Домашнее задание (модуль 25)

25.5 Практическая работа

Цель практической работы

Применить полученные знания об электронной почте и VPN.

Задание 1. Настройка алертов на email

Что нужно сделать

Используя полученные знания, самостоятельно настройте для Alertmanager отправку сообщений на электронную почту.

Что оценивается

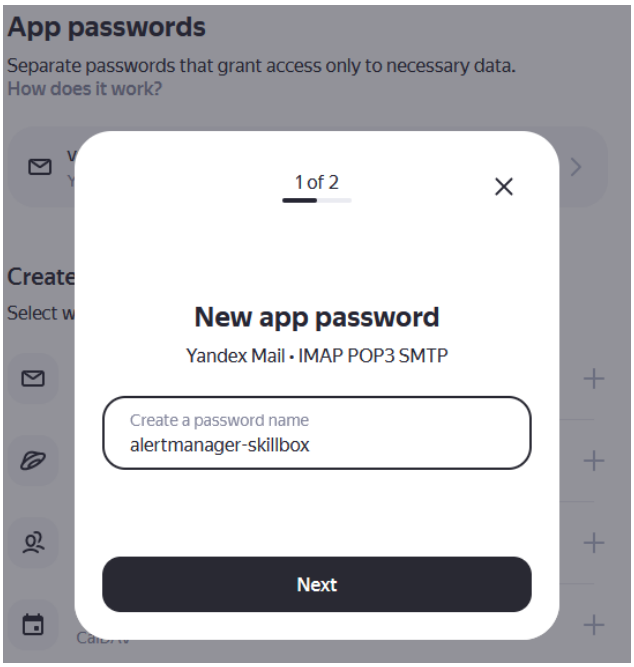
Оповещение от сработавшего алерта успешно отправляется на электронную почту.

Как отправить задание на проверку

Приложите скриншоты с конфигурационными файлами и с алертом, сработавшим в веб-интерфейсе Alertmanager и веб-интерфейсе почты.

Ответ

Создадим отдельный пароль для altermanager на почте Яндекс360:



Создадим docker-compose файл:

```
vgusev@vgusev:/srv/alertmanager$ cat docker-compose.yml
version: '3.8'
services:
  alertmanager:
    container_name: alertmanager-skillbox
    image: prom/alertmanager:v0.26.0
    restart: unless-stopped
    ports:
      - "9093:9093"
    volumes:
      - "./config:/config"
      - alertmanager-data:/data
    command: --config.file=/config/alertmanager.yml
    environment:
      TZ: "Europe/Moscow"


volumes:
  alertmanager-data:
```

Создадим минимальный файл конфигурации для alertmanager (пароль укажем отдельный, который мы специально сгенерировали ранее):

```
vgusev@vgusev:/srv/alertmanager$ cat config/alertmanager.yml
route:
  receiver: 'mail'
  repeat_interval: 4h
  group_by: [ alertname ]


receivers:
  - name: 'mail'
    email_configs:
      - smarthost: 'smtp.yandex.ru:587'
        auth_username: [redacted]
        auth_password: [redacted]
        from: 'vgusev2007@yandex.ru'
        to: 'vgusev2007@yandex.ru'
```

Запускаем контейнер:

```
vgusev@vgusev:/srv/alertmanager$ docker compose up -d
```

Если что-то пошло не так, смотрим статус контейнера, командой: docker ps и логи:

```
vgusev@vgusev:/srv/alertmanager$ docker logs alertmanager-skillbox
[default] 0:bash*
```

Исправляем все ошибки.

Посмотрим, какой ip адрес из внутренней подсети получил контейнер:

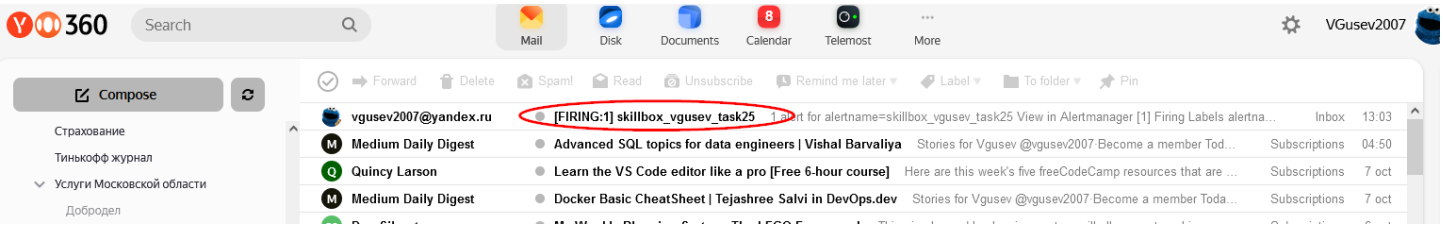
```
vgusev@vgusev:/srv/alertmanager$ docker inspect \
>   -f '{{range.NetworkSettings.Networks}}{{.IPAddress}}{{end}}' alertmanager-skillbox
172.20.0.2
```

Эмулируем, тревожный сигнал через api:

```
vgusev@vgusev:/srv/alertmanager$ curl -H 'Content-Type: application/json' -d '[{"labels":{"alertname":"skillbox_vgusev_task25"}}]' http://172.20
.0.2:9093/api/v1/alerts
{"status":"success"}vgusev@vgusev:/srv/alertmanager$
```

Видим успешный статус.

