ООО «БАЗИС»

СИСТЕМА УПРАВЛЕНИЯ ОБЛАЧНОЙ ПЛАТФОРМОЙ «ТИОНИКС»
ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ
ИНСТРУКЦИЯ ПО УСТАНОВКЕ
RU.НРФЛ.00001-01.96.01

ЛИСТОВ 33

Москва
2021

Оглавление

АННОТАЦИЯ

Настоящая инструкция предназначена для технического администратора ПО и содержит подробное описание работ, необходимых для установки ПО.

1 Термины и определения

В настоящем документе используются термины и основные понятия области информационных технологий. Термины и определения, представлены в таблице 1.

Таблица 1 – Термины и определения

Термин, сокращение	Определение
ansible	Система управления конфигурациями, написанная на языке программирования Python, с использованием декларативного языка разметки для описания конфигураций. Используется для автоматизации настройки и развертывания программного обеспечения. Используется для управления Linux-узлами, также поддерживается Windows
API	англ. application programming interface — программный интерфейс приложения) — описание способов (набор классов, процедур, функций, структур или констант), которыми одна компьютерная программа может взаимодействовать с другой программой
Bare Metal	Физический сервер с одним арендатором
CPU	(англ. Central Processing Unit - центральное обрабатывающее устройство) – процессор виртуальной машины
Email	(англ. Email – электронная почта) – технология и служба по пересылке и получению электронных сообщений между пользователями компьютерной сети (в том числе – Интернета)
HDD	(англ. hard (magnetic) disk drive – жёсткий диск, винчестер) — запоминающее устройство (устройство хранения информации) произвольного доступа, основанное на принципе магнитной записи
IaaS	(англ. Infrastructure as a Service – инфраструктура как услуга) - одна из форм облачных вычислений, в модели обслуживания которой предполагается большая свобода действий — потребитель может собственноручно управлять предоставляемыми сервисами.
IP-адрес	(англ. Internet Protocol Address — адрес Интернет-протокола) — уникальный сетевой адрес узла в компьютерной сети, построенной на основе стека протоколов TCP/IP

Термин, сокращение	Определение
NAT	(англ. Network Address Translation — «преобразование сетевых адресов») — это механизм в сетях TCP/IP, позволяющий преобразовывать IP-адреса транзитных пакетов
NFS	(англ. Network File System) — протокол сетевого доступа к файловым системам
NFSaaS	Сервис NFS
OpenStack	Проект по разработке платформы для построения программно-конфигурируемых ЦОД с открытым исходным кодом
Proxy	(англ. proxy — «представитель», «уполномоченный»; в связи с этим общеупотребительным является сокращение термина просто до Прокси), иначе – Прокси-сервер , сервер-посредник — промежуточный сервер (комплекс программ) в компьютерных сетях, выполняющий роль посредника между пользователем и целевым сервером (при этом о посредничестве могут как знать, так и не знать обе стороны), позволяющий клиентам как выполнять косвенные запросы (принимая и передавая их через прокси-сервер) к другим сетевым службам, так и получать ответы
SAS	Диски, используемые для задач, которые чувствительны к скорости и требуют многопоточного доступа
SATA	Диски с большим объемом хранимой информации
SNAT	Static NAT
SSD	Диски, обеспечивающие максимально возможную скорость чтения и записи, что позволяет использовать их для любых высоконагруженных проектов
Terraform	Инструмент (open-source), разработанный HashiCorp в 2014 году. Подход «инфраструктура как код» позволяет описывать облачную инфраструктуру через набор конфигурационных файлов и тем самым задавать правила настроек.
VPN	(англ. Virtual Private Network — виртуальная частная сеть) — обобщённое название технологий, позволяющих обеспечить одно или несколько сетевых соединений (логическую сеть) поверх другой сети (например, Интернет)
Администратор	Роль пользователя ОП с правами управления всеми заказами
БД	База данных
ВМ	Виртуальная машина

Термин, сокращение	Определение
ВХ	Виртуальное Хранилище
ВЦОД	Виртуальный центр обработки данных
Инфраструктура	Совокупность компьютерного оборудования (серверы, системы хранения данных, коммутационное оборудование и др.) и системного ПО, включая средства виртуализации
ОП	Облачная платформа
Оператор	Роль пользователя в СУОП с правами управления определенными заказами
Оркестратор	Компонент по управлению рабочими процессами в центре обработки данных
ПО	Программное обеспечение
Предбиллинг	Компонент прикладной подсистемы ОП, предназначенный для управления тарифными планами
Расширенный ВЦОД	Расширенный виртуальный центр обработки данных
Система	ПО Портала Облачной Платформы
СУБД	Система управления базами данных
СУ ОП	Система управления облачной Платформы
ЦОД	Центр обработки данных

2 Установка дистрибутива продукта

2.1 Установка приложения

Приложение базируется на узлах двух типов: инфраструктурные узлы (включая web-прокси для компонента «портал») и узлы приложения.

Операционная система – одна из удовлетворяющих требованиям к программному обеспечению: Astra Linux Special Edition, Centos, Red Hat Enterprise Linux, Ubuntu, Debian. Рассмотрим на примере Centos (RHEL).

Разворачивание инфраструктуры и компонентов приложения описано в ресурсных файлах terraform и плейбуках Ansible, доступных в репозитории.

2.1.1 Подготовка окружения

Terraform

Виртуальные машины стенда создаются с помощью сервиса terraform. Файлы ресурсов (для провайдера Openstack) находятся в директории stand-terraform в корневом каталоге репозитория.

Адреса созданных виртуальных машин помещаются в файл hosts (в директории stand-terraform), который используется как inventory ansible.

Для запуска выполните команды:

```
terraform init
terraform plan
terraform apply
```

Ansible

Подготовка окружения производится путём выполнения плейбуков в директории stand-playbooks. Порядок выполнения описан ниже.

Перед запуском плейбуков вносятся изменения в конфигурационные файлы:

– ./ansible.cfg - содержит настройки ansible (пользователь на управляемых узлах, способ аутентификации и т.д.);

– ./inventory/hosts - содержит список управляемых узлов в составе групп и ряд переменных, индивидуальных для узлов (используйте сформированный terraform из директории stand-terraform).

