

МИНИСТЕРСТВО ЦИФРОВОГО РАЗВИТИЯ, СВЯЗИ И  
МАССОВЫХ КОММУНИКАЦИЙ РОССИЙСКОЙ  
ФЕДЕРАЦИИ

«Сибирский государственный университет  
телекоммуникаций и информатики»  
(СибГУТИ)

ОТЧЕТ  
по дисциплине  
«*WEB-технологии*»

по теме:  
Освоение Ansible на ВМ с ОС Linux

Студент:  
*Группы ИКС-432*

*А.А. Пастухов*

Преподаватель:

*А.В. Андреев*

Новосибирск 2025

## Содержание

ВВЕДЕНИЕ .....	3
1. ПРАКТИЧЕСКАЯ ЧАСТЬ.....	4
ЗАКЛЮЧЕНИЕ .....	10

## ВВЕДЕНИЕ

Создадим топологию сети, состоящую из трёх машин. Используем ОС 24.10 семейства Linux, одну машину будем использовать как сервер, остальные – как клиенты, а также установим и настроим Ansible, напишем playbook, при запуске которого на сервер в папку /etc/ansible/web9 с клиентов будет собираться следующая информация:

- IP адреса клиентов;
- версия операционной системы клиентов;
- имена клиентов;
- количество свободного места на диске.

Наша топология сети:

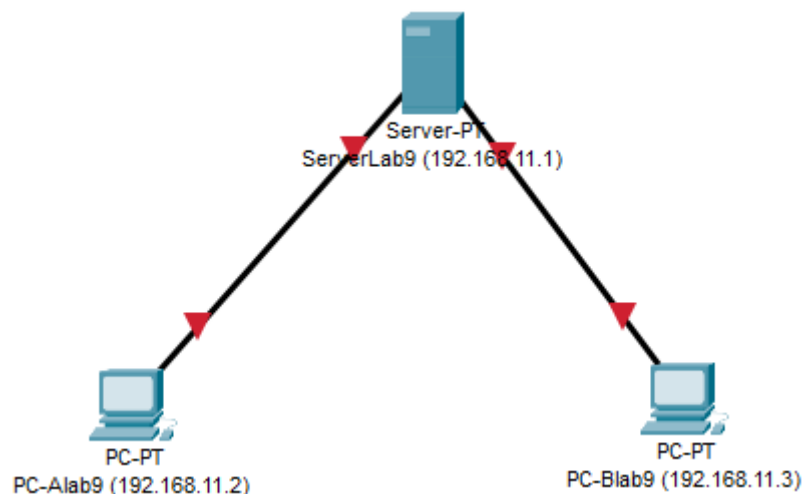


Рисунок 1 – Топология сети в Cisco Packet Tracer

Для визуализации топологии сети была использована программа Cisco Packet Tracer. На схеме ServerLab9 – наш Ansible сервер с будущим ipv4 адресом 192.168.11.1  
Клиентам Alab9 и Blab9 в дальнейшем будут присвоены адреса 192.168.11.2 и 192.168.11.3 соответственно.

# 1. ПРАКТИЧЕСКАЯ ЧАСТЬ

На каждой виртуальной машине будут включены 2 сетевых адаптера

- NAT – для доступа в интернет (enp0s3)
- Internal network (Внутренняя сеть) – для общения устройств между собой (enp0s8)

