Знакомство с облачной инфраструктурой. Google Cloud Platform

План самостоятельной работы

- Создание учетной записи в GCP
- Создание в веб-интерфейсе инстансов ВМ и подключение к ним по SSH
- Рассмотрим варианты подключения к хостам через бастион-хост и VPN

Проверка домашнего задания

- Создайте новую ветку `cloud-bastion` в вашем репозитории `<yourname>_infra` для выполнения данного ДЗ
- Прием ДЗ будет производиться через *Pull Request* ветки с ДЗ к ветке `master` вашего репозитория
- После того, как будет получен approve пул реквеста, ветку с ДЗ **нужно** смерджить и закрыть PR

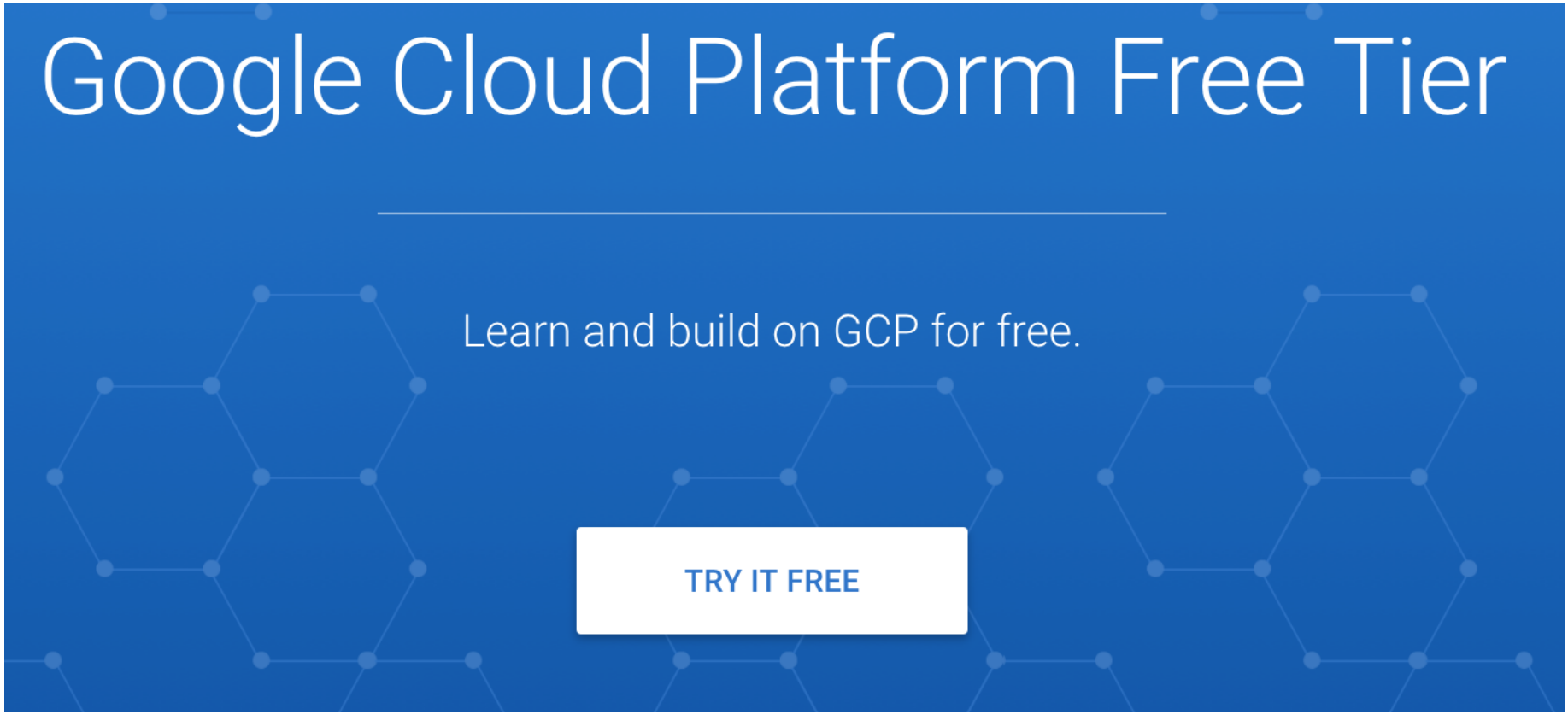
Создание учетной записи в Google Cloud Platform

Необходимо зарегистрироваться в GCP по ссылке:

- https://cloud.google.com/free/

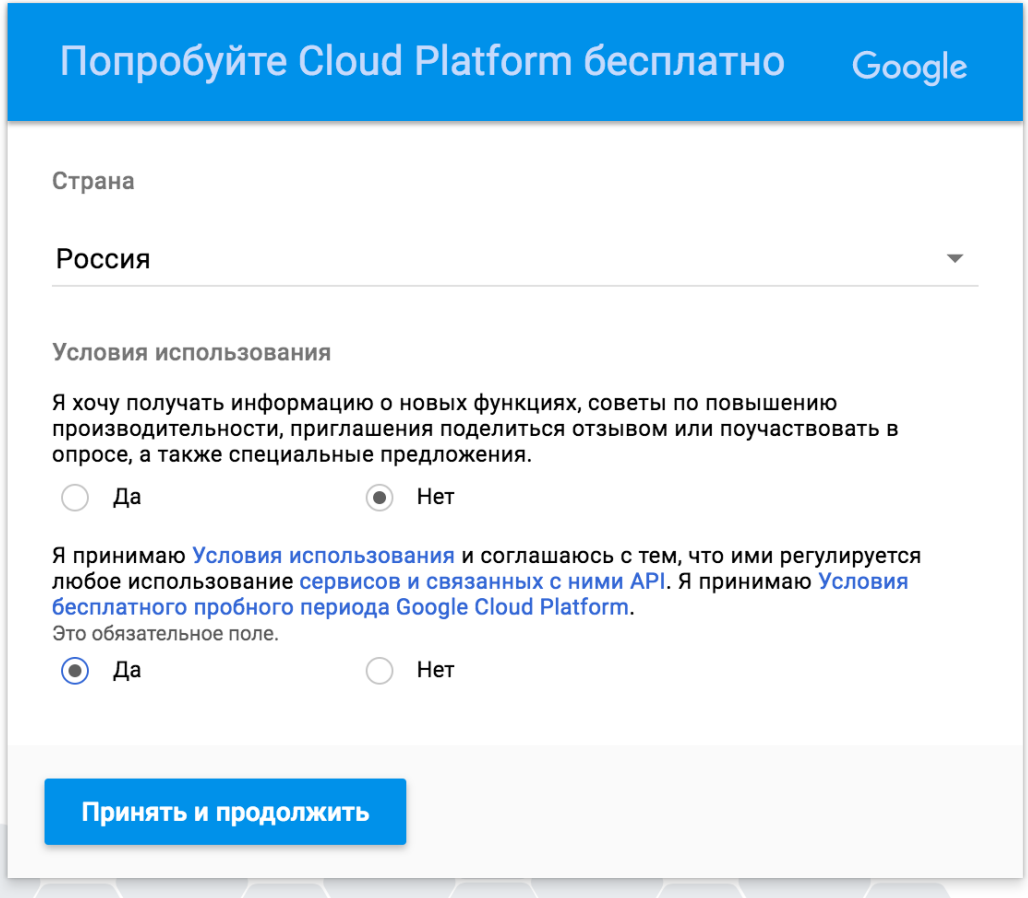
> Для регистрации рекомендуется использовать новую и отдельную учетную запись Google

Создание учетной записи в Google Cloud Platform



Создание учетной записи в Google Cloud Platform

Google Cloud Platform



Создание учетной записи в Google Cloud Platform

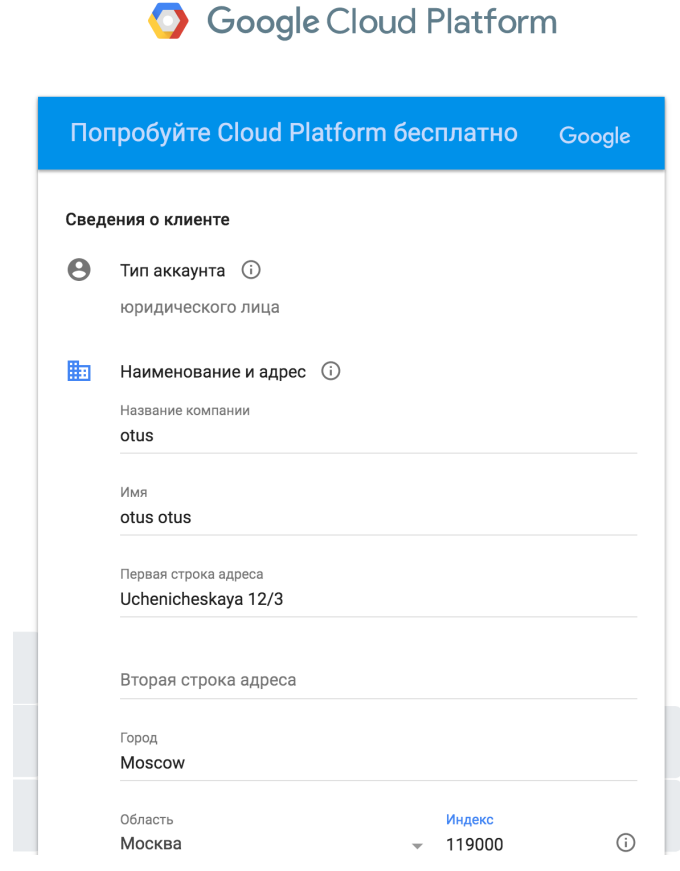
Во время регистрации Google может запросить ввести данные платежной карты, это единственное платежное требование.