Проверяем почту:



Всё отлично. Если почта не пришла, смотрим логи:

```
vgusev@vgusev:/srv/alertmanager$ docker logs alertmanager-skillbox
[default] 0:bash*
```

Изначально, почта не приходила, т.к. я указывал порт 465 яндекс, а там, похоже не работает STARTLTS (по всей видимости, именно эту команду сначала посылает alertmanager) только TLS, пришлось немного разбираться и сменить порт.

Задание 2. Настройка VPN-сервера

Что нужно сделать

Воспользовавшись продемонстрированной спикером инструкцией, самостоятельно настройте VPN-сервер и VPN-клиента на вашем компьютере.

Что оценивается

VPN-клиент успешно подключается к VPN-серверу. IP-адрес, который отображает сервис, наподобии myip.ru или yandex.ru/internet, совпадает с IP-адресом VPN-сервера.

Как отправить задание на проверку

Приложите скриншоты, на которых видна конфигурация VPN-сервера и VPN-клиента, скриншоты IP-адреса и успешно подключившегося к серверу VPN-клиента лог-файла.

Docker

Создадим свой контейнер для управления облачной VM в gcp:

Но сперва прочитаем вот это: и увидим, что Россия и Беларусь, не имеют права скачивать продукты terraform:

https://github.com/hashicorp/terraform/issues/30591

Купим VPS в Нидерландах, и настроим там http прокси.

Настроим dumbproxy

vi /etc/systemd/system/dumbproxy.service

vi /etc/default/dumbproxy

systemctl restart dumbproxy

Пропишем прокси как во время создания контейнера, для успешной установки terraform, так и после, для установки провайдера google для terraform. Всё пропишем в .env файле.

```
vgusev@vgusev:/srv/skillbox/adm/manage$ cat docker-compose.yml
version: '3.8'
services:
  cloud:
    build:
      context: ./base_img
      args:
        - UBUNTU_VERSION=22.04
        - ANSIBLE_VERSION=8.5.0-1ppa~jammy
        - GOOGLE_CLOUD_CLI_VERSION=451.0.0-0
        - TERRAFORM_VERSION=1.6.2-1
        - http_proxy
        - https_proxy
    tty: true
    image: base_img
    user: vgusev2007
    restart: unless-stopped
    volumes:
      - cloud_home_data:/home
    environment:
      - TZ="Europe/Moscow"
      - http_proxy
      - https_proxy

volumes:
  cloud_home_data:
```

Создадим Dockerfile:

```
ARG UBUNTU_VERSION

FROM ubuntu:${UBUNTU_VERSION}

ARG DEBIAN_FRONTEND=noninteractive
ARG ANSIBLE_VERSION
ARG TERRAFORM_VERSION
ARG GOOGLE_CLOUD_CLI_VERSION

RUN useradd -m vgusev2007 && echo "cd ~" >> /etc/bash.bashrc
RUN apt-get update && \
      apt-get -y install \
      sudo vim mc software-properties-common apt-transport-https ca-certificates gnupg curl openssh-client git wget
# Add ansible gcloud and terraform repos
RUN apt-add-repository -y ppa:ansible/ansible
RUN echo "deb [signed-by=/usr/share/keyrings/cloud.google.asc] https://packages.cloud.google.com/apt cloud-sdk main" | \
    tee -a /etc/apt/sources.list.d/google-cloud-sdk.list && curl https://packages.cloud.google.com/apt/doc/apt-key.gpg | \
    tee /usr/share/keyrings/cloud.google.asc

RUN wget -O- https://apt.releases.hashicorp.com/gpg | gpg --dearmor | tee /usr/share/keyrings/hashicorp-archive-keyring.gpg && \
    echo "deb [signed-by=/usr/share/keyrings/hashicorp-archive-keyring.gpg] https://apt.releases.hashicorp.com $(lsb_release -cs) main" | \
    tee /etc/apt/sources.list.d/hashicorp.list
RUN apt-get update && apt-get -y install ansible=$ANSIBLE_VERSION google-cloud-cli=${GOOGLE_CLOUD_CLI_VERSION} terraform=${TERRAFORM_VERSION}

COPY --chmod=0640 ./config/01-vgusev2007 /etc/sudoers.d
```

