Программное обеспечение оборудования коммутации и маршрутизации пакетов информации сетей передачи данных YACORE

Инструкция по установке

1

дата: 2026-05-09

Версия: <none>:0:1

1 Аннотация

Настоящий документ содержит описание работ по установке программного обеспечения «Программное обеспечение оборудования коммутации и маршрутизации пакетов информации сетей передачи данных YACORE» (далее - ПО).

Раздел «Общие сведения» содержит базовую информацию о ПО и процессе установки ПО.

Раздел «» содержит описание работ по созданию образа ПО в формате, пригодном к развертыванию на аппаратных средствах изделия.

Раздел «Установка образа ПО» содержит описание работ по развертыванию образа ПО.

2 Общие сведения

Установка ПО предполагает следующие этапы:

1. Сборка экземпляра ПО в виде образа, пригодного для развертывания на аппаратной платформе.
2. Установка образа ПО на аппаратную платформу.

3 Установка образа ПО

3.1 Требования к программно-аппаратным средствам

Установка образа ПО на аппаратную платформу осуществляется при помощи сервисного ноутбука, соответствующей требованиям завода-изготовителя.

Для успешной установки и настройки ПО необходимы:

1. Ноутбук с операционной системой Linux (желательно Ubuntu 22.04) или с виртуальной машиной Linux;
2. Наличие интерфейса 1Гбит/с RJ-45;
3. Наличие следующего установленного программного обеспечения:
    - docker не ниже версии 26.1.1;
    - python не ниже версии 3.10;
    - ansible не ниже версии 2.15.10.

Обязательным условием установки ПО является наличие подключения к хранилищу данных, в котором хранятся образы ПО.

Для корректной работы после установки, необходимо наличие базы данных ПО Тарантул Плюс, которая не является частью ПО и приобретается заказчиком отдельно.

3.2 Процесс установки

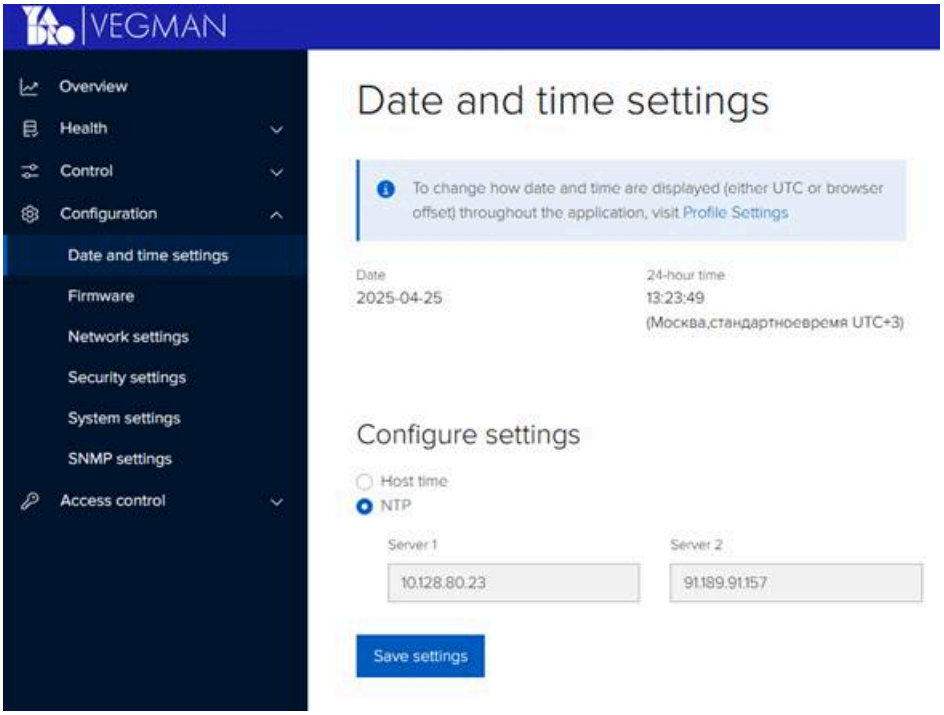
Подготовка к установке

1. **Настройка BMC**

   Для настройки BMC необходимо выполнить следующие шаги:

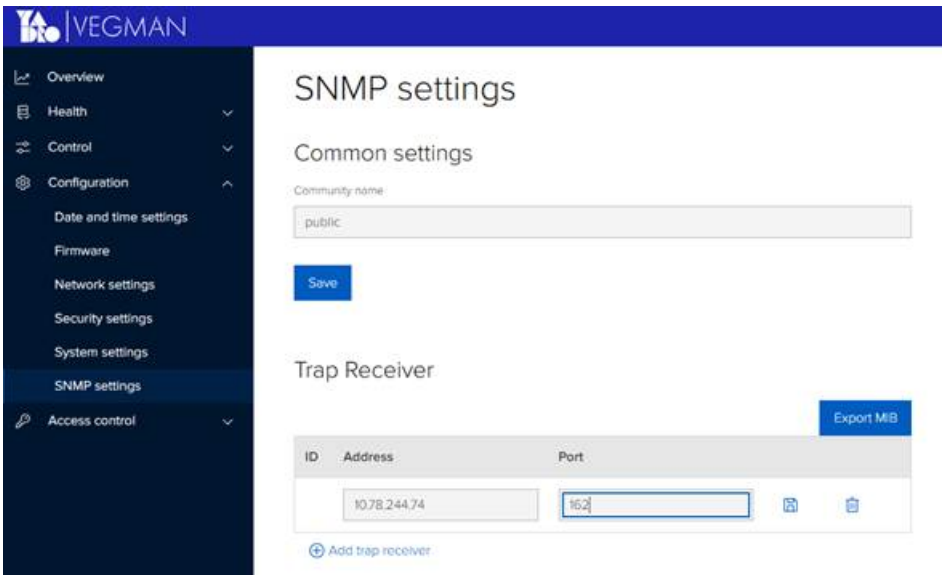
   - **Настройка NTP**

     В разделе Configation - Date and time settings необходимо прописать адреса серверов для NTP.



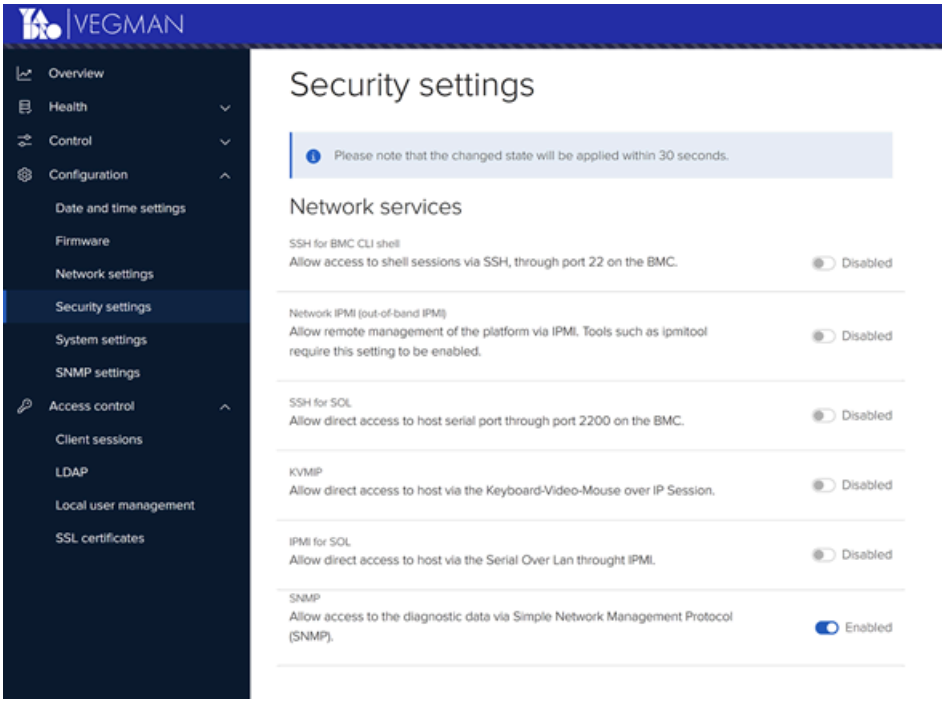
   - **Настройка SNMP**

     В разделе Configation - SNMP settings необходимо прописать адреса серверов для SNMP.



   - **Отключение неиспользуемых сетевых сервисов**

В разделе Configuration - Security settings необходимо выставить переключатели, как показано на рисунке ниже.



2. **Подготовка BIOS**

   Перед установкой ОС необходимо в BIOS создать RAID10 массив, отключить загрузку по сети и Hyper Threading.

   Для подготовки BIOS необходимо открыть GUI BMC сервера, включить питание, в SOC (Serial over LAN) консоли BMC UI зайти в BIOS и выполнить следующие шаги:

   - **настройка RAID10:**
     1. Перейти в раздел Advanced - AVAGO MegaRAID - Configuraiton Utility.
     2. Проверить количество дисков, подключенных к контроллеру. Значение должно быть равным четырем.
     3. Перейти во вкладку Configure - Create Virtual Drive.
     4. В поле Select RAID Level выбрать RAID10.
     5. Добавить два диска в Span1. Зайти в Span 1 - Select Drives. Выбрать тип SSD, добавить два диска и применить изменения.
     6. Создать еще один Span и по аналогии добавить два оставшихся диска.
     7. Сохранить конфигурацию.
     8. Если сохранение прошло успешно, то должно появиться сообщение, как показано на рисунке ниже.
     9. Нажать ESC и перейти к дальнейшему конфигурированию;
   - **установка корректного времени**

     (перед установкой системы необходимо правильно установить время в bios в разделе Main);
   - **отключение загрузки по сети**

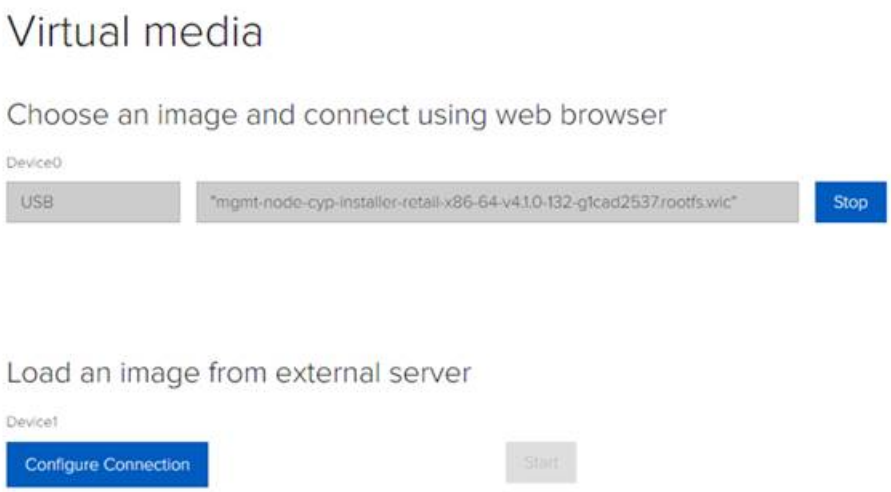
     (перейти в Advanced - Network Stack Configuration. Выставить значения полей в Disabled, как показано на рисунке ниже);
   - **отключение Hyper Threading**

     (в разделе Socket Configuration - Processor Configuration отключить Hyper Threading).

