Репозиторий на GitHub: https://github.com/pashokitsme/net-tech-2s

Этот репозиторий, как и этот отчёт - второй этап braim.org/network-contest

Изначально оформлено в .md и предполагалось к просмотру в README репозитория.

В ходе выполнения использовался пакетный (проектный) менеджер uv. Поэтому для вызова команд используется uv run <command>.

В качестве машин использовались три docker-контейнера с базовым образом alt linux, а также установленным openssh и подготовленными ключами.

Контроллер доменов не был реализован (соответственно, и ввод клиентов в него)

TL;DR

> `uv` заменить на необходимый пакетный менеджер

```
git clone https://github.com/pashokitsme/net-tech-2s
cd net-tech-2s

docker compose up -d
uv run ansible-playbook -i inventory.yaml playbooks/install.yaml
```

Особенности реализации из-за docker-контейнеров

1. Всё, что связано с systemd - не работает, поскольку сам systemd не инициализируется в контейнерах. Из-за этого в плейбуке не используется `ansible.builtin.service` для управления сервисами, процессы убиваются и запускаются вручную
2. Традиционно (`bridge`/`host` сети) контейнеры запускаются с заранее заданным ip. Поэтому для работы dhcp-сервера была использована сеть `macvlan`. Кроме неё, используется `bridge` для доступа к контейнерам с хост машины по сети - иначе бы пришлось настраивать `macvlan` и на хосте тоже
3. В docker compose версии `v2.34.0-desktop.1` не работает опция `gw_priority` ((https://github.com/docker/compose/issues/12574)), поэтому нужный интерфейс ищется по-месту - иногда он может быть eth0, а иногда eth1

Окружение

Dockerfile

Предполагается, что на целевой машине уже установлен SSH-сервер и указаны ключи для доступа

```
FROM alt:latest

RUN apt-get update
RUN apt-get install -y openssh-server

RUN ssh-keygen -A

RUN sed -i 's/#PermitRootLogin prohibit-password/PermitRootLogin yes/' /etc/openssh/sshd_config
&& \
  sed -i 's/#PubkeyAuthentication/PubkeyAuthentication/' /etc/openssh/sshd_config && \
  printf 'AuthorizedKeysFile %s\n' /root/.ssh/authorized_keys >> /etc/openssh/sshd_config

CMD ["/usr/sbin/sshd", "-D"]
```

docker compose

Разворачивает минимальное окружение, в котором будет происходить выполнение playbook

```
x-networks: &networks
```

```
    - net-internal
    - net-external

services:
  srv01:
    build: .
    container_name: srv01
    hostname: srv01
    volumes:
      - ./keys/srv01.pub:/root/.ssh/authorized_keys
    cap_add:
      - NET_ADMIN
    ports:
      - 6001:22
    networks: *networks

  client01:
    build: .
    container_name: client01
    hostname: client01
    volumes:
      - ./keys/client01.pub:/root/.ssh/authorized_keys
    cap_add:
      - NET_ADMIN
    ports:
      - 6002:22
    networks: *networks

  client02:
    build: .
    container_name: client02
    hostname: client02
    volumes:
      - ./keys/client02.pub:/root/.ssh/authorized_keys
    cap_add:
      - NET_ADMIN
    ports:
      - 6003:22
    networks: *networks

networks:
  net-internal:
    driver: macvlan
    driver_opts:
      parent: eth0
    ipam:
      config:
        - subnet: 10.10.0.0/24
          gateway: 10.10.0.1

  net-external:
    driver: bridge
    ipam:
      config:
        - subnet: 10.100.0.0/24
          gateway: 10.100.0.1
```

inventory.yaml

Из-за того, что все машины - контейнеры, хост всегда `127.0.0.1`, а порты `600*`

```
all:
  vars:
    ansible_python_interpreter: /usr/bin/python3
    ansible_user: root
    ansible_host_key_checking: false

  children:
    servers:
      hosts:
        srv01:
          ansible_host: 127.0.0.1
          ansible_port: 6001
          ansible_ssh_private_key_file: ./keys/srv01

    clients:
      vars:
        ansible_host: 127.0.0.1

      hosts:
```

```
        client01:
          ansible_port: 6002
          ansible_ssh_private_key_file: ./keys/client01

        client02:
          ansible_port: 6003
          ansible_ssh_private_key_file: ./keys/client02
```

Playbook

В этой главе освещаны не все задачи этого плейбука, а лишь главное.