После внесения нужных изменений в указанные файлы, выполнить команду:

```
ansible -m ping all
```

Команда выполняется из директории с данным плейбуком. В корректном выводе команды должен содержаться ответ от всех управляемых узлов вида:

```
portal | SUCCESS => {
"ansible_facts": {
"discovered_interpreter_python": "/usr/bin/python"
},
"changed": false,
"ping": "pong"
```

В противном случае, необходимо проверить настройки ansible и доступность узлов.

После успешного ответа от всех управляемых узлов, можно приступать к развертыванию окружения.

Переменные

Переменные содержатся в файлах:

– ./cloud/defaults.yml

– Примечание. Имеет смысл изменить переменную cloud_user, и только в случае, если её значение не совпадает с именем пользователя, от которого осуществляется логин на управляемый узел.

– ./cloud/vars/database.yml

Также переменные содержатся в файлах плейбуков:

- all_hosts_prepare.yml
- infra_hosts_prepare.yml
- app_hosts_prepare.yml
- deploy_billing.yml
- deploy_cas.yml
- deploy_iaas.yml
- deploy_orch.yml
- deploy_pcs.yml
- deploy_pdp.yml
- deploy_portal.yml

После подготовки переменных можно приступать к развертыванию окружения.

2.1.2 Развертывание окружения

Подготовка узлов

Для начала нужно подготовить все узлы, выполнив плейбук - all_hosts_prepare.yml

Данный плейбук выполнит следующие действия:

- установку hostname;
- внесение имён всех узлов в /etc/hosts на каждом узле;
- dist-upgrade и установку полезных для администрирования пакетов;
- установку временной зоны (Europe/Moscow);
- установку и настройку ntp клиента chrony;

Подготовка инфраструктурных узлов

Подготовка инфраструктурных узлов осуществляется путём выполнения плейбука infra_hosts_prepare.yml

Данный плейбук выполнит следующие действия:

- установку и настройку сервера mysql;
- создание БД перечисленных в файле настроек cloud/vars/database.yml;
- установку и настройку сервера rabbitmq;
- установку и настройку сервера nfs;
- установку и настройку сервера redis;
- установку и настройку nginx;
- подключение директории /var/cloud/data с сервера nfs на proxy сервер и сервер компонента portal.

Подготовка узлов приложения

Подготовка узлов приложения осуществляется путём выполнения плейбука app_hosts_prepare.yml.

Данный плейбук выполнит следующие действия:

- подключение репозитория docker для Centos;
- установка необходимых для установки и запуска docker-compose пакетов;
- установка docker-compose;
- выполнение docker login на registry-сервер, содержащий образы docker с релизами компонентов приложений (в соответствии с переменной app_name в inventory);

– подключение директории /var/cloud/data с сервера nfs на узлы из группы appsrv.

По завершении процесса подготовка узлов закончена.

2.1.3 Развертывание приложения

Выполнить команду:

```
ansible-playbook deploy_portal.yml deploy_pdp.yml deploy_billing.yml
deploy_cas.yml deploy_iaas.yml deploy_orch.yml
```

В плейбуки включены инструкции для инициализации базы данных для компонентов portal, cas, billing и orchestrator.

После развертывания необходимо произвести подключение ресурсов (датацентров и т.д.).

Результатом успешного развертывания приложения является доступность веб-интерфейса портала по адресу proxy сервера.

3 Первоначальная настройка приложения

3.1.1 Инициализация БД

Для компонентов, использующих базу данных (portal,cas,billing,orchestrator) нужно выполнить инструкции:

```
portal, cas, orchestrator, billing:
bundle exec rake db:schema:load
bundle exec rake db:migrate
```

3.1.2 Инициализация данных в портале

Для наполнения компонента portal тестовыми данными, в консоли портала (bundle exec rails c) выполнить скрипт для конкретной ОП.