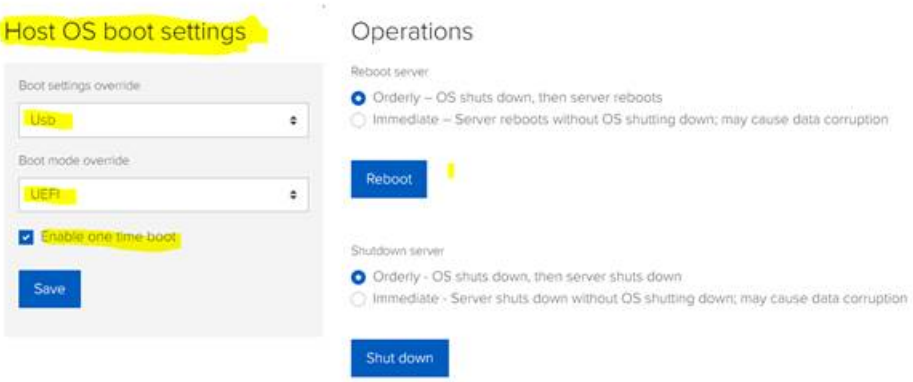
Установка ОС

Контрольная сумма установочного образа находится в файле с расширением .sha256. Перед монтирование образа необходимо сверить значение контрольной суммы.

В разделе Virtual media подключить установочный образ через браузер или через внешний сервер (если есть ftp или nginx).



В разделе Server power operations установить необходимые настройки Host OS boot settings.

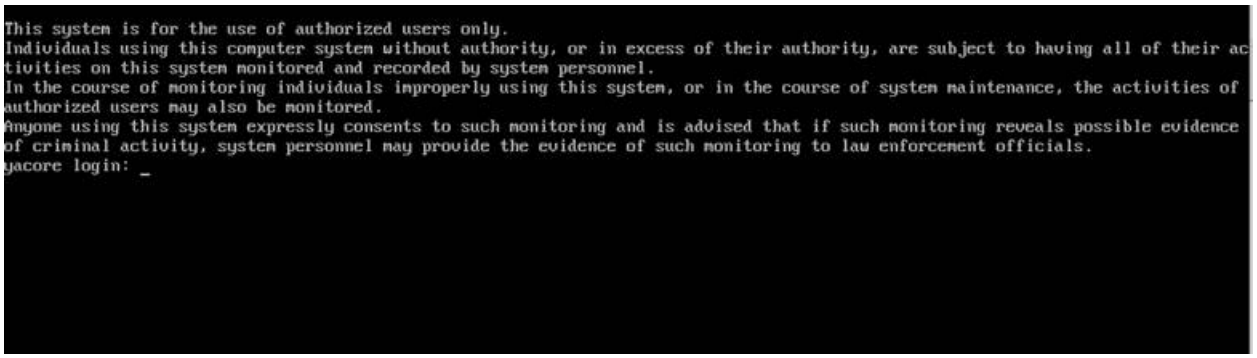


Нажать Reboot для перезагрузки сервера, дождаться начала загрузки образа.

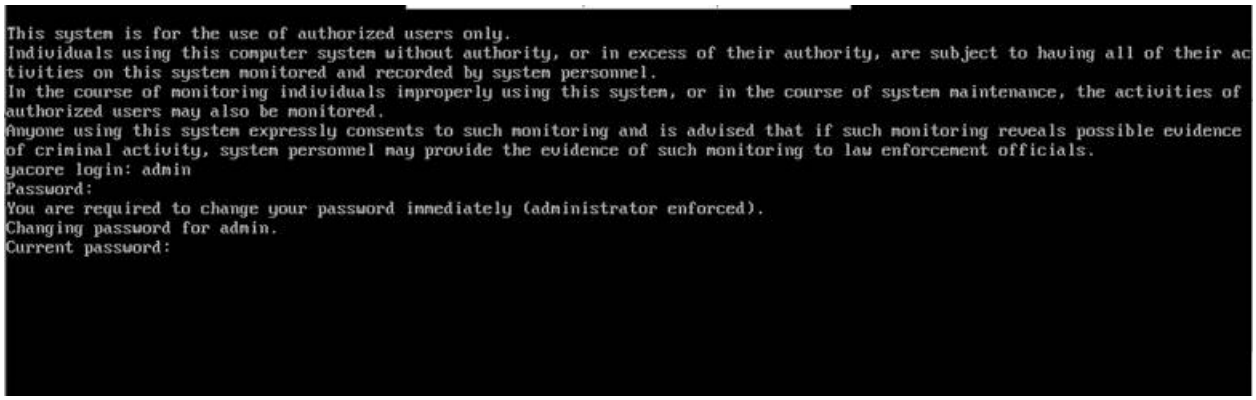
В SOC (Serial over LAN) консоли BMC UI после перезагрузки запустится инсталлятор загрузочного образа.

Дождаться окончания работы инсталлятора.

После успешного развертывания образа инсталлятором появится приглашение в систему.

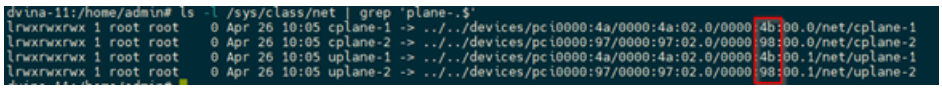


Далее необходимо осуществить первый вход со стандартным паролем и после входа изменить пароль.



Скорректировать PCI.

Если команда ls -l /sys/class/net | grep 'plane-.$' выводит PCI отличный от стандартных 0000:4b:00 / 0000:17:00 (см. пример ниже), то необходимо отразить корректный PCI в файлах /etc/systemd/network/*plane-*.link и произвести reboot сервера.



Распаковка пакета поставки

Пакет поставки представляет из себя два tar.gz архива:

- playbook-worker-node - ansible плейбук, содержащий роли по конфигурированию YACORE worker серверов;
- playbook-management-node - ansible плейбук, содержащий роли для конфигурирования YACORE management серверов.

1. Для того, чтобы распаковать плейбуки необходимо создать рабочую директорию и распаковать в нее содержимое архива:

   ```
   YACORE_DEPLOY=~/yacore_deploy
   ```

   ```
   mkdir -p $YACORE_DEPLOY && tar -xzvf yacore_deployment_package_<release_version>.tar.gz -C $YACORE_DEPLOY
   ```

2. Далее необходимо перейти в директорию с деплоем и запустить скрипт предварительной конфигурации:

   ```
   cd $YACORE_DEPLOY &&./predeploy.sh
   ```

   Пример вывода скрипта predeploy.sh

   ```
   Define package manager
   ```

   ```
   Check if openssl exists
   ```

   ```
   Openssl has been already installed
   ```

```
Error: ATE docker image with version 1.9.0 is not loaded
205c1a638f9c: Loading layer [==================================================>]
229.6MB/229.6MB
875493cc44d3: Loading layer [==================================================>]
1.536kB/1.536kB
c6763ed194d2: Loading layer [==================================================>]
18.12MB/18.12MB
5fd1e8d35303: Loading layer [==================================================>]
2.56kB/2.56kB
628e063141c0: Loading layer [==================================================>]
542.2MB/542.2MB
91ba7c9e7f2c: Loading layer [==================================================>]
140.3kB/140.3kB
5598f8e33e2e: Loading layer [==================================================>]
1.357MB/1.357MB
1f4450d89b4c: Loading layer [==================================================>]
311.8kB/311.8kB
ce7f23e3c1b4: Loading layer [==================================================>]
19.46kB/19.46kB
7b3b3d1e4299: Loading layer [==================================================>]
247.8kB/247.8kB
4db16d358990: Loading layer [==================================================>]
2.265MB/2.265MB
Loaded image: ansible-tarantool-enterprise: 1.9.0
```

Конфигурация сервера

Поставка YACORE состоит из двух типов серверов: Management и Worker.

Настройка каждого типа серверов осуществляется с помощью Ansible. Для каждого типа сервера предназначен свой плейбук.

- playbook-worker-node - ansible плейбук для worker нод;
- playbook-management-node - ansible плейбук для management нод.

Перед запуском плейбука необходимо подготовить inventory файлы согласно имеющемуся LLD.

Далее необходимо создать файл с переменными *host vars* и заполнить переменные данными для настройки сервера, название файла должно соответствовать имени хоста в файле *hosts.yml*.

Подготовленный файл с переменными поместить в папку *host_vars* плейбука.

Ниже описаны переменные плейбука, которые требуются для корректной настройки хоста. Полный список переменных можно посмотреть в файле README.md каждой роли.

Пример готового файла с переменными можно посмотреть в host_vars/cyp-yacore-vm-1.tel.yadro.com-.yml.

Условное выполнение ролей в плейбуках

Часть ролей в плейбуках выполняется только если необходимо развернуть конкретный компонент на сервере. К таким ролям относятся: tarantool, vpp, bird.

Для того чтобы данные роли выполнились, необходимо добавить их в переменную **host_groups**:

```
host_groups:
                - tarantool-udr
                - tarantool-udsf
```

или

```
host_groups:
                - vpp
                - bird
```

tarantool-udr и tarantool-udsf это разные группы и указываются отдельно. Группы для развертывания tarantool не существует.