Полностью его посмотреть можно в репозитории.

Установка Python

Изначально ни на одной машине нет Python. Следовательно, Ansible не сможет полноценно выполнять функции. Этот Play выводит версию Python и устанавливает в случае его отсутствия.

```
- name: Install python
  hosts: all
  become: true
  gather_facts: false
  vars:
    python_version: 3
    executable: /bin/bash

  pre_tasks:
    - name: Check if python is installed
      ansible.builtin.raw: python{{ python_version }} --version
      args:
        executable: '{{ executable }}'
      register: python_installed
      failed_when: false
      changed_when: false

    - name: Update package repository
      ansible.builtin.raw: apt-get update
      args:
        executable: '{{ executable }}'
      when: 'python_installed.rc'
      changed_when: 'python_installed.rc'

    - name: Install python{{ python_version }}
      ansible.builtin.raw: apt-get install -y python{{ python_version }}
      args:
        executable: '{{ executable }}'
      when: 'python_installed.rc'
      changed_when: 'python_installed.rc'

  tasks:
    - name: Gather installed python version
      ansible.builtin.raw: python{{ python_version }} --version
      args:
        executable: '{{ executable }}'
      register: result
      changed_when: false

    - name: Display python version
      ansible.builtin.debug:
        msg: '{{ result.stdout.strip() }}'
```

Установка web-сервера

Используется Angie - идейный форк Nginx, поскольку он разрабатывается российскими разработчиками (что требовала задача - российский софт).

Устанавливается только если не был уже установлен. После этого тестируется конфиг, запускается и отправляется http запрос на порт 80 для проверки.

```
- name: Setup web-server
  hosts: servers
  become: true
  gather_facts: false
```

```
  tasks:
    - name: Check if Angie is installed
      ansible.builtin.command:
        cmd: angie -v
      register: angie_installed
      failed_when: false
      changed_when: false

    - name: Download Angie keys & add repository
      ansible.builtin.shell:
        cmd: # (... сокращено, скрипт с доки)
      when: angie_installed.rc
      changed_when: angie_installed.rc

    - name: Install packages
      ansible.builtin.package:
        name:
          - angie
        update_cache: true
        state: present_not_latest
      when: angie_installed.rc

    - name: Test Angie conf
      ansible.builtin.command:
        cmd: angie -t
      register: angie_conf
      changed_when: false

    - name: Stop Angie
      ansible.builtin.shell:
        cmd: |
          if [ -f /run/angie.pid ]; then
            angie -s stop
            exit 0
          fi
          exit 1
      register: angie_stop
      when: angie_conf.rc == 0
      changed_when: angie_stop.rc == 0
      failed_when: angie_stop.rc > 1

    - name: Start Angie
      ansible.builtin.command:
        cmd: angie
      when: angie_conf.rc == 0
      changed_when: angie_conf.rc == 0

    - name: Test Angie
      ansible.builtin.uri:
        url: http://localhost:80
        method: GET
      changed_when: false
```