После окончания trial-периода Google **не будет** автоматически снимать средства с карты, **можно не волноваться**.

От предложений использовать карту после окончания trial-периода рекомендуется отказываться.

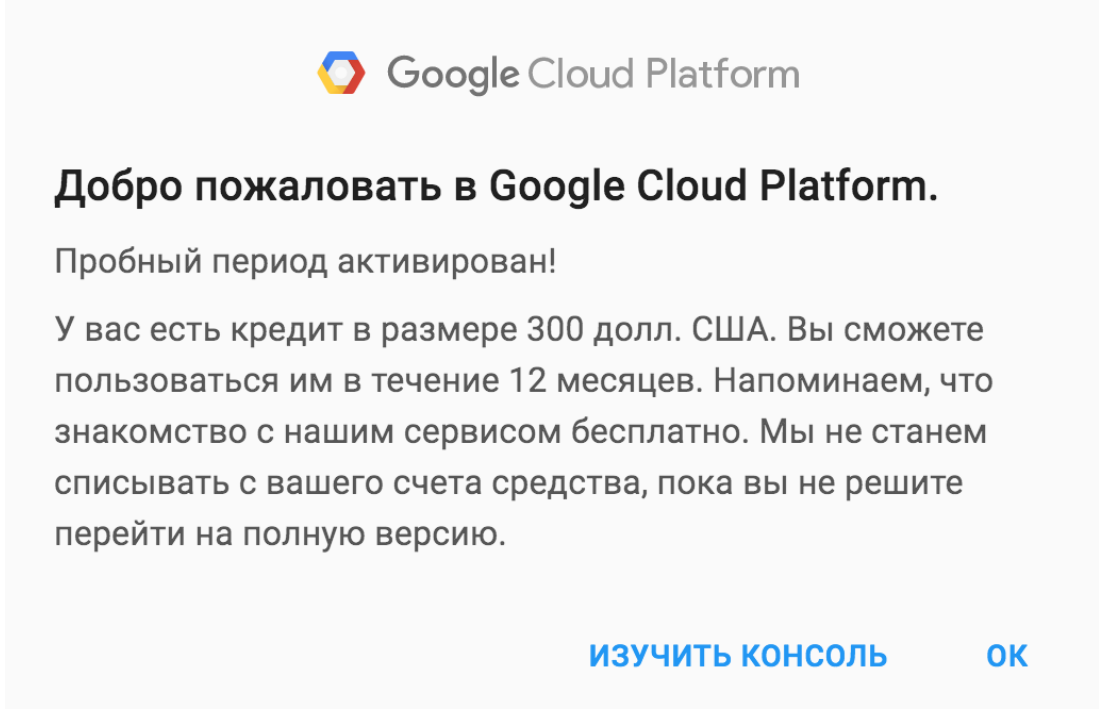
Создание учетной записи в Google Cloud Platform

Заполните поля, необходимые для регистрации

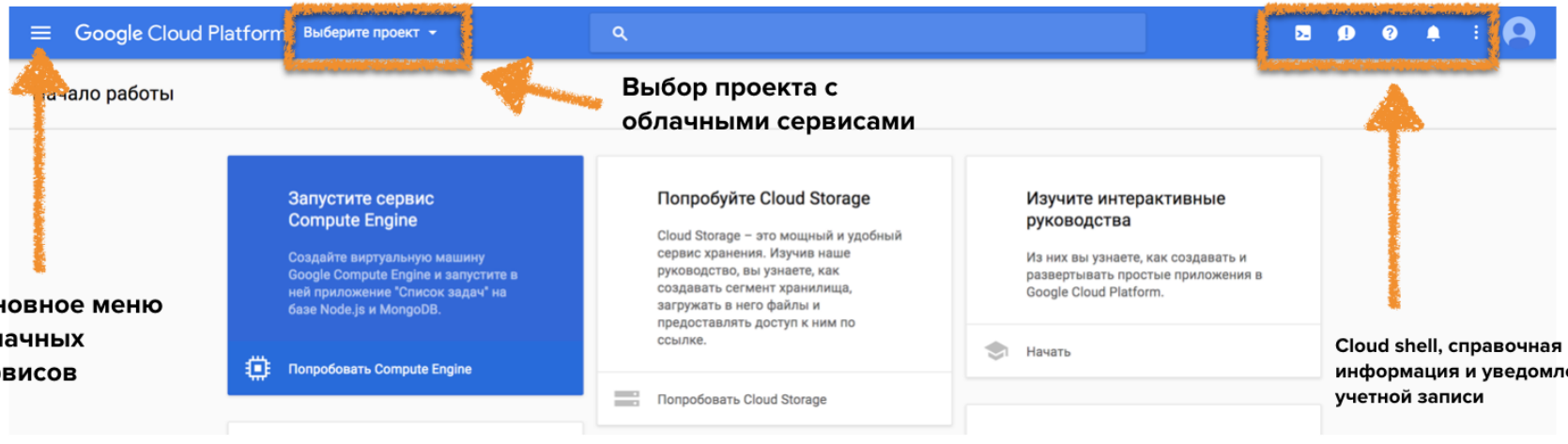


Создание учетной записи в Google Cloud Platform

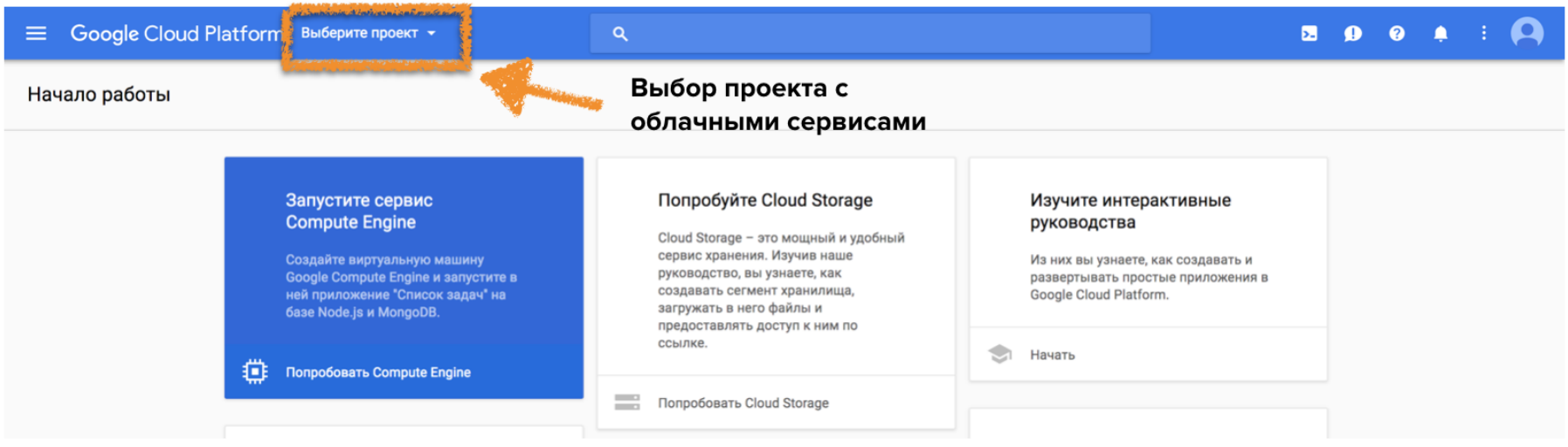
В конце успешной регистрации должно отобразиться окно приветствия