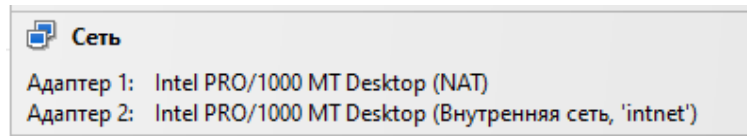


Рисунок 2 – адаптеры на виртуальных машинах

Отредактируем конфигурационный файл 00-installer-config.yaml на виртуальных машинах для присваивания ipv4 адресов клиентам и серверу

```
GNU nano 8.1 /etc/netplan/00-installer-config.yaml
network:
  ethernets:
    enp0s3:
      dhcp4: no
      addresses:
        - 10.0.2.15/24
      gateway4: 10.0.2.2
      nameservers:
        addresses: [8.8.8.8]
    enp0s8:
      dhcp4: no
      addresses:
        - 192.168.11.1/24

[ Wrote 13 lines ]
^C Help      ^O Write Out ^F Where Is  ^K Cut      ^T Execute  ^C Location
^X Exit      ^R Read File ^\ Replace  ^U Paste    ^J Justify  ^_ Go To Line
root@arown:/home/sasha# ping ya.ru
PING ya.ru (5.255.255.242) 56(84) bytes of data.
64 bytes from ya.ru (5.255.255.242): icmp_seq=1 ttl=255 time=48.7 ms
64 bytes from ya.ru (5.255.255.242): icmp_seq=2 ttl=255 time=52.5 ms

[1]+  Stopped                  ping ya.ru
root@arown:/home/sasha# ping 8.8.8.8
PING 8.8.8.8 (8.8.8.8) 56(84) bytes of data.
64 bytes from 8.8.8.8: icmp_seq=1 ttl=255 time=63.1 ms
^Z
[2]+  Stopped                  ping 8.8.8.8
root@arown:/home/sasha#
```

```
GNU nano 8.1 /etc/netplan/00-installer-config.yaml
network:
  ethernets:
    enp0s3:
      dhcp4: no
      addresses:
        - 10.0.2.15/24
      gateway4: 10.0.2.2
      nameservers:
        addresses: [8.8.8.8]
    enp0s8:
      dhcp4: no
      addresses:
        - 192.168.11.2/24

[ Read 13 lines ]
^G Help      ^O Write Out ^F Where Is  ^K Cut      ^T Execute   ^C Location
^X Exit      ^R Read File ^\ Replace   ^U Paste     ^J Justify   ^_ Go To Line

PING ya.ru (5.255.255.242) 56(84) bytes of data.
64 bytes from ya.ru (5.255.255.242): icmp_seq=1 ttl=255 time=49.9 ms
^Z
[2]+  Stopped                  ping ya.ru
root@arown:/home/sasha# ping 8.8.8.8
PING 8.8.8.8 (8.8.8.8) 56(84) bytes of data.
64 bytes from 8.8.8.8: icmp_seq=1 ttl=255 time=70.9 ms
^Z
[3]+  Stopped                  ping 8.8.8.8

GNU nano 8.1 /etc/netplan/00-installer-config.yaml
network:
  ethernets:
    enp0s3:
      dhcp4: no
      addresses:
        - 10.0.2.15/24
      gateway4: 10.0.2.2
      nameservers:
        addresses: [8.8.8.8]
    enp0s8:
      dhcp4: no
      addresses:
        - 192.168.11.3/24

[ Wrote 13 lines ]
^G Help      ^O Write Out ^F Where Is  ^K Cut      ^T Execute   ^C Location
^X Exit      ^R Read File ^\ Replace   ^U Paste     ^J Justify   ^_ Go To Line

root@arown:/home/sasha# ping 8.8.8.8
PING 8.8.8.8 (8.8.8.8) 56(84) bytes of data.
64 bytes from 8.8.8.8: icmp_seq=1 ttl=255 time=71.2 ms
^Z
[1]+  Stopped                  ping 8.8.8.8
root@arown:/home/sasha# ping ya.ru
PING ya.ru (77.88.55.242) 56(84) bytes of data.
64 bytes from ya.ru (77.88.55.242): icmp_seq=1 ttl=255 time=52.5 ms
^Z
[2]+  Stopped                  ping ya.ru
root@arown:/home/sasha#
```

Рисунок 3-8 – настройка 00-installer-config.yaml

Установим Ansible на наш Server с помощью утилиты APT

```
root@arown:/home/sasha# apt update && apt install ansible
```