Пример скрипта:

```
# мрф
Mrf.delete_all
mrf = Mrf.create!(name: 'Виртуальный', mrf_str_id: 'VIRT')

# биллинг
Billing.delete_all
billing = mrf.billings.create!(code: 10000, str_code: 'VIRT_TEST', title:
'Виртуальный филиал', is_default_virt_asr_in_region: true)

# Создание регионов
Region.delete_all
Region.create!([ { code: "01", name: "Адыгея", socr: "Респ", title: "Адыгея
Республика", subject: "014010" },
  { code: "04", name: "Алтай", socr: "Респ", title: "Алтай Республика",
subject: nil },
  { code: "22", name: "Алтайский", socr: "край", title: "Алтайский край",
subject: "016050" },
  { code: "28", name: "Амурская", socr: "обл", title: "Амурская область",
subject: "017040" },
  { code: "29", name: "Архангельская", socr: "обл", title: "Архангельская
область", subject: "012010" },
  { code: "30", name: "Астраханская", socr: "обл", title: "Астраханская
область", subject: "014020" },
```

```
  { code: "02", name: "Башкортостан", socr: "Респ", title: "Башкортостан
Республика", subject: "013061" },
  { code: "31", name: "Белгородская", socr: "обл", title: "Белгородская
область", subject: "011010" },
  { code: "32", name: "Брянская", socr: "обл", title: "Брянская область",
subject: "011020" },
  { code: "03", name: "Бурятия", socr: "Респ", title: "Бурятия Республика",
subject: "016030" }])
Region.create!([ { code: "33", name: "Владимирская", socr: "обл", title:
"Владимирская область", subject: "011040" },
  { code: "34", name: "Волгоградская", socr: "обл", title: "Волгоградская
область", subject: "014030" },
  { code: "35", name: "Вологодская", socr: "обл", title: "Вологодская
область", subject: "012020" },
  { code: "36", name: "Воронежская", socr: "обл", title: "Воронежская
область", subject: "011050" },
  { code: "05", name: "Дагестан", socr: "Респ", title: "Дагестан Республика",
subject: "014110" },
  { code: "79", name: "Еврейская", socr: "Аобл", title: "Еврейская АО",
subject: nil },
  { code: "75", name: "Забайкальский", socr: "край", title: "Забайкальский
край", subject: "016120" },
  { code: "37", name: "Ивановская", socr: "обл", title: "Ивановская область",
subject: nil },
  { code: "06", name: "Ингушетия", socr: "Респ", title: "Ингушетия
Республика", subject: "014120" },
  { code: "38", name: "Иркутская", socr: "обл", title: "Иркутская область",
subject: "016070" }])
Region.create!([ { code: "07", name: "Кабардино-Балкарская", socr: "Респ",
title: "Кабардино-Балкарская Республика", subject: "014040" },
  { code: "39", name: "Калининградская", socr: "обл", title: "Калининградская
область", subject: "012030" },
  { code: "08", name: "Калмыкия", socr: "Респ", title: "Калмыкия Республика",
subject: "014050" },
  { code: "40", name: "Калужская", socr: "обл", title: "Калужская область",
subject: "011060" },
  { code: "41", name: "Камчатский", socr: "край", title: "Камчатский край",
subject: "017030" },
  { code: "09", name: "Карачаево-Черкесская", socr: "Респ", title:
"Карачаево-Черкесская Республика", subject: "014060" },
```

```
  { code: "10", name: "Карелия", socr: "Респ", title: "Карелия Республика",
subject: "012040" },
  { code: "42", name: "Кемеровская", socr: "обл", title: "Кемеровская
область", subject: "016080" },
  { code: "43", name: "Кировская", socr: "обл", title: "Кировская область",
subject: "013020" },
  { code: "11", name: "Коми", socr: "Респ", title: "Коми Республика",
subject: "012110" }])
Region.create!([ { code: "44", name: "Костромская", socr: "обл", title:
"Костромская область", subject: nil },
  { code: "23", name: "Краснодарский", socr: "край", title: "Краснодарский
край", subject: "014071" },
  { code: "24", name: "Красноярский", socr: "край", title: "Красноярский
край", subject: nil },
  { code: "91", name: "Крым", socr: "Респ", title: "Крым Республика",
subject: nil },
  { code: "45", name: "Курганская", socr: "обл", title: "Курганская область",
subject: "015030" },
  { code: "46", name: "Курская", socr: "обл", title: "Курская область",
subject: "011070" },
  { code: "47", name: "Ленинградская", socr: "обл", title: "Ленинградская
область", subject: "012122" },
  { code: "48", name: "Липецкая", socr: "обл", title: "Липецкая область",
subject: "011080" },
  { code: "49", name: "Магаданская", socr: "обл", title: "Магаданская
область", subject: "017020" },
  { code: "12", name: "Марий Эл", socr: "Респ", title: "Марий Эл Республика",
subject: "013090" }])
Region.create!([ { code: "13", name: "Мордовия", socr: "Респ", title:
"Мордовия Республика", subject: "013100" },
  { code: "77", name: "Москва", socr: "г", title: "Москва", subject: "010030"
},
  { code: "50", name: "Московская", socr: "обл", title: "Московская область",
subject: "011092" },
  { code: "51", name: "Мурманская", socr: "обл", title: "Мурманская область",
subject: "012050" },
  { code: "83", name: "Ненецкий", socr: "АО", title: "Ненецкий автономный
округ", subject: nil },
  { code: "52", name: "Нижегородская", socr: "обл", title: "Нижегородская
область", subject: "013030" },
```

```
  { code: "53", name: "Новгородская", socr: "обл", title: "Новгородская
область", subject: "012060" },
  { code: "54", name: "Новосибирская", socr: "обл", title: "Новосибирская
область", subject: "016090" },
  { code: "55", name: "Омская", socr: "обл", title: "Омская область",
subject: "016100" },
  { code: "56", name: "Оренбургская", socr: "обл", title: "Оренбургская
область", subject: "013040" }])
Region.create!([ { code: "57", name: "Орловская", socr: "обл", title:
"Орловская область", subject: "011100" },
  { code: "58", name: "Пензенская", socr: "обл", title: "Пензенская область",
subject: "013050" },
  { code: "59", name: "Пермский", socr: "край", title: "Пермский край",
subject: "015020" },
  { code: "25", name: "Приморский", socr: "край", title: "Приморский край",
subject: "017010" },
  { code: "60", name: "Псковская", socr: "обл", title: "Псковская область",
subject: "012080" },
  { code: "61", name: "Ростовская", socr: "обл", title: "Ростовская область",
subject: "014080" },
  { code: "62", name: "Рязанская", socr: "обл", title: "Рязанская область",
subject: "011110" },
  { code: "63", name: "Самарская", socr: "обл", title: "Самарская область",