Роли плейбука playbook-worker-node

Курсивом указаны роли, которые выполняются только если у хоста в переменных есть соответствующая группа

Роль	Описание
hostname	Устанавливает hostname в системе
fqdn	Добавляет или удаляет записи fqdn/ip в файле /etc/hosts
network	Настраивает сетевые интерфейсы и маршруты, включая VLAN
ntp	Настройка клиента NTP
node-exporter	Устанавливает и настраивает Prometeus node exporter
yacore-service	Настраивает и запускает сервисы Yacore
snmpd	Настраивает службу snmpd
vpp	Настраивает и запускает сервис vpp
bird	Настраивает и запускает сервис bird (необходим для поддержки BGP)
tarantool	Подготавливает хост к установке кластера Tarantool. Если на хосте будут работать два кластера: и для UDR и для UDSF, то роль запускается два раза с разными настройками

Роли плейбука playbook-management-node

Курсивом указаны роли, которые выполняются только если у хоста в переменных есть соответствующая группа

Роль	Описание
hostname	Устанавливает hostname в системе
network	Настраивает сетевые интерфейсы и маршруты, включая VLAN
fqdn	Добавляет или удаляет записи fqdn/ip в файле /etc/hosts
node-exporter	Устанавливает и настраивает Prometeus node exporter
snmp-exporter	Устанавливает и настраивает snmp exporter
vmagent	Устанавливает и настраивает vmagent для victoriametrics
victoriametrics	Устанавливает и настраивает Victoriametrics
clickhouse	Устанавливает и настраивает БД Clickhouse (для yacore-collector)
grafana	Устанавливает и настраивает Grafana

Роль	Описание
ntp	Настройка клиента NTP
rsyslogd	Настраивает службу rsyslog для логгирования
nginx	Устанавливает и настраивает NGINX (для LMT)
core-oam	Настраивает OAM agent
snmpd	Настраивает службу snmpd
yacore-collector	Устанавливает и настраивает yacore-backend и yacore-fronted для сервиса yacore-collector
keepalived	Настраивает keepalived сервис
yacore-service	Настраивает и запускает сервисы Yacore (ProvGW на management node)
tarantool	Подготавливает хост к установке кластера Tarantool. Если на хосте будут работать два кластера: и для UDR и для UDSF, то роль запускается два раза с разными настройками

Генерация сертификатов

Для деплоя ETCD, Tarantool, OAM LMT требуется генерация сертификатов для работы по протоколу TLS.

Сертификаты генерируются ролью certificates. Рекомендуется создать отдельный инвентори файл, содержащий настройки всех необходимых сертификатов.

Генерация сертификатов осуществляется через запуск плейбука.

```
ansible-playbook certificates.yml
                - i  host_vars/etcd-certificates.yml  --tags
                'create-certs'--connection=local
```

Сертификаты будут сгенерированы в директорию */usr/share/certs/YADRO/* на машине, с которой запускался плейбук.

Сертификаты для etcd

Пример инвентори файла генерации сертификатов для кластера ETCD приведен ниже.

etcd_certificates.yml

```
---
certificates:
hosts:
localhost:
ansible_ssh_host: 127.0.0.1
ansible_user: yuser
certificates_list_ca:
- domain: "root"
days: 3650
hash_alg: sha384
default_bits: 2048
country: RU
organization: yadro.mnc32.mcc901.3gppnetwork.org
common_name: YADRO TLS Issuing CA
ca_password: yadro
regenerate: false
intermediate_certificates:
- issuer: "root"
domain: "tnt"
days: 3650
hash_alg: sha384
default_bits: 2048
country: RU
organization: node.mnc32.mcc901.3gppnetwork.org
common_name: NODE TLS Issuing CA
intermediate_password: yadro
regenerate: false
create_chained: true
domains_to_sign:
- domain: "etcd"
```

```
                intermediate: "tnt"
                days: 3650
                hash_alg: sha384
                default_bits: 2048
                country: RU
                organization: node.mnc32.mcc901.3gppnetwork.org
                common_name: etcd.node.mnc32.mcc901.3gppnetwork.org
                intermediate_password: yadro
                ca: root
                pkcs12_password: 123456789
                regenerate: false
                - domain: "etcd1"
                intermediate: "tnt"
                days: 3650
                hash_alg: sha384
                default_bits: 2048
                country: RU
                organization: node1.mnc032.mcc901.3gppnetwork.org
                common_name: etcd.node1.mnc032.mcc901.3gppnetwork.org
                intermediate_password: yadro
                ca: root
                pkcs12_password: 123456789
                regenerate: false
                subject_alt_name:
                ip: 2001:ab8:efdb::100a
                fqdn: etcd.node1.mnc032.mcc901.3gppnetwork.org
                - domain: "etcd2"
                intermediate: "tnt"
                days: 3650
                hash_alg: sha384
                default_bits: 2048
                country: RU
                organization: node2.mnc032.mcc901.3gppnetwork.org
                common_name: etcd.node2.mnc032.mcc901.3gppnetwork.org
                intermediate_password: yadro
                ca: root
                pkcs12_password: 123456789
                regenerate: false
                subject_alt_name:
                ip: 2001:ab8:efdb::200a
                fqdn: etcd.node2.mnc032.mcc901.3gppnetwork.org
                - domain: "etcd3"
                intermediate: "tnt"
                days: 3650
                hash_alg: sha384
                default_bits: 2048
                country: RU
                organization: node3.mnc032.mcc901.3gppnetwork.org
                common_name: etcd.node3.mnc032.mcc901.3gppnetwork.org
                intermediate_password: yadro
                ca: root
                pkcs12_password: 123456789
                regenerate: false
                subject_alt_name:
                ip: 2001:ab8:efdb::a00a
                fqdn: etcd.node3.mnc032.mcc901.3gppnetwork.org
```

Сертификаты для ПО Тарантул Плюс

Пример инвентори файла генерации сертификатов для ПО Тарантул Плюс приведен ниже.

tnt_certificates.yml

```
                ---
                certificates:
                hosts:
                localhost:
                ansible_ssh_host: 127.0.0.1
                ansible_user: yuser
                certificates_list_ca:
                - domain: "root"
                days: 3650
                hash_alg: sha384
                default_bits: 2048
                country: RU
                organization: yadro.mnc32.mcc901.3gppnetwork.org
                common_name: YADRO TLS Issuing CA
                ca_password: yadro
                regenerate: false
                intermediate_certificates:
                - issuer: "root"
                domain: "tnt"
                days: 3650
                hash_alg: sha384
                default_bits: 2048
                country: RU
```

```
organization: node.mnc32.mcc901.3gppnetwork.org
common_name: NODE TLS Issuing CA
intermediate_password: yadro
regenerate: false
create_chained: true
domains_to_sign:
- domain: "provgw"
intermediate: "tnt"
days: 3650
hash_alg: sha384
default_bits: 2048
country: RU
organization: node.mnc32.mcc901.3gppnetwork.org
common_name: provgw.node.mnc32.mcc901.3gppnetwork.org
intermediate_password: yadro
ca: root
pkcs12_password: 12345678
regenerate: false
- domain: "udr-udsf"
intermediate: "tnt"
days: 3650
hash_alg: sha384
default_bits: 2048
country: RU
organization: node.mnc32.mcc901.3gppnetwork.org
common_name: udr-udsf.node.mnc32.mcc901.3gppnetwork.org
intermediate_password: yadro
ca: root
pkcs12_password: 123456789
regenerate: false
- domain: "router1"
intermediate: "tnt"
days: 3650
hash_alg: sha384
default_bits: 2048
country: RU
organization: node.mnc32.mcc901.3gppnetwork.org
common_name: router1.tntrouter.node.mnc32.mcc901.3gppnetwork.org
intermediate_password: yadro
ca: root
pkcs12_password: 123456789
regenerate: false
- domain: "router2"
intermediate: "tnt"
days: 3650
hash_alg: sha384
default_bits: 2048
country: RU
organization: node.mnc32.mcc901.3gppnetwork.org
common_name: router2.tntrouter.node.mnc32.mcc901.3gppnetwork.org
intermediate_password: yadro
ca: root
pkcs12_password: 123456789
regenerate: false
- domain: "storage1"
intermediate: "tnt"
days: 3650
hash_alg: sha384
default_bits: 2048
country: RU
organization: node.mnc32.mcc901.3gppnetwork.org
common_name: storage1.tntstorage.node.mnc32.mcc901.3gppnetwork.org
intermediate_password: yadro
ca: root
pkcs12_password: 123456789
regenerate: false
- domain: "storage2"
intermediate: "tnt"
days: 3650
hash_alg: sha384
default_bits: 2048
country: RU
organization: node.mnc32.mcc901.3gppnetwork.org
common_name: storage2.tntstorage.node.mnc32.mcc901.3gppnetwork.org
intermediate_password: yadro
ca: root
pkcs12_password: 123456789
regenerate: false
- domain: "storage3"
intermediate: "tnt"
days: 3650
hash_alg: sha384
default_bits: 2048
country: RU
organization: mnc32.node.mcc901.3gppnetwork.org
common_name: storage3.tntstorage.node.mnc32.mcc901.3gppnetwork.org
intermediate_password: yadro
```

```
                ca: root
                pkcs12_password: 123456789
                regenerate: false
                - domain: "trusted-client"
                intermediate: "tnt"
                days: 3650
                hash_alg: sha384
                default_bits: 2048
                country: RU
                organization: mnc32.node.mcc901.3gppnetwork.org
                common_name: trusted-client.tntstorage.node.mnc32.mcc901.3gppnetwork.org
                intermediate_password: yadro
                ca: root
                pkcs12_password: 123456789
                regenerate: false
```

Сертификаты для LMT, OAM, BSS, Colba

Можно объединить инвентори для всех сертификатов в один файл и запускать одной командой.

