

- Домашняя работа № 1
  - По предмету Контейнеризация
    - Ростислав Ромашин - группа 4992
    - Задания
    - Решение
      - 1. chroot
      - 2. ip namespace
      - 3. unshare
        - 3.1. PID
        - 3.2 NET
        - 3.3 USER
        - 3.4. MOUNT
        - 3.5. IPC
        - 3.6. UTS
    - unshare и несколько неймспейсов

## Домашняя работа № 1

---

## По предмету Контейнеризация

---

### Ростислав Ромашин - группа 4992

### Задания

1. Сделать chroot для /bin/bash и перенести в новый корень программу ls - Хорошо.
2. Повторить последовательность команд с ip как на семинаре. - Отлично
3. Повторить последовательность команд с unshare как на семинаре. - Отлично

Формат сдачи ДЗ: предоставить доказательства выполнения задания посредством файла PDF или ссылки на google-документ с правами на комментирование/редактирование. Результатом работы будет: текст объяснения, логи выполнения, история команд и скриншоты (важно придерживаться такой последовательности). В названии работы должны быть указаны ФИ, номер группы и номер урока

# Решение

## 1. chroot

Воспользовался в своем решении легковесным `busybox`, который занимает около 700КБ на диске.

```
apt install busybox
```

**BusyBox** - это исполняемый файл командной строки, включающий большой набор системных утилит (`ls`, `rm`, `md`), разработанный для использования во встраиваемых системах и ограниченных окружениях, а так же в образах *Docker*.

Благодаря тому что *busybox* собран методом статической линковки, ему не нужны сторонние библиотеки, т.к. они находятся внутри самого бинарного файла.

```
Rostislav:~$ pwd
/home/Rostislav
Rostislav:~$ mkdir -p gb/bin
Rostislav:~$ cp /usr/bin/busybox gb/bin
Rostislav:~$ gb/bin/busybox --install gb/bin/
Rostislav:~$ ls gb/bin
['
[['
acpid      cmp        egrep      hostid     ln         more       rdate      sleep      tftp       uptime
adjtimex   cp         env        hostname  loadfont  mount      readlink   sort       time       usleep
ar         cpio       expand     httpd     loadkmap  mv         reboot     ssl_client timeout    uudecode
arch       crond      expr       hwclock   logger    mv         reboot     start-stop-daemon top        uuencode
arp        cronstab  factor     i2cdetect login      nameif     renice     stat       touch      vconfig
brctl      crontab   falldate   i2cdump   logname   nc         reset      static-sh  tr         vi
brp        cttyhack  false      i2cget    logread   netstat    resume     strings    traceroute w
arping     cut       fatattr    i2cset    losetup   nl         rev        stty       traceroute6 watch
ash        date      fdisk      id         ls         nologin    rm         su         true       watchdog
awk        dc         fgrep      ifconfig  lsmode    nproc      rmdir      sulogin    truncate   wc
basename  dd         find       ifdown    lsscsi    nsenter    rmmod      svc        tty        wget
bc         deallocvt fold       ifup      lzcat     nslookup   route      svok       tuncctl    which
blkdiscard depmod    free       init       lzma       nuke       rpm        swapoff    ubirename  who
blockdev  devmem   freeramdisk insmod     lzop       od         rpm2cpio   swapon     udhcpc     whoami
brctl     df         fsfreeze   ionice    md5sum    openvt     run-init   switch_root udhcpd     xargs
bunzip2   diff      fstrim     ip        mdev      partprobe  run-parts  sync       uevent     xxd
busybox   dirname   ftpget     ipcalc    microcom  passwd     sed        sysctl     umount     xz
bzip2     dmesg     ftpput     ipneigh   mkdir     patch      seq        syslogd    uname      xzcat
cal       dos2unix  getopt     kill       mkdosfs    patch      setkeycodes tac         uncompress yes
cat       dpkg      getty      killall    mke2fs     pidof      setpriv    tail       unexpand   zcat
chgrp     dpkg-deb grep       klogd     mkfifo     ping       setsid     tar        uniq
chmod     du        gunzip     less       mkpasswd  ping6      sh         taskset    unix2dos   unlink
chown     dumpkmap gzip       link       mkswap     poweroff   sha256sum  tee        unlnma     unshare
chpasswd  dumpleases halt       linux32    mktemp     printf     sha512sum  telnet     telnetd   unxz
chroot    echo      head       linux64    modinfo    ps         shred      telnetd    unz
chvt      ed        hexdump    linuxrc    modprobe   pwd        shuf       test       unzip
Rostislav:~$ sudo chroot gb /bin/sh

BusyBox v1.30.1 (Ubuntu 1:1.30.1-4ubuntu6.4) built-in shell (ash)
Enter 'help' for a list of built-in commands.

/ # ls
bin
/ # █
```