subject: "013062" },
  { code: "78", name: "Санкт-Петербург", socr: "г", title: "Санкт-Петербург",
subject: "012121" },
  { code: "64", name: "Саратовская", socr: "обл", title: "Саратовская
область", subject: "013070" }])
Region.create!([ { code: "14", name: "Саха (Якутия)", socr: "Респ", title:
"Саха (Якутия) Республика", subject: "017080" },
  { code: "65", name: "Сахалинская", socr: "обл", title: "Сахалинская
область", subject: "017050" },
  { code: "66", name: "Свердловская", socr: "обл", title: "Свердловская
область", subject: "015040" },
  { code: "92", name: "Севастополь", socr: "г", title: "Севастополь",
subject: nil },
  { code: "15", name: "Северная Осетия - Алания", socr: "Респ", title:
"Северная Осетия - Алания Республика", subject: "014090" },
  { code: "67", name: "Смоленская", socr: "обл", title: "Смоленская область",
subject: "011120" },
```

```
  { code: "26", name: "Ставропольский", socr: "край", title: "Ставропольский
край", subject: "014104" },
  { code: "68", name: "Тамбовская", socr: "обл", title: "Тамбовская область",
subject: "011130" },
  { code: "16", name: "Татарстан", socr: "Респ", title: "Татарстан
Республика", subject: "013130" },
  { code: "69", name: "Тверская", socr: "обл", title: "Тверская область",
subject: "011140" }])
Region.create!([ { code: "70", name: "Томская", socr: "обл", title: "Томская
область", subject: "016110" },
  { code: "71", name: "Тульская", socr: "обл", title: "Тульская область",
subject: "011150" },
  { code: "17", name: "Тыва", socr: "Респ", title: "Тыва Республика",
subject: "016042" },
  { code: "72", name: "Тюменская", socr: "обл", title: "Тюменская область",
subject: "015060" },
  { code: "18", name: "Удмуртская", socr: "Респ", title: "Удмуртская
Республика", subject: "013110" },
  { code: "73", name: "Ульяновская", socr: "обл", title: "Ульяновская
область", subject: "013080" },
  { code: "27", name: "Хабаровский", socr: "край", title: "Хабаровский край",
subject: nil },
  { code: "19", name: "Хакасия", socr: "Респ", title: "Хакасия Республика",
subject: "016041" },
  { code: "86", name: "Ханты-Мансийский Автономный округ - Югра", socr: "АО",
title: "Ханты-Мансийский АО", subject: "015070" },
  { code: "74", name: "Челябинская", socr: "обл", title: "Челябинская
область", subject: "015050" },
  { code: "20", name: "Чеченская", socr: "Респ", title: "Чеченская
Республика", subject: "014101" },
  { code: "21", name: "Чувашская Республика", socr: "Чувашия", title:
"Чувашская Республика", subject: "013120" },
  { code: "87", name: "Чукотский", socr: "АО", title: "Чукотский автономный
округ", subject: "017001" },
  { code: "89", name: "Ямало-Ненецкий", socr: "АО", title: "Ямало-Ненецкий
АО", subject: "015080" },
  { code: "76", name: "Ярославская", socr: "обл", title: "Ярославская
область", subject: "011160" } ])
Region.find_each do |region|
  region.billings << billing
```

```
end


User.delete_all
# создание тестового юзера
user = User.new(
  last_name: "Пользователь", first_name: "Самый", middle_name: "Первый",
user_type: "P", name_id: "_03299c68506b08357e7fd3d9d4d5ce7b", phone: nil,
mobile: nil, org_type: nil, org_inn: nil, org_ogrn: nil,
  name: "Самый Первый Пользователь", user_name: "sunop-test03@tionix.ru",
uuid: "2bec9c9074520130f3bc0050568100d4", snils: nil, org_kpp: nil,
sia_org_id: nil,
  org_admin: true, org_chief: false, light: true, sia_user_id: "1000000132",
default_account_id: nil,
  mpz_elk_fields: nil
)
user.emails.build(address: 'sunop-test03@tionix.ru')
user.save!

# создание тестовой организации для роли клиент
client_org = User.create!(
  last_name: nil, first_name: nil, middle_name: nil, user_type: "O", name_id:
nil, created_at: "2013-03-21 12:38:35", updated_at: "2017-05-18 15:42:53",
  phone: "+7(999)911-11-99", mobile: "+7(999)000-27-61", org_type: nil,
org_inn: nil, org_ogrn: nil,
  name: "Виртуальное предприятие", user_name: nil, uuid:
"2bfc0ee074520130f3be0050568100d4",
  snils: nil, org_kpp: nil, sia_org_id: "1000000133", org_admin: nil,
org_chief: nil, sia_user_id: nil, light: true, default_account_id: nil,
  mpz_elk_fields: {
    "ext_region"=>"77","city"=>"Москва", "contact_name"=>"Иванов Иван
Иванович",
    "email"=>"sunop-test03@tionix.ru", "phone"=>"+7(999)911-11-99",
"comment"=>""
  }
)
# создание тестовой организации для роли оператор
operator_org = User.create!(
  last_name: nil, first_name: nil, middle_name: nil, user_type: "O", name_id:
nil, created_at: "2013-03-21 12:38:35", updated_at: "2017-05-18 15:42:53",
```

```
  phone: "+7(999)911-11-99", mobile: "+7(999)000-27-61", org_type: nil,
org_inn: nil, org_ogrn: nil,
  name: "Операторы сервисов”, user_name: nil, uuid:
"2bfc0ee074520130f3be0050568100d4",
  snils: nil, org_kpp: nil, sia_org_id: "1000000096", org_admin: nil,
org_chief: nil, sia_user_id: nil, light: true, default_account_id: nil,
  mpz_elk_fields: {
    "ext_region"=>"77","city"=>"Москва", "contact_name"=>"Иванов Иван
Иванович",
    "email"=>"sunop-test03@tionix.ru", "phone"=>"+7(999)911-11-99",
"comment"=>""
  }
)


Assignment.delete_all
assignment = Assignment.create!(org_id: client_org.id, employee_id: user.id)
assignment = Assignment.create!(org_id: operator_org.id, employee_id:
user.id)

OperatorKey.create(billing_id: billing.first.id, key: '7A3S-38TR-RCE6-SUCZ',
billing_code: billing.first.code, assignment_id: assignment.id)

Account.delete_all
# создание аккаунта
client_org.accounts.create!(
  ils: "va_2bfaa02074520130f3bd0050568100d4", billing_code: 10000, client_id:
"virt",
  blocked: false, nls: nil, nls_switch_date: nil
)
operator_org.accounts.create!(
  ils: "va_dc3ra02074520130f3bd005056819afg", billing_code: 10000, client_id:
"virt",
  blocked: false, nls: nil, nls_switch_date: nil
)

# Заполнение классификатора Программный продукт
ClassifierSoftwareService.delete_all
[
  { code: "cis_datacenter", name: "CIS Datacenter, 1 процессор" },
```

```
  { code: "cis_standart", name: "CIS Standard, 1 процессор" },
  { code: "dynamics_ax_enterprize_user", name: "Dynamics AX Enterprise, 1
пользователь" },
  { code: "dynamics_ax_enterprize_device", name: "Dynamics AX Enterprise, 1
