# Работа с deployment (в 2-х частях)

Перед началом работ убедитесь, что у вас установлены:

- docker
- kubectl
- k3d

Установка `docker`:

```
$ sudo apt-get install \
   ca-certificates \
   curl \
   gnupg \
   lsb-release

$ sudo mkdir -p /etc/apt/keyrings

$ curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo
gpg --dearmor -o /etc/apt/keyrings/docker.gpg

$ echo \
  "deb [arch=$(dpkg --print-architecture)
signed-by=/etc/apt/keyrings/docker.gpg]
https://download.docker.com/linux/ubuntu \
  $(lsb_release -cs) stable" | sudo tee
/etc/apt/sources.list.d/docker.list > /dev/null

$ sudo apt-get update

$ sudo apt-get install docker-ce docker-ce-cli containerd.io
```

Установка `kubectl` с валидацией корректности установки:

```
$ curl -LO "https://dl.k8s.io/release/$(curl -L -s
https://dl.k8s.io/release/stable.txt)/bin/linux/amd64/kubectl"

$ curl -LO "https://dl.k8s.io/$(curl -L -s
https://dl.k8s.io/release/stable.txt)/bin/linux/amd64/kubectl.sha2
56"

$ echo "$(cat kubectl.sha256)  kubectl" | sha256sum --check
```

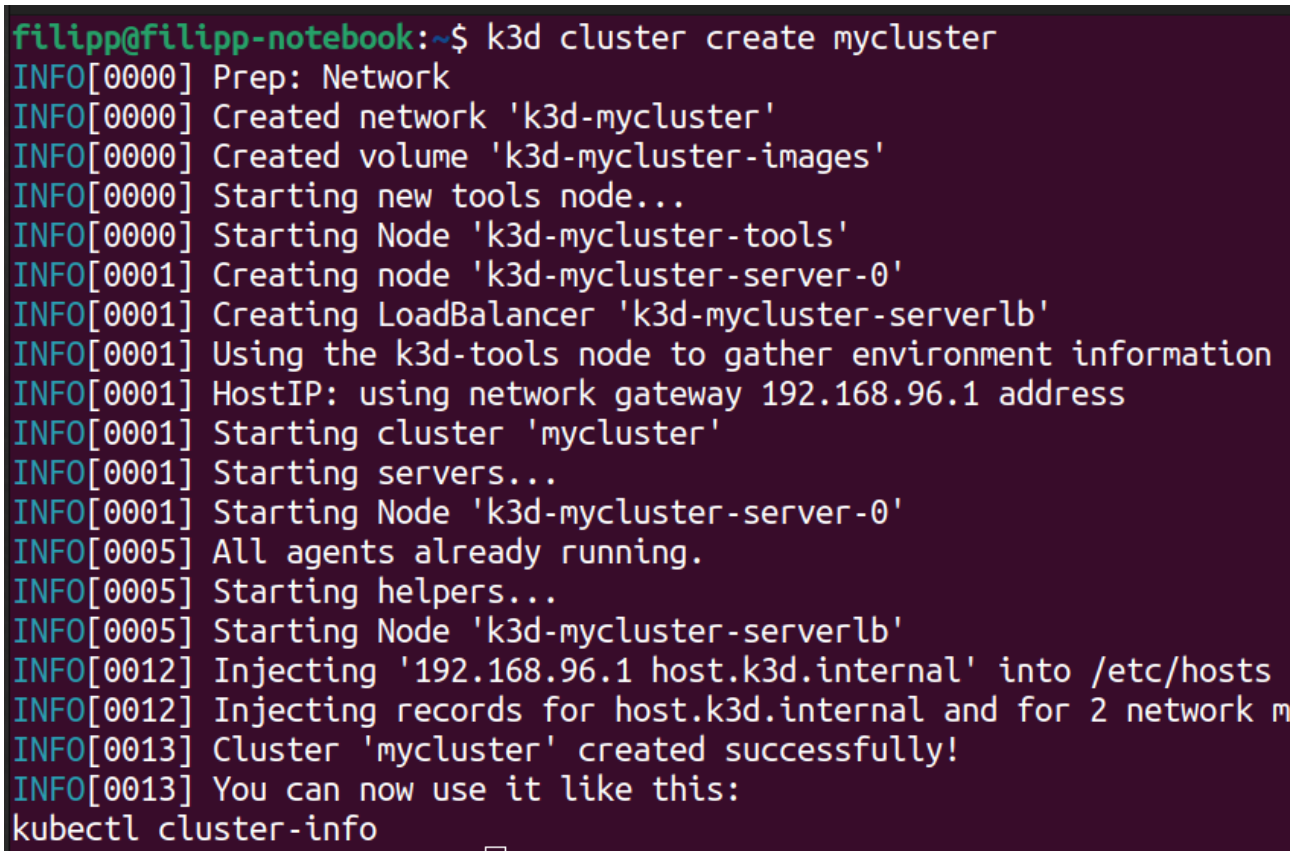
Установка `k3d`:

```
$ wget -q -O -
https://raw.githubusercontent.com/k3d-io/k3d/main/install.sh |
bash
```