Как видно из картинки процесс настройки *chroot* с полноценным набором команд занимает гораздо меньше места нежели процесс с поиском и копированием библиотек для каждой утилиты и команд *linux*.

Только в качестве командного интерпретатора выбирается встроенный в *busybox* классический **sh**.

Настройка *busybox* выполняется командой

```
busybox --install bin
```

Создаются символические ссылки с именем каждой утилиты на сам *busybox*, т.к он в качестве аргумента принимает имя команды, например:

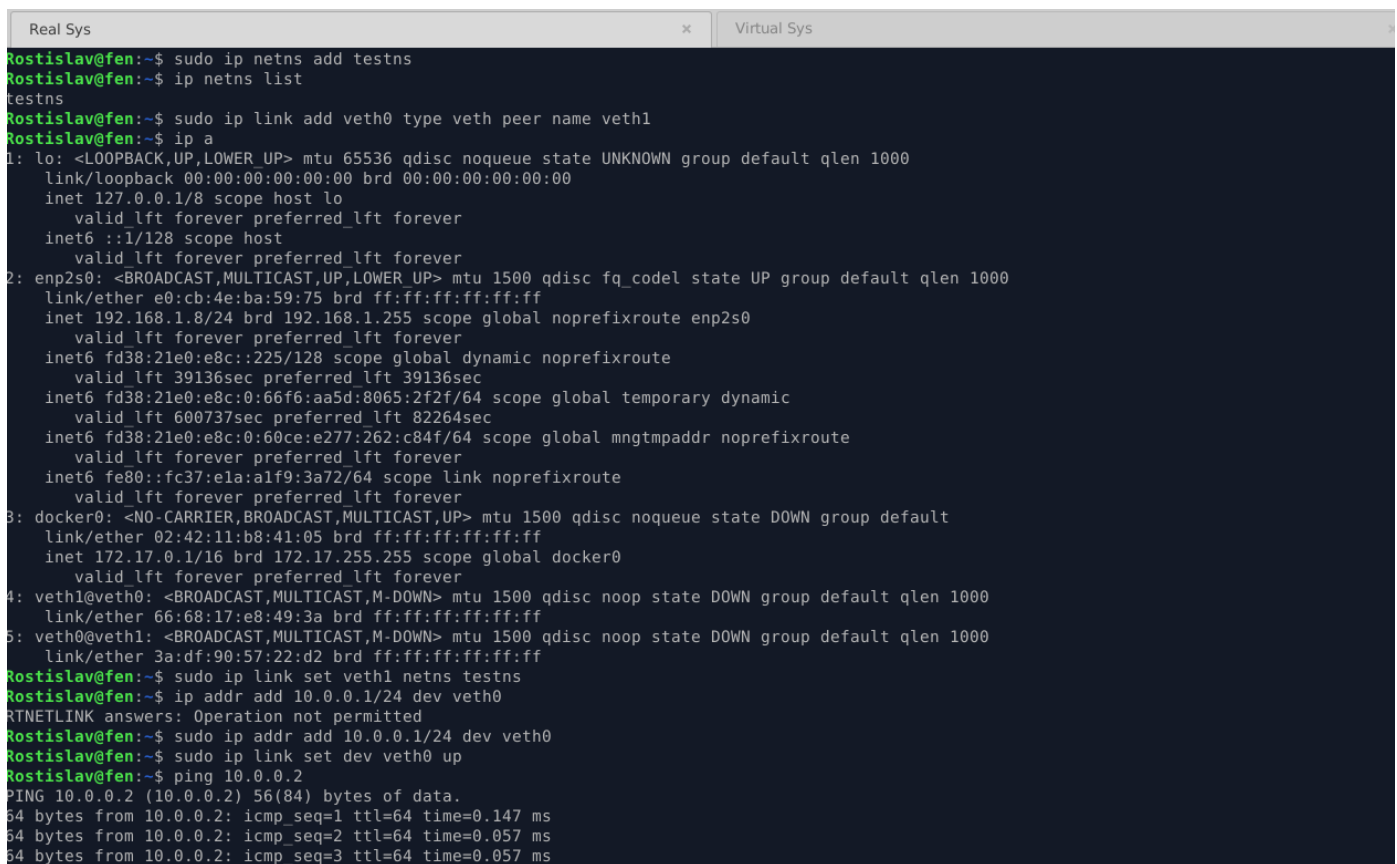
```
busybox ls
```

верхняя команда аналогична команде *ls* и т.д.

## 2. ip namespace

Для проведения эксперимента созданы две закладки в терминале:

- Real sys - Реальная система
- Virtual sys - Виртуальная система



```
Real Sys x Virtual Sys
Rostislav@fen:~$ sudo ip netns add testns
Rostislav@fen:~$ ip netns list
testns
Rostislav@fen:~$ sudo ip link add veth0 type veth peer name veth1
Rostislav@fen:~$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
        valid_lft forever preferred_lft forever
2: enp2s0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether e0:cb:4e:ba:59:75 brd ff:ff:ff:ff:ff:ff
    inet 192.168.1.8/24 brd 192.168.1.255 scope global noprefixroute enp2s0
        valid_lft forever preferred_lft forever
    inet6 fd38:21e0:e8c::225/128 scope global dynamic noprefixroute
        valid_lft 39136sec preferred_lft 39136sec
    inet6 fd38:21e0:e8c:0:66f6:aa5d:8065:2f2f/64 scope global temporary dynamic