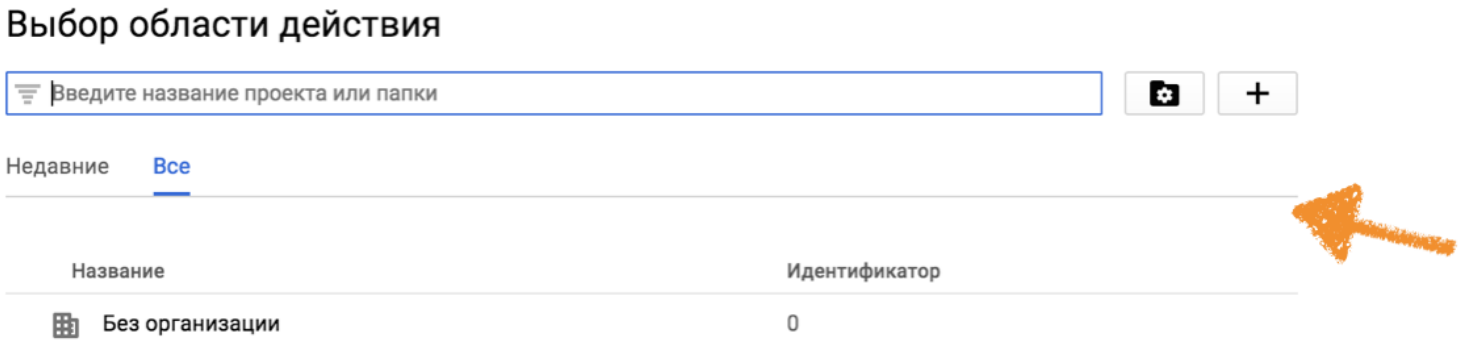
Основные элементы управления



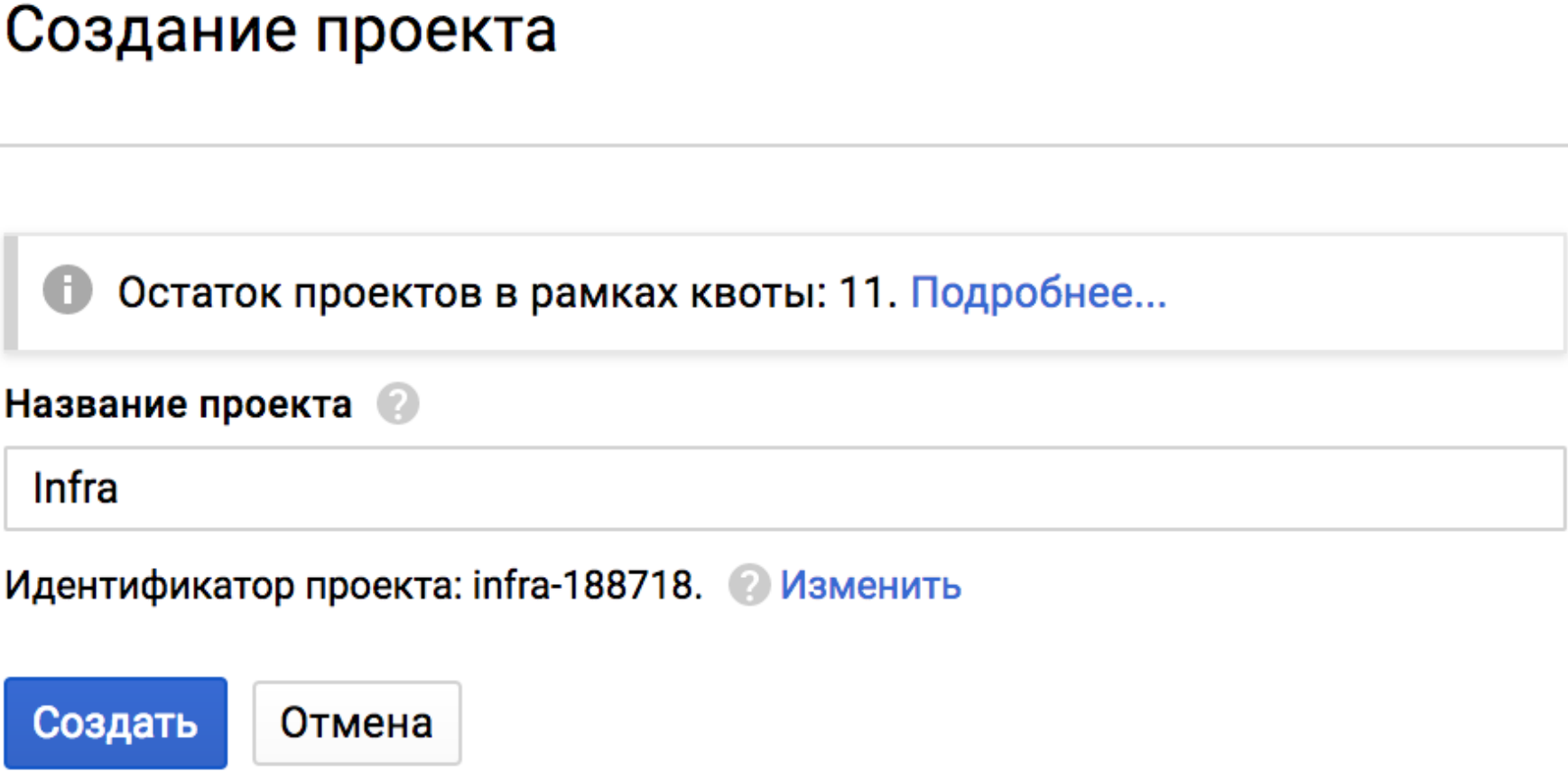
Создаем новый проект



Создаем новый проект



Создаем новый проект



Работа с Google Compute Engine

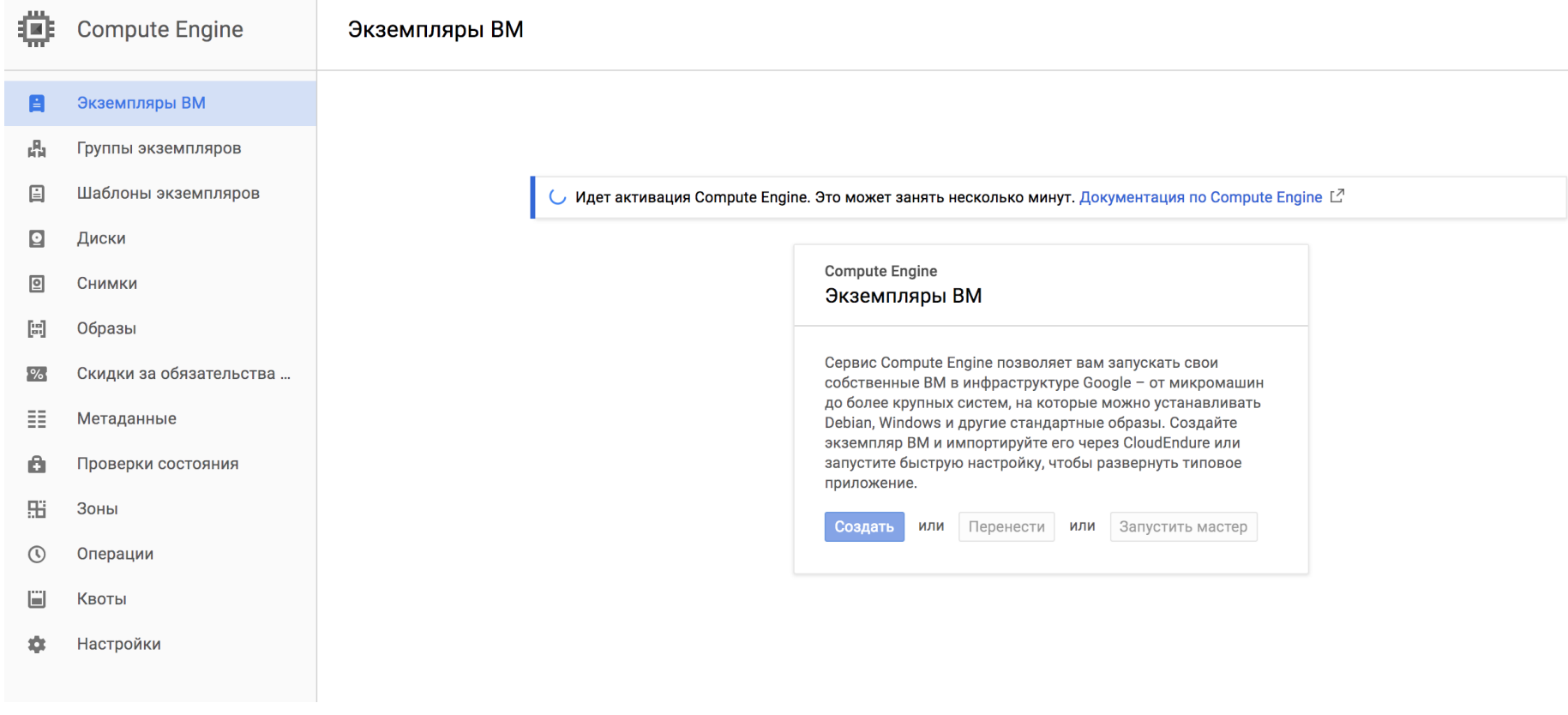
В следующих заданиях мы будем работать с IaaS слоями Google Compute Platform:

- VPC сети (Virtual Private Network)
- GCE Metadata (для управления ключами доступа к серверам)
- GCE VM (для создания и управления инстансами виртуальных машин)

А также создадим пару ключей и привяжем ее к метаданным GCE (Google Compute Engine) для последующего получения доступа на виртуальные машины