устройство" },
  { code: "dynamics_ax_functional_user", name: "Dynamics AX Functional, 1
пользователь"},
  { code: "dynamics_ax_functional_device", name:"Dynamics AX Functional, 1
устройство"},
  { code: "dynamics_ax_selfserve_user", name:"Dynamics AX SelfServe, 1
пользователь"},
  { code: "dynamics_ax_selfserve_device", name:"Dynamics AX SelfServe, 1
устройство"},
  { code: "dynamics_ax_server", name:"Dynamics AX Server, 4 ядра"},
  { code: "dynamics_ax_strsvr_user", name:"Dynamics AX StrSvr, 1
пользователь"},
  { code: "dynamics_ax_task_user", name:"Dynamics AX Task, 1 пользователь"},
  { code: "dynamics_ax_task_device", name:"Dynamics AX Task, 1 устройство"},
  { code: "dynamics_crm", name:"Dynamics CRM, 1 пользователь"},
  { code: "dynamics_crm_basic", name:"Dynamics CRM Basic, 1 пользователь"},
  { code: "dynamics_crm_essentials", name:"Dynamics CRM Essentials, 1
пользователь"},
  { code: "exchange_basic", name:"Exchange Basic, 1 пользователь"},
  { code: "exchange_enterprise", name:"Exchange Enterprise, 1 пользователь"},
  { code: "exchange_enterprise_plus", name:"Exchange Enterprise Plus, 1
пользователь"},
  { code: "exchange_standart", name:"Exchange Standard, 1 пользователь"},
  { code: "exchange_standart_plus", name:"Exchange Standard Plus, 1
пользователь"},
  { code: "forefront_identity_manager", name:"Forefront Identity Manager, 1
пользователь"},
  { code: "lync_server_enterprise", name:"Lync Server Enterprise, 1
пользователь"},
  { code: "lync_server_enterprise_plus", name:"Lync Server Enterprise Plus, 1
пользователь"},
  { code: "lync_server_plus", name:"Lync Server Plus, 1 пользователь"},
  { code: "lync_server_standart", name:"Lync Server Standard, 1
пользователь"},
  { code: "office_professional_plus", name:"Office Professional Plus, 1
пользователь"},
```

```
  { code: "office_standart", name:"Office Standard, 1 пользователь"},
  { code: "productivity_suite", name:"Productivity Suite, 1 пользователь"},
  { code: "project", name:"Project, 1 пользователь"},
  { code: "project_professional", name:"Project Professional с лицензией
Project Server, 1 пользователь"},
  { code: "project_server", name:"Project Server, 1 пользователь"},
  { code: "sharepoint_hosting", name:"SharePoint Hosting, 1 сервер"},
  { code: "sharepoint_server_enterprise", name:"SharePoint Server Enterprise,
1 пользователь"},
  { code: "sharepoint_server_standart", name:"SharePoint Server Standard, 1
пользователь"},
  { code: "sql_server_business_intelligence", name:"SQL Server Business
Intelligence, 1 пользователь"},
  { code: "sql_server_enterprise", name:"SQL Server Enterprise, 2 ядра"},
  { code: "sql_server_standart_user", name: "SQL Server Standard. 1
пользователь" },
  { code: "sql_server_standart_core", name: "SQL Server Standard, 2 ядра" },
  { code: "sql_server_web", name: "SQL Server Web, 2 ядра" },
  { code: "system_center_datacenter", name: "System Center Datacenter, 1
процессор" },
  { code: "system_center_standard", name: "System Center Standard, 1
процессор" },
  { code: "windows_professional", name: "Windows Professional, 1
пользователь" },
  { code: "windows_remote_desktop_services", name: "Windows Remote Desktop
Services, 1 пользователь" },
  { code: "windows_server_datacenter", name: "Windows Server Datacenter, 1
процессор" },
  { code: "windows_server_essentials", name: "Windows Server Essentials, 1
процессор" },
  { code: "windows_server_standart", name: "Windows Server Standard, 1
процессор" }
].each do |software|
  ClassifierSoftwareService.create!(software)
end

# Заполнение продуктов, таблица products
Product.delete_all
all_products = []
# Виртуальное хранилище
```

```
all_products += Product.create!([
  { title: 'Виртуальный ЦОД', service_type: 'vdc', full_description: '',
short_description: 'Виртуальный центр обработки данных организации.',
functions: ''},
  { title: 'Виртуальный ЦОД', service_type: 'xvdc', full_description: '',
short_description: 'Расширенный Виртуальный ЦОД', functions: ''},
  { title: 'Виртуальное хранилище', service_type: 'cloud_storage',
full_description: '', short_description: 'Услуга построена на базе
программно-аппаратного комплекса Hitachi Content Platform.', functions: ''},
])

# Подсервисы продукта для xvdc и vdc
subservice_params = [
  { title: 'Управление DNS', service_type: 'dns_zone', full_description: '',
functions: '', subservice: true, visible: false, short_description: '' },
  { title: 'Мониторинг', service_type: 'monitoring', full_description: '',
functions: '', subservice: true, visible: false, short_description: '' },
  { title: 'Балансировка', service_type: 'load_balancer', full_description:
'', functions: '', subservice: true, visible: false, short_description: '' },
  { title: 'Резервное копирование', service_type: 'baas', full_description:
'', functions: '', subservice: true, visible: false, short_description: '' },
  { title: 'Удаленный доступ по VPN', service_type: 'vpn', full_description:
'', functions: '', subservice: true, visible: false, short_description: '' },
  { title: 'Выделенный физический сервер', service_type: 'baremetal_server',
functions: '', full_description: '', subservice: true, visible: false,
short_description: '' },
  { title: 'Программные услуги', service_type: 'software', functions: '',
full_description: '', subservice: true, visible: false, short_description: ''
},
]
subservices = Product.create!(subservice_params)
all_products += subservices
all_products.first(2).each do |parent_product|
  parent_product.subservices << subservices
end

# Datacenter
dc = Datacenter.create!(code: 'tionix', name: 'Tionix DC', virtualization:
%w(openstack))
dc.products << all_products
```

```
ProductDatacenter.delete_all
vdc = Product.find_by(service_type: 'vdc')
vdc.product_datacenters.create!([
  { datacenter_code: 'tionix',  virtualization: %w(openstack)}
])

xvdc = Product.find_by(service_type: 'xvdc')
xvdc.product_datacenters.create!([
  { datacenter_code: 'tionix',  virtualization: %w(openstack)}
])

Banner.delete_all

HddType.delete_all
HddType.create!([
  {code: "fast", name: "SAS", plural: "SAS диски", sort_order: 1},