## Часть 1. Практическая работа по созданию деплоймента

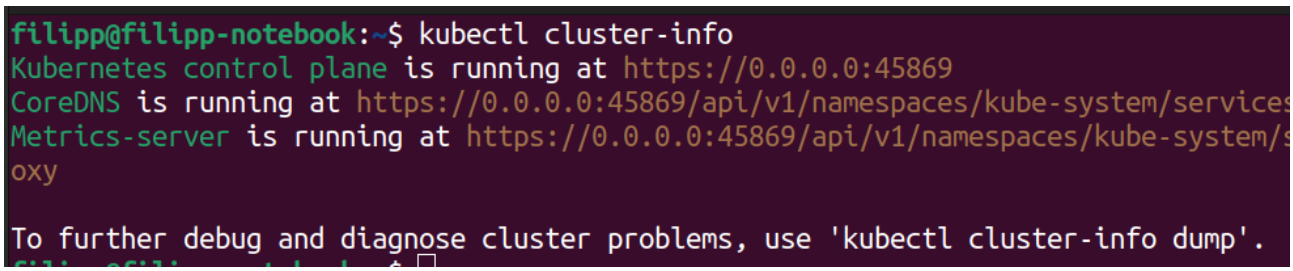
Для начала запустим кластер `k3d` с именем "`mycluster`":

```
$ k3d cluster create mycluster
```




Убедимся, что кластер установлен успешно:

```
$ kubectl cluster-info
```




--

Создадим неймспейс "`mobsf`" для работы с будущим деплойментом:

```
$ kubectl create ns mobsf
```

```
filipp@filipp-notebook:~$ kubectl create ns mobsf
namespace/mobsf created
```

Создадим деплоймент "mobsf" в ранее подготовленном одноименном неймспейсе, на основе docker-образа mobsf версии v3.1.1:

```
$ kubectl create deployment mobsf --image=opensecurity/mobile-security-framework-mobsf:v3.1.1 -n mobsf
```

```
filipp@filipp-notebook:~$ kubectl create deployment mobsf --image=opensecurity/mobile-security-framework-mobsf:v3.1.1 -n mobsf
deployment.apps/mobsf created
```

Создадим службу для деплоймента mobsf в том же неймспейсе:

```
$ kubectl expose deployment mobsf --port=8000 --type=NodePort -n mobsf
```

```
filipp@filipp-notebook:~$ kubectl expose deployment mobsf --port=8000 --type=NodePort -n mobsf
service/mobsf exposed
```