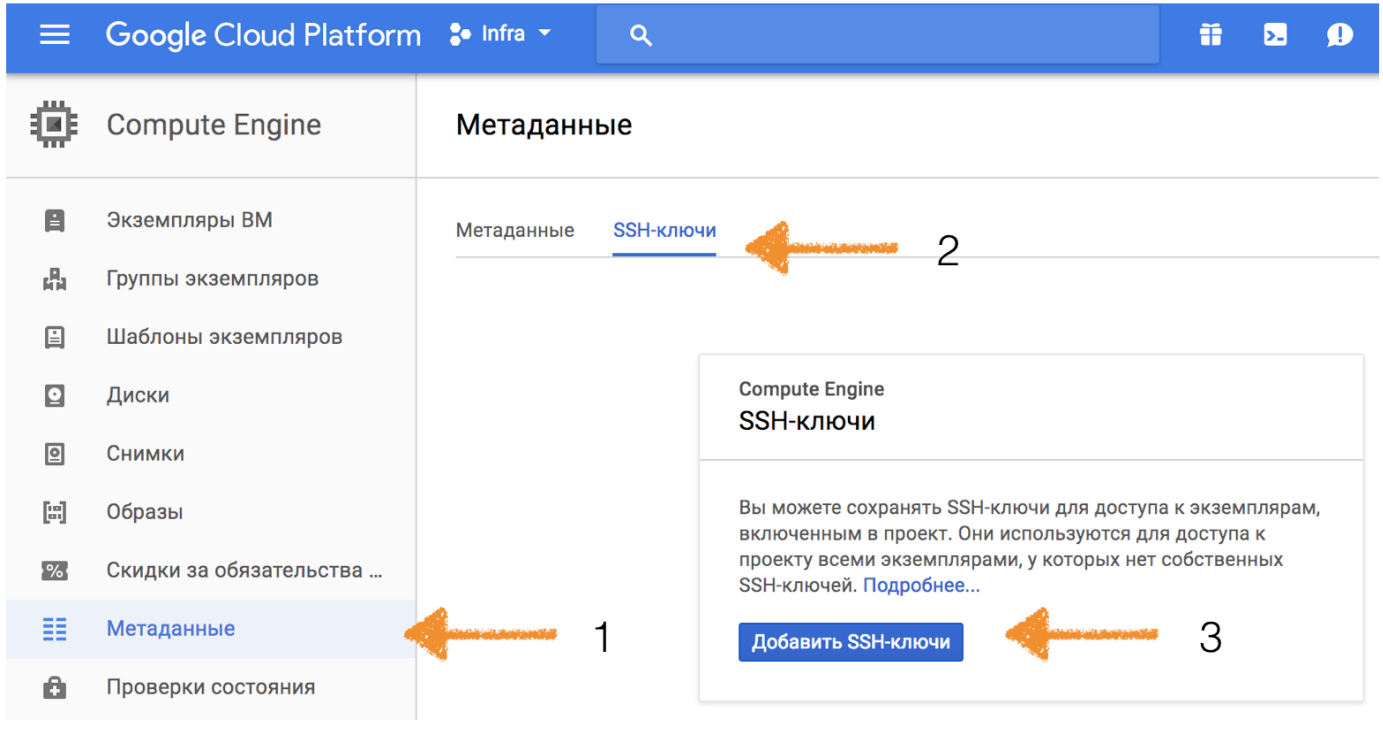
Работа с Google Compute Engine

Из основного меню переходим в **Compute Engine (GCE)**, дожидаемся активации



Работа с GCE: Метаданные

Для начала перейдем в раздел меню **Метадата**, выберем вкладку **SSH ключи** и нажмем **Добавить SSH-ключи**



Генерация пары ключей

На вашей пользовательской Linux/Unix системе сгенерировать пару ключей можно при помощи утилиты ssh-keygen (часть ssh-agent) (пример на следующем слайде)

После того как команда отработает, мы получаем пару из приватного и публичного ключей в домашнем каталоге текущего пользователя системы

- **Приватный ключ:**
  - `~/.ssh/appuser`
- **Публичный ключ:**
  - `~/.ssh/appuser.pub`

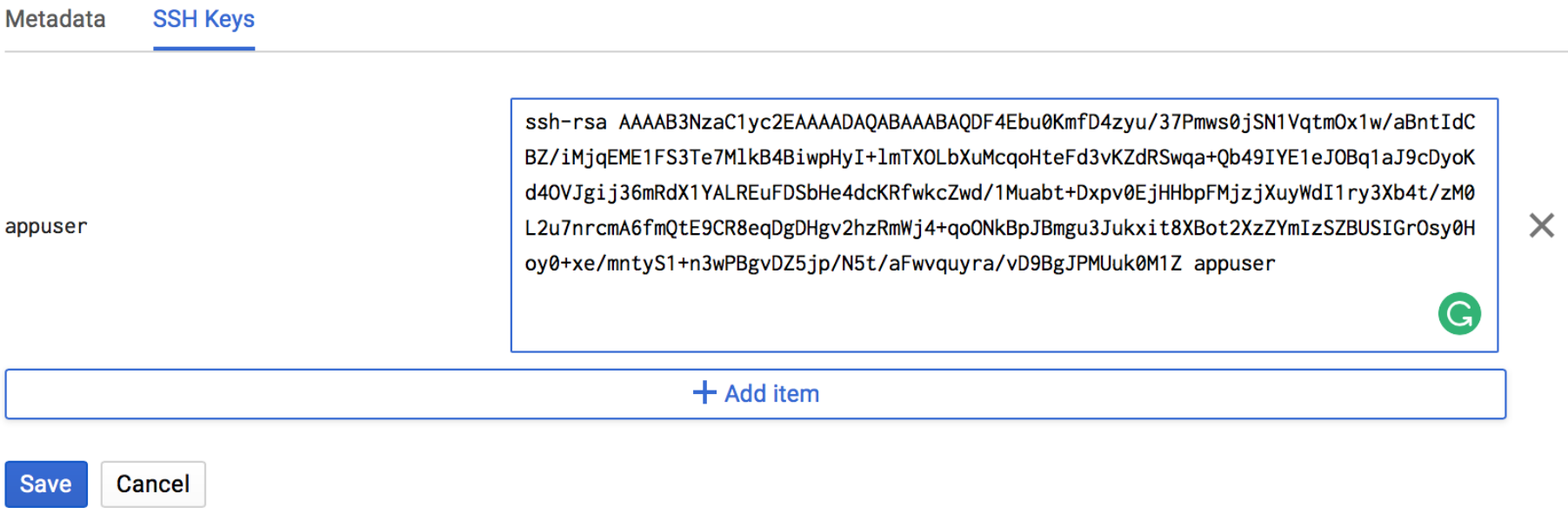
Генерация пары ключей

Пример вывода ssh-keygen (для пользователя *appuser*):

```
> ssh-keygen -t rsa -f ~/.ssh/appuser -C appuser -P ""
Generating public/private rsa key pair.
Your identification has been saved in /Users/Express42/.ssh/appuser.
Your public key has been saved in /Users/Express42/.ssh/appuser.pub.
The key fingerprint is:
SHA256:m006lywMIY72dA2kpNEtKktb10jg2pvrd2ajkRHoG6c appuser
The key's randomart image is:
+---[RSA 2048]----+
|  .oo..          |
|  .+++.          |
|  .++o=          |
|..=+ +.=         |
|.+=++oo S .      |
|.o oB.oo B .     |
|   E.o  B =      |
|    ...= +       |
|  .o..= .        |
+----[SHA256]-----+
```

Вносим в форму публичный ключ

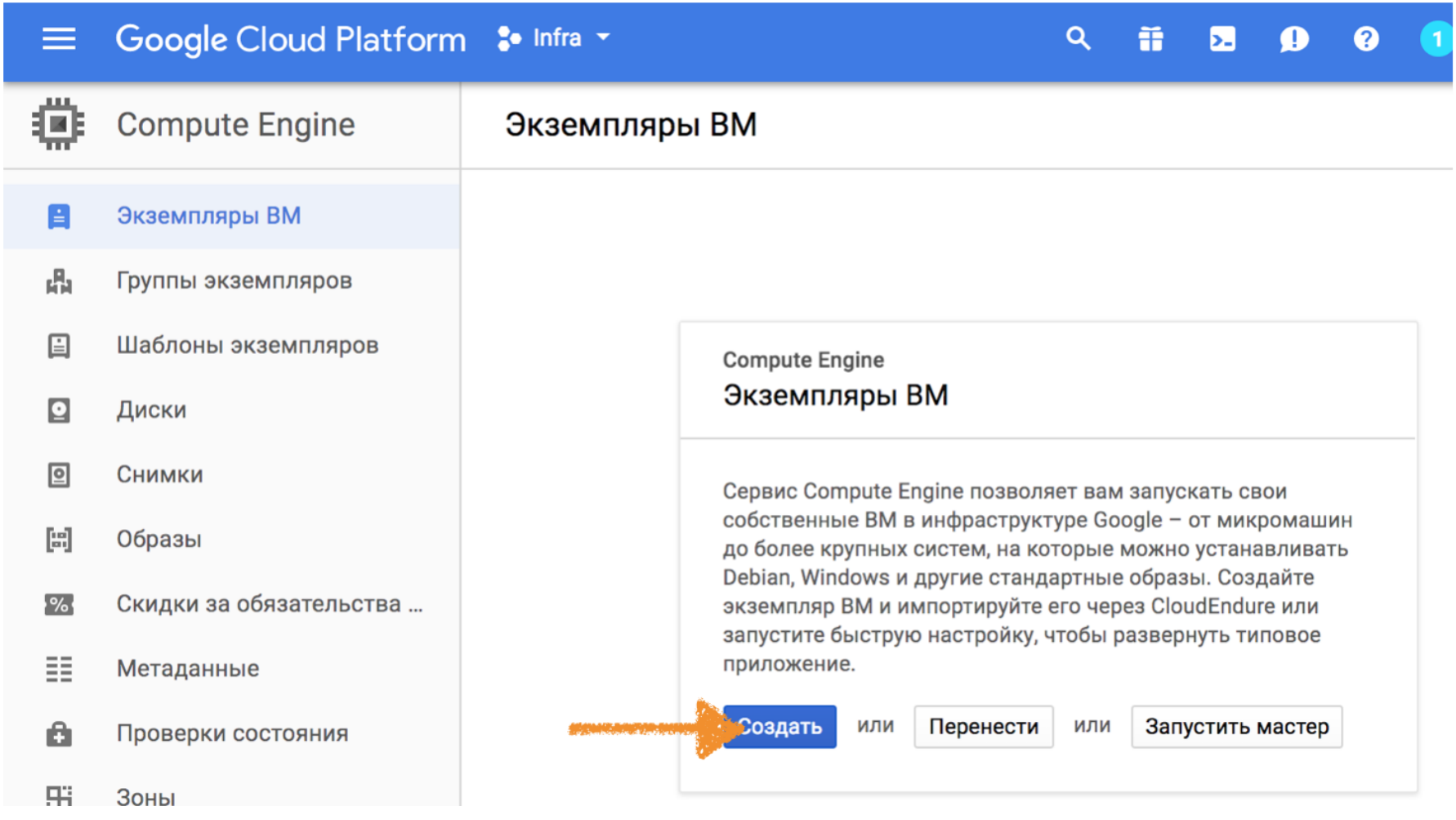
Содержимое файла `~/.ssh/appuser.pub` вносим в форму ввода ключа и сохраняем