        valid_lft 600737sec preferred_lft 82264sec
    inet6 fd38:21e0:e8c:0:60ce:e277:262:c84f/64 scope global mngtmpaddr noprefixroute
        valid_lft forever preferred_lft forever
    inet6 fe80::fc37:ela:alf9:3a72/64 scope link noprefixroute
        valid_lft forever preferred_lft forever
3: docker0: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc noqueue state DOWN group default
    link/ether 02:42:11:b8:41:05 brd ff:ff:ff:ff:ff:ff
    inet 172.17.0.1/16 brd 172.17.255.255 scope global docker0
        valid_lft forever preferred_lft forever
4: veth1@veth0: <BROADCAST,MULTICAST,M-DOWN> mtu 1500 qdisc noop state DOWN group default qlen 1000
    link/ether 66:68:17:e8:49:3a brd ff:ff:ff:ff:ff:ff
5: veth0@veth1: <BROADCAST,MULTICAST,M-DOWN> mtu 1500 qdisc noop state DOWN group default qlen 1000
    link/ether 3a:df:90:57:22:d2 brd ff:ff:ff:ff:ff:ff
Rostislav@fen:~$ sudo ip link set veth1 netns testns
Rostislav@fen:~$ ip addr add 10.0.0.1/24 dev veth0
RTNETLINK answers: Operation not permitted
Rostislav@fen:~$ sudo ip addr add 10.0.0.1/24 dev veth0
Rostislav@fen:~$ sudo ip link set dev veth0 up
Rostislav@fen:~$ ping 10.0.0.2
PING 10.0.0.2 (10.0.0.2) 56(84) bytes of data.
64 bytes from 10.0.0.2: icmp_seq=1 ttl=64 time=0.147 ms
64 bytes from 10.0.0.2: icmp_seq=2 ttl=64 time=0.057 ms
64 bytes from 10.0.0.2: icmp_seq=3 ttl=64 time=0.057 ms
```

В реальной системе с помощью утилиты *ip* создаем namespace *testns*

```
sudo ip netns add testns
```