Создадим контейнер:

```
vgusev@vgusev:/srv/skillbox/adm/manage$ docker compose down && docker compose up --build -d
[+] Building 336.3s (13/13) FINISHED                                                                                    docker:default
 => [cloud internal] load build definition from Dockerfile                                                                        0.0s
 => => transferring dockerfile: 1.32kB                                                                                            0.0s
 => [cloud internal] load .dockerignore                                                                                           0.0s
 => => transferring context: 2B                                                                                                   0.0s
 => [cloud internal] load metadata for docker.io/library/ubuntu:22.04                                                             0.0s
 => [cloud 1/8] FROM docker.io/library/ubuntu:22.04                                                                               0.0s
 => [cloud internal] load build context                                                                                           0.0s
 => => transferring context: 139B                                                                                                 0.0s
 => [cloud 2/8] RUN useradd -m vgusev2007 && echo "cd ~" >> /etc/bash.bashrc                                                      0.4s
 => [cloud 3/8] RUN apt-get update &&       apt-get -y install       sudo vim mc software-properties-common apt-transport-https ca-cert  147.2s
 => [cloud 4/8] RUN apt-add-repository -y ppa:ansible/ansible                                                                     6.2s
 => [cloud 5/8] RUN echo "deb [signed-by=/usr/share/keyrings/cloud.google.asc] https://packages.cloud.google.com/apt cloud-sdk main" |     0.7s
 => [cloud 6/8] RUN wget -O- https://apt.releases.hashicorp.com/gpg | gpg --dearmor | tee /usr/share/keyrings/hashicorp-archive-keyring.g  0.8s
 => [cloud 7/8] RUN apt-get update && apt-get -y install ansible=8.5.0-1ppa~jammy google-cloud-cli=451.0.0-0 terraform=1.6.2-1           146.6s
 => [cloud 8/8] COPY --chmod=0640 ./config/01-vgusev2007 /etc/sudoers.d                                                           0.1s
 => [cloud] exporting to image                                                                                                   34.1s
 => => exporting layers                                                                                                          34.1s
 => => writing image sha256:8d90112233aac7aa3e941950a9188addb2014bed89bae2397e45dbd1e4cab00d                                      0.0s
 => => naming to docker.io/library/base_img                                                                                       0.0s
[+] Running 2/2
 ✔ Network manage_default    Created                                                                                              0.1s
 ✔ Container manage-cloud-1  Started                                                                                              0.1s
```

Проверим, что мы ходим через **Нидерланды** внутри контейнера:

```
vgusev2007@77a0029261b1:~$ curl ifconfig.me
109.23[redacted]vgusev2007@77a0029261b1:~$
```

Настроим gcloud:

```
vgusev@vgusev:/srv/skillbox/adm/manage$ docker exec -it manage-cloud-1 bash
vgusev2007@5d4e511640c7:~$ gcloud init
Welcome! This command will take you through the configuration of gcloud.

Your current configuration has been set to: [default]

You can skip diagnostics next time by using the following flag:
  gcloud init --skip-diagnostics

Network diagnostic detects and fixes local network connection issues.
Checking network connection...done.
Reachability Check passed.
Network diagnostic passed (1/1 checks passed).

You must log in to continue. Would you like to log in (Y/n)?

Go to the following link in your browser:

    https://accounts.google.com/o/oauth2/auth?response_type=code&client_id=3
.cloud.google.com%2Fauthcode.html&scope=openid+https%3A%2F%2Fwww.googleapis.
2Fcloud-platform+https%3A%2F%2Fwww.googleapis.com%2Fauth%2Fappengine.admin+h
F%2Fwww.googleapis.com%2Fauth%2Fcompute+https%3A%2F%2Fwww.googleapis.com%2Fa
nsent&access_type=offline&code_challenge=tMvyGD9HRMEDqpsU2RfCjTtb795i0I_6UEi

Enter authorization code: 
```

```
Pick cloud project to use:
 [1] static-lens-400903
 [2] Enter a project ID
 [3] Create a new project
Please enter numeric choice or text value (must exactly match list item):

Enter a Project ID. Note that a Project ID CANNOT be changed later.
Project IDs must be 6-30 characters (lowercase ASCII, digits, or
hyphens) in length and start with a lowercase letter. skillbox
```

Создадим проект:

```
vgusev2007@5d4e511640c7:~$ gcloud projects create dvp-project-vgusev
Create in progress for [https://cloudresourcemanager.googleapis.com/v1/projects/dvp-project-vgusev].
Waiting for [operations/cp.8024969754618833943] to finish...done.
Enabling service [cloudapis.googleapis.com] on project [dvp-project-vgusev]...
Operation "operations/acat.p2-573621570778-02172e23-8e12-4432-b75b-df06b50bdf60" finished successfully.
```

Активируем только что созданный проект:

```
vgusev2007@5d4e511640c7:~$ gcloud projects list
PROJECT_ID          NAME                PROJECT_NUMBER
dvp-project-vgusev  dvp-project-vgusev  573621570778
vgusev2007@5d4e511640c7:~$ gcloud config set project dvp-project-vgusev
Updated property [core/project].
```

Создадим сервисный аккаунт:

```
vgusev2007@5d4e511640c7:~$ gcloud iam service-accounts create terraform-user --display-name="Service Account for terraform"
Created service account [terraform-user].
```

Установим роль admin.role и account user в проекте для сервисного аккаунта:

```
vgusev2007@5d4e511640c7:~$ gcloud projects add-iam-policy-binding dvp-project-vgusev --member='serviceAccount:terraform-user@dvp-project-vgusev.
iam.gserviceaccount.com' --role=roles/compute.admin
Updated IAM policy for project [dvp-project-vgusev].
bindings:
- members:
  - serviceAccount:terraform-user@dvp-project-vgusev.iam.gserviceaccount.com
  role: roles/compute.admin
- members:
  - user:VGusev2007@gmail.com
  role: roles/owner
```

```
gcloud projects add-iam-policy-binding dvp-project-vgusev --member="serviceAccount:terraform-
user@dvp-project-vgu sev.iam.gserviceaccount.com" --role="roles/iam.serviceAccountUser"
```