SSH-ключи в Метаданных проекта GCP

- Действуют на все виртуальные машины в проекте
- Могут быть переопределены при создании виртуальной машины
- Могут быть заблокированы при создании виртуальной машины - чтобы ни один из описанных в метадате проекта ключей не использовался

Создаем Экземпляр VM (инстанс)



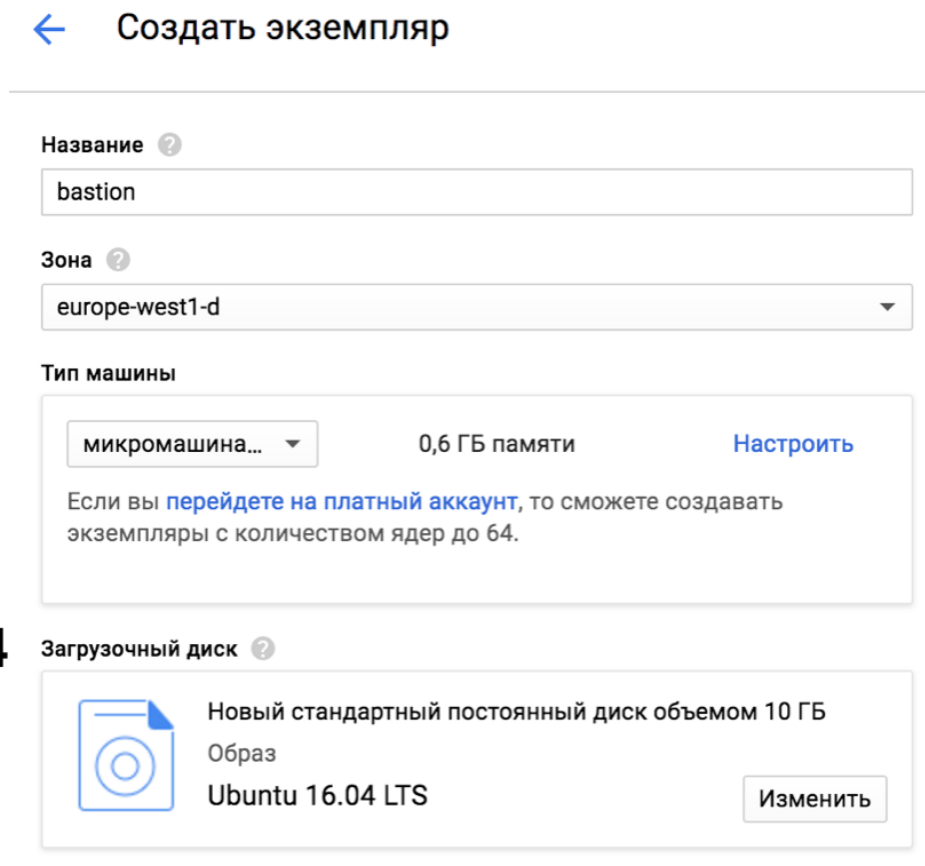
Создаем Экземпляр VM (инстанс)

Вбиваем имя хоста: **bastion**

Зона: **europe***

Тип машины: **микромашина**

Загрузочный диск: **Ubuntu 16.04**



Создаем Экземпляр VM (инстанс)

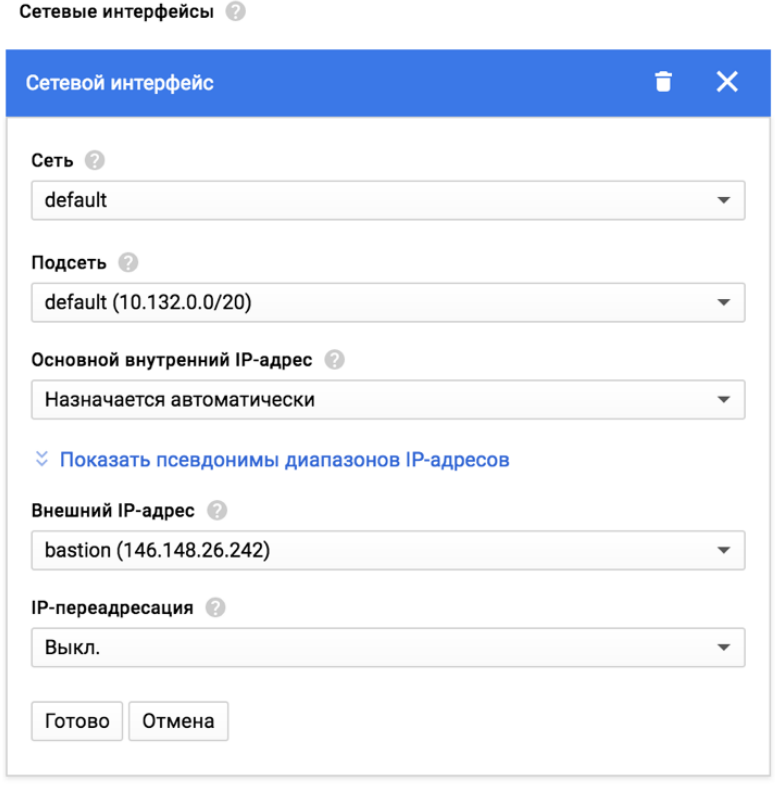
Настройка параметров сети -> Сеть -> Сетевые интерфейсы

Оставляем сеть **default**

Внешний IP: **Создать адрес**

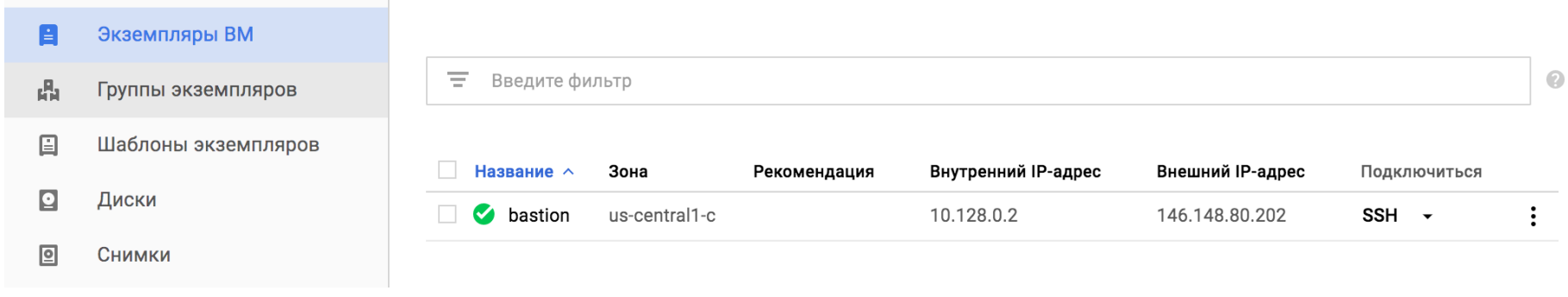
Название адреса: **bastion**

default сеть может отличаться от той, что на скриншоте, это нормально



Создаем Экземпляр VM (инстанс)

Нажимаем **Создать** и дожидаемся готовности VM на панели Compute Engine



Проверяем подключение по полученному внешнему адресу

Проверяем из локальной консоли подключение к созданной VM: `ssh -i ~/.ssh/appuser appuser@<внешний IP VM>`

```
$ ssh -i ~/.ssh/appuser appuser@<внешний IP VM>
The authenticity of host '<внешний IP VM> (<внешний IP VM>)' can't be established.
ECDSA key fingerprint is SHA256:Vp9dwnLwlyfLbWPpgR/bcFze0UIvuqCqhnG6uRjMzRo.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '<внешний IP VM>' (ECDSA) to the list of known hosts.
Welcome to Ubuntu 16.04.3 LTS (GNU/Linux 4.10.0-42-generic x86_64)
* Documentation:  https://help.ubuntu.com
* Management:     https://landscape.canonical.com
* Support:        https://ubuntu.com/advantage
Get cloud support with Ubuntu Advantage Cloud Guest:
    http://www.ubuntu.com/business/services/cloud
0 packages can be updated.
0 updates are security updates.
appuser@bastion:~$ cat /etc/issue Ubuntu 16.04.3 LTS \n \l
```