убеждаемся что неймспейс создан

```
ip netns list
```

создаем два виртуальных сетевых интерфейса

```
sudo ip link add veth0 type veth peer name veth1
```

убеждаемся, что интерфейсы созданы

```
ip a
```

присвоим первому сетевому интерфейсу *veth0* айпи-адрес

```
sudo ip addr add 10.0.0.1/24 dev veth0
```

переведем сетевой интерфейс в активное состояние

```
sudo ip link set dev veth0 up
```

На вкладке виртуальной системы создадим папку *docker* и перейдем в нее

```
Real Sys Virtual Sys
Rostislav@fen:~$ mkdir docker
Rostislav@fen:~$ cd docker
Rostislav@fen:~/docker$ sudo ip netns exec testns bash
[sudo] пароль для Rostislav:
root@fen:/home/Rostislav/docker# ip a
1: lo: <LOOPBACK> mtu 65536 qdisc noop state DOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
root@fen:/home/Rostislav/docker# ip addr add 10.0.0.2/24 dev veth1
root@fen:/home/Rostislav/docker# ip link set dev veth1 up
root@fen:/home/Rostislav/docker# ip a
1: lo: <LOOPBACK> mtu 65536 qdisc noop state DOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
2: veth1@if5: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP group default qlen 1000
    link/ether 66:68:17:e8:49:3a brd ff:ff:ff:ff:ff:ff link-netnsid 0
    inet 10.0.0.2/24 scope global veth1
        valid_lft forever preferred_lft forever
    inet6 fe80::6468:17ff:fee8:493a/64 scope link
        valid_lft forever preferred_lft forever
root@fen:/home/Rostislav/docker# ip link set dev veth1 up
root@fen:/home/Rostislav/docker# ping 10.0.0.1
PING 10.0.0.1 (10.0.0.1) 56(84) bytes of data:
64 bytes from 10.0.0.1: icmp_seq=1 ttl=64 time=0.088 ms
64 bytes from 10.0.0.1: icmp_seq=2 ttl=64 time=0.061 ms
64 bytes from 10.0.0.1: icmp_seq=3 ttl=64 time=0.059 ms
64 bytes from 10.0.0.1: icmp_seq=4 ttl=64 time=0.060 ms
64 bytes from 10.0.0.1: icmp_seq=5 ttl=64 time=0.060 ms
64 bytes from 10.0.0.1: icmp_seq=6 ttl=64 time=0.059 ms
64 bytes from 10.0.0.1: icmp_seq=7 ttl=64 time=0.059 ms
64 bytes from 10.0.0.1: icmp_seq=8 ttl=64 time=0.059 ms
^C
--- 10.0.0.1 ping statistics ---
8 packets transmitted, 8 received, 0% packet loss, time 7144ms
rtt min/avg/max/mdev = 0.059/0.063/0.088/0.009 ms
root@fen:/home/Rostislav/docker# ping ya.ru
ping: ya.ru: Временный сбой в разрешении имен
root@fen:/home/Rostislav/docker# ping 8.8.8.8
ping: connect: Сеть недоступна
root@fen:/home/Rostislav/docker#
```

Применим нетспейс *testns* к текущему процессу *bash*

```
sudo ip link set veth1 netns testns
```

как видим кроме "обратной петли" ничего в виртуальной системе не осталось

```
ip a
```

после того как на реальной системе добавили виртуальные интерфейсы, они уже не входят в ранее примененный нетспейс. Второй интерфейс добавим в виртуальную систему прописав ай-пи

```
ip addr add 10.0.0.2/24 dev veth1
```

поднимем сетевой интерфейс для виртуальной системы

```
ip link set dev veth1 up
```

Адреса реальной и виртуальной систем пингуются в обе стороны.