Создадим ключ ssh:

```
vgusev2007@77a0029261b1:~$ ssh-keygen -t ed25519
Generating public/private ed25519 key pair.
Enter file in which to save the key (/home/vgusev2007/.ssh/id_ed25519):
Created directory '/home/vgusev2007/.ssh'.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/vgusev2007/.ssh/id_ed25519
Your public key has been saved in /home/vgusev2007/.ssh/id_ed25519.pub
The key fingerprint is:
SHA256:GGAfylTasa0JT5dJOlKpGcDRinf+bizSPwTnynSInhQ vgusev2007@77a0029261b1
The key's randomart image is:
+--[ED25519 256]--+
| .o++.=..        |
|  .+oB.B o       |
| . .*+O =        |
|. E ==.B         |
| . = == S        |
|  o + +          |
| o = *           |
|  + = =          |
|   . =o.         |
+----[SHA256]-----+
vgusev2007@77a0029261b1:~$
```

Загрузим его в OS login:

```
vgusev2007@77a0029261b1:~$ gcloud compute os-login ssh-keys add --key-file=.ssh/id_ed25519.pub
API [oslogin.googleapis.com] not enabled on project [dvp-project-vgusev]. Would you like to enable and retry (this will take a few minutes)?
(y/N)?  y

Enabling service [oslogin.googleapis.com] on project [dvp-project-vgusev]...
Operation "operations/acat.p2-573621570778-81913802-5608-4f8e-8641-46e5bb83e527" finished successfully.
loginProfile:
  name: '114316261966728082367'
  posixAccounts:
  - accountId: dvp-project-vgusev
    gid: '1956538576'
    homeDirectory: /home/vgusev2007_gmail_com
    name: users/vgusev2007@gmail.com/projects/dvp-project-vgusev
    operatingSystemType: LINUX
    primary: true
    uid: '1956538576'
    username: vgusev2007_gmail_com
  sshPublicKeys:
    b484817f8c80f4ec69eb37e97e8ca890091b803db79777198548062e5cde109e:
      fingerprint: b484817f8c80f4ec69eb37e97e8ca890091b803db79777198548062e5cde109e
      key: |
        ssh-ed25519 AAAAC3NzaC1lZDI1NTE5AAAAIKTHstn95rr5AXViBHEo7Rjjr3SN4RHQ3Hd9fra+LGZt vgusev2007@77a0029261b1
      name: users/vgusev2007@gmail.com/sshPublicKeys/b484817f8c80f4ec69eb37e97e8ca890091b803db79777198548062e5cde109e
```

Сгенерируем json файл с ключом доступа, для сервисного аккаунта (потребуется для terraform):

```
vgusev2007@5d4e511640c7:~$ gcloud iam service-accounts keys create service_account.json --iam-account=terraform-user@dvp-project-vgusev.iam.gser
viceaccount.com
created key [842a930004e57c610e1a8b960b3ea6b5bc5198e3] of type [json] as [service_account.json] for [terraform-user@dvp-project-vgusev.iam.gserv
iceaccount.com]
```

terraform:

```
vgusev2007@5d4e511640c7:~$ mkdir gcp
vgusev2007@5d4e511640c7:~$ mv service_account.json gcp/
vgusev2007@5d4e511640c7:~$ cd gcp
vgusev2007@5d4e511640c7:~/gcp$ l
service_account.json
```

Создадим файл main.tf, и укажем путь до файла с ключами от сервисного аккаунта:

```
vgusev2007@5d4e511640c7:~/gcp$ l
main.tf  service_account.json
```

Не забудем подключить os login (не зря ведь изучали его возможности), настроить сеть ит.п. для всего проекта:

```
provider "google" {
  credentials = file("service_account.json")

  project = "dvp-project-vgusev"
  region  = "us-central1"
  zone    = "us-central1-c"
}

resource "google_compute_project_metadata" "dvp-project-vgusev" {
  metadata = {
    enable-oslogin: "TRUE"
  }
}

resource "google_compute_instance" "vm_instance" {
  name         = "openvpn-instance"
  machine_type = "e2-micro"
  tags = ["ovpn"]

  boot_disk {
    initialize_params {
      image = "ubuntu-os-cloud/ubuntu-2204-lts"
    }
  }

  network_interface {
    # A default network is created for all GCP projects
    network = "default"
    access_config {
    }
  }
}

resource "google_compute_firewall" "default" {
  name    = "ovpn-firewall"
  network = "default"


    allow {
      protocol = "udp"
      ports    = ["1194"]
    }

  source_ranges = ["0.0.0.0/0"]
  target_tags = ["ovpn"]
}
```