  {code: "slow", name: "SATA", plural: "SATA диски", sort_order: 2},
  {code: "ultrafast", name: "SSD", plural: "SSD диски", sort_order: 3},
  {code: "archive", name: "Архивное хранение", plural: "Диски архивного
хранения", sort_order: 4}
])

hdd_types = HddType.where(code: 'fast')
Datacenter.find_each do |datacenter|
  datacenter.hdd_types << hdd_types
  datacenter.hdd_type_datacenters.each {|hdd_type_datacenter|
hdd_type_datacenter.update(virtualization: datacenter.virtualization,
product_service_type: 'vdc') }
end
Datacenter.find_each do |datacenter|
  datacenter.hdd_types << hdd_types
  datacenter.hdd_type_datacenters.where(product_service_type: nil).each
{|hdd_type_datacenter| hdd_type_datacenter.update(virtualization:
datacenter.virtualization, product_service_type: 'xvdc') }
end

# обязательные классификаторы, без которых не одобрить заказа
# «Центр финансовой ответственности»
ClassifierCfo.delete_all
```

```
classifier_cfo = ClassifierCfo.create(code: "011", name: "Департамент внешних
коммуникаций", is_default: true, is_archived: false,
not_visible_to_primecost: false)

# «Бизнес-процесс»
ClassifierBusinessProcess.delete_all
ClassifierBusinessProcess.create(code: "bp_01", name: "Бизнес-процесс №1",
is_default: true, is_archived: false)

#«ШПП»
ClassifierShpp.delete_all
ClassifierShpp.create(code: "SHPP_01", name: "ШПП №1", is_default: true,
is_archived: false)

# «Расходная статья»
ClassifierDepartment.delete_all
ClassifierDepartment.create(code: "001", name: "Главная расходная статья",
is_default: true, is_archived: false)

# «Доходная статья»
ClassifierRevenueArticle.delete_all
ClassifierRevenueArticle.create(code: "R493304", name: "Виртуальный ЦОД",
is_default: true, is_archived: false)

# «Проект»
ClassifierProduct.delete_all
ClassifierProduct.create(code: "001", name: "Тестовый стенд", is_default:
true, is_archived: false)

OperatorCfoKey.create(classifier_cfo_code: classifier_cfo.code, key: '5QFH-
DT55-W7TF-KY3X', assignment_id: assignment.id)

# Создание ресурсов
Resource.delete_all
Resource.create(code: "cpu_openstack", name: "Количество ядер процессора,
KVM", total_amount: 0, is_saas: false)
Resource.create(code: "ram_openstack", name: "Объем оперативной памяти, ГБ,
KVM", total_amount: 0, is_saas: false)
Resource.create(code: "hdd_openstack_fast", name: "Объем диска, ГБ,
OpenStack-KVM, SAS", total_amount: 0, is_saas: false)
```

```
Resource.create(code: "hdd_openstack_slow", name: "Объем диска, ГБ,
OpenStack-KVM, SATA", total_amount: 0, is_saas: false)
Resource.create(code: "hdd_openstack_ultrafast", name: "Объем диска, ГБ,
OpenStack-KVM, SSD", total_amount: 0, is_saas: false)
Resource.create(code: "vlan_openstack", name: "VLAN, KVM", total_amount: 0,
is_saas: false)
Resource.create(code: "external_ip_openstack", name: "Внешний IP, OpenStack-
KVM", total_amount: 0, is_saas: false)
Resource.create(code: "internal_ip_openstack", name: "Маршрутизируемый IP,
OpenStack-KVM", total_amount: 0, is_saas: false)
Resource.create(code: "iops_openstack", name: "Производительность диска,
IOPS, OpenStack-KVM", total_amount: 0, is_saas: false)
Resource.create(code: "bandwidth", name: "Полоса пропускания", total_amount:
10000, is_saas: true)
# создаём количества доступных ресурсов в датацентре на определённой
виртуализации (openstack)
ResourceAmount.delete_all
ResourceAmount.create(resource_code: "cpu_openstack", region: "tionix",
amount: 2500, cluster: nil, can_be_oversold: false, reseller_code: "default")
ResourceAmount.create(resource_code: "ram_openstack", region: "tionix",
amount: 5400, cluster: nil, can_be_oversold: false, reseller_code: "default")
ResourceAmount.create(resource_code: "hdd_openstack_fast", region: "tionix",
amount: 8210, cluster: nil, can_be_oversold: true, reseller_code: "default")
ResourceAmount.create(resource_code: "hdd_openstack_slow", region: "tionix",
amount: 3000, cluster: nil, can_be_oversold: true, reseller_code: "default")
ResourceAmount.create(resource_code: "hdd_openstack_ultrafast", region:
"tionix", amount: 20000, cluster: nil, can_be_oversold: true, reseller_code:
"default")
ResourceAmount.create(resource_code: "vlan_openstack", region: "tionix",
amount: 1000, cluster: nil, can_be_oversold: false, reseller_code: "default")
ResourceAmount.create(resource_code: "external_ip_openstack", region:
"tionix", amount: 70, cluster: nil, can_be_oversold: false, reseller_code:
"default")
ResourceAmount.create(resource_code: "internal_ip_openstack", region:
"tionix", amount: 101, cluster: nil, can_be_oversold: false, reseller_code:
"default")
ResourceAmount.create(resource_code: "iops_openstack", region: "tionix",
amount: 20000, cluster: nil, can_be_oversold: false, reseller_code:
"default")
```

```
ResourceAmount.create(resource_code: "bandwidth", region: "tionix", amount:
10000, cluster: nil, can_be_oversold: false, reseller_code: "default")

# Создание лимитов для ограничения при заказе с ручными настройками
Limit.delete_all
Resource.pluck(:code).uniq.each do |resource_code|
  %w[vdc xvdc].each do |service_type|
    %w[admin operator client].each do |role|
      %w[comm test].each do |mode|
        parse_resource =
resource_code.match(/(?<resource>[\w_]+)_openstack_?(?<hdd_type>\w+)?/)
        params = { service_type: service_type, role: role, datacenter_code:
'tionix', virtualization: 'openstack',
                   mode: mode, resource: parse_resource.try(:[], :resource)
|| resource_code, reseller_code: 'default', }

        if parse_resource.try(:[], :resource) == 'hdd'
          Limit.find_or_create_by(params.merge(min: 0, max: 1000, step: 10,
init_value: 10, hdd_type: parse_resource[:hdd_type]))
        else
          Limit.find_or_create_by(params.merge(min: 1, max: 100, step: 10,
init_value: 1, hdd_type: nil))
        end
      end
    end
  end
end

# Создание настроек
Setting.create!([
  { name: 'notify_new_order_to', data: 'sunop-test03@tionix.ru',
setting_type: 'notification' },
  { name: 'notify_order_processed', data: 'sunop-test03@tionix.ru',
setting_type: 'notification' },
  { name: 'notify_new_ticket', data: 'sunop-test03@tionix.ru', setting_type:
'notification' }
])
```