Пример инвентори файла генерации сертификатов для работы LMT, OAM, NMS, BSS и Colba приведен ниже.

oambss_certificates.yml

```
                ---
                certificates:
                hosts:
                localhost:
                ansible_ssh_host: 127.0.0.1
                ansible_user: yuser
                certificates_list_ca:
                - domain: "root"
                days: 3650
                hash_alg: sha384
                default_bits: 2048
                country: RU
                organization: yadro.mnc32.mcc901.3gppnetwork.org
                common_name: YADRO TLS Issuing CA
                ca_password: yadro
                regenerate: false
                intermediate_certificates:
                - issuer: "root"
                domain: "bss"
                days: 3650
                hash_alg: sha384
                default_bits: 2048
                country: RU
                organization: node.mnc32.mcc901.3gppnetwork.org
                common_name: NODE TLS Issuing CA
                intermediate_password: yadro
                regenerate: false
                create_chained: true
                domains_to_sign:
                - domain: "nms"
                intermediate: "bss"
                days: 3650
                hash_alg: sha384
                default_bits: 2048
                country: RU
                organization: node.mnc32.mcc901.3gppnetwork.org
                common_name: nms.node.mnc32.mcc901.3gppnetwork.org
                intermediate_password: yadro
                ca: root
                pkcs12_password: 12345678
                regenerate: false
                - domain: "oam"
                intermediate: "bss"
                days: 3650
                hash_alg: sha384
                default_bits: 2048
                country: RU
                organization: node.mnc32.mcc901.3gppnetwork.org
                common_name: oam.node.mnc32.mcc901.3gppnetwork.org
                intermediate_password: yadro
                ca: root
                pkcs12_password: 12345678
                regenerate: false
                - domain: "lmt-server"
                intermediate: "bss"
                days: 3650
                hash_alg: sha384
                default_bits: 2048
                country: RU
                organization: node.mnc32.mcc901.3gppnetwork.org
```

```
                common_name: lmt-server.node.mnc32.mcc901.3gppnetwork.org
                intermediate_password: yadro
                ca: root
                pkcs12_password: 123456789
                subject_alt_name:
                ip: 10.78.228.231
                regenerate: false
                - domain: "lmt-client"
                intermediate: "bss"
                days: 3650
                hash_alg: sha384
                default_bits: 2048
                country: RU
                organization: node.mnc32.mcc901.3gppnetwork.org
                common_name: lmt-client.node.mnc32.mcc901.3gppnetwork.org
                intermediate_password: yadro
                ca: root
                pkcs12_password: 123456789
                regenerate: false
                - domain: "colba-ui"
                intermediate: "bss"
                days: 3650
                hash_alg: sha384
                default_bits: 2048
                country: RU
                organization: node.mnc32.mcc901.3gppnetwork.org
                common_name: colba-ui.node.mnc32.mcc901.3gppnetwork.org
                intermediate_password: yadro
                ca: root
                pkcs12_password: 123456789
                regenerate: false
```

Описание переменных для деплоя

Общие переменные

Общие переменные используются во всех плейбуках, это переменные, описывающие доступ к серверу.

```
ansible_user: admin
                ansible_ssh_host: 192.168.100.111
                ansible_password: YaDr0@)24Core
                ansible_become_pass: "{{ansible_password}}"
```

Также необходимо указать OAM IP адрес, который будет сконфигурирован на сервере, и его хостнейм (по умолчанию совпадает с именем хоста в hosts.yml).

```
mgmt_ip: "10.78.244.3"
                # configure hostname
                hostname_name: "{{inventory_hostname}}"
```

playbook-worker-node и playbook-management-node имеют некоторое общее множество используемых ansible roles, поэтому некоторые переменные для конфигурации этих ролей идентичны.

Переменные для настройки NTP

```
# ntp configuration example ntp_servers:
                address:
                - 91.189.91.157
                - 10.128.80.23
                minpoll: 4
                maxpoll: 8
```

Переменные для настройки FQDN

Для добавления записей fqdn/ip в файл /etc/hosts необходимо в host_vars определить словарный список fqdn_add в котором необходимо будет перечислить список fqdn и соответствующих им IP адресов с помощью переменных host и ip:

Переменная	Описание
host	Значение FQDN

Переменная	Описание
ip	Соответствующий IP адрес для FQDN

Для удаления записей fqdn/ip в файл /etc/hosts необходимо в host_vars определить словарный список fqdn_del в котором аналогично, как с fqdn_add, необходимо будет перечислить список fqdn и соответствующих им IP адресов с помощью переменных host и ip. Пример конфигурации представлен ниже:

```
fqdn_del:
                -  host: test1.yadro
                ip: 127.0.1.2
                -  host: test2.yadro
                ip: 11.11.11.2
                -  host: test3.yadro
                ip: 11.11.11.3


                fqdn_add:
                -  host: test1.yadro
                ip: 127.0.1.1
                -  host: test2.yadro
                ip: 11.11.11.2
                -  host: test3.yadro
                ip: 11.11.11.3
```

Переменные для настройки node-exporter

Для конфигурации сервиса node-exporter используются следующие переменные:

Переменная	Описание
node_exporter_listen_address	Прослушиваемый IP адрес и номер порта
node_exporter_telemetry_path	Путь по которому отдаются метрики

Пример конфигурации представлен ниже:

```
node_exporter_listen_address: {{ mgmt_ip }}:9100
                node_exporter_telemetry_path: /metrics
```

Переменные для настройки SNMPD

Для конфигурации сервиса snmpd используются следующие переменные:

Переменная	Описание
ip_oam	IP адрес прослушиваемый snmpd сервисом
ip_for_traps	IP адрес целевого сервера для отправки трапов
community	Парольная фраза для отдачи метрик по SNMP
sysLocation	Физическое расположение устройства
sysServices	Тип сервисов предоставляемый устройством
sysContact	Контакт администратора устройства
guest_user	Имя пользователя для доступа по SNMP
MD5	Пароль аутентификации для пользователя {{ gues_user }} по алгоритму MD5
DES	Пароль для шифрования SNMP трафика пользователя {{ gues_user }} по алгоритму DES

Пример конфигурации представлен ниже:

```
ip_oam: {{mgmt_ip}}
                ip_for_traps: 10.78.244.10
                community: Bureau1440
                sysLocation:  "Bureau-1440"
                sysContact:  "admin@example.com"
                sysServices: "72"
                guest_user: "yadro"
                MD5: "123456"
                DES: "123456"
```

Переменные для настройки ПО Тарантул Плюс

Настройка ПО Тарантул Плюс для YACORE состоит из двух этапов:

1. Подготовка сервера к развертыванию картриджа ПО Тарантул Плюс с помощью плейбука деплоя.
2. Установка картриджа ПО Тарантул Плюс на подготовленные сервера с помощью плейбука от VK (ПО Тарантул Плюс не входит в пакет поставки и приобретается заказчиком отдельно).

В данном разделе представлены переменные, необходимые для выполнения первого этапа - подготовки сервера.

Кластер ETCD разворачивается через плейкук деплоя ПО Тарантул Плюс от VK.

ПО Тарантул Плюс и ETCD разворачиваются на хостах средствами ATE (Ansible Tarantool Enterprise).

Заполняем переменные, необходимые чтобы подготовить сервер под размещение кластера ПО Тарантул Плюс и ETCD.

Необходимо указать в переменной tarantool_ssh_public_key публичный ключ, для копирования на целевой хост (плейбук деплоя ATE получает доступ на хост через этот ключ): `tarantool_ssh_public_key: "{{playbook_dir}}/files/ssh/tarantool_rsa.pub"`.

Отдельно необходимо настраить переменные для TLS.

```
tarantool_copy_certs_from_local: true
                tarantool_tls_params:
                - domain: tnt
                ca: "/etc/opt/tarantool/pki/ca/chained-tnt.pem"
                cert: "/etc/opt/tarantool/pki/cert/router1.pem"
                key: "/etc/opt/tarantool/pki/key/router1.key"
                - domain: tnt
                ca: "/etc/opt/tarantool/pki/ca/chained-tnt.pem"
                cert: "/etc/opt/tarantool/pki/cert/storage1.pem"
                key: "/etc/opt/tarantool/pki/key/storage1.key"
                - domain: tnt
                ca: "/etc/opt/tarantool/pki/ca/chained-tnt.pem"
                cert: "/etc/opt/tarantool/pki/cert/storage2.pem"
                key: "/etc/opt/tarantool/pki/key/storage2.key"
                - domain: tnt
                ca: "/etc/opt/tarantool/pki/ca/chained-tnt.pem"
                cert: "/etc/opt/tarantool/pki/cert/storage3.pem"
                key: "/etc/opt/tarantool/pki/key/storage3.key"
                - domain: tnt
                ca: "/etc/opt/tarantool/pki/ca/tnt.pem"
                cert: "/etc/opt/tarantool/pki/cert/trusted-router.pem"
                key: "/etc/opt/tarantool/pki/key/trusted-router.key"
                - domain: tnt
                ca: "/etc/opt/tarantool/pki/ca/tnt.pem"
                cert: "/etc/opt/tarantool/pki/cert/trusted-storage.pem"
                key: "/etc/opt/tarantool/pki/key/trusted-storage.key"
```

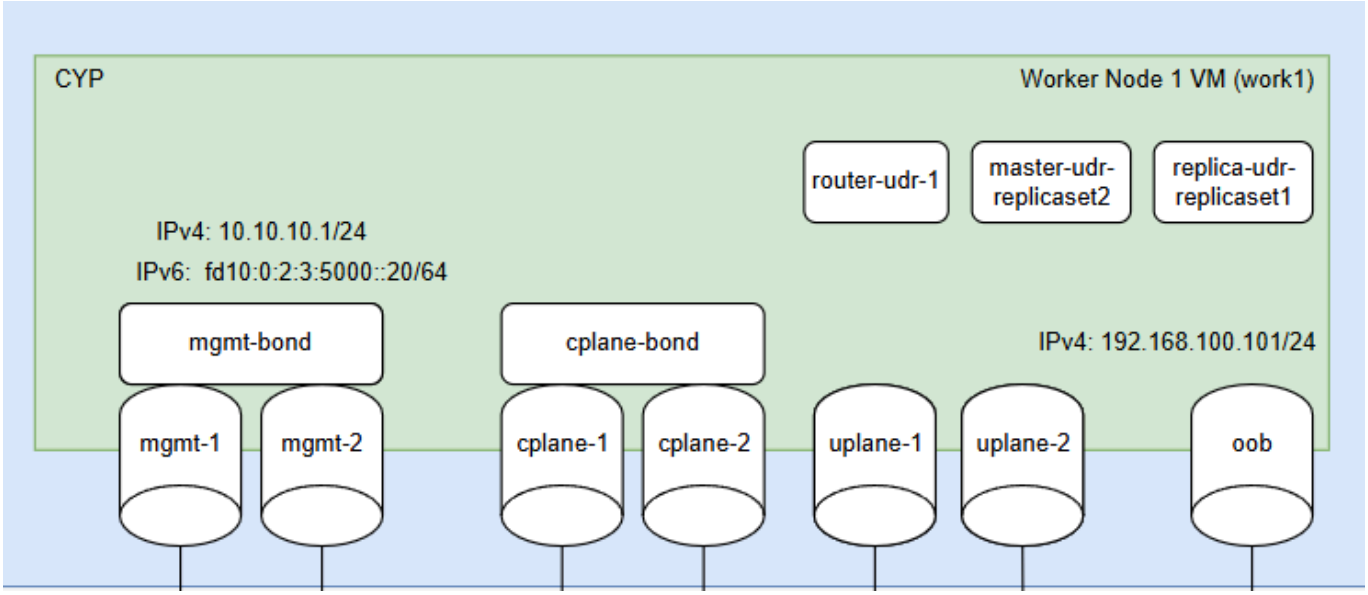
Необходимо добавить группы для тарантула в инвентори: `host_groups: - tarantool`.