На виртуальной системе, нет доступа к любым другим сетевым ресурсам, что видно на примере

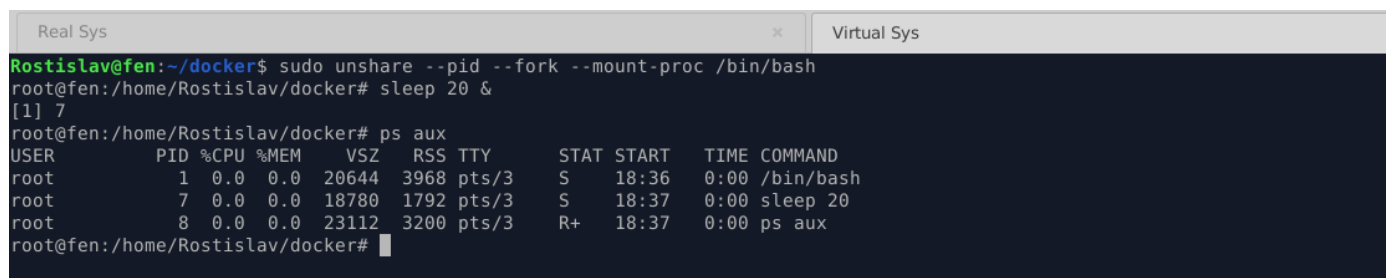
```
ping 8.8.8.8
```

Все icmp-пакеты потеряны

### 3. unshare

#### 3.1. PID

Активируем неймспейс изолирующий процесс *bash* виртуальной системы



```
Real Sys Virtual Sys
Rostislav@fen:~/docker$ sudo unshare --pid --fork --mount-proc /bin/bash
root@fen:/home/Rostislav/docker# sleep 20 &
[1] 7
root@fen:/home/Rostislav/docker# ps aux
USER      PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
root         1  0.0  0.0  20644  3968 pts/3    S   18:36   0:00 /bin/bash
root         7  0.0  0.0  18780  1792 pts/3    S   18:37   0:00 sleep 20
root         8  0.0  0.0  23112  3200 pts/3    R+  18:37   0:00 ps aux
root@fen:/home/Rostislav/docker#
```

для этого создадим с помощью команды *unshare* новое дерево процессов во вновь созданной и смонтированной псевдо директории *proc*

```
sudo unshare --pid --fork --mount-proc
```

на картинке виден запущенный в фоне процесс

```
sleep 20 &
```

виден только в виртуальной системе

```
ps aux
```

Но, данного процесса *sleep* не видно на реальной системе даже с привилегиями супер пользователя

```
Real Sys Virtual Sys
Rostislav@fen:~$ sudo ps aux | grep slee[p]
root      10372  0.0  0.0 18780 1792 pts/3    S    18:37   0:00  sleep 20
Rostislav@fen:~$
```

### 3.2 NET

Действие аналогично действию утилиты *ip* из пункта 2.

```
Real Sys Virtual Sys
Rostislav@fen:~/docker$ sudo /bin/unshare --net
root@fen:/home/Rostislav/docker# ip a
lo: <LOOPBACK> mtu 65536 qdisc noop state DOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
root@fen:/home/Rostislav/docker#
```

после применения команды сетевого нетспейса к процессам виртуальной системы

```
sudo unshare --net /bin/bash
```

остался только loopback интерфейс

### 3.3 USER

На реальной системе команда *id* отображает полную информацию о текущем пользователе

```
Real Sys Virtual Sys
Rostislav@fen:~$ id
uid=1002(Rostislav) gid=1002(Rostislav) rгруппы=1002(Rostislav),4(adm),24(cdrom),27(sudo),46(plugdev),122(lpadmin),133(lxd),134(sambashare)
Rostislav@fen:~$ id
uid=1002(Rostislav) gid=1002(Rostislav) rгруппы=1002(Rostislav),4(adm),24(cdrom),27(sudo),46(plugdev),122(lpadmin),133(lxd),134(sambashare)
Rostislav@fen:~$
```

На виртуальной системе после команды

```
sudo unshare --user /bin/bash
```

для текущего процесса *bash* применился неймспейс изолирующий от остальных пользователей

```
Real Sys Virtual Sys
Rostislav@fen:~/docker$ sudo unshare --user /bin/bash
nobody@fen:/home/Rostislav/docker$ id
uid=65534(nobody) gid=65534(nogroup) группы=65534(nogroup)
nobody@fen:/home/Rostislav/docker$
```