Инициализируем terraform (**убедимся, что загрузка ПО прошла через Нидерланды и ошибок более нет**):

```
vgusev2007@77a0029261b1:~/gcp$ terraform init

Initializing the backend...

Initializing provider plugins...
- Finding latest version of hashicorp/google...
- Installing hashicorp/google v5.3.0...
- Installed hashicorp/google v5.3.0 (signed by HashiCorp)

Terraform has created a lock file .terraform.lock.hcl to record the provider
selections it made above. Include this file in your version control repository
so that Terraform can guarantee to make the same selections by default when
you run "terraform init" in the future.

Terraform has been successfully initialized!

You may now begin working with Terraform. Try running "terraform plan" to see
any changes that are required for your infrastructure. All Terraform commands
should now work.

If you ever set or change modules or backend configuration for Terraform,
rerun this command to reinitialize your working directory. If you forget, other
commands will detect it and remind you to do so if necessary.
```

Смотрим план:

```
vgusev2007@77a0029261b1:~/gcp$ terraform plan

Terraform used the selected providers to generate the following execution plan. Resource actions are indic
  + create

Terraform will perform the following actions:

  # google_compute_instance.vm_instance will be created
  + resource "google_compute_instance" "vm_instance" {
      + can_ip_forward       = false
      + cpu_platform         = (known after apply)
      + current_status       = (known after apply)
      + deletion_protection  = false
      + effective_labels     = (known after apply)
      + guest_accelerator    = (known after apply)
      + id                   = (known after apply)
      + instance_id          = (known after apply)
      + label_fingerprint    = (known after apply)
      + machine_type         = "e2-micro"
      + metadata_fingerprint = (known after apply)
      + min_cpu_platform     = (known after apply)
      + name                 = "openvpn-instance"
      + project              = "dvp-project-vgusev"
      + self_link            = (known after apply)
      + tags_fingerprint     = (known after apply)
      + terraform_labels     = (known after apply)
      + zone                 = "us-central1-c"

      + boot_disk {
          + auto_delete                = true
          + device_name                = (known after apply)
          + disk_encryption_key_sha256 = (known after apply)
          + kms_key_self_link          = (known after apply)
          + mode                       = "READ_WRITE"
```

Применяем план:

terraform apply

```
          + internal_ipv6_prefix_length = (known after app
          + ipv6_access_type            = (known after app
          + ipv6_address                = (known after app
          + name                        = (known after app
          + network                     = "default"
          + network_ip                  = (known after app
          + stack_type                  = (known after app
          + subnetwork                  = (known after app
          + subnetwork_project          = (known after app

          + access_config {
              + nat_ip       = (known after apply)
              + network_tier = (known after apply)
            }
        }
    }

Plan: 1 to add, 0 to change, 0 to destroy.

Do you want to perform these actions?
  Terraform will perform the actions described above.
  Only 'yes' will be accepted to approve.

  Enter a value: yes
```

```
google_compute_instance.vm_instance: Creating...
google_compute_instance.vm_instance: Still creating... [10s elapsed]
google_compute_instance.vm_instance: Still creating... [20s elapsed]
google_compute_instance.vm_instance: Still creating... [30s elapsed]
google_compute_instance.vm_instance: Creation complete after 39s [id=projects/dvp-project-vgusev/zones/us-central1-c/instances/openvpn-instance]

Apply complete! Resources: 1 added, 0 changed, 0 destroyed.
```

VM instances

Filter Enter property name or value

Status	Name	Zone	Recommendations	In use by	Internal IP	External IP	Connect
✓	openvpn-instance	us-central1-c			10.128.0.2 (nic0)	34.72.43.252 (nic0)	SSH

Ansible:

```
vgusev2007@77a0029261b1:~$ mkdir -p ansible ; cd $_
vgusev2007@77a0029261b1:~/ansible$
```

```
vgusev2007@77a0029261b1:~/ansible$ ansible-config init --disabled > ansible.cfg
vgusev2007@77a0029261b1:~/ansible$
```

Создадим playbook:

```
---
-  hosts: openvpnservers
   vars:
     OVPN_DATA: ovpn-data-vgusev
     VPN_DNS_NAME: VPN.TECHENERGOANALIT.RU

   roles:
     - geerlingguy.docker
   tags:
   - setup
   - never

   tasks:
     - name: Install required system packages
       apt:
         pkg:
           - python3-pip
         state: latest
         update_cache: true
       tags:
       - setup
       - never

     - name: Install Docker Module for Python
       pip:
         name: docker
       tags:
       - setup
       - never

     - name: Create OVPN volume
       community.docker.docker_volume:
         name: "{{ OVPN_DATA }}"
       tags:
       - setup
       - never
```

```
    - name: Del ovpn_genconfig container
      docker_container:
        name: ovpn_genconfig
        state: absent
      tags:
      - setup
      - never

    - name: Make script for ovpn initpki
      ansible.builtin.lineinfile:
        state: present
        dest: /usr/local/bin/ovpn_initpki.sh
        line: "{{ item }}"
        create: yes
        mode: 0755
      with_items:
        - "#!/bin/bash"
        - "docker run -v {{ OVPN_DATA }}:/etc/openvpn --rm -it kylemanna/openvpn ovpn_initpki"
      tags:
      - always
```

Выполняем ping:

```
vgusev2007@77a0029261b1:~/ansible$ ansible -i hosts all -m ping
34.72.43.252 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
```

Настроим docker:

Установим необходимые роли:

```
vgusev2007@77a0029261b1:~/ansible$ ansible-galaxy role install geerlingguy.docker
Starting galaxy role install process
- downloading role 'docker', owned by geerlingguy
- downloading role from https://github.com/geerlingguy/ansible-role-docker/archive/7.0.1.tar.gz
- extracting geerlingguy.docker to /home/vgusev2007/.ansible/roles/geerlingguy.docker
- geerlingguy.docker (7.0.1) was installed successfully
```

```
vgusev2007@77a0029261b1:~/ansible$ mkdir roles
vgusev2007@77a0029261b1:~/ansible$ cd $_
vgusev2007@77a0029261b1:~/ansible/roles$ ansible-galaxy init geerlingguy.docker
- Role geerlingguy.docker was created successfully
vgusev2007@77a0029261b1:~/ansible/roles$
```

Настроим и применим роль для подготовки среды docker в облаке:

```
vgusev2007@77a0029261b1:~$ ansible-galaxy role install geerlingguy.docker
[default] 0:docker*
```

Установим все необходимые переменные:

roup_vars/openvpnservers.yml

```
ansible_connection: ssh
ansible_user: vgusev2007_gmail_com
ansible_become: true
docker_install_compose: false
docker_users:
    - vgusev2007_gmail_com
docker_packages:
    - "docker-{{ docker_edition }}=5:24.0.6-1~ubuntu.22.04~jammy"
    - "docker-{{ docker_edition }}-cli=5:24.0.6-1~ubuntu.22.04~jammy"
    - "docker-{{ docker_edition }}-rootless-extras=5:24.0.6-1~ubuntu.22.04~jammy"
```

Применяем роль:

```
vgusev2007@77a0029261b1:~/ansible$ ansible-playbook playbook.yaml

PLAY [openvpnservers] ***********************************************************************************

TASK [Gathering Facts] ***********************************************************************************
ok: [34.72.43.252]

TASK [geerlingguy.docker : Load OS-specific vars.] *******************************************************
ok: [34.72.43.252]

TASK [geerlingguy.docker : include_tasks] ****************************************************************
skipping: [34.72.43.252]

TASK [geerlingguy.docker : include_tasks] ****************************************************************
included: /home/vgusev2007/.ansible/roles/geerlingguy.docker/tasks/setup-Debian.yml for 34.72.43.252

TASK [geerlingguy.docker : Ensure old versions of Docker are not installed.] *****************************
ok: [34.72.43.252]

TASK [geerlingguy.docker : Ensure dependencies are installed.] *******************************************
changed: [34.72.43.252]
```

...

```
PLAY RECAP ***********************************************************************************************
34.72.43.252               : ok=16   changed=6    unreachable=0    failed=0    skipped=9    rescued=0    ignored=0
```

Добавляем необходимые модули в playbook для запуска контейнера:

```
TASK [Install required system packages] ***********
changed: [34.72.43.252]

TASK [Install Docker Module for Python] ***********
changed: [34.72.43.252]

PLAY RECAP ****************************************
34.72.43.252                : ok=14   changed=2    u

TASK [Pull default Docker image] **
changed: [34.72.43.252]
```

Настройка ovpn сервера в облаке: внутри VM, и внутри этой VM docker

Создадим свой CA:

```
vgusev2007_gmail_com@openvpn-instance:~$ ovpn_initpki.sh

init-pki complete; you may now create a CA or requests.
Your newly created PKI dir is: /etc/openvpn/pki


Using SSL: openssl OpenSSL 1.1.1g  21 Apr 2020

Enter New CA Key Passphrase:
Re-Enter New CA Key Passphrase:
Generating RSA private key, 2048 bit long modulus (2 primes)
.......+++++
...............................+++++
e is 65537 (0x010001)
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Common Name (eg: your user, host, or server name) [Easy-RSA CA]:vpn.techenergoanalit.ru

CA creation complete and you may now import and sign cert requests.
Your new CA certificate file for publishing is at:
/etc/openvpn/pki/ca.crt


Using SSL: openssl OpenSSL 1.1.1g  21 Apr 2020
Generating DH parameters, 2048 bit long safe prime, generator 2
This is going to take a long time
.....................................................................................
.........+.......................+.....................................................
  '


Using SSL: openssl OpenSSL 1.1.1g  21 Apr 2020
Generating a RSA private key
.................................+++++
..+++++
writing new private key to '/etc/openvpn/pki/easy-rsa-72.caPccG/tmp.ONkMGP'
-----
Using configuration from /etc/openvpn/pki/easy-rsa-72.caPccG/tmp.lgCBKH
Enter pass phrase for /etc/openvpn/pki/private/ca.key:
Check that the request matches the signature
Signature ok
The Subject's Distinguished Name is as follows
commonName            :ASN.1 12:'VPN.TECHENERGOANALIT.RU'
Certificate is to be certified until Jan 29 10:13:51 2026 GMT (825 days)

Write out database with 1 new entries
Data Base Updated

Using SSL: openssl OpenSSL 1.1.1g  21 Apr 2020
Using configuration from /etc/openvpn/pki/easy-rsa-147.iJnOGA/tmp.KLDPGi
```

```
Enter pass phrase for /etc/openvpn/pki/private/ca.key:

An updated CRL has been created.
CRL file: /etc/openvpn/pki/crl.pem
```