DHCP-сервер

Устанавливается дефолтный dhcp-сервер из пакетного реестра alt linux. Конфиг берётся из `playbooks/templates/dhcpd.conf`

```
authoritative;
default-lease-time 600;
max-lease-time 7200;

subnet 10.10.0.0 netmask 255.255.255.0 {
    range 10.10.0.110 10.10.0.120;
    option routers 10.10.0.1;
    option domain-name-servers 10.10.0.2;
    option domain-name "internal";
}
```

```
- name: Install DHCP server
  hosts: servers
  gather_facts: false

  tasks:
    - name: Install DHCP server
      ansible.builtin.package:
        name:
          - dhcp-server
```

```
        - dhcp-client
      state: present_not_latest

  - name: Stop DHCP server
    ansible.builtin.command:
      cmd: pkill -F /run/dhcpd.pid
    register: dhcpd_stop
    failed_when: dhcpd_stop.rc > 1
    changed_when: dhcpd_stop.rc == 0

  - name: Configure DHCP server
    ansible.builtin.template:
      src: dhcpd.conf
      dest: /etc/dhcp/dhcpd.conf
      owner: root
      group: root
      mode: '0640'

  - name: Test DHCP server conf
    ansible.builtin.command:
      cmd: dhcpd -t
    become: true
    register: dhcpd_conf
    changed_when: false

  - name: Start DHCP server
    ansible.builtin.command:
      cmd: dhcpd
    when: dhcpd_conf.rc == 0
    changed_when: dhcpd_conf.rc == 0

  - name: Assign static IP to server
    ansible.builtin.shell:
      cmd: |
        set -o pipefail
        INTERFACE=$(ip -o addr show | awk '/10.10.0.*brd 10.10.0.255/{print $2}')
        ip addr flush dev $INTERFACE
        ip addr add 10.10.0.2/24 dev $INTERFACE
        ip link set $INTERFACE up
    async: 1
    poll: 0
    changed_when: true
```

DHCP-клиент

```
- name: Install DHCP client
  hosts: clients
  gather_facts: false
  tasks:
    - name: Install DHCP client
      ansible.builtin.package:
        name:
          - dhcp-client
        state: present_not_latest

    - name: Retrieve interface name
    # (...сокращено)

    - name: Set interface fact
    # (...сокращено)

    # selected_interface = eth0 или eth1

    - name: Display interface
      ansible.builtin.debug:
        msg: 'Selected interface: {{ selected_interface }}'

    - name: 'Release addresses: {{ selected_interface }}'
      ansible.builtin.shell:
        cmd: |
          set -o pipefail
          dhclient -r {{ selected_interface }}
          ip addr flush dev {{ selected_interface }}
      changed_when: true

    - name: Run DHCP client
      ansible.builtin.command:
        cmd: dhclient {{ selected_interface }}
      changed_when: true

    - name: Gather claimed IP addresses
```

```
      ansible.builtin.shell:
        cmd: |
          set -o pipefail
          ip -br -f inet addr show {{ selected_interface }} | awk '{print $3}'
      register: claimed_addresses
      changed_when: false

    - name: Display claimed addresses
      ansible.builtin.debug:
        msg: 'Claimed addresses: {{ claimed_addresses.stdout.strip() }}'
```

DNS-сервер

В качестве dns-сервера используется bind.

Файлы с зонами, конфигурациями и так далее всё так же расположены в `playbooks/templates/`.

Зона .internal - db.internal

```
$TTL 86400
@       IN      SOA     ns.internal. admin.internal. (
                        2023040101      ; Serial
                        3600            ; Refresh
                        1800            ; Retry
                        604800          ; Expire
                        86400 )         ; Minimum TTL

; Name servers
@       IN      NS      ns.internal.

; A records
@       IN      A       10.10.0.2
ns      IN      A       10.10.0.2
server  IN      CNAME   ns.internal.
```

Реверс-зона db.10.10.rev

```
$TTL 86400
@       IN      SOA     ns.internal. admin.internal. (
                        2023040101      ; Serial
                        3600            ; Refresh
                        1800            ; Retry
                        604800          ; Expire
                        86400 )         ; Minimum TTL

; Name servers
@       IN      NS      ns.internal.

; PTR records
2.0     IN      PTR     ns.internal.
```

named.conf

```
options {
    directory "/etc/bind";
    allow-query { any; };
    forwarders { 1.1.1.1; };
    recursion yes;
    dnssec-validation no;
};

zone "internal" {
    type master;
    file "/etc/bind/zones/db.internal";
    allow-transfer { none; };
};

zone "10.10.in-addr.arpa" {
    type master;
    file "/etc/bind/zones/db.10.10.rev";
    allow-transfer { none; };
};
```

resolv.conf

Для клиентов устанавливается следующий `resolv.conf`

```
nameserver 10.10.0.2
```