Рисунок 9 – установка ansible

Укажем наших клиентов в файле /etc/ansible/hosts

```
[client]
192.168.11.2  ansible_user=root
192.168.11.3  ansible_user=root
```

Рисунок 10 – клиенты в /etc/ansible/hosts

Для сбора информации о клиентах Ansible требуется подключение по SSH. Установим на каждую машину SSH с помощью утилиты APT. Запустим сервис SSH и поставим его в автозагрузку с помощью systemctl. Проверим статус сервиса:

```
root@arown:/home/sasha# systemctl start ssh
root@arown:/home/sasha# systemctl enable ssh
Synchronizing state of ssh.service with SysV service script with /usr/lib/systemd/systemd-sysv-install.
Executing: /usr/lib/systemd/systemd-sysv-install enable ssh
Created symlink '/etc/systemd/system/ssh.service' -> '/usr/lib/systemd/system/ssh.service'.
Created symlink '/etc/systemd/system/multi-user.target.wants/ssh.service' -> '/usr/lib/systemd/system/ssh.service'.
root@arown:/home/sasha# systemctl status ssh
● ssh.service - OpenBSD Secure Shell server
   Loaded: loaded (/usr/lib/systemd/system/ssh.service; enabled; preset: enabled)
   Active: active (running) since Wed 2025-04-02 10:47:38 +07; 16s ago
     Invocation: 0d67b1bd83b6408398a9e1d503d773fc
   TriggeredBy: ● ssh.socket
      Docs: man:sshd(8)
            man:sshd_config(5)
    Main PID: 6941 (sshd)
       Tasks: 1 (limit: 10293)
      Memory: 1.3M (peak: 1.5M)
         CPU: 51ms
        CGroup: /system.slice/ssh.service
                └─6941 "sshd: /usr/sbin/sshd -D [listener] 0 of 10-100 startups"

Apr 02 10:47:38 arown systemd[1]: Starting ssh.service - OpenBSD Secure Shell server...
Apr 02 10:47:38 arown sshd[6941]: Server listening on 0.0.0.0 port 22.
Apr 02 10:47:38 arown sshd[6941]: Server listening on :: port 22.
Apr 02 10:47:38 arown systemd[1]: Started ssh.service - OpenBSD Secure Shell server.
root@arown:/home/sasha#
```

```
root@arown:/home/sasha# systemctl start ssh
root@arown:/home/sasha# systemctl enable ssh
Synchronizing state of ssh.service with SysV service script with /usr/lib/systemd/systemd-sysv-install.
Executing: /usr/lib/systemd/systemd-sysv-install enable ssh
Created symlink '/etc/systemd/system/ssh.service' -> '/usr/lib/systemd/system/ssh.service'.
Created symlink '/etc/systemd/system/multi-user.target.wants/ssh.service' -> '/usr/lib/systemd/system/ssh.service'.
root@arown:/home/sasha# systemctl status ssh
● ssh.service - OpenBSD Secure Shell server
   Loaded: loaded (/usr/lib/systemd/system/ssh.service; enabled; preset: enabled)
   Active: inactive (dead)
     Invocation: a884b543c8534e08b4f8cdeb9b3df9bd
   TriggeredBy: ● ssh.socket
      Docs: man:sshd(8)
            man:sshd_config(5)
    Main PID: 8469 (sshd)
       Tasks: 1 (limit: 10293)
      Memory: 1.3M (peak: 1.5M)
         CPU: 61ms
        CGroup: /system.slice/ssh.service
                └─8469 "sshd: /usr/sbin/sshd -D [listener] 0 of 10-100 startups"

Apr 02 10:48:13 arown systemd[1]: Starting ssh.service - OpenBSD Secure Shell server...
Apr 02 10:48:13 arown sshd[8469]: Server listening on 0.0.0.0 port 22.
Apr 02 10:48:13 arown sshd[8469]: Server listening on :: port 22.
Apr 02 10:48:13 arown systemd[1]: Started ssh.service - OpenBSD Secure Shell server.
root@arown:/home/sasha#
```

```

root@arown:/home/sasha# systemctl start ssh
root@arown:/home/sasha# systemctl enable ssh
Synchronizing state of ssh.service with SysV service script with /usr/lib/systemd/systemd-sysv-install.
Executing: /usr/lib/systemd/systemd-sysv-install enable ssh
Created symlink '/etc/systemd/system/ssh.service' → '/usr/lib/systemd/system/ssh.service'.
Created symlink '/etc/systemd/system/multi-user.target.wants/ssh.service' → '/usr/lib/systemd/system/ssh.service'.
root@arown:/home/sasha# systemctl status ssh
● ssh.service - OpenBSD Secure Shell server
   Loaded: loaded (/usr/lib/systemd/system/ssh.service; enabled; preset: enabled)
   Active: active (running) since Wed 2025-04-02 10:48:51 +07; 10s ago
     Invocation: 5bc98d655dfb4f7fa7c05f40217fe44a
   TriggeredBy: ● ssh.socket
      Docs: man:sshd(8)
           man:sshd_config(5)
    Main PID: 6280 (sshd)
      Tasks: 1 (limit: 10293)
     Memory: 1.3M (peak: 1.6M)
        CPU: 53ms
      CGroup: /system.slice/ssh.service
              └─6280 "sshd: /usr/sbin/sshd -D [listener] 0 of 10-100 startups"

Apr 02 10:48:51 arown systemd[1]: Starting ssh.service - OpenBSD Secure Shell server...
Apr 02 10:48:51 arown sshd[6280]: Server listening on 0.0.0.0 port 22.
Apr 02 10:48:51 arown sshd[6280]: Server listening on :: port 22.
Apr 02 10:48:51 arown systemd[1]: Started ssh.service - OpenBSD Secure Shell server.
root@arown:/home/sasha#