Запускаем openvpn server:

```
vgusev2007_gmail_com@openvpn-instance:~$ docker run --restart unless-stopped -v ovpn-data-vgusev:/etc/openvpn -d --name ovpn_srv -p 1194:1194/udp --cap-add=NE
T_ADMIN kylemanna/openvpn
bd0c1cb5198a9cf686c05911f3f24e4dd7791e5f38711110e546c3bf8fab0af1
```

Создаем клиентский сертификат:

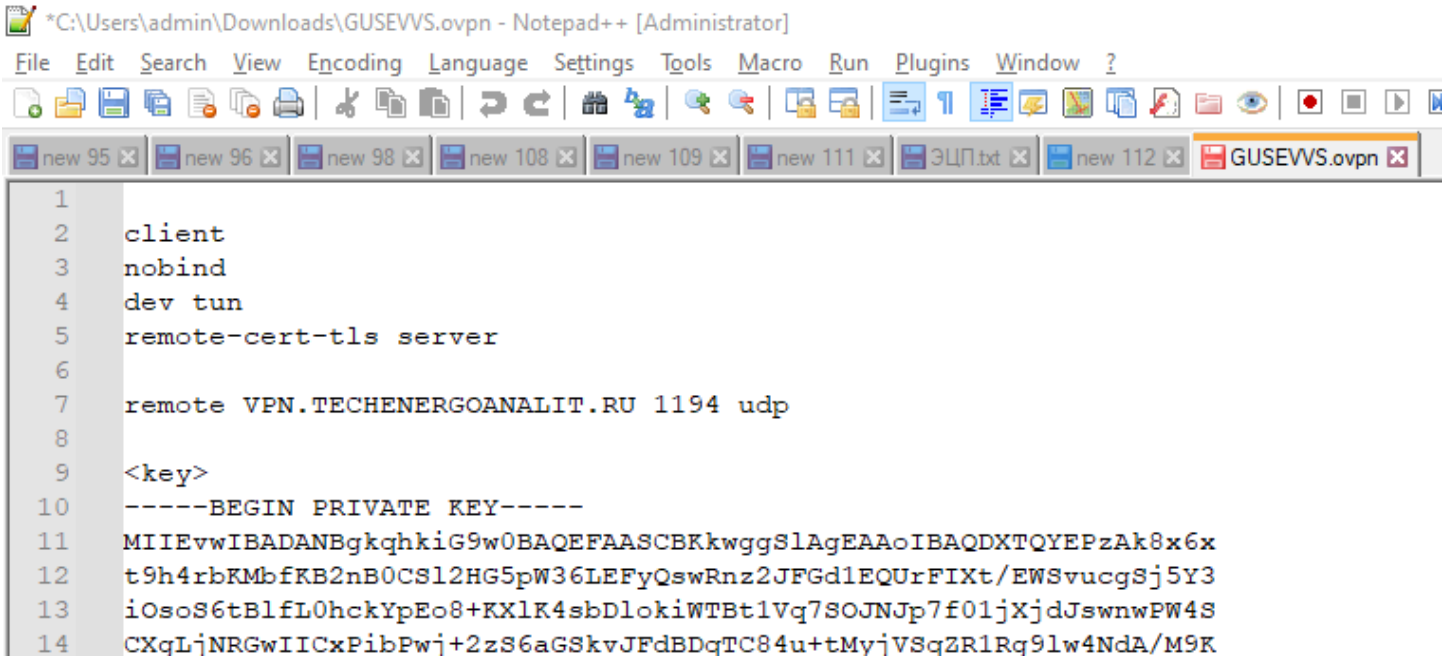
```
vgusev2007_gmail_com@openvpn-instance:~$ docker run -v ovpn-data-vgusev:/etc/openvpn --rm -it kylemanna/openvpn easyrsa build-client-full GUSEVVS nopass
Using SSL: openssl OpenSSL 1.1.1g  21 Apr 2020
Generating a RSA private key
..................+++++
...............+++++
writing new private key to '/etc/openvpn/pki/easy-rsa-1.lejAHl/tmp.edkFhD'
-----
Using configuration from /etc/openvpn/pki/easy-rsa-1.lejAHl/tmp.lBPjLd
Enter pass phrase for /etc/openvpn/pki/private/ca.key:
Check that the request matches the signature
Signature ok
The Subject's Distinguished Name is as follows
commonName            :ASN.1 12:'GUSEVVS'
Certificate is to be certified until Jan 31 09:29:20 2026 GMT (825 days)

Write out database with 1 new entries
Data Base Updated
```