В конфигурационном файле product_configs.yml в секции shared, в available_products добавить vdc и xvdc, в vm добавить тип виртуализации openstack:

```
shared:
  # доступные в новом портале типы продуктов
  available_products:
    - vdc
    - xvdc
...
vm:
  # Порядок следования виртуализаций в заказе Вирт ЦОДа берется отсюда,
  # а переводы из локали.
  virtualization:
    - openstack
  ...
  hdd_types:
    tionix:
      openstack:
        fast:
          name: Быстрый
          default_value: 100
  ...
  os:
    openstack:
      - name: "Cirros"
```

Создать шаблоны писем, выполнив команду:

```
rake notifications:create_notifications_templates['default']
```

При пропуске CAS (SKIP_CAS=true), в host_configs.yml установить значение operator_org.sia_org_id

```
operator_sia_org_id: '1000000096'
```

Настройки в pdp

Для доступа к панели администратора для созданного в портале пользователя нужно в конфигурационном файле policy_store.yml в portal_admins добавить uuid пользователя, которого создали в портале.

Инициализация данных в биллинге

Для наполнения компонента billing данными, выполняем следующие задачи в консоли биллинга.

```
billing:
bundle exec rake tariff_classes:init_data
bundle exec rake tariff_plans:init_data
bundle exec rake tariff_plan_datacenters:init_data
bundle exec rake product_configs:init_data
bundle exec rake resources:init_data
bundle exec rake vdc:init_datara
```

Для пункта «bundle exec rake tariff_plans:init_data» необходим доступ до портала. Перед выполнением данного пункта выполнить в консоли биллинга:

```
TariffPlan.where(service_type: %w(vdc xvdc), reseller_code:
nil).update_all(reseller_code: 'default', check_status: true)
```

Для сбора из портала тарифных планов биллинга надо организовать связь пользователей между компонентами.

В консоли компонента billing (bundle exec rails c) выполнить следующую команду:

```
billing_user = User.create(email: 'sunop-test03@tionix.ru', password:
'11111111', password_confirmation: '11111111')
# получаем токен пользователя
billing_user.authentication_token
```

В компонент portal, в файл **.env**, в качестве значения переменной BILLING_ACCESS_TOKEN надо внести полученный токен авторизации. К примеру, если billing_user.authentication_token равен e402a643a7e25a465c321960c45d7d68, то вносим его в файл. В итоге должно получиться:

```
BILLING_ACCESS_TOKEN=e402a643a7e25a465c321960c45d7d68
```

Инициализация данных в оркестраторе

Для наполнения компонента orchestrator тестовыми данными, в консоли оркестратора (bundle exec tux) выполнить скрипт.

Пример скрипта:

```
# Создание настроек
Orchestrator::Models::Config.create!([
  { key: 'iaas_tionix_openstack_active-network-id', value: '', v_type:
'string' },
  { key: 'iaas_tionix_openstack_active-cluster', value: '', v_type: 'string'
},
```

```
  { key: 'iaas_tionix_openstack_active-network-pool', value: '', v_type:
'string' },
  { key: 'sms_attempts-one-order', value: '3', v_type: 'string' },
  { key: 'sms_attempts-one-day', value: '5', v_type: 'string' },
  { key: 'sms_block-list', value: '', v_type: 'array' },
  { key: 'sms_white-list', value: '', v_type: 'array' },
  { key: 'sms_message-template', value: 'Код для активации заказа №
%{order_id}: %{activation_code}.', v_type: 'string' },
  { key: 'sms_activation-code-expire-time', value: '60', v_type: 'string' },
  { key: 'sms_approver-operator', value: 'smsapproveuuid', v_type: 'string'
},
  { key: 'dns_block-list', value: '', v_type: 'array' },
  { key: 'load-balancer_tionix_load-balancer_active-network-id', value: '',
v_type: 'string' },
  { key: 'load-balancer_tionix_load-balancer-ssl_active-network-id', value:
'', v_type: 'string' },
  { key: 'load-balancer_tionix_load-balancer-addos_active-network-id', value:
'', v_type: 'string' },
  { key: 'load-balancer_tionix_lb-class', value: '', v_type: 'string' },
  { key: 'fraud_white-list', value: '', v_type: 'array' },
  { key: 'fraud_block-list', value: '', v_type: 'array' },
  { key: 'fraud_popular-domain-list', value:
'@gmail.com$,@yandex.ru$,@mail.ru$', v_type: 'array' },
  { key: 'fraud_popular-domain-registratons-first-threshold', value: '10',
v_type: 'string' },
  { key: 'fraud_popular-domain-registratons-second-threshold', value: '20',
v_type: 'string' },
  { key: 'fraud_custom-domain-registratons-first-threshold', value: '3',
v_type: 'string' },
  { key: 'fraud_custom-domain-registratons-second-threshold', value: '7',
v_type: 'string' },
  { key: 'time_days-before-deprovision', value: '14', v_type: 'string' },
  { key: 'iaas_tionix_openstack_baremetal-active-network-id', value: '',
v_type: 'string' },
])

# Настройки сетей
openstack_net = Orchestrator::Models::IntNetwork.create
```

```
openstack_net.save_params(nil, network: '10.10.20.0/24', vlan: '121', region:
'tionix', virtualization: 'openstack', gateway_ip: '10.10.20.1',
virtualization_network_id: '97fde46f-6cfc-4e87-b610-36b5436b6d71')
(10..250).each do |oct|
  nat = Orchestrator::Models::Nat.create
  nat.save_params(nil, { int_ip: "10.10.20.#{oct}", ext_ip:
"46.61.160.#{oct}", int_network_id: openstack_net.id, cp_status:
'preconfigured' }, nil)
  int_ip = Orchestrator::Models::IntIp.create
  int_ip.save_params(nil, { ip: "10.10.20.#{oct}", int_network_id:
openstack_net.id, status: 'preconfigured'}, nil)
end
Orchestrator::Models::Config.find_by_key('iaas_tionix_openstack_active-
network-id').update_attribute(:value, openstack_net.id)
```