VPC

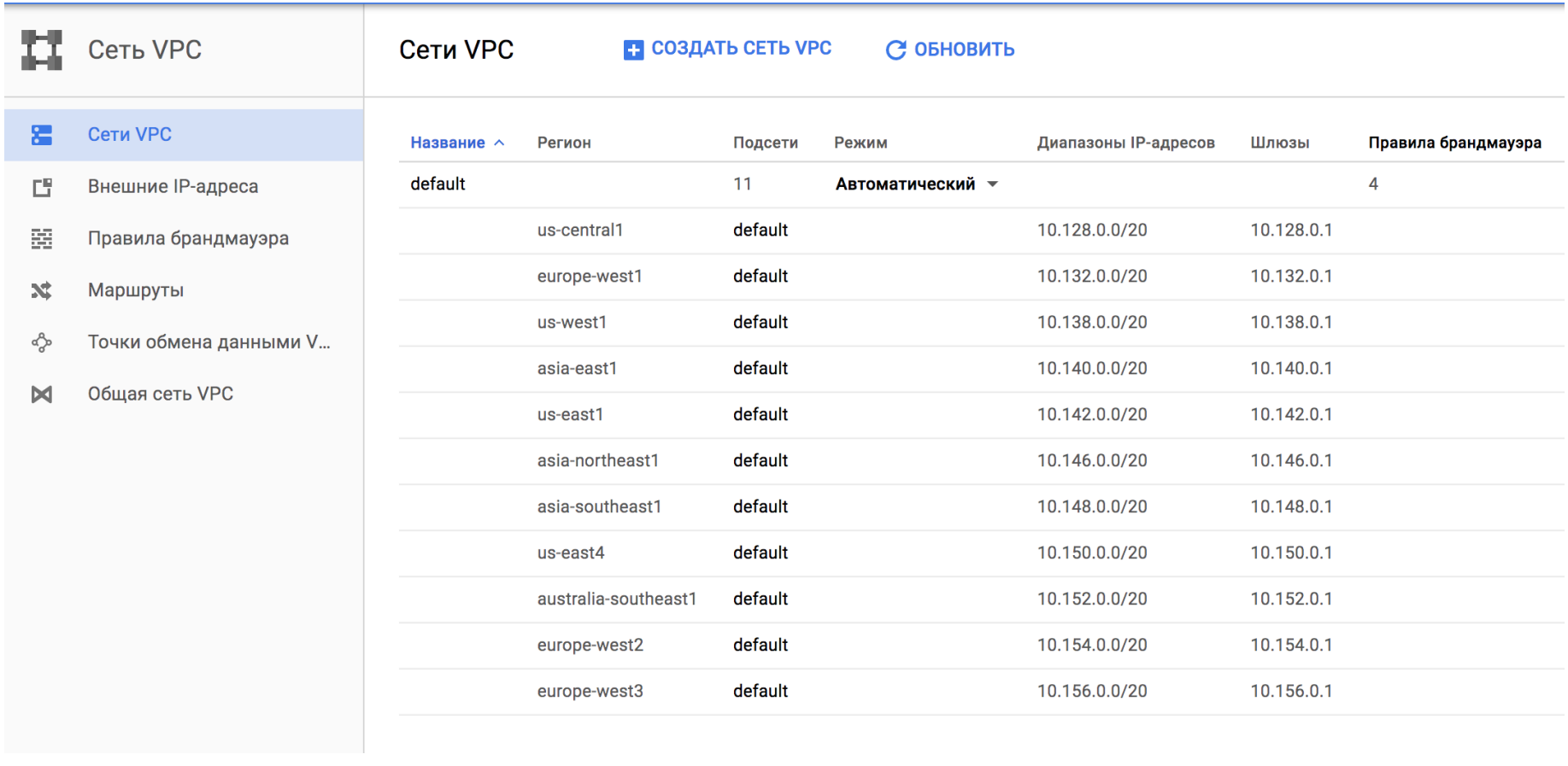
Часть IaaS состоит из сетевого слоя, но это не значит, что управление им должно выполняться отдельно через раздел **VPC**

В нашем случае, при создании проектов автоматом было создано по одной приватной сети на каждый регион и шлюз для выхода в интернет

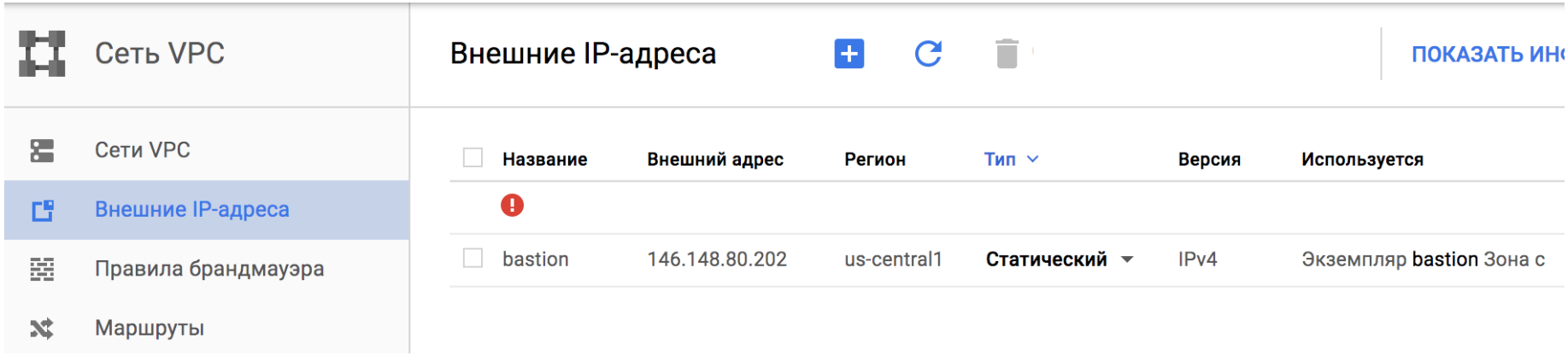
При создании VM мы также указали, что хотим завести статический внешний IP **bastion**, который также будет виден в разделе VPC

VPC: Приватные сети

Из главного меню, откройте раздел VPC:



VPC: Публичные адреса



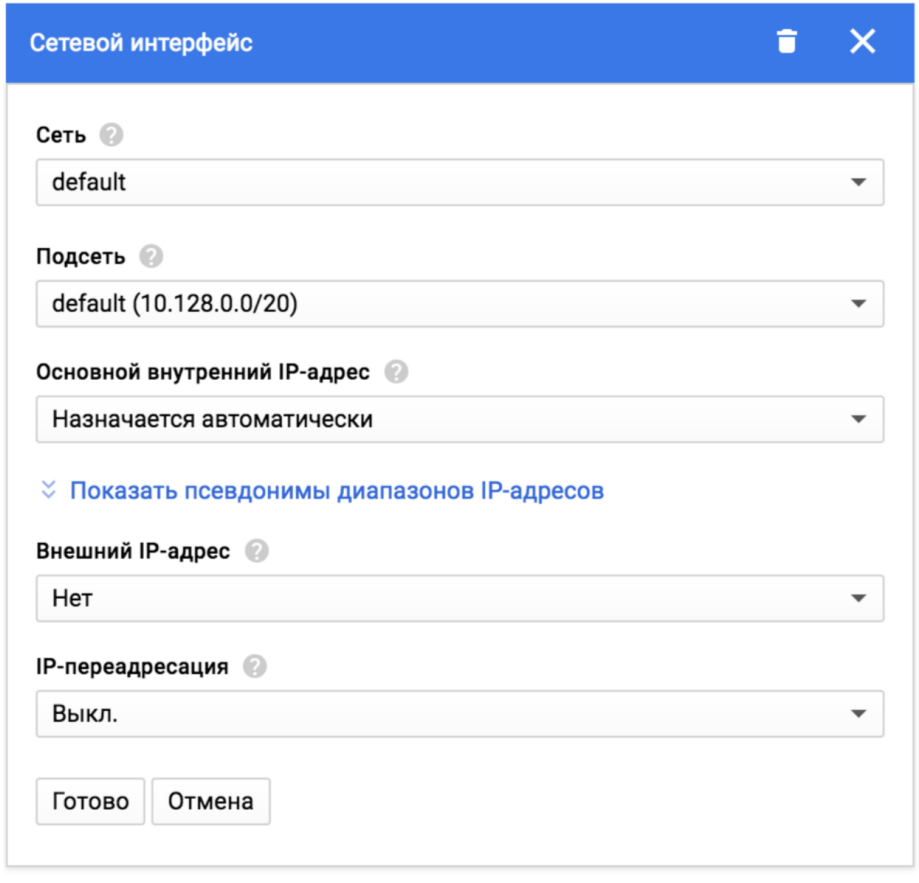
VPC

Примеры управления сетями и разделение топологий будут описаны в будущих лекциях, с использованием инструментов, работающих через API