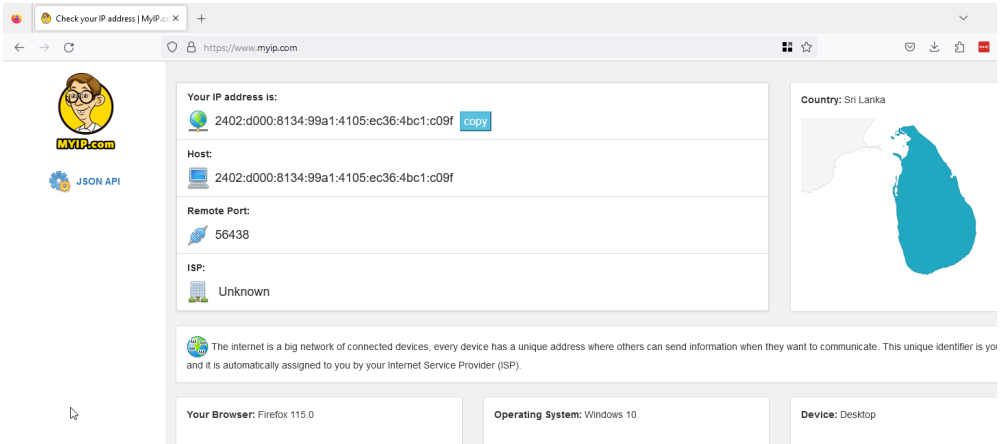
Создаем конфигурационный файл openvpn:

```
vgusev2007_gmail_com@openvpn-instance:~$ docker run -v ovpn-data-vgusev:/etc/openvpn --rm kylemanna/openvpn ovpn_getclient GUSEVVS > GUSEVVS.ovpn
vgusev2007_gmail_com@openvpn-instance:~$
```

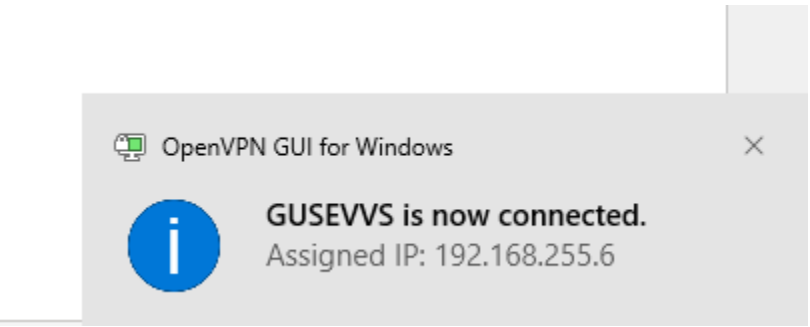
Посмотрим, что у нас в конфигурационном файле:



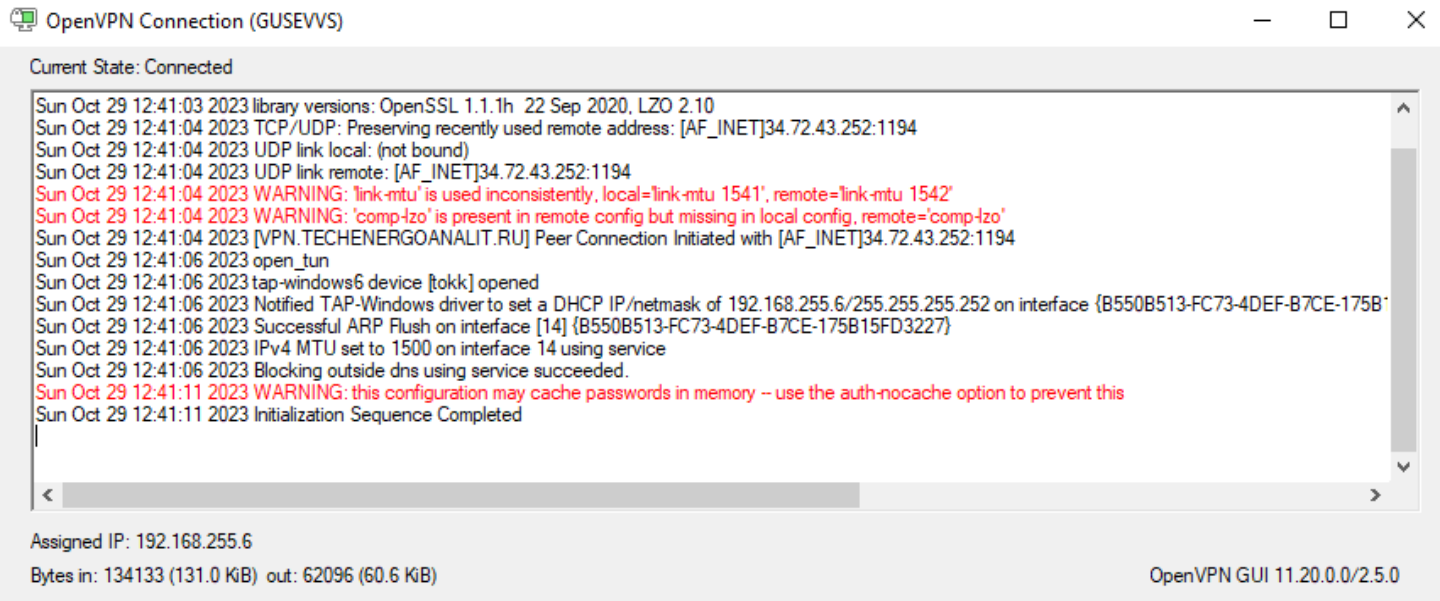
Перед подключением (Страна моего проживания Шри-Ланка: ipv6):



После подключения (Страна, США):



Лог подключения, немного отличается mtu, надо разобраться бы, но, пока не принципиально:



США: