

Инструкция по установке Q.RunneR

Exported on Jul 08, 2019

1	Предварительная настройка системы.....	3
1.1	Отключение файла подкачки.....	3
1.2	Установка зависимостей	3
1.3	Сетевые порты	3
2	Установка	4

1 Предварительная настройка системы

1.1 Отключение файла подкачки

Отключить файл подкачки в текущей сессии командой:

```
sudo swapoff -a
```

Чтобы swap после перезагрузки не включался - надо в `/etc/fstab` закомментировать его в монтирование, например:

```
#/swap.img none swap sw 0 0
#/swapfile swap swap defaults 0 0
```

1.2 Установка зависимостей

Установить обязательные пакеты, необходимы для работы:

- Q.PorteR, в соответствии с [инструкцией по установке](#)
- etcd из стандартного репозитория пакетов используемой операционной системы:
 - Debian/Ubuntu: `sudo apt-get install -y etcd`
 - CentOS: `yum -y installetcd`

1.3 Сетевые порты

В тестовом окружении можно отключить фаервол, в production-окружении так делать нельзя, необходимо открыть следующие порты:

Узел с контрольной панелью (мастер)

Протокол	Направление	Диапазон портов	Назначение
TCP	Входящие	6443	Q.Runner API server
TCP	Входящие	2379-2380	etcd server client API
TCP	Входящие	10250	QRunlet API
TCP	Входящие	10251	qrun-scheduler
TCP	Входящие	10252	qrun-controller-manager

Рабочие узлы

Протокол	Направление	Диапазон портов	Назначение
TCP	Входящие	10250	QRunlet API
TCP	Входящие	30000-32767	NodePort Services**

** Диапазон портов по умолчанию

Также следует учитывать, что в каждом конкретном случае список открываемых портов будет изменяться.

2 Установка

Установить пакеты (deb или rpm, в зависимости от операционной системы):

```
# Debian/Ubuntu
sudo dpkg -i <package-filename.deb>
# CentOS
sudo yum localinstall <package-filename.rpm>
```

Список

Создать пользователя, от имени которого будут выполняться процессы

```
sudo groupadd --system qrunner
sudo useradd -s /sbin/nologin --system -g qrunner qrunner
```

Создать каталог для настроек:

```
sudo mkdir -p /etc/qrunner
sudo chown qrunner:qrunner /etc/qrunner
```

Создать рабочий каталог для qrunlet:

```
sudo mkdir -p /var/lib/qrunlet
sudo chown qrunner:qrunner /var/lib/qrunlet
```

Создать каталог /var/run/qrunner

```
sudo mkdir -p /var/run/qrunner
sudo chown qrunner:qrunner /var/run/qrunner
```

Создать файл настроек /etc/qrunner/config:

```

###
# Q.Runner system config
#
# The following values are used to configure various aspects of all
# Q.Runner services, including
#
# qrun-apiserver.service
# qrun-controller-manager.service
# qrun-scheduler.service
# qrunlet.service
# qrun-proxy.service
# logging to stderr means we get it in the systemd journal
KUBE_LOGTOSTDERR="--logtostderr=true"

# journal message level, 0 is debug
KUBE_LOG_LEVEL="--v=0"

# Should this cluster be allowed to run privileged docker containers
KUBE_ALLOW_PRIV="--allow-privileged=false"

# How the controller-manager, scheduler, and proxy find the apiserver
KUBE_MASTER="--master=http://<MASTER HOSTNAME/IP>:8080"

```

Code Block 1 config

Создать файл настроек /etc/qrunner/apiserver:

```

###
# kubernetes system config
#
# The following values are used to configure the kube-apiserver
#
# The address on the local server to listen to.
KUBE_API_ADDRESS="--insecure-bind-address=0.0.0.0"

# The port on the local server to listen on.
# KUBE_API_PORT="--port=8080"

# Port minions listen on
KUBELET_PORT="--kubelet-port=10250"

# Comma separated list of nodes in the etcd cluster
KUBE_ETCD_SERVERS="--etcd-servers=http://127.0.0.1:2379"

# Address range to use for services
KUBE_SERVICE_ADDRESSES="--service-cluster-ip-range=10.254.0.0/16"

# default admission control policies
KUBE_ADMISSION_CONTROL="--admission-
control=NamespaceLifecycle,LimitRanger,SecurityContextDeny,ServiceAccount,ResourceQuota"

# Add your own!
KUBE_API_ARGS=""

```

Code Block 2 apiserver:

Создать файл автозапуска сервиса /etc/systemd/system/qrun-apiserver.service

```
[Unit]
Description=Q.Runner API Server
Documentation=
After=network.target
After=etcd.service

[Service]
EnvironmentFile=-/etc/qrunner/config
EnvironmentFile=-/etc/qrunner/apiserver
User=qrunner
ExecStart=/usr/bin/qrun-apiserver \
    $KUBE_LOGTOSTDERR \
    $KUBE_LOG_LEVEL \
    $KUBE_ETCD_SERVERS \
    $KUBE_API_ADDRESS \
    $KUBE_API_PORT \
    $KUBELET_PORT \
    $KUBE_ALLOW_PRIV \
    $KUBE_SERVICE_ADDRESSES \
    $KUBE_ADMISSION_CONTROL \
    $KUBE_API_ARGS
Restart=on-failure
Type=notify
LimitNOFILE=65536

[Install]
WantedBy=multi-user.target
```

Code Block 3 qrun-apiserver.service

Создать файл автозапуска сервиса /etc/systemd/system/qrunlet.service

```
[Unit]
Description=Q.Runner QRunlet Server
Documentation=
After=porter.service
Requires=porter.service

[Service]
WorkingDirectory=/var/lib/qrunlet
EnvironmentFile=/etc/qrunner/config
EnvironmentFile=/etc/qrunner/qrunlet
ExecStart=/usr/bin/qrunlet \
    $KUBE_LOGTOSTDERR \
    $KUBE_LOG_LEVEL \
    $KUBELET_KUBECONFIG \
    $KUBELET_ADDRESS \
    $KUBELET_PORT \
    $KUBELET_HOSTNAME \
    $KUBE_ALLOW_PRIV \
    $KUBELET_ARGS
Restart=on-failure
KillMode=process

[Install]
WantedBy=multi-user.target
```

Code Block 4 qrunlet.service

Обновить информацию о сервисах командой

```
sudo systemctl daemon-reload
```

Запустить apiserver:

```
sudo systemctl enable qrun-apiserver
sudo systemctl start qrun-apiserver
```

Проверить статус сервиса

```
systemctl status qrun-apiserver
```

Результат ответа в случае успешного запуска:

```
admin@digitaldemodb:~$ systemctl status qrun-apiserver
● qrun-apiserver.service - Q.Runner API Server
   Loaded: loaded (/etc/systemd/system/qrun-apiserver.service; enabled; vendor preset: enabled)
   Active: active (running) since Fri 2019-06-28 14:46:06 MSK; 1min 32s ago
     Main PID: 21868 (qrun-apiserver)
        Tasks: 14 (limit: 9486)
      CGroup: /system.slice/qrun-apiserver.service
             └─21868 /usr/bin/qrun-apiserver --logtostderr=true --v=0 --etcd-servers=http://127.0.0.1:2379
```

Создать файл настроек /etc/qrunner/qrunlet:

```

###
# Q.Runner qrunlet (minion) config

# The address for the info server to serve on (set to 0.0.0.0 or "" for all interfaces)
# KUBELET_ADDRESS="--address=127.0.0.1"
KUBELET_ADDRESS="--address=0.0.0.0"

# The port for the info server to serve on
# KUBELET_PORT="--port=10250"

# You may leave this blank to use the actual hostname
# KUBELET_HOSTNAME="--hostname-override=127.0.0.1"

# Edit the qrunlet.kubeconfig to have correct cluster server address
KUBELET_KUBECONFIG="--kubeconfig /etc/qrunner/qrunlet.kubeconfig"

# Add your own!
# CGroup Driver in Q.Porter and QRunlet must be identical
KUBELET_ARGS="--cgroup-driver=cgroupfs --fail-swap-on=false"

```

Code Block 5 qrunlet

Создать файл настроек /etc/qrunner/qrunlet.kubeconfig:

```

apiVersion: v1
clusters:
- cluster:
  server: http://127.0.0.1:8080
  name: qrunner-systemd
contexts:
- context:
  cluster: qrunner-systemd
  user: system:node:<HOSTNAME>
  name: system:node:<HOSTNAME>@qrunner-systemd
current-context: system:node:<HOSTNAME>@qrunner-systemd
kind: Config
preferences: {}
users:
- name: system:node:<HOSTNAME>

```

Code Block 6 qrunlet.kubeconfig

Запустить qrunlet:

```

sudo systemctl enable qrunlet
sudo systemctl start qrunlet

```

Проверить статус сервиса

```

systemctl status qrunlet

```

Результат ответа в случае успешного запуска:

```
admin@digitaldemodb:~$ systemctl status qrunlet
● qrunlet.service - Q.Runner QRunlet Server
   Loaded: loaded (/etc/systemd/system/qrunlet.service; enabled; vendor preset: enabled)
   Active: active (running) since Fri 2019-06-28 15:24:49 MSK; 52min ago
 Main PID: 23842 (qrunlet)
    Tasks: 18 (limit: 9486)
   CGroup: /system.slice/qrunlet.service
           └─23842 /usr/bin/qrunlet --logtostderr=true --v=0 --kubeconfig /etc/qrunner/qrunlet.ku
```