Переменные для worker-node

Настройка сетевых интерфейсов

Целевая схема сети для сервера worker-node представлена ниже.

Схема сети в LLD может отличаться от представленной в данном руководстве!



После установки образа CYP необходимо сконфигурировать IP адреса, VLAN и необходимые IP маршруты на интерфейсах mgmt-bond и cplane-bond.

Описание переменных

Переменная	Описание
mgmt_network	Конфигурация сети для management
cplane_networks	Конфигурация сети для CPLANE
vrfs	Конфигурация VRF для DNN

Переменные для конфигурации VLAN's

VLAN конфигурируются внутри *mgmt_network* или *cplane_networks* в переменной *vlans*.

Переменная	Описание
name	Имя VLAN'а
vlan_id	Номер VLAN
description	Описание
sshd_listen	Открыть доступ по SSH на интерфейсе (применима только для VLAN mgmt-bond)
addresses	Список IP адресов для VLAN
routes	Список маршрутов для VLAN

Для OAM VLAN обязательно указать параметр sshd_listen.

Переменные маркировки egress трафика IP пакетов и кадров ethernet метками DSCP и PCP

Маркировка конфигурируется внутри переменной *vlans*.

Переменная	Описание
egress_dscp	номер метки DSCP для IP пакета
egress_pcp	номер метки PCP для кадра Ethernet

Пример настройки сети для менеджмента.

```
# network interfaces configuration
                mgmt_network:
                bond_address: "{{mgmt_ip}}/24"
                bond_gateway: 10.78.244.1
                vlans:
                - name: oam
                vlan_id: 100
                description: "oam network interface"
                sshd_listen: yes
                addresses:
                - 217.18.134.6/24
                routes:
                - gateway: 217.18.134.1
                destination: 217.18.134.0/24
                marking:
                egress_dscp: 40
                egress_pcp: 5
```

Можно назначить IP адрес на самом интерфейсе, можно создать VLAN (если это заложено в LLD), и прописать дополнительную адресацию там.

Настройка VLAN для CPLANE производится аналогично, создается согласно имеющегося LLD.

```
cplane_networks:
                vlans:
                - name: n2
                vlan_id: 10
                description: "n2 network interface"
                addresses:
                - 2001:ab8:ef02::2001/96
                routes:
                - gateway: 2001:ab8:ef02::1
                destination: 3101:ab8:ef02::/32
                - name: n4
                vlan_id: 40
                description: "n4 network interface"
                addresses:
                - 2001:ab8:ef04::4001/96
                routes:
                - gateway: 2001:ab8:ef04::111
                destination: 2001:abc:ef04::/64
                - gateway: 2001:ab8:ef04::111
                destination: 2001:ab0:ef04::/64
                - name: sbi
                vlan_id: 20
                description: "sbi network interface"
                addresses:
                - 2001:ab8:ef0b::4001/96
                marking:
                egress_dscp: 40
                egress_pcp: 5
                - name: db
                vlan_id: 30
                description: "db network interface"
                addresses:
                - 2001:ab8:efdb::4001/96
```

Переменные для настройки VRF

Если в LLD предусмотрены VRF для DNN, то указывать их в переменной *vrfs* при настройке сети.

```
vrfs:
                - id: internet
                table_id: 3131
                - id: yadro
                table_id: 3132
```

Переменные для настройки yacore-service

Задать переменные, общие для всех NF Yacore.

Переменная	Описание
yacore_service_config_path	Путь до файлов конфигурации Yacore на сервере
yacore_mgmt_node_ip	IP адрес Management ноды с OAM агентом, Grafana, etc.

Переменная	Описание
keepalived_virtual_ip	Отказоустойчивый виртуальный IP который выдается из сети otel и управляется keepalived
yacore_oam_ip	OAM IP адрес для yacore-app Yacore (обычно совпадает с mgmt IP самого сервера)
yacore_slice_cpus	Номера CPU, выделенных под работу NF Yacore
yacore_mgmt_node_fqdn	FQDN, который резолвится в IP адрес Management ноды с OAM агентом, Grafana, etc.
yacore_oam_ip_fqdn	FQDN, который резолвится в OAM IP адрес для yacore-app Yacore (обычно совпадает с mgmt IP самого сервера)

```
yacore_service_config_path: "/etc/opt/yacore/config"
                yacore_mgmt_node_ip: 10.78.244.4
                yacore_oam_ip: "{{mgmt_ip}}"
                yacore_slice_cpus: "1-18"
```

Создать переменные, общие для всех NF Yacore, чтобы не дублировать код.

Настройки OAM агента представлены ниже.

```
yacore_oam_agent:
                client:
                ipAddr:
                ipv4: "{{ yacore_oam_ip }}"
                server:
                ipAddrWithPort:
                ipv4WithPort:
                addr: "{{ yacore_mgmt_node_ip }}"
                port: 3380
```

Перед настройкой секций логирования и трэйсинга, необходимо убедиться в корректности заданной переменной keepalived_virtual_ip.

Настройки трейсинга и метрик для OTEL.

```
yacore_otel_config:
                tracing:
                collectorAddressList:
                - ipAddr:
                ipv4: "{{ keepalived_virtual_ip }}"
                exportingPeriodMilliseconds: 100
                port: 14317
                metrics:
                collectorAddressList:
                - ipAddr:
                ipv4: "{{ yacore_mgmt_node_ip }}"
                reportingPeriodMinutes: 1
```

Настройки логгирования.

```
yacore_log_config:
                level: "Debug"
                handlers:
                crash-handler:
                handlerType: "crash"
                config:
                path: "/var/log/yacore"
                maxFilesCountPerShard: 10
                needResolveStacktrace: true
                stdout-handler:
                handlerType: "stdout"
                config:
                severity: "Debug"
                RsyslogYacoreHandler:
                handlerType: "rsyslog"
                config:
                destination: "{{ yacore_mgmt_node_fqdn }}"
                source: "{{ yacore_oam_fqdn }}"
                port: 514
                severity: "Debug"
                transportLayer: "UDP"
```

```
                RsyslogYacoreHandler-Collector:
                handlerType: "rsyslog"
                config:
                destination: "{{ yacore_mgmt_node_fqdn }}"
                source: "{{ yacore_oam_fqdn }}"
                port: 22514
                severity: "Debug"
                transportLayer: "TCP"
```

Значения этих переменных могут отличаться!

Заполняем переменную yacore_services данными для yacore-app каждой NF.

Переменная содержит в себе yacore-app конфиг в виде YAML. Описание конкретных полей можно узнать в соответствующих спецификациях.

Ниже пример для двух NF, обратите внимание - в конфигах используются общие переменные, созданные выше.

```
debugging:
                disable_http2_dynamic_headers_table: true
```

Пример заполнения переменной *yacore_services* для нескольких NF.

```
yacore_services:
  - yacore_service_name: "amf"
    yacore_service_config:
      name: AMF01
      nfConfigFilePath: /etc/opt/yacore/config/amf-config.json
      platform:
        cpuset: "1"
        memory: 40960Mi
      plugins:
        - name: amf
      debugging:
        disable_http2_dynamic_headers_table: true
      otel: "{{yacore_otel_config}}"
      oam:
        server:
          address:
            ipAddrWithPort:
              ipv4WithPort:
                addr: "{{ yacore_oam_ip }}"
                port: 3381
          ca:
            - ca1
            - ca2
            - ca3
        agent: "{{ yacore_oam_agent }}"
      log: "{{ yacore_log_config }}"
      connections:
        total: 100
        UDSF: 10
        smptest: 20
  - yacore_service_name: "smf"
    yacore_service_config:
      name: SMF01
      nfConfigFilePath: /etc/opt/yacore/config/smf-config.json
      platform:
        cpuset: "2"
        memory: 40960Mi
      plugins:
        - name: smf
      otel: "{{yacore_otel_config}}"
      oam:
        server:
          address:
            ipAddrWithPort:
              ipv4WithPort:
                addr: "{{ yacore_oam_ip }}"
                port: 3382
          ca:
            - ca1
            - ca2
            - ca3
        agent: "{{ yacore_oam_agent }}"
      log: "{{ yacore_log_config }}"
      connections:
        total: 100
        UDSF: 10
        smptest: 20
  - yacore_service_name: "udsf"
```

```
    yacore_copy_certs_from_local: true
    yacore_service_config:
      name: UDSF01
      nfConfigFilePath: /etc/opt/yacore/config/udsf-config.json
      platform:
        cpuset: "6,10,11"
        memory: 40960Mi
      plugins:
        - name: udsf
          params:
            db:
              password: 8TP6WP4znss
              username: udsf
              tarantoolRouters:
                - address:
                    fqdn: "router1udsf.tntrouter.rcys1.mnc032.mcc901.3gppnetwork.org"
                  port: 4501
                  priority: 1
                - address:
                    fqdn: "router2udsf.tntrouter.rcys1.mnc032.mcc901.3gppnetwork.org"
                  port: 4503
                  priority: 2
            tls:
              - domain: "tnt"
                params:
                  caFile: "{{ yacore_cert_path }}/udsf/caFile/chained-tnt.pem"
                  cert: "{{ yacore_cert_path }}/udsf/cert/udr-udsf.pem"
                  key: "{{ yacore_cert_path }}/udsf/key/udr-udsf.key"
      debugging:
        disable_http2_dynamic_headers_table: true
      otel: "{{yacore_otel_config}}"
      oam:
        server:
          address:
            ipAddrWithPort:
              ipv4WithPort:
                addr: "{{ yacore_oam_ip }}"
                port: 3386
          ca:
            - ca1
            - ca2
            - ca3
        agent: "{{ yacore_oam_agent }}"
      log: "{{ yacore_log_config }}"
      connections:
        total: 100
        UDSF: 10
        smptest: 20
```

Переменные для настройки vpp

Описание переменных представлено ниже.