Play

```yaml
# Сервер
- name: Install DNS server
  hosts: servers
  gather_facts: false

  tasks:
    - name: Install DNS server
      ansible.builtin.package:
        name:
          - bind
        state: present_not_latest

    - name: Create bind user
      ansible.builtin.user:
        name: bind
        state: present

    - name: Create bind directories
      ansible.builtin.file:
        path: '{{ item }}'
        state: directory
        owner: root
        group: bind
        mode: '0750'
      loop:
        - /etc/bind
        - /etc/bind/zones

    - name: Configure DNS server
      ansible.builtin.template:
        src: '{{ item.from }}'
        dest: '{{ item.to }}'
        owner: root
        group: bind
        mode: '0750'
      loop:
        - from: named.conf
          to: /etc/bind/named.conf
        - from: db.internal
          to: /etc/bind/zones/db.internal
        - from: db.10.10.rev
          to: /etc/bind/zones/db.10.10.rev

    - name: Test DNS server conf
      ansible.builtin.command:
        cmd: named-checkconf /etc/bind/named.conf
      register: named_conf
      changed_when: false

    - name: Start DNS server
      ansible.builtin.command:
        cmd: named
      register: named_start
      when: named_conf.rc == 0
      changed_when: named_conf.rc == 0
      failed_when: named_start.rc != 0

# Клиенты
- name: Configure DNS client
  hosts: clients
  gather_facts: false

  tasks:
    - name: Copy resolv.conf
      ansible.builtin.template:
        src: resolv.conf
        dest: /tmp/resolv.conf
        owner: root
        group: root
        mode: '0640'

    - name: Configure resolv.conf
```

```
      ansible.builtin.shell:
        cmd: |
          set -o pipefail
          cat /tmp/resolv.conf > /etc/resolv.conf
          rm /tmp/resolv.conf
      changed_when: true

- name: Test DNS server
  hosts: clients
  gather_facts: false

  tasks:
    - name: Test DNS server
      ansible.builtin.command:
        cmd: nslookup {{ item.host }} {{ item.dns }}
      loop:
        - host: ya.ru
          dns: 8.8.8.8
        - host: server.internal
          dns: 10.10.0.2
        - host: server.internal
          dns: ''
      changed_when: false
      register: nslookup_result

    - name: Display nslookup result
      ansible.builtin.debug:
        msg: "{{ nslookup_result.results | map(attribute='stdout') | list }}"
```

По результатам установки проводится тестирование: клиенты должны иметь возможность получить ip-адрес сервера по доменному имени `server.internal`

Межсетевой экран

Следующая конфигурация iptables разрешает только некоторый трафик. Сконфигурирован под работу с AD, но AD не был реализован.

Конфигурация

```
*filter
:INPUT ACCEPT [0:0]
:FORWARD ACCEPT [0:0]
:OUTPUT ACCEPT [0:0]

-A INPUT -i lo -j ACCEPT

-A INPUT -p icmp -j ACCEPT
-A INPUT -m state --state ESTABLISHED,RELATED -j ACCEPT
-A INPUT -p tcp --dport 22 -j ACCEPT

# http
-A INPUT -p tcp --dport 80 -j ACCEPT
-A INPUT -p tcp --dport 443 -j ACCEPT

# dns
-A INPUT -p tcp --dport 53 -j ACCEPT
-A INPUT -p udp --dport 53 -j ACCEPT

# kerberos
-A INPUT -p tcp --dport 88 -j ACCEPT
-A INPUT -p udp --dport 88 -j ACCEPT
-A INPUT -p udp --dport 123 -j ACCEPT

# rpc endpoint mapper
-A INPUT -p tcp --dport 135 -j ACCEPT

# netbios name service
-A INPUT -p udp --dport 137 -j ACCEPT
-A INPUT -p tcp --dport 137 -j ACCEPT

# netbios datagram service
-A INPUT -p udp --dport 138 -j ACCEPT

# netbios session service
-A INPUT -p tcp --dport 139 -j ACCEPT

# ldap
-A INPUT -p tcp --dport 389 -j ACCEPT
-A INPUT -p udp --dport 389 -j ACCEPT
```

```
# smb over tcp
-A INPUT -p tcp --dport 445 -j ACCEPT

# kerberos password change
-A INPUT -p tcp --dport 464 -j ACCEPT
-A INPUT -p udp --dport 464 -j ACCEPT

# ldap ssl
-A INPUT -p tcp --dport 636 -j ACCEPT

# global catalog
-A INPUT -p tcp --dport 3268 -j ACCEPT
-A INPUT -p tcp --dport 3269 -j ACCEPT

# dynamic rpc ports
-A INPUT -p tcp --dport 49152:65535 -j ACCEPT

# default drop all other incoming connections
-A INPUT -j DROP
COMMIT
```