В конфигурационном файле orchestrator.yml требуется добавить типы дисков для ДЦ tionix (где ДЦ tionix - сущность внутри приложения, сопоставляется с реальным датацентром или площадкой внутри него и предназначается для хранения настроек специфичных для данного датацентра).

Пример файла:

```
hdd_types:
  tionix:
    openstack:
      - 'fast'
      - 'slow'
      - 'ultrafast'

# параметры интеграции с API openstack
openstack:
  tionix:
    skip: false
```

В конфигурационном файле poller.yml требуется указать host rabbitmq.

Пример файла:

```
pids_dir: "/orchestrator"
logs_dir: "/orchestrator/log"
log_level: "debug"
rabbitmq_server: "rabbit"
```

В конфигурационном файле config/config.d/iaas.yml требуется указать routing_key, по которому сообщения будут попадать в очередь AMQP.

Пример файла:

```
routing_key:
  tionix: "tionix.adapters.iaas"
vm_users:
  openstack:
    Cirros: 'cirros'
```

Настройки в iaas

Конфигурационный файл iaas_adapter.yml заполнить аналогично примеру:

```
pids_dir: "/iaas"
logs_dir: "/iaas/log"
routing_key: "tionix.adapters.iaas"
queue_name: "tionix.iaas"
rabbitmq_server: "rabbit"
# Массив систем виртуализаций которые обрабатываются данным экземпляром
iaas_adapter.
# Допустимые значения: 'vmware', 'openstack'
virtualizations:
  - 'openstack'
```

Виртуализация openstack настраивается в конфигурационном файле openstack.yml.

Пример файла:

```
connection:
  username: "admin"
  password: "123456"
  auth_url: "http://10.38.18.125:5000/v3"
  auth_tenant: "admin"
  public_url: "https://10.38.18.125/dashboard"
  instance_path: "/var/lib/nova/instances/"
  admin_username: "admin"
volume_type:
  fast:
    - fast
availability_zone: nova

user_default_roles:
```

```
  - 'user'
username_roles:
  - 'user'
  - 'admin'
admin_username_roles:
  - 'admin'

# до трёх IP адресов DNS-серверов для площадки
dns_nameservers:
  - '77.88.8.88'
  - '77.88.8.2'

web_services:
  open_timeout: 30
  read_timeout: 60

# список device_owners для портов, игнорируемых при удалении
ignored_device_owners:
  # при удалении портов поотдельности (до удаления роутера)
  tenant_ports:
    - 'network:router_interface'
    - 'network:router_interface_distributed'
  # при удалении портов перед самим удалением роутера
  router_ports:
    - 'network:router_gateway'
    - 'network:router_centralized_snat'
```

Таймауты настраиваются в конфигурационном файле tiomeouts.yml.

Пример файла:

```
default:
  default: 2700 # in seconds
openstack:
  default: 2700
  create_vm: 3600 # in seconds
  power_on_vm: 2700
  power_off_vm: 2700
  soft_reboot_vm: 2700
  hard_reboot_vm: 2700
  destroy_vm: 2700
```

```
  vm_add_ext_ip: 2700
  vm_remove_ext_ip: 2700
  vm_update_security_profile: 2700
  vm_add_volume: 2700
  vm_remove_volume: 2700
  find_vm: 2700
  create_user: 2700
  delete_user: 2700
  change_user_password: 2700
  allocate_resource: 2700
  reallocate_resource: 2700
  deallocate_resource: 2700
  power_off_resources: 2700
  power_on_resources: 2700
  allocate_network_resource: 2700
  reallocate_network_resource: 2700
  update_bandwidth: 2700
  deallocate_network_resource: 2700
```

Шаблоны ВМ настраиваются в конфигурационном файле vm_templates.yml.

Пример файла:

```
kvm:
  Cirros:
    image_name: '23da6f11-33b8-453c-895f-9f6536878e09'
    deployment_method: 'default'
```

При возникновении ошибки Unable to connect to manage.tcmp нужно в файле /etc/hosts добавить запись анологичную примеру: 10.38.18.125 manage.tcmp

3.1.3 Подключение ресурсов (датацентров и т.д.)

Openstack

Необходимо обеспечить:

– корректную настройку учётной записи с правами админа, идентичную таковой в переменных плейбука deploy_iaas.yml;

– существование проекта openstack типа admin, идентичного таковому в переменных плейбука deploy_iaas.yml;

– существование "внешней" сети, создание разработчиками соответствующих записей в БД (да, вручную);

– существование образа glance с именем "cirros",и id, соответствующим таковому в переменных плейбука deploy_iaas.yml;

– существование "типа диска" с именем "fast";

– сетевую связность между узлом с компонентом iass и узлом управления облачной платформой с доступностью портов:

```
35357
5000
8041
8774
8776
8778
9292
9360
9362
9363
9364
9696
10001
```

Далее. Настраиваем компоненты orchestrator, portal и iaas.

Лист регистрации изменений

Номера листов (страниц)					Всего листов (страниц) в докум.	№ документа	Входящий № сопроводительного докум. и дата	Подп.	Дата
Изм.	измененных	замененных	новых	аннулированных					