На выводе после команды *id* виден пользователь *nobody* группы *nogroup*

### 3.4. MOUNT

На виртуальной системе после команды

```
sudo unshare --mount /bin/bash
```

применился неймспейс изолирующий все устройства смонтированные под данным процессом *bash*. Мы можем только тут наблюдать свежесмонтированный диск */dev/sda1* по лейблу *BACKUP* в директорию */mnt*

```
Real Sys Virtual Sys
Rostislav@fen:~/docker$ sudo /bin/unshare --mount /bin/bash
root@fen:/home/Rostislav/docker# df -mh
Файл.система  Размер  Использовано  Дост  Использовано%  Смонтировано в
/dev/sdd1      112G      12G      101G      11% /
tmpfs          7,9G      0      7,9G      0% /dev/shm
tmpfs          1,6G      1,6M      1,6G      1% /run
tmpfs          5,0M      4,0K      5,0M      1% /run/lock
tmpfs          1,6G      80K      1,6G      1% /run/user/1002
tmpfs          1,6G      88K      1,6G      1% /run/user/1000
root@fen:/home/Rostislav/docker# mount LABEL=BACKUP /mnt
root@fen:/home/Rostislav/docker# df -mh
Файл.система  Размер  Использовано  Дост  Использовано%  Смонтировано в
/dev/sdd1      112G      12G      101G      11% /
tmpfs          7,9G      0      7,9G      0% /dev/shm
tmpfs          1,6G      1,6M      1,6G      1% /run
tmpfs          5,0M      4,0K      5,0M      1% /run/lock
tmpfs          1,6G      80K      1,6G      1% /run/user/1002
tmpfs          1,6G      88K      1,6G      1% /run/user/1000
/dev/sda1      110G      84G      21G      81% /mnt
root@fen:/home/Rostislav/docker#
```

Под реальной системой раздела */mnt* не видно

```
Real Sys Virtual Sys
Rostislav@fen:~$ df -mh
Файл.система  Размер  Использовано  Дост  Использовано%  Смонтировано в
tmpfs          1,6G      1,6M      1,6G      1% /run
/dev/sdd1      112G      12G      101G      11% /
tmpfs          7,9G      0      7,9G      0% /dev/shm
tmpfs          5,0M      4,0K      5,0M      1% /run/lock
tmpfs          1,6G      80K      1,6G      1% /run/user/1002
tmpfs          1,6G      88K      1,6G      1% /run/user/1000
Rostislav@fen:~$ ls /mnt
Rostislav@fen:~$ sudo dmesg | grep sda1
[sudo] пароль для Rostislav:
[ 1.920801] sda: sda1
[ 6656.731250] EXT4-fs (sda1): mounted filesystem 768b608e-acc8-442f-8eec-7cf99f32d07b with ordered data mode. Quota mode: none.
Rostislav@fen:~$
```

Но в логах информация о монтировании видна.

### 3.5. IPC

Создадим в реальной системе очередь сообщений межпроцессного взаимодействия с помощью команды *ipcrm*

```
ipcrm -Q 1234
```



```
Real Sys Virtual Sys
Rostislav@fen:~$ ipcmk -Q 1234
Идентификатор очереди сообщений: 0
Rostislav@fen:~$ ipc
ipcmk ipcrm ipcs
Rostislav@fen:~$ ipcs -q

----- Очереди сообщений -----
ключ  msqid      владелец права исп. байты сообщения
0x8b93d8f4 0          Rostislav 644      0          0

Rostislav@fen:~$ python3
Python 3.10.12 (main, Jun 11 2023, 05:26:28) [GCC 11.4.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> import sysv_ipc
>>> key = 0x8b93d8f4
>>> message = b"Hello world"
>>> queue = sysv_ipc.MessageQueue(key)
>>> queue.send(message)
>>> quit()
Rostislav@fen:~$ ipcs -q

----- Очереди сообщений -----
ключ  msqid      владелец права исп. байты сообщения
0x8b93d8f4 0          Rostislav 644      11         1

Rostislav@fen:~$ █
```

Очередь с ключом 1234 в шестнадцатичном счислении 0x8b93d8f4 после создания имеет 0 сообщений.