Переменная	Описание
vpp_cpu_main_core	CPU для главного обработчика
vpp_cpu_corelist_workers	Список CPU для обработчиков трафика (не должны пересекаться с vpp_cpu_main_core)
vpp_dpdk_interface	PCI адреса DPDK сетевых интерфейсов
vpp_cfg	Конфиг VPP

При указании DPDK PCI адресов в VPP будут автоматически созданы соответствующие интерфейсы .

Например мы указали интерфейс с адресом *"0000:98:00.1"* и *"0000:17:00.1"*.

```
vpp_dpdk_interface:
                - address: "0000:98:00.1"
                receive_queues: 2
                - address: "0000:17:00.1"
                receive_queues: 2
```

В VPP будет создан интерфейсы с именем *TwentyFiveGigabitEthernet98/0/1* и *TwentyFiveGigabitEthernet17/0/1* . При дальнейшей конфигурации интерфейсов **an** и **cn** следует ссылаться на DPDK интерфейс по этому имени.

Описание структуры vpp_cfg

Переменная	Описание
host_iface	Описание интерфейсов хоста, на которых будет работать VPP (включая создание Bond)
an	Параметры интерфейсов типа **an (N3)**
cn	Параметры интерфейсов типа **cn (N6)**

```
vpp_cfg:
            host_iface:
            hw:
            - name: TwentyFiveGigabitEthernet98/0/1
            mtu: 1500
            - name: TwentyFiveGigabitEthernet17/0/1
            mtu: 1500
            bond:
            - name: BondEthernet0
            members:
            - TwentyFiveGigabitEthernet98/0/1
            - TwentyFiveGigabitEthernet17/0/1
```

Описание структуры an (N3) и cn (N6)

Переменная	Описание
vrf	Имя VRF, к которому привязан интерфейс(если его нет, то не указываем переменную)
iface	Привязка к интерфейсу хоста
iface.name	Имя интерфейса
iface.vlan	VLAN ID
iface.mtu	Число MTU для интерфейса
table	Номер таблицы VRF из списка vrfs
ips	Конфигурация IP адресов
routes	Маршруты

В случае деплоя YACORE с поддержкой BGP статические маршруты создавать не нужно.

```
an:
                - iface:
                name: BondEthernet0
                vlan: 16
                table: 16
                ips:
                - 100.0.3.2/30
                - 2001:ab8:1440::1/64
                routes:
                - dst: 11.0.0.0/16
                gw: 11.0.0.1
                - dst: 1002:1002::/64
                gw: 2002:2002::1

                cn:
                - vrf: dnn1
                table: 31
                iface:
                name: BondEthernet0
                vlan: 31
                mtu: 1531
                ips:
                - 31.31.31.1/24
                - 2001:ff01::1/96
                routes:
                - dst: 10.10.10.1/32
                gw: 31.31.31.254
```

Пример конфига с полной настройкой VPP для нескольких dnn

```
vpp_cpu_main_core: 15
vpp_cpu_corelist_workers: 16-19
vpp_dpdk_interface:
  - address: "0000:4b:00.1"
    receive_queues: 2
  - address: "0000:98:00.1"
    receive_queues: 2

vpp_cfg:
  host_iface:
    hw:
      - name: TwentyFiveGigabitEthernet4b/0/1
        mtu: 1500
      - name: TwentyFiveGigabitEthernet98/0/1
        mtu: 1500
    bond:
      - name: BondEthernet0
        members:
          - TwentyFiveGigabitEthernet4b/0/1
          - TwentyFiveGigabitEthernet98/0/1
  an:
    - iface:
        name: BondEthernet0
        vlan: 3102
      table: 3102
      ips:
        - 169.254.254.254/32
        - 2001:ab8:ef03::3001/96
  cn:
    - vrf: internet
      table: 3131
      iface:
        name: BondEthernet0
        vlan: 3131
        mtu: 1531
      ips:
        - 100.65.7.1/29
    - vrf: yadro
      table: 3132
      iface:
        name: BondEthernet0
        vlan: 3132
        mtu: 1531
      ips:
        - 100.65.7.9/29
    - vrf: static
      table: 3133
      iface:
        name: BondEthernet0
        vlan: 3133
        mtu: 1531
      ips:
        - 100.65.7.17/29
```

Переменные для настройки bird

Служба bird реализует протокол динамической маршрутизации BGP, а также протокол обнаружения дефектов связности между маршрутизаторами BFD/.

Все настройки хранятся в переменной *bird* ниже будут описаны структуры данных, хранящиеся внутри этой переменной.

Все структуры данных расположены внутри *bgp.protocols*.

```
bird:
               protocols:
               kernel:
               static:
               bgp:
               bfd:
               yard:
```

Переменные относящиеся к настройке vrf должны быть определены в переменной *vrfs*.

Переменная	Описание
id	Название VRF

Переменная	Описание
table_id	Номер в таблице VRF

Пример конфигурации представлен ниже (важно, чтобы используемая переменная ***vrfs.id*** совпадала с используемой переменной **vrf_id** при конфигурации протоколов в **bird** внутри переменной **protocols**).

```
vrfs:
                - id: internet
                table_id: 10
                - id: yadro
                table_id: 20
                - id: static
                table_id: 20
```

Описание структуры kernel

Содержит список VRF. Настройка внутреннего псевдо-протокола позволяет добавлять маршруты, полученные от BGP соседа в определенном VRF во внутренние таблицы маршрутизации.

```
bird:
                protocols:
                kernel:
                - vrf_id: internet
                - vrf_id: yadro
                - vrf_id: static
```

Описание структуры static

Содержит список статических IPv4, IPv6 маршрутов для каждого VRF. Статические маршруты представляют из себя IP-пулы UE, анонсируемые в дальнейшем по протоколу BGP.

Переменная	Описание
vrf_id	Строковый идентификатор VRF
routes_ipv4, routes_ipv6	Список IPv4, IPV6 подсетей

```
bird:
                protocols:
                static:
                - vrf_id: internet
                routes_ipv4:
                - 10.0.0.0/16
                routes_ipv6:
                - vrf_id: yadro
                routes_ipv4:
                - 10.1.0.0/16
                routes_ipv6:
                - vrf_id: static
                routes_ipv4:
                - 10.2.0.0/16
                routes_ipv6:
```

Описание структуры bgp

Содержит следующие настройки протокола BGP.

Переменная	Описание
session_id	Уникальный строковый идентификатор BGP сессии
vrf_id	Строковый идентификатор VRF, в котором будет подниматься BGP сессия
router_id	Уникальный идентификатор роутера
description	Строковое описание сессии
local_ip	IP адрес, используемый в качестве source адреса в BGP сессии

Переменная	Описание
local_as	Номер локальной автономной системы
neighbor_ip	IP адрес хоста, с которым планируется установить BGP сессию
neighbor_as	Номер автономной системы хоста, с которым планируется установить сессию
hold_time	Время в секундах, необходимое для ожидания сообщения Keepalive от другой стороны, прежде чем считать соединение устаревшим. Эффективное значение согласовывается во время установления сессии и является минимальным из этого настроенного значения и значения, предложенного соседним узлом
keepalive	Время в секундах, обозначающее с какой периодичностью необходимо отправлять сообщения Keepalive
bfd	Включает автоматическую активацию BFD сессии

```
bird:
                protocols:
                bgp:
                - session_id: bgp_internet_PE1
                vrf_id: internet
                router_id: 192.168.100.101
                description: "Some session"
                local_ip: 100.65.8.1
                local_as: 65001
                neighbor_ip: 100.65.8.2
                neighbor_as: 65002
                hold_time: 180
                keepalive: 60
                bfd: true
                - session_id: bgp_internet_PE2
                vrf_id: internet
                router_id: 192.168.100.101
                description: "Some session"
                local_ip: 100.65.8.1
                local_as: 65001
                neighbor_ip: 100.65.8.3
                neighbor_as: 65002
                hold_time: 60
                keepalive: 20
                bfd: true
```

Описание структуры bfd

Содержит список экземпляров протокола BFD.

Переменная	Описание
instance_name	Уникальный строковый идентификатор экземпляра BFD
vrf_id	Строковый идентификатор VRF, в котором будет подниматься BFD сессия
interfaces	Определяет список интерфейсов и параметров BFD для настройки на каждом интерфейсе
interfaces.name	Имя сетевого интерфейса
interfaces.min_rx_interval	Определяет минимальный RX интервал, который анонсируется соседу и учитывается последним при ограничении скорости генерирования контрольный пакетов BFD
interfaces.min_tx_interval	Определяет минимальный TX интервал, который ограничивает скорость отправляемых контрольный пакетов.

Переменная	Описание
interfaces.idle_tx_interval	Чтобы ограничить ненужный трафик в случаях, когда сосед недоступен или не работает BFD, скорость генерируемых пакетов управления BFD ниже, когда сеанс BFD не запущен.
interfaces.multiplier	Определяет интервал multipler*(min_rx_interval/min_tx_interval), по истечении которого, если от соседа не приходит контрольного пакета, сессия считается закрытой
neighbors	Задает список соседей, для которых необходимо явно создать BFD сессию.
neighbors.remote_ip	Определяет IP адреса BFD соседа, для которого необходимо в явном виде создать BFD сессию

```
bird:
                protocols:
                bfd:
                - instance_name: bfd_internet
                vrf_id: internet
                interfaces:
                - name: "dn.3131"
                min_rx_interval: 30
                min_tx_interval: 50
                idle_tx_interval: 300
                multiplier: 10
                neighbors:
                - remote_ip: 100.65.8.2
                - remote_ip: 100.65.8.3
                - instance_name: bfd_yadro
                vrf_id: yadro
                interfaces:
                - name: "dn.3132"
                min_rx_interval: 20
                min_tx_interval: 50
                idle_tx_interval: 300
                multiplier: 10
                neighbors:
                - remote_ip: 100.65.8.10
```

Описание структуры yard

Содержит настройки сервиса для добавления /32 маршрутов, представляющих статические IP адреса, назначаемые UE. Не рекомендуется изменять.

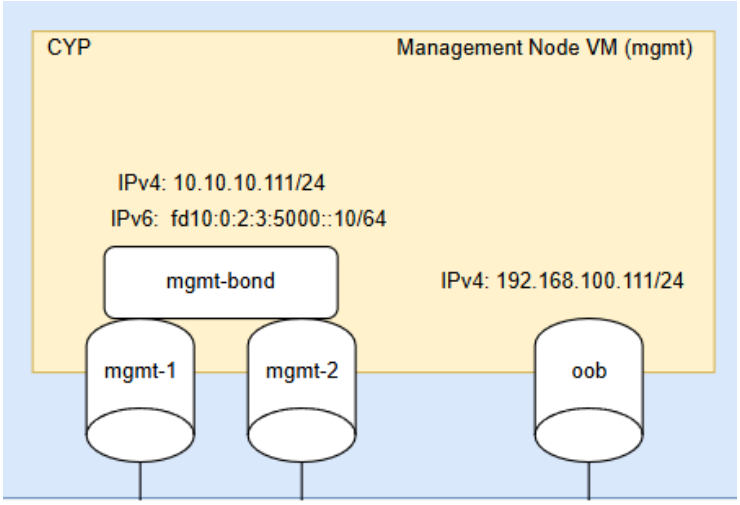
```
yard:
                port: 4641
                address: ::1
```

Переменные для management-node

Настройка сетевых интерфейсов

Целевая схема сети для сервера management-node представлена ниже.

Схема сети в LLD может отличаться от представленной в данном руководстве!



После установки образа CYP необходимо сконфигурировать IP адреса, VLAN и необходимые IP маршруты на интерфейсе **mgmt-bond**

Для OAM VLAN обязательно указать параметр sshd_listen.

Пример настройки сети для менеджмента.

```
# network interfaces configuration
                mgmt_network:
                vlans:
                - name: oam
                vlan_id: 100
                description: "oam network interface"
                sshd_listen: yes
                addresses:
                - "{{mgmt_ip}}/24"
                routes:
                - gateway: 10.78.244.1
                destination: 10.78.244.0/24
                marking:
                egress_dscp: 40
                egress_pcp: 5
```

Можно назначить IP адрес на самом интерфейсе, или создать VLAN (если это заложено в LLD), и прописать дополнительную адресацию там.

Настроить VLAN DB для Provisioning Gateway.

```
cplane_networks:
                vlans:
                - name: db
                vlan_id: 30
                description: "db network interface"
                addresses:
                - 2001:ab8:efdb::a005/96
                - 2001:ab8:efdb::a006/96
                - 2001:ab8:efdb::a00a/96
```

Переменные для настройки snmp-exporter

Для конфигурации сервиса **snmp-exporter** используются следующие переменные:

Переменная	Описание
snmp_exporter_web_listhen_address	Прослушиваемый IP адрес и номер порта
snmp_exporter_module_concurrency	Количество модулей опроса
snmp_exporter_default_path_files	Путь к источнику файла с конфигурацией snmp

Пример конфигурации представлен ниже.

```
snmp_exporter_default_path_files: ../files/snmp/
                snmp_exporter_module_concurrency: 2
                snmp_exporter_web_listhen_address: {{ mgmt_ip }}:8888
```

Переменные для настройки vmagent

Для конфигурации сервиса **vmagent** используются следующие переменные.

Переменные	Описание
vmagent_default_path_files	Путь источника к подкладываемому scrap конфигу vmagent
vmagent_victoriametrics_host	IP адрес и номер порта VictoriaMetrics (должен совпадать с victoriametrics_httpListenAddr)

Пример конфигурации представлен ниже.

```
vmagent_default_path_files: files/vmagent/
                vmagent_victoriametrics_host: {{victoriametrics_httpListenAddr}}
```

Путь, который был определен в переменной **vmagent_default_path_files**, должен содержать конфигурационный файл scrap для vmagent с названием **vmagent_promscrape_config.yaml**. В этом конфигурационном файле должна быть настроена сборка метрик **vmagent** с экспортеров. Ниже представлен пример scrap конфига для **vmagent**.

```
global:
                scrape_interval: 1m
                scrape_timeout: 30s

                scrape_configs:
                - job_name: "node"
                static_configs:
                - targets:
                - 127.0.0.1:9100

                - job_name: "ngc"
                static_configs:
                - targets:
                - 127.0.0.1:9101

                metric_relabel_configs:
                - source_labels: [__name__]
                regex: 'yacore_(.*)'
                replacement: $1
                target_label: __name__

                - job_name: "snmp"
                static_configs:
                - targets:
                - 127.0.0.1
                metrics_path: /snmp
                params:
                auth: [my_secure_v3]
                module:
                - zes
                - ifmib

                relabel_configs:
                - source_labels: [__address__]
                target_label: __param_target
                - source_labels: [__param_target]
                target_label: instance
                - target_label: __address__
                replacement: 127.0.0.2:9116
```

Существует несколько типов job:

1. node - предназначены для сбора метрик с экземпляров node_exporter, расположенных на разных нодах в т.ч. и на локальной ноде.
2. ngc - предназначены для сбора метрик с NF, расположенных на worker нодах.
3. tarantool-udr - предназначены для сбора метрик с экземпляров Tarantool UDR, расположенных на worker нодах.
4. tarantool-udsf - предназначены для сбора метрик с экземпляров Tarantool UDSF, расположенных на worker нодах.

В поле targets каждой ноды необходимо указывать список IP адресов и портов экспортеров того или иного типа.

Переменные для настройки victoriametrics

Для конфигурации сервиса victoriametrics используются следующие переменные.

Переменная	Описание
victoriametrics_httpListenAddr	IP адрес и номер порта прослушиваемый victoriametrics
victoriametrics_retentionPeriod	Период хранения метрик в месяцах

Пример конфигурации представлен ниже.

```
victoriametrics_httpListenAddr: 127.0.0.1:8428
                victoriametrics_retentionPeriod: 1
```

Настройка пользователей ClickHouse

По умолчанию пользователи не создаются и используется учетная запись *default* без пароля. Для создания учетных записей необходимо задать переменную **clickhouse_db_users**.

Значения этой переменной буду задействованы в роли *clickhouse* и *collector*.

```
clickhouse_db_users:
                default:
                username: default
                password: "kkZEqPzVk2P9g4DRK"
                profile: default
                otel:
                username: otel
                password: "xaB3IEWzacJMnGkNI"
                profile: default
                grants:
                - "GRANT SELECT ON otel.*"
                - "GRANT INSERT ON otel.*"
                colba:
                username: colba
                password: "vb3OAXF70YJNaZLVf"
                profile: default
                grants:
                - "GRANT SELECT ON otel.*"
                - "GRANT CREATE TEMPORARY TABLE ON *.*"
                - "GRANT REMOTE ON *.*"
```

Переменные для настройки Grafana

Для конфигурации сервиса Grafana используются следующие переменные.

Переменная	Описание
grafana_http_addr	IP адрес прослушиваемый Grafana
grafana_http_port	Номер порта прослушиваемый Grafana
grafana_url_victoria_ip	IP адрес VictoriaMetrics
grafana_url_victoria_port	Номер порта VictoriaMetrics

Переменные **grafana_url_victoria_ip** и **grafana_url_victoria_port** должны совпадать со значением переменной **victoriametrics_httpListenAddr**. Пример конфигурации представлен ниже.

```
grafana_http_addr: "{{mgmt_ip}}"
                grafana_http_port: 3000
                grafana_url_victoria_ip: "{{victoriametrics_httpListenAddr}}"
                grafana_url_victoria_port: 8428
```

Переменные для настройки rsyslog

На managment-node так же разворачивается сервис логирования событий **rsyslog.** Для конфигурации прослушиваемых входящих соединений используются следующие переменные.

Переменная	Описание
rsyslogd_imudp_address	IP адрес прослушиваемый rsyslog для UDP
rsyslogd_imudp_port	Номер UDP порта
rsyslogd_imtcp_address	IP адрес прослушиваемый rsyslog для TCP
rsyslogd_imtcp_port	Номер TCP порта

Для настройки отправки аудит логов на сервер SIEM необходимо определить словарь **rsyslogd_forward**, который должен включать в себе следующие переменные:

Переменная	Описание
address	IP адрес целевого сервера SIEM
port	Номер порта целевого сервера SIEM
protocol	Тип используемого протокола (UDP | TCP)

Для настройки ротации логов накопленных **rsylog** необходимо определить словарь **rsyslogd_pool**. Данный словарь включает в себе еще два словаря:

- **worker_nodes** - Словарь для конфигурации логов полученных по UDP/TCP сокетам;
- **system_logs** - Словарь для конфигурации аудит логов полученных по UNIX сокету.

Данные словари содержат ряд переменных, по настройке хранения логов и их ротации, ниже представлены перемененные, которые необходимо определить внутри **worker_nodes и system_logs**.

Переменная	Описание
dir	Директория в которую будут записываться логи
log_file	Название лог файла
config_file	Абсолютную путь к конфигурационному файлу логов для worker_nodes и system_logs
group_owner	Имя группы пользователей, которая будет иметь доступ к логам
rotate	Словарь для конфигурирования ротации логов

Для изменения ротации логов, рекомендуется менять только значения переменных **count** и **size**, которые определены внутри словаря **rotate**, ниже представлено описание этих переменных.

Переменная	Описание
count	Максимально число ротации логов
size	Размер лога, при котором будет выполняться его ротация
config_file	Абсолютный путь к скрипту ротации логов

Пример конфигурации сервиса **rsyslog** представлен ниже.

```
rsyslogd_imudp_address: "{{mgmt_ip}}"
                rsyslogd_imudp_port: 514
                rsyslogd_imtcp_address: "{{mgmt_ip}}"
                rsyslogd_imtcp_port: 514

                rsyslogd_forward:
```

```
address: 10.78.244.156 # You have to specify the IP of the target SIEM server
port: 6514
protocol: "udp"

#The rsyslog and logrotate configuration block for worker_nodes and system_audit logs rules
rsyslogd_pool:
worker_nodes:
dir: /data/logs/rsyslog
log_file: "%HOSTNAME%-%FROMHOST-IP%.log"
config_file: "{{ rsyslogd_config_path }}/server.conf"
group_owner: rsyslog
rotate:
count: 30
size: 300M
config_file: "{{ rsyslogd_scripts_path }}/logrotate.data.logs"
system_logs:
dir: /data/logs/system/audit
log_file: "audit.log"
config_file: "{{ rsyslogd_config_path }}/system_cef_audit.conf"
group_owner: www-data
rotate:
count: 10
size: 10M
config_file: "{{ rsyslogd_scripts_path }}/logrotate.system_audit.logs"
```

Переменные для настройки nginx

Для конфигурации прослушиваемого порта и адреса nginx используются следующие переменные.

Переменная	Описание
nginx_virt_host_listen_ip	Прослушиваемый IP адрес для nginx
nginx_virt_host_listen_port	Номер прослушиваемого порта для nginx

В случае если необходимо настроить TLS для LMT так же необходимо определить словарь **nginx_certificates_list** со следующими переменными.

Переменная	Описание
ca	Абсолютный путь к подкладываемому файлу сертификата CA
cert	Абсолютный путь к подкладываемому сертификату LMT
key	Абсолютную путь к подкладываемому приватному ключу LMT

Пример конфигурации nginx представлен ниже.

```
nginx_virt_host_listen_ip: "{{mgmt_ip}}"
nginx_virt_host_listen_port: 443
nginx_certificates_list:
ca: "/usr/share/cert/YADRO/intermediate-lmt/certs/chained-lmt-intermediate.pem"
cert: "/usr/share/cert/YADRO/intermediate-lmt/certs/lmt-server.pem"
key: "/usr/share/cert/YADRO/intermediate-lmt/provate/lmt-server.key"
```

Переменные для настройки core-oam

Для конфигурации сервиса core-oam используются следующие переменные.

Переменная	Описание
core_mgr_http_ip	IP адрес прослушиваемый сервисом OAM Core Manager
core_mgr_http_port	Номер порта прослушиваемый сервисом
core_mgr_grafana_ip	IP адрес, который прослушивает Grafana (должен совпадать с grafana_http_addr)
core_mgr_grafana_port	Номер порта, который прослушивает Grafana (должен совпадать с grafana_http_port)

Переменная	Описание
core_mgr_mkpasswd_method	Алгоритм хеширования пароля пароля для netopeer
core_mgr_ignore_netopeer_password	Включение/отключение функции смены пароля пароля для Netopeer
core_oam_net_iface_bh	Имя интерфейса управления, который использует OAM для подключения к NMS
netopeer_server_listen_address	IP адрес прослушиваемый сервисом netopeer

При выполнении роли **core-oam** в **playbook-management-node** от вас потребуется в интерактивном режиме изменить пароль по умолчанию для Netooper.

Пример конфигурации представлен ниже.

```
core_mgr_http_ip: "{{mgmt_ip}}"
                core_mgr_http_port: 3380
                core_mgr_grafana_ip: "{{grafana_http_addr}}"
                core_mgr_grafana_port: "{{grafana_http_port}}"
                core_mgr_mkpasswd_method: "sha512crypt"
                core_oam_net_iface_bh: "oam"
                netopeer_server_listen_address: "0.0.0.0"
```

Переменные для yacore-collector

Переменные для otel-collector

Для настройки получения логов и трейсов, необходимо определить словари **yacore_collector_grps**, **yacore_collector_syslog** и **yacore_collector_syslog_audit**, в которых можно осуществлять сетевую конфигурацию с помощью следующих переменных.

Переменная	Описание
listen_address	Прослушиваемый IP адрес
listen_port	Номер прослушиваемого порта

Так же для настройки производительности **ote-collector** необходимо определить словарь **yacore_collector_performance,** в котором должны быть следующие переменные.

Переменная	Описание
send_batch_size	Количество отправляемых записей в одном пакете otel-collector
memory_limit	Ограничение количество выделяемой памяти в МБ

Пример конфигурации представлен ниже.

```
yacore_collector_grps:
                listen_address: "{{mgmt_ip}}"
                listen_port: "14317"

                yacore_collector_syslog:
                listen_address: "{{mgmt_ip}}"
                listen_port: "22514"

                yacore_collector_syslog_audit:
                listen_address: "{{mgmt_ip}}"
                listen_port: "22614"

                yacore_collector_performance:
                send_batch_size: 10000
                memory_limit: 4000
```

Переменные так же будут задействованы в роли keepalived, который будет проверять работоспособность сервиса otel-collector по указанным адресам и портам.

Переменные для внешних серверов clickhouse

В переменной **yacore_collector_other_clickhouse_endpoint** необходимо указать все конечные точки используемых серверов clickhouse. Содержимое этой переменной будет записано в конец конфигурационного файла colba-backend ("/etc/opt/collector/colba-backend.conf") без какой либо валидации.

localhost:9000 указывать не требуется, он уже задан в исходном коде colba-backend.

```
yacore_collector_other_clickhouse_endpoint:
                - address: "10.10.10.217:9000"
                type: rcnc
                - address: "10.10.10.218:9000"
                type: rcnc
```

Переменные для настройки хранения логов и трейсов в clickhouse

Для конфигурирования параметров хранения логов и трейсов в clickhouse используются следующие переменные.

Переменная	Описание
yacore_collector_clickhouse_ttl_logs	Время жизни логов в днях
yacore_collector_clickhouse_ttl_traces	Время жизни трейсов в днях

Пример конфигурация указан ниже.

```
yacore_collector_clickhouse_ttl_logs: 5
                yacore_collector_clickhouse_ttl_traces: 5
```

Переменные для colba-frontend

В словаре **yacore_collector_frontend** необходимо указать и переменные **intermediate_domain** и **endpoint_domain** в случае если будет использоваться TLS соединение. Переменную **content_path** изменять не нужно - оставить по умолчанию.

В словаре **yacore_collector_frontend** необходимо включить словарь **nginx**, в котором необходимо настроить прослушиваемый порт и адрес интерфейса с помощью переменных **listen_port** и **listen_address** (должен совпадать с IP адресом OAM интерфейса). Переменные **cert_path** и **include_dir** рекомендуется оставить по умолчанию. Пример конфигурации colba-frontend приставлен ниже.

```
yacore_collector_frontend:
                content_path: "/opt/colba-frontend/browser/"
                intermediate_domain: "bss" # Specify if you want to enable tls for colba-frontend
                endpoint_domain: "colba-ui" # Specify if you want to enable tls for colba-frontend
                nginx:
                include_dir: "/etc/opt/nginx/include"
                listen_port: 8085
                listen_address: "217.18.134.6" # Specify IP address OAM interface
                cert_path: "/etc/opt/nginx/certificates/colba"
```

Переменные для настройки keepalived

Для бесперебойной работы колбы необходим сервис keepalived.

Для его работы выделяется отдельный vlan интерфейс. Его конфигурацию необходимо описать в переменной **mgmt_network**. Он настраивается на mgmt и worker нодах.

```
  vlans:
                - name: otel
                vlan_id: 200
                description: "otel-collector network"
                addresses:
                - 10.100.1.1/24
```

Имя интерфейса затем указываем в переменной **keepalived_interface_name,** а виртуальный ИП в **keepalived_virtual_ip**, который должен быть в той же подсети.

Переменные для настройки сервиса keepalived представлены ниже.

Переменная	Описание
keepalived_node_type	Тип ноды (MASTER или BACKUP)
keepalived_interface_name	Имя vvrp интерфейса
keepalived_virtual_ip	IP адрес для vvrp

Ниже представлен пример конфигурации.

```
keepalived_interface_name: otel
               keepalived_virtual_ip: 10.100.1.10
               keepalived_node_type: MASTER
```

Переменные для настройки yacore-service

На management node может разворачивается сервис ProvGW.

Переменные для деплоя ProvGW очень похожи на деплой UDR.

Рекомендуется сменить пароль для доступа к базе данных UDR. Пользователь и пароль в конфигурации ProvGW должен совпадать с аналогичным в секции credentials при настройке ПО Тарантул Плюс кластера.

```
yacore_services:
  - yacore_service_name: "provgw"
    yacore_service_config:
      name: PROVGW01
      nfConfigFilePath: /etc/opt/yacore/config/provgw-config.json
      platform:
        cpuset: "1"
        memory: 40960Mi
      plugins:
        - name: provgw
          params:
            db:
              password: QgKD7pi26oq
              username: udr
              tarantoolRouters:
                - address:
                    ipAddress:
                      ipv6: 2001:ab8:efdb::2001
                  port: 4403
                  priority: 1
                - address:
                    ipAddress:
                      ipv6: 2001:ab8:efdb::1001
                  port: 4401
                  priority: 2
      otel: "{{yacore_otel_config}}"
      prometheus:
        enable: true
        address: "{{ yacore_oam_ip }}"
        port: 9187
        timeout: 3
        bufferSize: 3
      oam:
        server:
          address:
            ipv4:
              addr: "{{ yacore_oam_ip }}"
              port: 3387
          ca:
            - ca1
            - ca2
            - ca3
        agent: "{{ yacore_oam_agent }}"
      log: "{{ yacore_log_config }}"
      connections:
        total: 100
        UDSF: 10
        smptest: 20
```

При деплое management node большее количество переменных можно оставить в значении по умолчанию.

Запуск плейбуков

После подготовки inventory файла запускаем плейбуки с ролями конфигурации серверов.

Убедиться, что файл с переменными лежит в папке *host_vars* плейбука, название файла соответствует имени хоста в файле *hosts.yml*.

Команда для запуска плейбука по конфигурации *worker-node* представлена ниже.

```
ansible-playbook worker-node.yml
```

Команда для запуска плейбука по конфигурации management-node представлена ниже.

```
ansible-playbook management-node.yml
```

После окончания работы плейбука, в логах в сводке выполнения убеждаемся что нет задач в статусе *unreachable* или *failed*.

```
PLAY RECAP

 ***********************************************************************************************************************
                        mgmt-cyp-yacore: ok=48   changed=37   unreachable=0
                        failed=0    skipped=12   rescued=0
                   ignored=0
```

Смена пароля по умолчанию для Grafana

После развертывания плейбука для конфигурации mgmt-node веб-интерфейс Grafana становится доступным для подключения под учётной записью admin с паролем по умолчанию — admin. Использование пароля по умолчанию представляет серьёзную угрозу безопасности, поэтому при первой авторизации в веб-интерфейсе Grafana рекомендуется сменить пароль. Для этого необходимо:

1. Открыть веб-браузер.
2. Перейти к веб-интерфейсу Grafana (если настройки Grafana при развертывании плейбука для конфигурации mgmt-node не менялись, то по умолчанию Grafana будет слушать 3000-й порт на OAM-интерфейсе хоста).
3. Авторизоваться с использованием данных учетной записи по умолчанию (после первичной авторизации Grafana предложит сменить пароль по умолчанию).
4. Ввести новый пароль в поле New password.
5. Повторить новый пароль в поле Confirm new password.
6. Нажать Submit.

По завершении на экране отобразится основной интерфейс Grafana.

3.3 Результат установки

В случае успешной установки и активации инициируется перезагрузка аппаратной платформы.

Если в ходе установки произошла ошибка, то функция активации будет недоступна. Требуется проверить логи, устранить причину ошибки и повторить последовательность установки.