--

Выполним команду для отображения EXTERNAL-IP нашего системного ингресс-контроллера traefik (службы с типом loadbalancer):

```
$ kubectl get svc -n kube-system
```

```
filipp@filipp-notebook:~$ kubectl get svc -n kube-system
NAME             TYPE           CLUSTER-IP      EXTERNAL-IP    PORT(S)                      AGE
kube-dns         ClusterIP      10.43.0.10      <none>         53/UDP,53/TCP,9153/TCP       5m6s
metrics-server   ClusterIP      10.43.153.45    <none>         443/TCP                      5m4s
traefik          LoadBalancer   10.43.70.122    192.168.96.3   80:30033/TCP,443:30280/TCP   3m41s
```

Выполним команду просмотра службы mobsf в одноименном неймспейсе и запомним выделенный данной службе порт (порт может принимать значение из определенного диапазона - пятизначный порт, начинающийся с цифры "3"):

```
$ kubectl get svc -n mobsf
```

```
filipp@filipp-notebook:~$ kubectl get svc -n mobsf
NAME    TYPE       CLUSTER-IP     EXTERNAL-IP   PORT(S)          AGE
mobsf   NodePort   10.43.105.36   <none>        8000:30052/TCP   3m25s
```

Убедимся, что pod в неймспейсе `mobsf` запущен и находится в статусе "running":

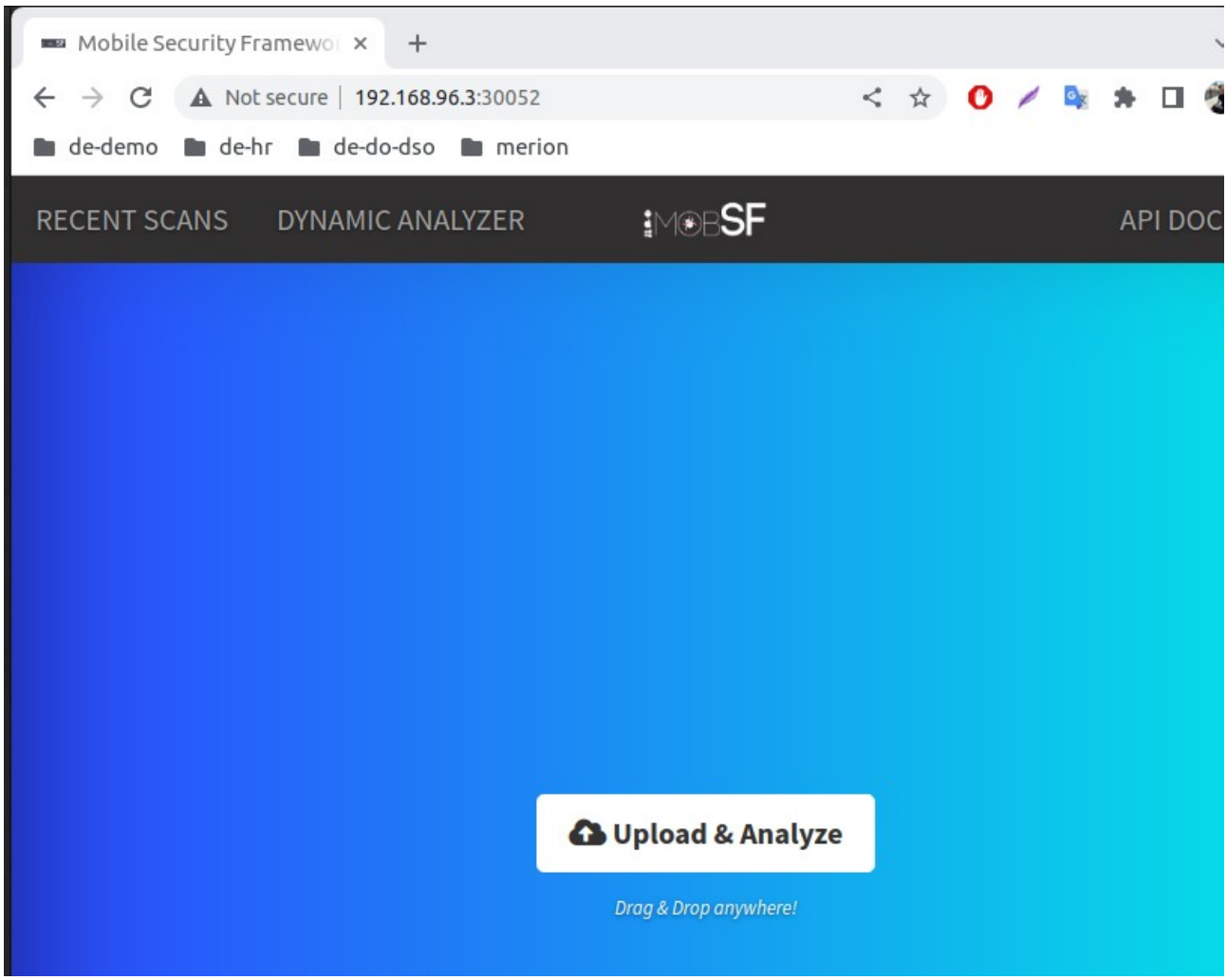
```
$ kubectl get pods -n mobsf
```

```
filipp@filipp-notebook:~$ kubectl get pods -n mobsf
NAME                     READY   STATUS    RESTARTS   AGE
mobsf-5db69649fc-lhdrn   1/1     Running   0          6m14s
```