Play

```
- name: Configure iptables
  hosts: all
  become: true
  gather_facts: false

  tasks:
    - name: Install iptables
      ansible.builtin.package:
        name:
          - iptables
        state: present_not_latest

    - name: Configure iptables
      ansible.builtin.template:
        src: iptables
        dest: /etc/sysconfig/iptables
        owner: root
        group: root
        mode: '0640'

    - name: Apply iptables
      ansible.builtin.command:
        cmd: iptables-restore /etc/sysconfig/iptables
      changed_when: true
```

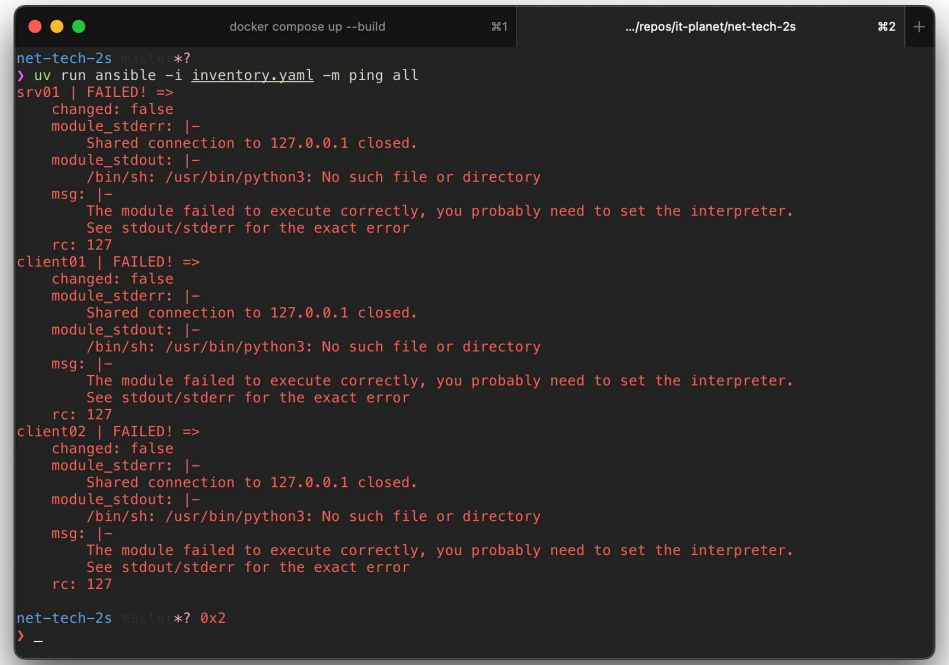
Выполнение

```
docker compose up
```

```
net-tech-2s master*?
❯ docker compose up --build
[+] Building 0.3s (15/15) FINISHED
 => [internal] load local bake definitions                                                0.0s
 => => reading from stdin 1.09kB                                                          0.0s
 => [srv01 internal] load build definition from Dockerfile                                0.0s
 => => transferring dockerfile: 478B                                                      0.0s
 => [srv01 internal] load metadata for docker.io/library/alt:latest                       0.0s
 => [client02 internal] load .dockerignore                                                0.0s
 => => transferring context: 2B                                                           0.0s
 => [client02 1/5] FROM docker.io/library/alt:latest@sha256:7e4727efeb6cbf10575622256de51852278cb  0.0s
 => => resolve docker.io/library/alt:latest@sha256:7e4727efeb6cbf10575622256de51852278cb41b5fe6a7  0.0s
 => CACHED [srv01 2/5] RUN apt-get update                                                 0.0s
 => CACHED [srv01 3/5] RUN apt-get install -y openssh-server                              0.0s
 => CACHED [srv01 4/5] RUN ssh-keygen -A                                                  0.0s
 => CACHED [srv01 5/5] RUN sed -i 's/#PermitRootLogin prohibit-password/PermitRootLogin yes/' /et  0.0s
 => [client01] exporting to image                                                         0.0s
 => => exporting layers                                                                   0.0s
 => => exporting manifest sha256:14840ee86188a3526586893c100c78111ce19e333e86489415964eef417b3c34  0.0s
 => => exporting config sha256:b4fdedeee4978c38768bae406a4a0da6355f612dbfef6e58170d3b60d2bb3433    0.0s
 => => exporting attestation manifest sha256:fa98d4ce1ea6f41591333027715eb9cbf6f7e03a3d0abacd760f  0.0s
 => => exporting manifest list sha256:c87504f20d72f266832e2b55f0d2dc6da51779307d3ca932ad6daaf2e14  0.0s
 => => naming to docker.io/library/net-tech-2s-client01:latest                            0.0s