```

Рисунок 11-13 – ssh на машинах

Далее, сгенерируем публичный и приватный RSA ключи на сервере и передадим их клиентам для осуществления SSH подключения, заодно проверим, что Ansible может установить подключения к клиентам:

```

root@arown:/home/sasha# ssh-copy-id root@192.168.11.3
/usr/bin/ssh-copy-id: INFO: Source of key(s) to be installed: "/root/.ssh/id_rsa.pub"
/usr/bin/ssh-copy-id: INFO: attempting to log in with the new key(s), to filter out any
that are already installed
/usr/bin/ssh-copy-id: INFO: 1 key(s) remain to be installed -- if you are prompted now
it is to install the new keys
root@192.168.11.3's password:

Number of key(s) added: 1

Now try logging into the machine, with:  "ssh 'root@192.168.11.3'"
and check to make sure that only the key(s) you wanted were added.

root@arown:/home/sasha#

root@arown:/home/sasha# ansible all -m ping
192.168.11.2 | SUCCESS => {
  "ansible_facts": {
    "discovered_interpreter_python": "/usr/bin/python3"
  },
  "changed": false,
  "ping": "pong"
}
192.168.11.3 | SUCCESS => {
  "ansible_facts": {
    "discovered_interpreter_python": "/usr/bin/python3"
  },
  "changed": false,
  "ping": "pong"
}
root@arown:/home/sasha#

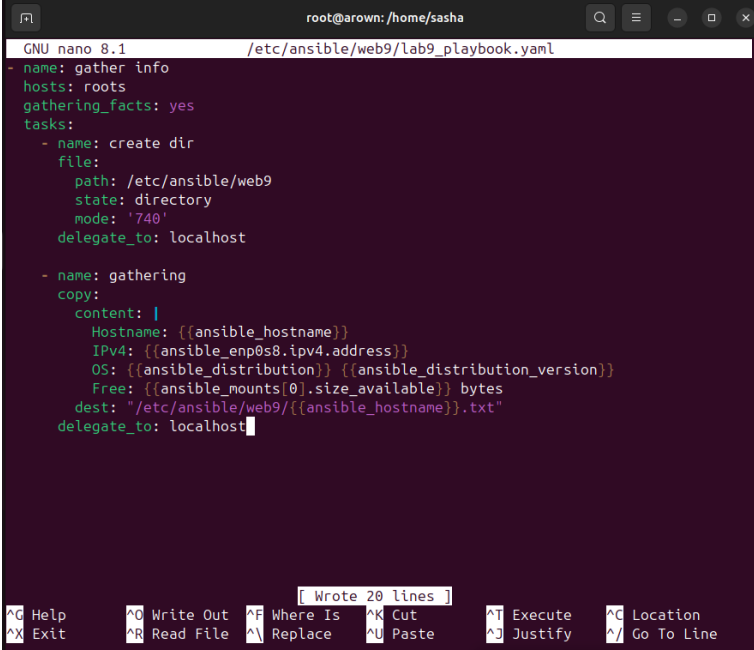
```