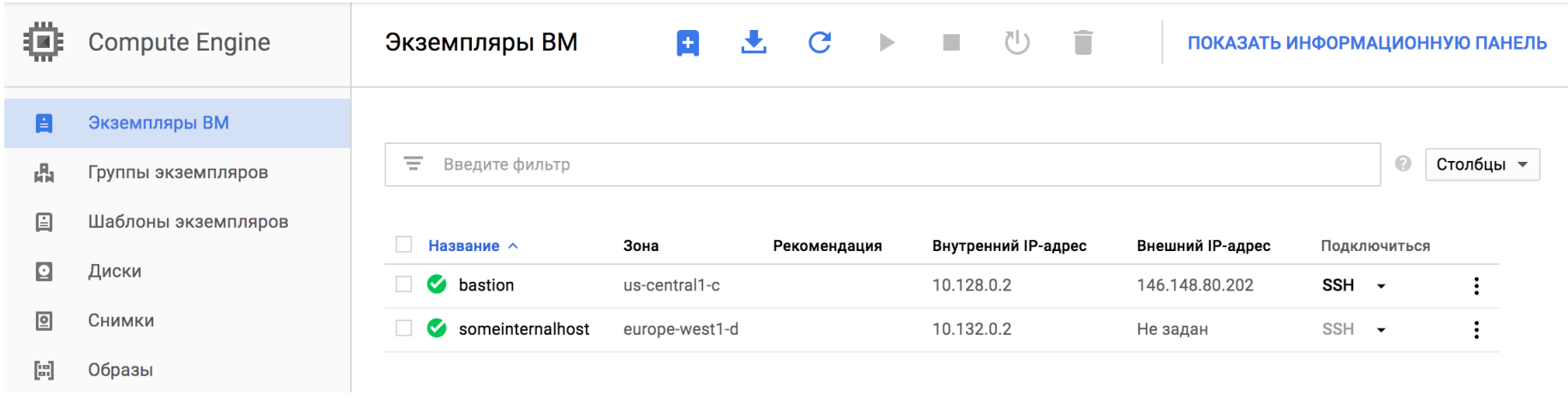
Создаем вторую VM без внешней сети

По аналогии с предыдущими шагами по созданию VM, инициировать создание второй машины с именем **someinternalhost**

Среди отличий в создании, в разделе управления сетями, **необходимо убрать создание публичного адреса** (см. скриншот)



Проверяем результат



Рассмотрим текущее состояние хостов

Для верности эксперимента, пробуем зайти по SSH на bastionhost, а с него по внутреннему адресу на someinternalhost (используйте ваши адреса)

```
$ ssh -i ~/.ssh/appuser appuser@146.148.80.202
Welcome to Ubuntu 16.04.3 LTS (GNU/Linux 4.10.0-32-generic x86_64)
...
0 packages can be updated.
0 updates are security updates.
Last login: Tue Aug 29 06:32:23 2017 from 79.164.31.28
To run a command as administrator (user "root"), use "sudo <command>". See "man
sudo_root" for details.
appuser@bastion:~$ ssh 10.132.0.2
Permission denied (publickey).
appuser@bastion:~$
```

Результат неудовлетворителен 😝

Используем Bastion host для прямого подключения к инстансам внутренней сети

Настроим SSH Forwarding на вашей локальной машине:

```
$ ssh-add -L
The agent has no identities.
```

Добавим приватный ключ в ssh агент авторизации:

```
$ ssh-add ~/.ssh/appuser
Identity added: /Users/otus/.ssh/appuser (/Users/otus/.ssh/appuser)
```

Используем Bastion host для сквозного подключения

Пробуем подключаться вновь, добавив в параметры подключения ключик –A, чтобы явно включить SSH Agent Forwarding

```
$ ssh -i ~/.ssh/appuser -A appuser@146.148.80.202
Welcome to Ubuntu 16.04.3 LTS (GNU/Linux 4.10.0-32-generic x86_64)
```

```
appuser@bastion:~$ ssh 10.132.0.2
Welcome to Ubuntu 16.04.3 LTS (GNU/Linux 4.10.0-32-generic x86_64)
...
Last login: Tue Aug 29 06:32:27 2017 from 10.128.0.2
To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.
```

Используем Bastion host для сквозного подключения

Убедимся, что попали на нужный хост:

```
appuser@someinternalhost:~$ hostname
someinternalhost
```

```
appuser@someinternalhost:~$ ip a show ens4
2: ens4: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1460 qdisc pfifo_fast state UP group
default qlen 1000
    link/ether 42:01:0a:84:00:02 brd ff:ff:ff:ff:ff:ff
    inet 10.132.0.2/32 brd 10.132.0.2 scope global ens4
    valid_lft forever preferred_lft forever
    inet6 fe80::4001:aff:fe84:2/64 scope link
    valid_lft forever preferred_lft forever
```

Успех, но не очень удобно 😉

Проверим отсутствие каких-либо приватных ключей на bastion машине:

```
appuser@bastion:~$ ls -la ~/.ssh/
total 16
drwx------ 2 appuser appuser 4096 Aug 29 06:07 .
drwxr-xr-x 4 appuser appuser 4096 Aug 29 06:07 ..
-rw------- 1 appuser appuser 397 Aug 29 05:50 authorized_keys
-rw-r--r-- 1 appuser appuser 222 Aug 29 06:03 known_hosts
```

Самостоятельное задание

Исследовать способ подключения к `someinternalhost` в одну команду из вашего рабочего устройства, проверить работоспособность найденного решения и внести его в **README.md** в вашем репозитории

Дополнительное задание:

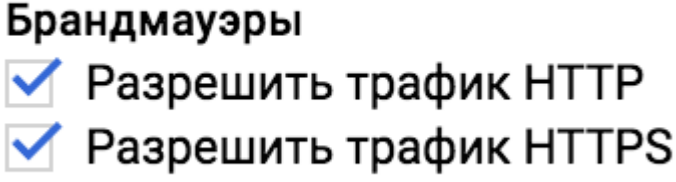
Предложить вариант решения для подключения из консоли при помощи команды вида `ssh someinternalhost` из локальной консоли рабочего устройства, чтобы подключение выполнялось по алиасу `someinternalhost` и внести его в **README.md** в вашем репозитории

Создаем VPN-сервер для серверов GCP

Не удаляя предыдущие серверы, создадим схему с VPN-сервером Pritunl, после сгенерируем конфигурацию VPN-клиента и подключимся к VPN-сети с последующим доступом в частную сеть облака

Создаем VPN-сервер для серверов GCP

Перед установкой перейдем в настройки `bastion` VM через Панель управления и разрешим в Брэндмауэре **HTTP/HTTPS-трафик**



Создаем VPN-сервер для серверов GCP

На хосте bastion выполняем команды Ссылка на gist

```
$ cat <<EOF> setupvpn.sh
#!/bin/bash
echo "deb http://repo.mongodb.org/apt/ubuntu xenial/mongodb-org/3.4 multiverse" > /etc/
apt/sources.list.d/mongodb-org-3.4.list
echo "deb http://repo.pritunl.com/stable/apt xenial main" > /etc/apt/sources.list.d/
pritunl.list
apt-key adv --keyserver hkp://keyserver.ubuntu.com --recv
0C49F3730359A14518585931BC711F9BA15703C6
apt-key adv --keyserver hkp://keyserver.ubuntu.com --recv
7568D9BB55FF9E5287D586017AE645C0CF8E292A
apt-get --assume-yes update
apt-get --assume-yes upgrade
apt-get --assume-yes install pritunl mongodb-org
systemctl start pritunl mongod
systemctl enable pritunl mongod
EOF
```

Создаем VPN-сервер для серверов GCP

Затем выполняем:

```
$ sudo bash setupvpn.sh
```

В результате:

- В текущей директории будет создан файл `setupvpn.sh`, описывающий установку VPN-сервера
- Будет установлена `mongodb` и VPN-сервер `pritunl`

Создаем VPN-сервер для серверов GCP

- Открываем в браузере ссылку: `https://<адрес bastion VM>/setup`
- Ошибку SSL пропускаем и доверяем этому сайту
- Следуем инструкциям на экране (запрашиваемые команды запускать через `sudo`)