--

Перейдем в браузер и обратимся к приложению mobsf, совместив `external-ip` службы traefik и порт службы `mobsf`:

в адресной строке браузера: `http://<external-ip>:<mobsf-service-port>/`

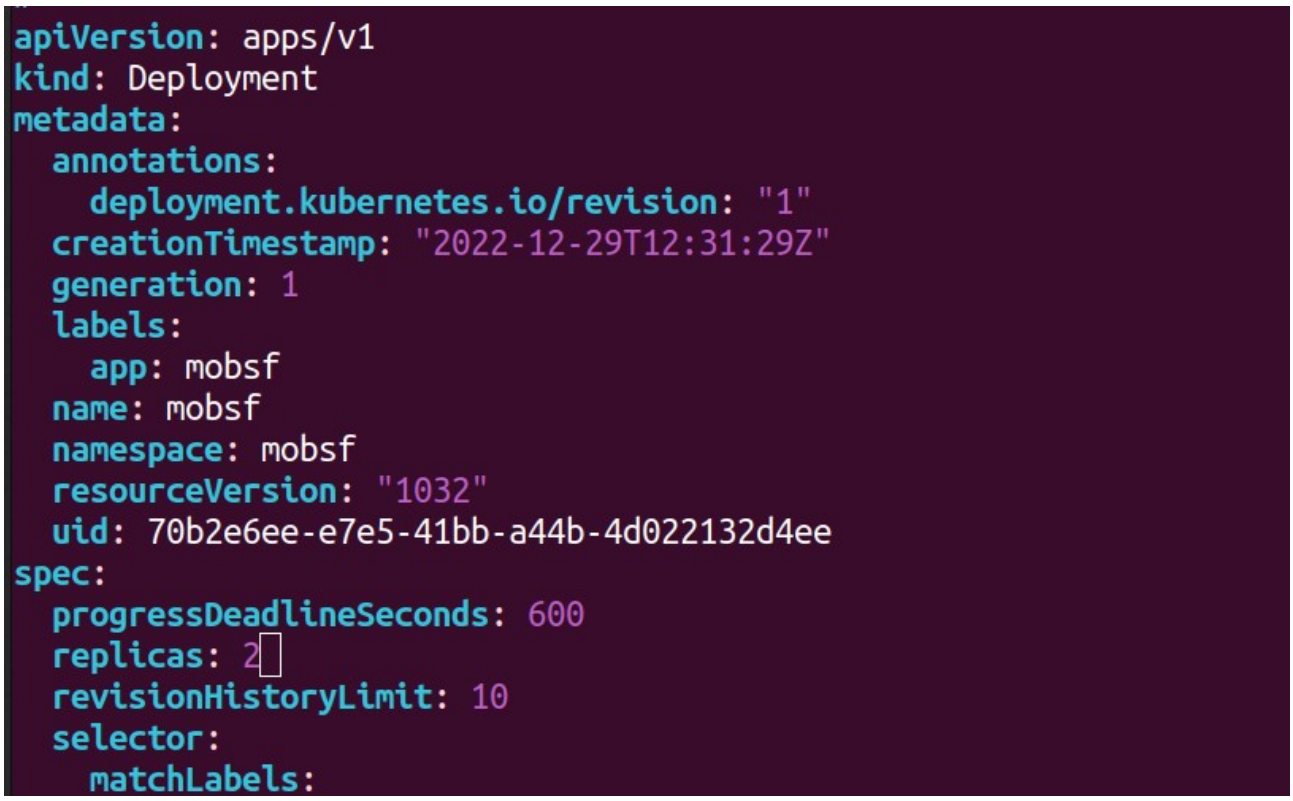



## Часть 2. Практическая работа по обновлению, откату и репликации деплоймента

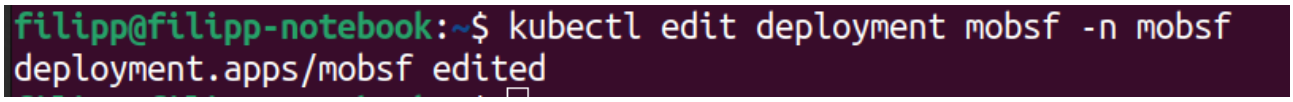
Изменим вручную количество репликаций деплоймента `mobsf`:

```
$ kubectl edit deployment mobsf -n mobsf
```

В открывшемся окне редактирования изменим количество `relicas` с `1` до `2`:

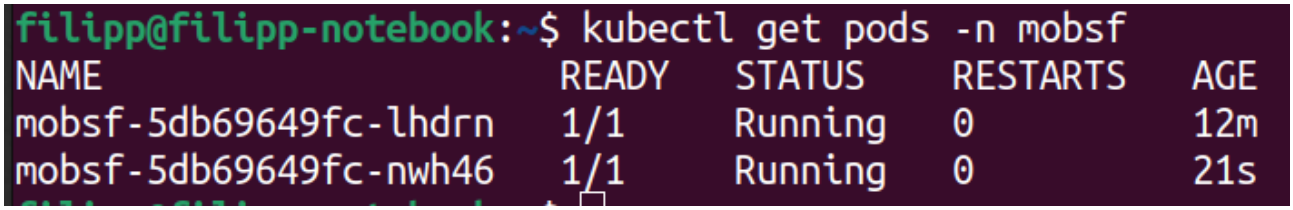



Сохраняем изменения




Выполним команду для просмотра количества pods в неймспейсе `mobsf`:

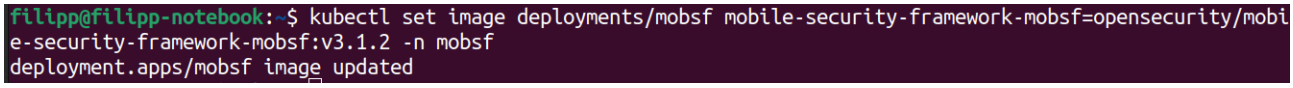
```
$ kubectl get pods -n mobsf
```




--

Произведем попытку обновления деплоймента `mobsf`, указав несуществующую версию docker-image:

```
$ kubectl set image deployments/mobsf mobile-security-framework-
mobsf=opensecurity/mobile-security-framework-mobsf:v3.1.2 -n mobsf
```



Выполним команду просмотра статуса текущего обновления деплоймента:

```
$ kubectl rollout status deployment/mobsf -n mobsf
```

```
filipp@filipp-notebook:~$ kubectl rollout status deployment/mobsf -n mobsf
Waiting for deployment "mobsf" rollout to finish: 1 out of 2 new replicas have been updated...
```

Выполним команду просмотра текущих pods в неймспейсе `mobsf`:

```
$ kubectl get pods -n mobsf
```

```
filipp@filipp-notebook:~$ kubectl get pods -n mobsf
NAME                     READY   STATUS         RESTARTS   AGE
mobsf-5db69649fc-lhdrn   1/1     Running        0          15m
mobsf-5db69649fc-nwh46   1/1     Running        0          3m24s
mobsf-84d89d9ff6-8gk2c   0/1     ErrImagePull   0          67s
```

Обратите внимание на идентификаторы pods: 2 pod в статусе running и 1 pod в статусе ErrImagePull

Выполним команду просмотра текущих replicasets (`rs`) в неймспейсе `mosf`:

```
$ kubectl get rs -n mobsf
```

```
filipp@filipp-notebook:~$ kubectl get rs -n mobsf
NAME               DESIRED   CURRENT   READY   AGE
mobsf-5db69649fc   2         2         2       17m
mobsf-84d89d9ff6   1         1         0       2m46s
```

идентификатор replicaset с desired и current равными 2 сооветствуют двум pods в статусе running, а диентификатор replicaset с desired и current равными 1 соответствуют одному pod в статусе ErrImagePull

Выполним rollback (`rollout undo`) текущего обновления деплоймента в неймспейсе `mobsf`:

```
$ kubectl rollout undo deployment/mobsf -n mobsf
```

```
filipp@filipp-notebook:~$ kubectl rollout undo deployment/mobsf -n mobsf
deployment.apps/mobsf rolled back
```

Выполним команду просмотра текущих `pods` в неймспейсе `mobsf`:

```
$ kubectl get pods -n mobsf
```

```
filipp@filipp-notebook:~$ kubectl get pods -n mobsf
NAME                      READY   STATUS    RESTARTS   AGE
mobsf-5db69649fc-lhdrn    1/1     Running   0          20m
mobsf-5db69649fc-nwh46    1/1     Running   0          8m1s
```

pod в статусе ErrImagePull больше не присутствует в выводе команды. На данный момент заупещено 2 pods со статусом running (рабочей версии v3.1.1)

--

Выполним еще одну попытку обновления деплоймента в неймспейсе `mobsf`, на этот раз обновим версию docker-image на существующую `v3.5.0`:

```
$ kubectl set image deployments/mobsf mobile-security-framework-mobsf=opensecurity/mobile-security-framework-mobsf:v3.5.0 -n mobsf
```

```
filipp@filipp-notebook:~$ kubectl set image deployments/mobsf mobile-security-framework-mobsf=opensecurity/mobile-security-framework-mobsf:v3.5.0 -n mobsf
deployment.apps/mobsf image updated
```

Выполним команду просмотра текущих `pods` в неймспейсе `mobsf` с флагом "`-w`" для просмотра всей цепочки обновления pods:

```
$ kubectl get pods -n mobsf -w
```

