-paste it.

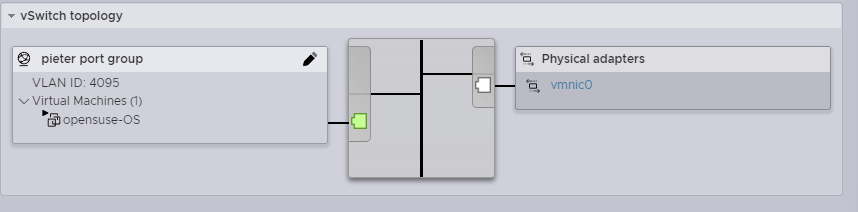
## VMWare

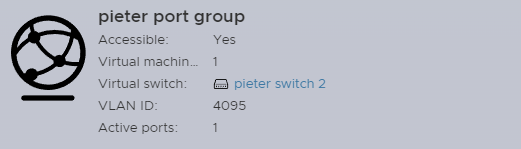
On Vmware there needs to be a setup for the second physical NIC.

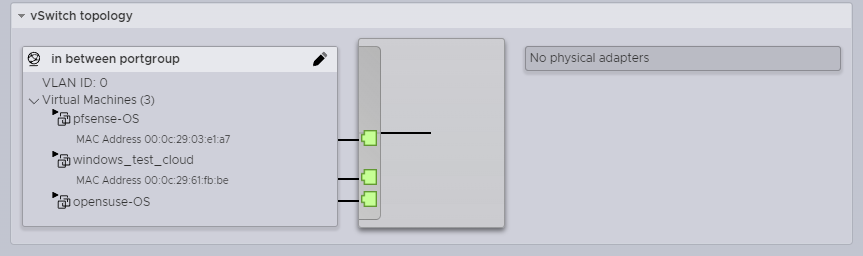
To enable the link:

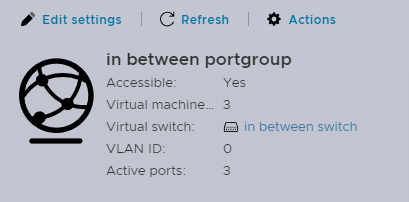
And then back to auto negotiate:

Create new virtual switch and attack the vmnic0 to it, also give the switch vlan 4095 so it will allow all vlans.

New port group with the virtual switch:

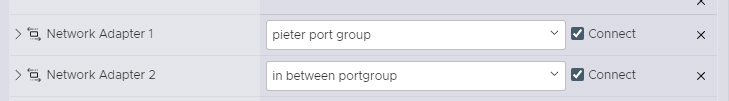
Create a new switch for the inbetween network that has no physical nics.

Also create a new portgroup for this switch (there are no vlans on the in between network so that doesnt matter).



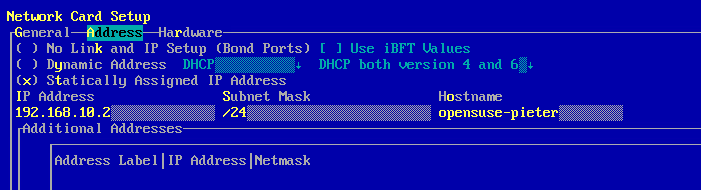
## Opensuse configuration

Connect the opensuse both to the new switch and the in between switch:

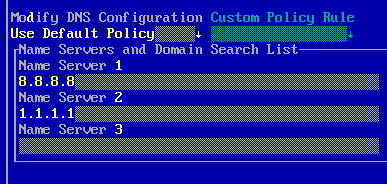


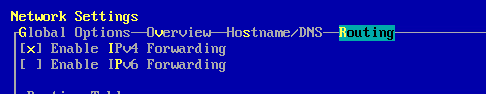
### Network connection

Set a static IP to eth1:

default gateway:

dns servers:

Ip forwarding:

eth0 does not need an ip as all the vlans are configured on there.

### Configure vlans

Script that generates vlans and gives each of them an address:

import ipaddress  
  
# Define the parent subnet (10.0.0.0/8)  
parent\_subnet = ipaddress.IPv4Network('10.0.0.0/8')  
  
# Calculate the subnet prefix length (/28)  
subnet\_prefix\_length = 28  
  
# Calculate the number of subnets to create  
num\_subnets = 200  
  
# Calculate the size of each subnet  
subnet\_size = 2 \*\* (32 - subnet\_prefix\_length)  
  
# Open a file for writing the configuration  
with open("opensuse-vlan.txt", "w") as config\_file:  
  
 # Check if there are enough addresses in the parent subnet  
 if num\_subnets \* subnet\_size > parent\_subnet.num\_addresses:  
 print("Error: Not enough addresses in the parent subnet to create 200 /28 subnets.")  
 else:  
 subnets = list(parent\_subnet.subnets(new\_prefix=subnet\_prefix\_length))  
  
 for i, subnet in enumerate(subnets[:num\_subnets]):  
 outputfile = f"ifcfg-vlan{i+2}"  
 vlan\_network = str(subnet.network\_address)  
 vlan\_ip = str(subnet.network\_address + 2)  
  
 config\_file.write(f"echo IPADDR=\\'{vlan\_ip}\\' >> {outputfile} \n")  
 config\_file.write(f"echo BOOTPROTO=\\'static\\' >> {outputfile} \n")  
 config\_file.write(f"echo STARTMODE=\\'hotplug\\' >> {outputfile} \n")  
 config\_file.write(f"echo NETMASK=\\'255.255.255.240\\' >> {outputfile} \n")  
 config\_file.write(f"echo ZONE=public >> {outputfile} \n")  
 config\_file.write(f"echo VLAN=\\'yes\\' >> {outputfile} \n")  
 config\_file.write(f"echo ETHERDEVICE=\\'eth0\\' >> {outputfile} \n")  
  
print("Configuration file generated: opensuse-vlan.txt")

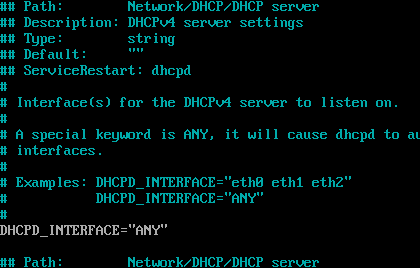
This creates a bash file, put it in /etc/sysconfig/network and run it.

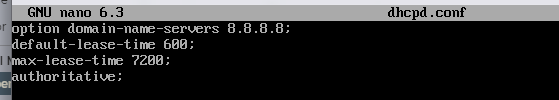
Then restart the network config. All the vlans should appear.

### DHCP config

Install dhcp through yast.

Configure the /etc/sysconfig/dhcpd file to allow any network interface:

Edit the dhcpd.conf file so it has a dns server:

Script that creates a script to append dhcp ranges to this file:

import ipaddress  
  
# Define the parent subnet (10.0.0.0/8)  
parent\_subnet = ipaddress.IPv4Network('10.0.0.0/8')  
  
# Calculate the subnet prefix length (/28)  
subnet\_prefix\_length = 28  
  
# Calculate the number of subnets to create  
num\_subnets = 200  
  
# Calculate the size of each subnet  
subnet\_size = 2 \*\* (32 - subnet\_prefix\_length)  
  
# Open a file for writing the configuration  
with open("dhcp\_config.txt", "w") as config\_file:  
 # Check if there are enough addresses in the parent subnet  
 if num\_subnets \* subnet\_size > parent\_subnet.num\_addresses:  
 print("Error: Not enough addresses in the parent subnet to create 200 /28 subnets.")  
 else:  
 subnets = list(parent\_subnet.subnets(new\_prefix=subnet\_prefix\_length))  
  
 for i, subnet in enumerate(subnets[:num\_subnets]):  
 vlan\_name = f"VLAN{i+2}"  
 vlan\_network = str(subnet.network\_address)  
 vlan\_default = str(subnet.network\_address + 2)  
 vlan\_ip\_start = str(subnet.network\_address + 3)  
 vlan\_ip\_end = str(subnet.network\_address + 14)  
  
 config\_file.write(f"echo subnet {vlan\_network} netmask 255.255.255.240 '{{' >> /etc/dhcpd.conf \n")  
 config\_file.write(f"echo \" range dynamic-bootp {vlan\_ip\_start} {vlan\_ip\_end};\" >> /etc/dhcpd.conf\n")  
 config\_file.write(f"echo \" option routers {vlan\_default};\" >> /etc/dhcpd.conf\n")  
 config\_file.write("echo } >> /etc/dhcpd.conf\n")  
  
  
print("Configuration file generated: dhcp\_config.txt")

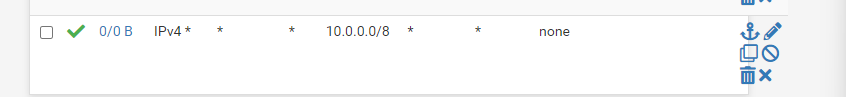
## Pfsense config:

Configure wan with static ip, set gateway and dns.

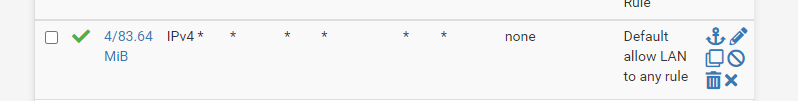
Wan rules:

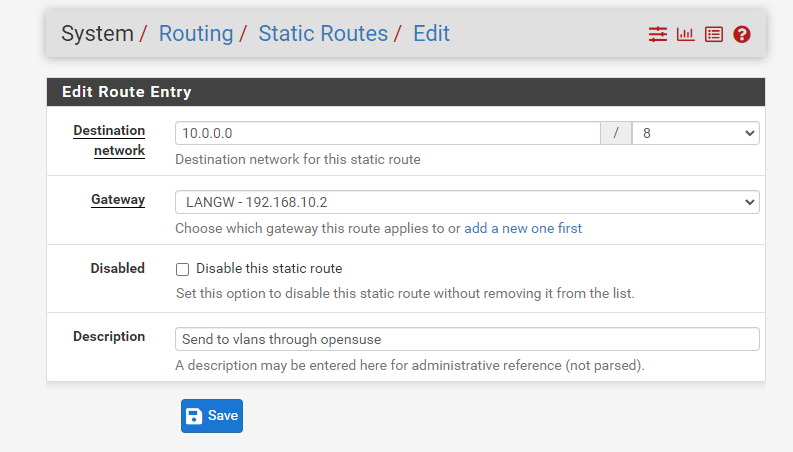
Allow WAN interface access:

Allow 10.0.0.0/8 traffic to return:

Lan rules:

change allow Ipv4 LAN net to any:

Static route to LAN:



Now if you connect to the switch through a port with an attached vlan, it will give your pc a dhcp address, dns and network access.

## Network schema