Обратимся к очереди с помощью пайтона и отправим сообщение *Hello World*

```
import sysv_ipc

key = 0x8b93d8f4
message = b"Hello world"
queue = sysv_ipc.MessageQueue(key)
queue.send(message)
```

Видно, что счетчик сообщений увеличился на единицу и его размер занимает 11 байт

```
ipcs -q
```

На виртуальной системе при изолированном *IPC* командой

```
sudo unshare --ipc /bin/bash
```

Не видны ни только сообщения, но и очереди межпроцессного взаимодействия

```
Real Sys Virtual Sys
Rostislav@fen:~$ sudo unshare --ipc /bin/bash
root@fen:/home/Rostislav# ipcs -q

----- Очереди сообщений -----
ключ  msqid      владелец права исп.  байты сообщения
root@fen:/home/Rostislav# exit
exit
Rostislav@fen:~$ ipcs -q

----- Очереди сообщений -----
ключ  msqid      владелец права исп.  байты сообщения
0x8b93d8f4 0      Rostislav  644      11      1

Rostislav@fen:~$ python3
Python 3.10.12 (main, Jun 11 2023, 05:26:28) [GCC 11.4.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> import sysv_ipc
>>> key = 0x8b93d8f4
>>> queue = sysv_ipc.MessageQueue(key)
>>> message, _ = queue.receive()
>>> print(message)
b'Hello world'
>>> exit()
Rostislav@fen:~$
```

После же выхода из изолированной среды видны и очередь и сообщение в нем

```
ipcs -a
```

С помощью пайтона мы можем прочесть сообщение

```
import sysv_ipc

key = 0x8b93d8f4
queue = sysv_ipc.MessageQueue(key)
message, _ = queue.receive()
print(message)
```

*Hello world*

### 3.6. UTS

На виртуальной системе для текущего процесса *bash* произведем изоляцию текущего имени хоста *fen*

```
sudo unshare --uts /bin/bash
```

```
Real Sys Virtual Sys
Rostislav@fen:~$ sudo unshare --uts /bin/bash
root@fen:/home/Rostislav# hostname
fen
root@fen:/home/Rostislav# echo "NewHostName" > /proc/sys/kernel/hostname
root@fen:/home/Rostislav# hostname
NewHostName
root@fen:/home/Rostislav# exit
exit
Rostislav@fen:~$ hostname
fen
Rostislav@fen:~$
```

Теперь поменяем имя хоста на новое

```
echo "NewHostName" > /proc/sys/kernel/hostname
```

Видим, что имя хоста сменилось на *NewHostName*

На реальной системе имя хоста осталось прежним - *fen*

```
Real Sys Virtual Sys
Rostislav@fen:~$ hostname
fen
Rostislav@fen:~$ cat /proc/sys/kernel/hostname
fen
Rostislav@fen:~$ hostname
fen
Rostislav@fen:~$
```

## unshare и несколько неймспейсов

Применим сразу несколько неймспейсов как было показано на семинаре № 1

```
sudo unshare --pid --net --fork --mount-proc /bin/bash
```

```
Real Sys Virtual Sys
Rostislav@fen:~$ sudo unshare --pid --net --fork --mount-proc /bin/bash
root@fen:/home/Rostislav# sleep 20 &
[1] 7
root@fen:/home/Rostislav# ps -fn
  PID TTY          STAT       TIME COMMAND
    1 pts/2    S          0:00 /bin/bash
    7 pts/2    S          0:00 sleep 20
    8 pts/2    R+         0:00 ps -fn
root@fen:/home/Rostislav# ip a
1: lo: <LOOPBACK> mtu 65536 qdisc noop state DOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
[1]+  Завершён                  sleep 20
root@fen:/home/Rostislav# ping 8.8.8.8
ping: connect: Сеть недоступна
root@fen:/home/Rostislav# ping localhost
ping: connect: Сеть недоступна
root@fen:/home/Rostislav# ping 127.0.0.1
ping: connect: Сеть недоступна
```