 => => unpacking to docker.io/library/net-tech-2s-client01:latest                         0.0s
 => [client02] exporting to image                                                         0.0s
 => => exporting layers                                                                   0.0s
 => => exporting manifest sha256:14840ee86188a3526586893c100c78111ce19e333e86489415964eef417b3c34  0.0s
 => => exporting config sha256:b4fdedeee4978c38768bae406a4a0da6355f612dbfef6e58170d3b60d2bb3433    0.0s
 => => exporting attestation manifest sha256:cf3bce7aa59c5241840817fab78df70c1d1d786dfc0abed3ba87  0.0s
 => => exporting manifest list sha256:b0c2637b210b78686c2af8ecf8875706763dd3bc4179a3f3080b29ac1a9  0.0s
 => => naming to docker.io/library/net-tech-2s-client02:latest                            0.0s
 => => unpacking to docker.io/library/net-tech-2s-client02:latest                         0.0s
 => [srv01] exporting to image                                                            0.0s
 => => exporting layers                                                                   0.0s
 => => exporting manifest sha256:14840ee86188a3526586893c100c78111ce19e333e86489415964eef417b3c34  0.0s
 => => exporting config sha256:b4fdedeee4978c38768bae406a4a0da6355f612dbfef6e58170d3b60d2bb3433    0.0s
 => => exporting attestation manifest sha256:080801447935aada917f2040852ab245225436f59be4340203d0  0.0s
 => => exporting manifest list sha256:99f7271cd9401c232f62328f47d4136db69ef014de0c61feeec1500acf6  0.0s
 => => naming to docker.io/library/net-tech-2s-srv01:latest                               0.0s
 => => unpacking to docker.io/library/net-tech-2s-srv01:latest                            0.0s
 => [client01] resolving provenance for metadata file                                     0.0s
 => [srv01] resolving provenance for metadata file                                        0.0s
 => [client02] resolving provenance for metadata file                                     0.0s
[+] Running 6/6
 ✔ client01            Built                                                            0.0s
 ✔ client02            Built                                                            0.0s
 ✔ srv01               Built                                                            0.0s
 ✔ Container srv01     Created                                                          0.0s
 ✔ Container client01  Created                                                          0.0s
 ✔ Container client02  Created                                                          0.0s
Attaching to client01, client02, srv01
```

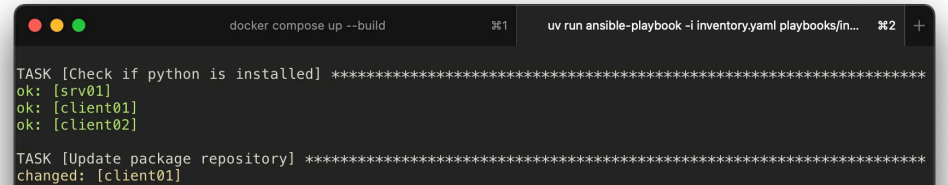
При попытке пинга, Ansible падает с ошибкой, поскольку Python не установлен