Рисунок 14-15 – создание ключа

Создадим Ansible playbook, который будет выполнять задачи по сбору следующей информации с клиентов:

- IP адреса клиентов;
- версия операционной системы клиентов;
- имена клиентов;
- количество свободного места на диске.

Информация о каждом клиенте будет записываться в файл `arown.txt`, расположенный в директории `/etc/ansible/web9` на ServerLab9:



```
GNU nano 8.1 /etc/ansible/web9/lab9_playbook.yaml
- name: gather info
  hosts: roots
  gathering_facts: yes
  tasks:
    - name: create dir
      file:
        path: /etc/ansible/web9
        state: directory
        mode: '740'
        delegate_to: localhost

    - name: gathering
      copy:
        content: |
          Hostname: {{ansible_hostname}}
          IPv4: {{ansible_enp0s8.ipv4.address}}
          OS: {{ansible_distribution}} {{ansible_distribution_version}}
          Free: {{ansible_mounts[0].size_available}} bytes
        dest: "/etc/ansible/web9/{{ansible_hostname}}.txt"
        delegate_to: localhost
```

[ Wrote 20 lines ]

^G Help ^O Write Out ^F Where Is ^K Cut ^T Execute ^C Location  
^X Exit ^R Read File ^\ Replace ^U Paste ^J Justify ^\_ Go To Line

Рисунок 16 – файл ansible playbook



Запустим Playbook и проверим, что всё работает корректно:

```
root@arown:/etc/ansible/web9# ansible-playbook lab9_playbook.yaml

PLAY [gather info] *****

TASK [Gathering Facts] *****
ok: [192.168.11.3]
ok: [192.168.11.2]

TASK [create dir] *****
changed: [192.168.11.3 -> localhost]
ok: [192.168.11.2 -> localhost]

TASK [gathering] *****
changed: [192.168.11.2 -> localhost]
changed: [192.168.11.3 -> localhost]

PLAY RECAP *****
192.168.11.2      : ok=3    changed=1    unreachable=0    failed=0    skipped=
0    rescued=0    ignored=0
192.168.11.3      : ok=3    changed=2    unreachable=0    failed=0    skipped=
0    rescued=0    ignored=0

root@arown:/etc/ansible/web9#
```

```
root@arown:/etc/ansible/web9# ls
arown.txt  lab9_playbook.yaml
root@arown:/etc/ansible/web9# cat arown.txt
Hostname: arown
IPv4: 192.168.11.3
OS: Ubuntu 24.10
Free: 14623801344 bytes
```

Рисунок 17 – содержимое файла arown.txt

## Заключение

В ходе выполнения работы была создана топология сети, состоящая из трёх машин: одного сервера и двух клиентских узлов. На сервере был установлен и настроен Ansible для централизованного управления клиентами.

Разработан Ansible playbook, который собирает следующую информацию с клиентских машин:

- IP-адреса клиентов,
- Версию операционной системы,
- Имена хостов,
- Количество свободного места на диске.

Собранные данные автоматически сохраняются на сервере в директории `/etc/ansible/web9`.