Создаем VPN-сервер для серверов GCP

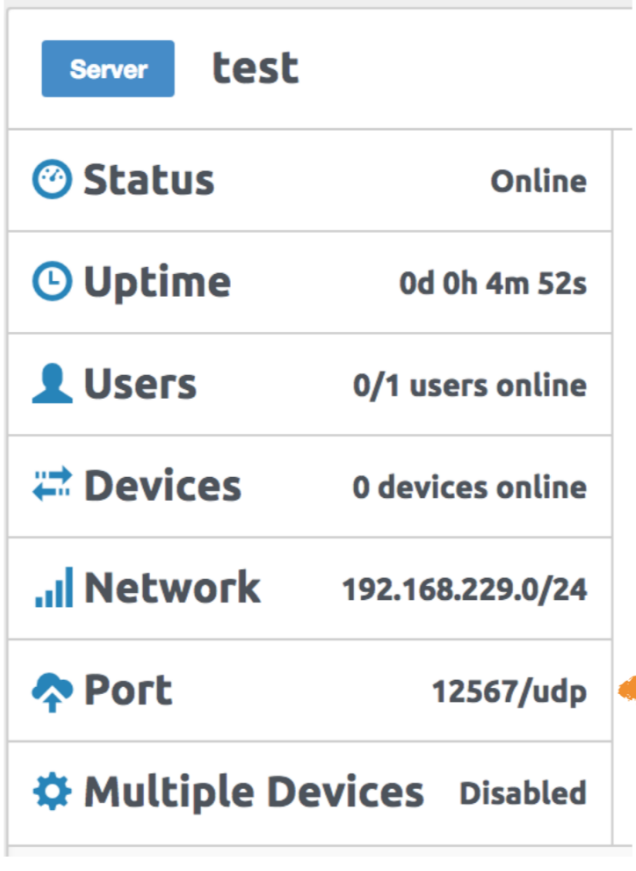
В конце установки авторизуемся, используя логин/пароль **pritunl/pritunl**

Далее добавляем в веб интерфейсе:

- Организацию
- Пользователя **test** с PIN **62141575072376783346705915562762**
- Сервер (затем привязываем его к организации и запускаем)

Подробнее: https://docs.pritunl.com/v1/docs/connecting

Создаем VPN-сервер для серверов GCP



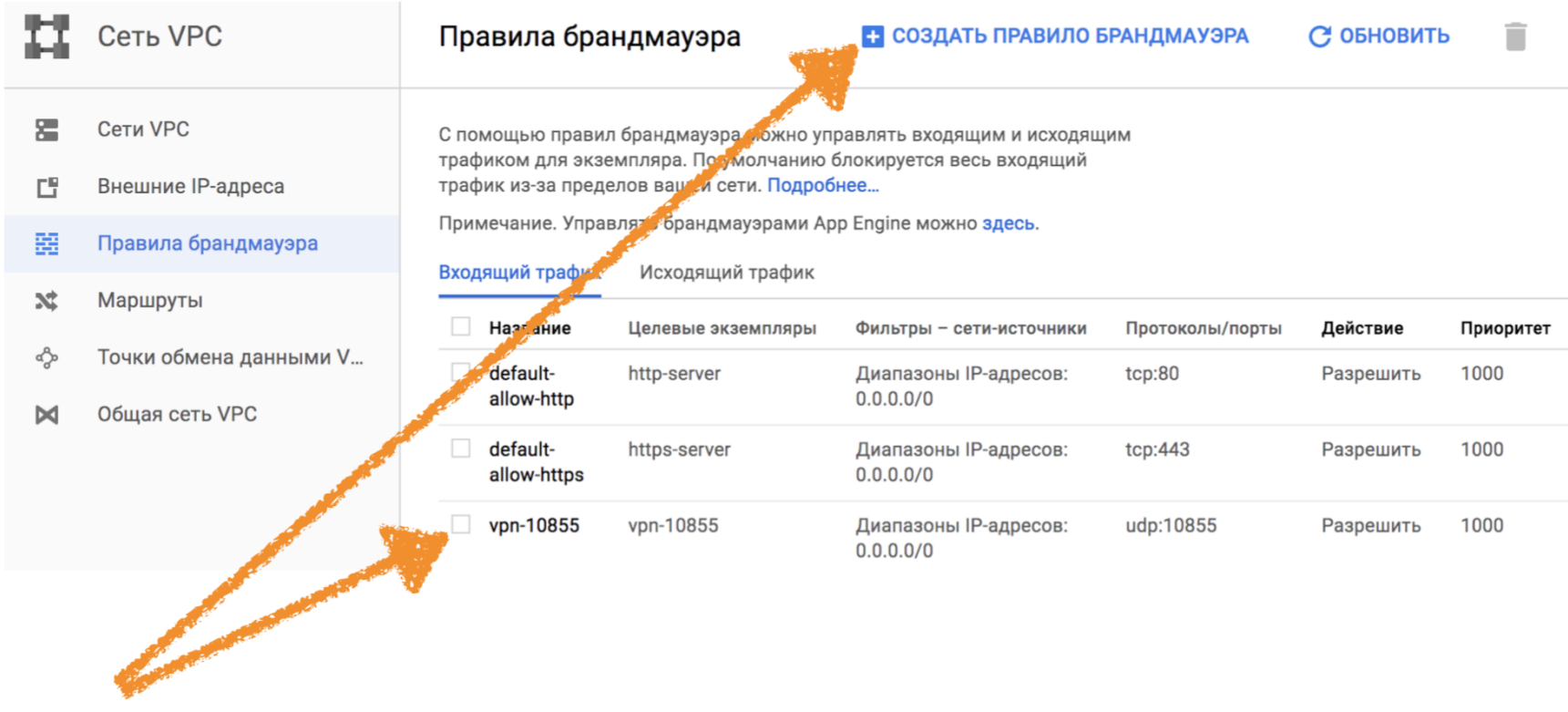
Запомните порт,
на котором поднялся сервер

Создаем VPN-сервер для серверов GCP

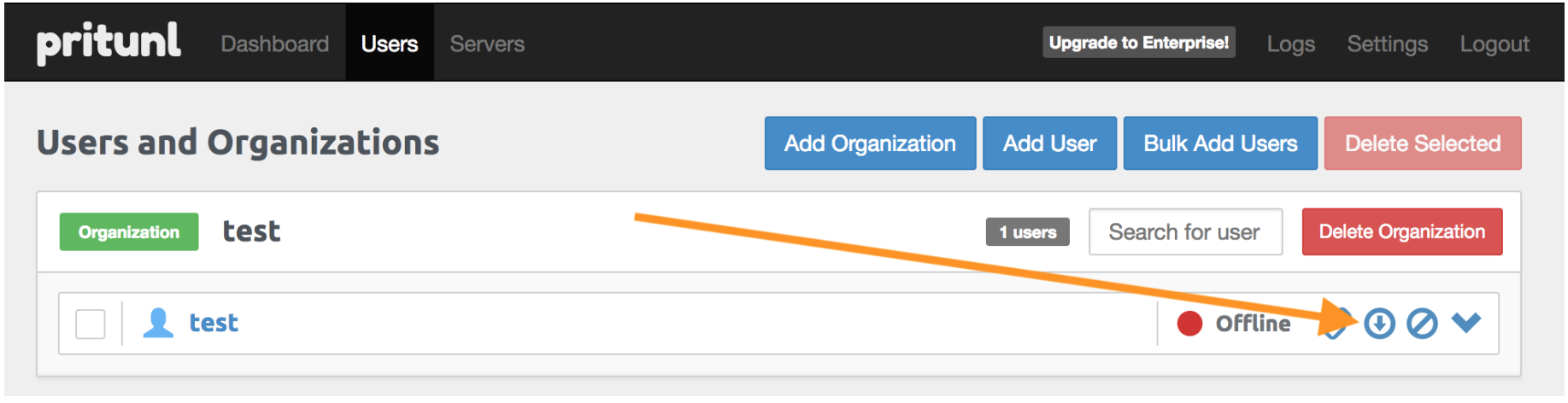
Далее:

- Создадим правило для открытия порта VPN, в данном случае `udp:10855` (порт указывается при создании сервера)
- Добавим в инстансе `bastion` в **Теги сети** наше новое правило

Создаем VPN-сервер для серверов GCP



Создаем VPN-сервер для серверов GCP



Таким образом, мы установили и сконфигурировали OpenVPN-сервер Pritunl

Теперь надо скачать конфигурационный файл для подключения (см. стрелку на скриншоте)

Проверяем подключение к VPN

- Добавьте полученный конфигурационный файл `*.ovpn` в клиент OpenVPN на вашем компьютере
- Проверьте подключение к VPN-серверу
- Проверьте возможность подключения к `someinternalhost` с вашего компьютера после подключения к VPN:

```
$ ssh -i ~/.ssh/appuser appuser@<внутренний IP someinternalhost>
```

Дополнительное задание

Сейчас веб-интерфейс VPN-сервера Pritunl работает с самоподписанным сертификатом. И браузер постоянно ругается на это.

С помощью сервисов sslip.io/xip.io и Let’s Encrypt реализуйте использование валидного сертификата для панели управления VPN-сервера

Задание

1. Выполните задание про подключение через бастион хост.
2. Добавьте в ваш репозиторий Infra (ветка `cloud-bastion`):
   - файл `setupvpn.sh`
   - конфигурационный файл для подключения к VPN (переименуйте `*.ovpn` в **cloud-bastion.ovpn**)
3. Опишите в `README.md` и получившуюся конфигурацию и данные для подключения в **следующем формате** (важно для проверки!):

```
bastion_IP = 35.198.167.169
someinternalhost_IP = 10.156.0.3
```

4. Добавьте "Labels" **cloud-bastion** к вашему Pull Request

Задание

- После успешного выполнения автоматической проверки в TravisCI можете удалить **оба инстанса** (`bastion` и `someinternalhost`) и **постоянный IP адрес**.