Обратите внимание на саму процедуру последовательного обновления: сперва добавляется и стартует новый pod с версией v.3.5.0 (его идентификатор отличается от идентификатора pods с версией v3.1.1), затем последовтаельно терминируется 1 из 2 pods с версией v3.1.1, только затем стартует и запускается новый pod с версией v3.5.0 (его идентификатор частично совпадает с уже работающим pod версии v3.5.0), после этого терминируется оставшийся 2 из 2 pod с версией v3.1.1)

```
filipp@filipp-notebook:~$ kubectl get pods -n mobsf -w
NAME                     READY   STATUS              RESTARTS   AGE
mobsf-5db69649fc-lhdrn   1/1     Running             0          23m
mobsf-5db69649fc-nwh46   1/1     Running             0          11m
mobsf-dcd84f7d6-d46rw    0/1     ContainerCreating   0          87s
mobsf-dcd84f7d6-d46rw    1/1     Running             0          6m35s
mobsf-5db69649fc-nwh46   1/1     Terminating         0          16m
mobsf-dcd84f7d6-b9wm7    0/1     Pending             0          0s
mobsf-dcd84f7d6-b9wm7    0/1     Pending             0          0s
mobsf-dcd84f7d6-b9wm7    0/1     ContainerCreating   0          0s
mobsf-dcd84f7d6-b9wm7    1/1     Running             0          1s
mobsf-5db69649fc-lhdrn   1/1     Terminating         0          29m
mobsf-5db69649fc-nwh46   0/1     Terminating         0          16m
mobsf-5db69649fc-lhdrn   0/1     Terminating         0          29m
mobsf-5db69649fc-lhdrn   0/1     Terminating         0          29m
mobsf-5db69649fc-lhdrn   0/1     Terminating         0          29m
mobsf-5db69649fc-nwh46   0/1     Terminating         0          17m
mobsf-5db69649fc-nwh46   0/1     Terminating         0          17m
```

Выполним команду для просмотра истории обновлений деплоймента (`revisions`):

```
$ kubectl rollout history deployment/mobsf -n mobsf
```

```
filipp@filipp-notebook:~$ kubectl rollout history deployment/mobsf -n mobsf
deployment.apps/mobsf
REVISION  CHANGE-CAUSE
2         <none>
3         <none>
4         <none>
```

Выполним команды для детального просмотра всех имеющихся ревизий:

```
$ kubectl rollout history deployment/mobsf -n mobsf --revision 2
$ kubectl rollout history deployment/mobsf -n mobsf --revision 3
$ kubectl rollout history deployment/mobsf -n mobsf --revision 4
```

```
filipp@filipp-notebook:~$ kubectl rollout history deployment/mobsf -n mobsf --revision 2
deployment.apps/mobsf with revision #2
Pod Template:
  Labels:       app=mobsf
        pod-template-hash=84d89d9ff6
  Containers:
   mobile-security-framework-mobsf:
    Image:      opensecurity/mobile-security-framework-mobsf:v3.1.2
    Port:       <none>
    Host Port:  <none>
    Environment:        <none>
    Mounts:     <none>
  Volumes:      <none>

filipp@filipp-notebook:~$ kubectl rollout history deployment/mobsf -n mobsf --revision 3
deployment.apps/mobsf with revision #3
Pod Template:
  Labels:       app=mobsf
        pod-template-hash=5db69649fc
  Containers:
   mobile-security-framework-mobsf:
    Image:      opensecurity/mobile-security-framework-mobsf:v3.1.1
    Port:       <none>
    Host Port:  <none>
    Environment:        <none>
    Mounts:     <none>
  Volumes:      <none>

filipp@filipp-notebook:~$ kubectl rollout history deployment/mobsf -n mobsf --revision 4
deployment.apps/mobsf with revision #4
Pod Template:
  Labels:       app=mobsf
        pod-template-hash=dcd84f7d6
  Containers:
   mobile-security-framework-mobsf:
    Image:      opensecurity/mobile-security-framework-mobsf:v3.5.0
    Port:       <none>
    Host Port:  <none>
    Environment:        <none>
    Mounts:     <none>
  Volumes:      <none>
```

Ревизия #2 - неуспешная попытка rolling-update на несуществующую версию v3.1.2
Ревизия #3 - rollback обратно на версию v3.1.1
Ревизия #4 - успешный rolling-update до версии v3.5.0

--

Выполним скалирование репликаций приложения до 3-х, но не в ручном режиме изменения конфигурации деплоймента, а командой:

```
$ kubectl scale deployment/mobsf --replicas=3 -n mobsf
```

```
filipp@filipp-notebook:~$ kubectl scale deployment/mobsf --replicas=3 -n mobsf
deployment.apps/mobsf scaled
```