`uv run ansible -i inventory.yaml -m ping all`



Запуск установки

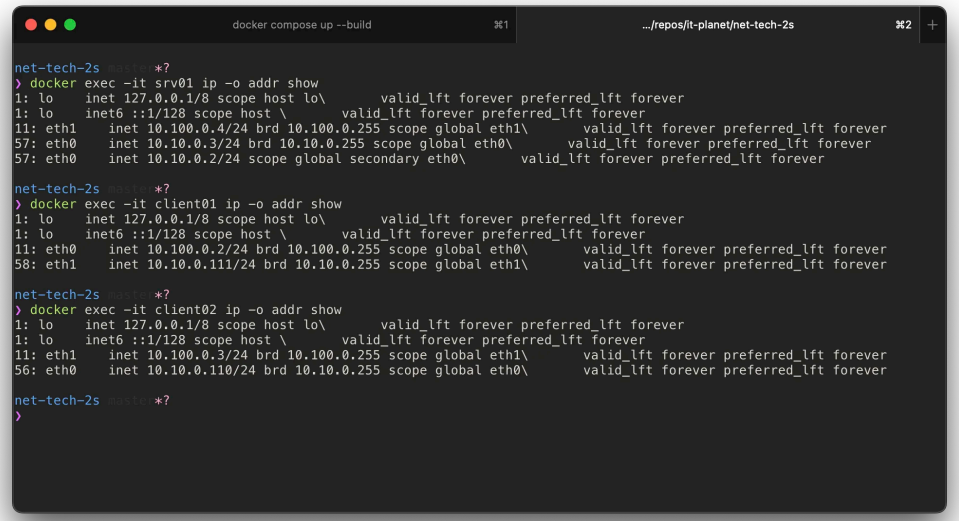
`uv run ansible-playbook -i inventory.yaml playbooks/install.yaml`



```
changed: [client02]
changed: [srv01]

TASK [Install python3] *********************************************************************************
changed: [srv01]
changed: [client01]
changed: [client02]

TASK [Gather installed python version] *****************************************************************
ok: [srv01]
ok: [client01]
ok: [client02]

TASK [Display python version] **************************************************************************
ok: [srv01] =>
    msg: Python 3.9.20
ok: [client01] =>
    msg: Python 3.9.20
ok: [client02] =>
    msg: Python 3.9.20

PLAY [Install packages] ********************************************************************************

TASK [Install helper packages] *************************************************************************
changed: [client02]
changed: [client01]
changed: [srv01]

PLAY [Setup web-server] ********************************************************************************

TASK [Check if Angie is installed] *********************************************************************
ok: [srv01]

TASK [Download Angie keys & add repository] ************************************************************
changed: [srv01]

TASK [Install packages] ********************************************************************************
changed: [srv01]

TASK [Test Angie conf] *********************************************************************************
ok: [srv01]

TASK [Stop Angie] **************************************************************************************
changed: [srv01]

TASK [Start Angie] *************************************************************************************
changed: [srv01]

TASK [Test Angie] **************************************************************************************
_
```

```
docker compose up --build  ⌘1    .../repos/it-planet/net-tech-2s  ⌘2  +

TASK [Install iptables] ********************************************************************************
ok: [client01]
ok: [srv01]
ok: [client02]

TASK [Configure iptables] ******************************************************************************
ok: [client01]
ok: [srv01]
ok: [client02]

TASK [Apply iptables] **********************************************************************************
changed: [srv01]
changed: [client02]
changed: [client01]

PLAY [Test DNS server] *********************************************************************************

TASK [Test DNS server] *********************************************************************************
ok: [client01]
ok: [client02]

PLAY RECAP *********************************************************************************************
client01                   : ok=20   changed=5    unreachable=0    failed=0    skipped=2    rescued=0    ignored=0
client02                   : ok=20   changed=5    unreachable=0    failed=0    skipped=2    rescued=0    ignored=0
srv01                      : ok=25   changed=8    unreachable=0    failed=0    skipped=4    rescued=0    ignored=0


net-tech-2s master * 58s
❯ _
```

В ходе выполнения playbook, он автоматически проверяет работу DNS сервера и выводит назначенные ip. Такие ip он назначил:



Проверка доступности HTTP-сервера по доменному имени при помощи curl: