N2N bridge Setup via Ansible

Beschreibung:

Hier ein script mit dem man relativ einfach ein setup aufbauen kann.
Einfach nur die hosts.ini richtig ausfüllen.
Diese kann um unendliche clients erweitert werden.
Wichtig ist nur das für jeden client eine eigene Gruppe erstellt wird, weil Gruppen Variablen nutzen um so flexibler zu sein.

Installation:

Es muss auf den hosts zugriff ber Schlüssel von nöten sein.
Die Netzwerkinterfaces ip-adressen gateways etc müssen bekannt sein und dem entsprechen in der hosts ini abgelegt sein.

Die host.ini

```
[client1]
peer1 ansible_host=167.235.xxx.xxx
[client2]
peer2 ansible_host=192.168.178.138

[all:vars]
community=layer2
password=ultrageheimeskennwort
supernode_port=5555
supernode_ip=167.235.xxx.xxx
ansible_user=root
debug=false

[client1:vars]
delete_cloud_init=true
bridgeaddress=167.235.xx.xxx
netmask=255.255.255.255
gateway=172.31.1.1
dns1=8.8.8.8
dns2=8.8.4.4
main_bridge_interface=eth0
```

```
n2n_ip=10.10.2.1
supernode=true
# Optional: Für die zweite Netzwerkkarte
secondary_interface=
secondary_ip=
secondary_netmask=
secondary_gateway=

[client2:vars]
delete_cloud_init=false
bridgeaddress=
netmask=
gateway=
dns1=8.8.8.8
dns2=8.8.4.4
main_bridge_interface=enp6s20
n2n_ip=10.10.2.2
supernode=false
# Optional: Für die zweite Netzwerkkarte

secondary_interface=enp6s18
secondary_ip=192.168.178.138
secondary_netmask=255.255.255.0
secondary_gateway=192.168.178.1
```

Kurze Erläuterung:

Die Variable delete_cloud_init besagt wenn die true ist, wird die cloud init Netzwerkkonfig gelöscht und deaktiviert.
Hier ist sie auf Hetzner VPS ausgelegt, ob die Konfig bei anderen VPS anbietern auch so ist, weiß ich nicht

Das main_bridge_interface sagt aus welche pysikalisch (bei ner vm die Netzwerkkarte) an die Bridge dran soll.
Ist eine zweite Netzwerkkarte ausgefüllt, so bekommt die bridge keine ip selbst wenn dieses ausgefüllt ist.
Soll keine zweite Karte genutzt werden, dann die variablen leer lassen.
Auch für die zweite Karte werden die DNS server verwendet die oben angeben sind, also ausfüllen bitte.

Die n2n_ip ist die interne ip, selbst wenn bei brücken keine interne ip von nöten ist, ist sie ein Pflichtfeld.

Möchten wir einen dritten client haben einfach einen client:vars abschnitt kopieren und einfügen und dann eine 3 draus machen.

in der Regel möchte man bei den anderen Clients immer eine separate Netzwerkkarte haben. Nur beim VPS-Server hat man in der regel nur eine.

Beispiel:

```
[client3:vars]
delete_cloud_init=false
bridgeaddress=
netmask=
gateway=
dns1=8.8.8.8
dns2=8.8.4.4
main_bridge_interface=enp6s20
n2n_ip=10.10.2.3
supernode=false
# Optional: Für die zweite Netzwerkkarte

secondary_interface=enp6s18
secondary_ip=192.168.178.139
secondary_netmask=255.255.255.0
secondary_gateway=192.168.178.1
```

Nun das template file

interfaces.j2

```
auto {{ main_bridge_interface }}
iface {{ main_bridge_interface }} inet manual

auto br0
iface br0 inet {% if secondary_interface %}manual{% else %}static{% endif %}

{% if not secondary_interface %}

    address {{ bridgeaddress }}
    netmask {{ netmask }}
    gateway {{ gateway }}
    dns-nameservers {{ dns1 }} {{ dns2 }}
```

```
{% endif %}

    bridge_ports {{ main_bridge_interface }}
    bridge_stp off
    bridge_fd 0
    bridge_maxwait 0
    {% if supernode %}

    post-up nohup /usr/bin/supernode -l {{ supernode_port }} > /var/log/supernode.log 2>&1 &
    pre-down /usr/bin/pkill supernode
    {% endif %}

    post-up /usr/sbin/edge -r -d n2n0 -c "{{ community }}" -k "{{ password }}" -a {{ n2n_ip }} -l "{{
supernode_ip }}:{{ supernode_port }}" -f > /var/log/n2n_edge
    post-up /bin/sh -c "brctl addif br0 n2n0"
    pre-down /bin/sh -c "brctl delif br0 n2n0"
    pre-down /usr/bin/pkill edge

{% if secondary_interface %}

auto {{ secondary_interface }}
iface {{ secondary_interface }} inet static
    address {{ secondary_ip }}
    netmask {{ secondary_netmask }}
    gateway {{ secondary_gateway }}
    dns-nameservers {{ dns1 }} {{ dns2 }}
{% endif %}
```

Und zum schluss das Playbook

```
---
- hosts: all
  become: yes
  tasks:
    - name: Zeige alle Variablen
      debug:
        var: hostvars[inventory_hostname]
      when: debug | default(false) | bool

    - name: Aktualisiere APT-Repository
```

```
    apt:
      update_cache: yes

  - name: Installiere n2n Paket
    apt:
      name: n2n
      state: present

  - name: Installiere bridge-utils Paket
    apt:
      name: bridge-utils
      state: present

  - name: Entferne cloud-init Netzwerkkonfiguration auf Debian
    file:
      path: /etc/network/interfaces.d/50-cloud-init.cfg
      state: absent
    when: delete_cloud_init | bool

  - name: Erstelle 99-cloud-init-disable-Datei
    copy:
      dest: /etc/cloud/cloud.cfg.d/99-disable-network-config.cfg
      content: "network: {config: disabled}\n"
    when: delete_cloud_init | bool



  - name: Template the interfaces file
    template:
      src: interfaces.j2
      dest: /etc/network/interfaces
    notify:
      - restart network

handlers:
  - name: restart network
    service:
      name: networking
      state: restarted
```

Fertig.
nun kann mittels, das playbook gestartet werden und ausgerollt werden

```
ansible-playbook -i hosts.ini setup_n2n.yml
```

Hier Dateien zum download:

interfaces.j2

setup_n2n.yml

Die Host.ini oben aus dem text kopieren.

Version #1
Erstellt: 29 Oktober 2023 14:40:17 von Admin
Zuletzt aktualisiert: 29 Oktober 2023 14:54:58 von Admin