Выполним команду просмотра текущих `pods` в неймспейсе `mobsf`:

```
$ kubectl get pods -n mobsf
```

```
filipp@filipp-notebook:~$ kubectl get pods -n mobsf
NAME                     READY   STATUS    RESTARTS   AGE
mobsf-dcd84f7d6-d46rw    1/1     Running   0          18m
mobsf-dcd84f7d6-b9wm7    1/1     Running   0          11m
mobsf-dcd84f7d6-pk5b8    1/1     Running   0          46s
```

Деплоймент успешно проскалирован до 3-х репликаций

--

Совершим `rollback` деплоймента до специфичной `revision` (возьмем ревизию #3 - версия приложения v3.1.1):

```
$ kubectl rollout undo deployment/mobsf -n mobsf --to-revision=3
```

```
filipp@filipp-notebook:~$ kubectl rollout undo deployment/mobsf -n mobsf --to-r
evision=3
deployment.apps/mobsf rolled back
```

rollback до конкретной версии ревизии успешно выполнен

Выполним команду просмотра списка текущих ревизий деплоймента `mobsf`:

```
$ kubectl rollout history deployment/mobsf -n mobsf
```

```
filipp@filipp-notebook:~$ kubectl rollout history deployment/mobsf -n mobsf
deployment.apps/mobsf
REVISION  CHANGE-CAUSE
2         <none>
4         <none>
5         <none>
```

Нумерация ревизий изменилась, выполним команды детального просмотра всех имеющихся в неймспейсе `mobsf` ревизий:

```
$ kubectl rollout history deployment/mobsf -n mobsf --revision 2
$ kubectl rollout history deployment/mobsf -n mobsf --revision 4
$ kubectl rollout history deployment/mobsf -n mobsf --revision 5
```

```
filipp@filipp-notebook:~$ kubectl rollout history deployment/mobsf -n mobsf --revision 2
deployment.apps/mobsf with revision #2
Pod Template:
  Labels:       app=mobsf
        pod-template-hash=84d89d9ff6
  Containers:
   mobile-security-framework-mobsf:
    Image:      opensecurity/mobile-security-framework-mobsf:v3.1.2
    Port:       <none>
    Host Port:  <none>
    Environment:        <none>
    Mounts:     <none>
  Volumes:      <none>

filipp@filipp-notebook:~$ kubectl rollout history deployment/mobsf -n mobsf --revision 4
deployment.apps/mobsf with revision #4
Pod Template:
  Labels:       app=mobsf
        pod-template-hash=dcd84f7d6
  Containers:
   mobile-security-framework-mobsf:
    Image:      opensecurity/mobile-security-framework-mobsf:v3.5.0
    Port:       <none>
    Host Port:  <none>
    Environment:        <none>
    Mounts:     <none>
  Volumes:      <none>

filipp@filipp-notebook:~$ kubectl rollout history deployment/mobsf -n mobsf --revision 5
deployment.apps/mobsf with revision #5
Pod Template:
  Labels:       app=mobsf
        pod-template-hash=5db69649fc
  Containers:
   mobile-security-framework-mobsf:
    Image:      opensecurity/mobile-security-framework-mobsf:v3.1.1
    Port:       <none>
    Host Port:  <none>
    Environment:        <none>
    Mounts:     <none>
  Volumes:      <none>
```

Ревизия #2 - неуспешная попытка rolling-update на несуществующую версию v3.1.2 (как и было до rollback на ревизию #3)
Ревизия #4 - успешный rolling-update до версии v3.5.0 (как и было до rollback на ревизию #3)
Ревизия #5 - успешный rollback до версии v3.1.1 (до бывшей ревизии #3 - после успешного rollback это теперь ревизия #5)

--