Пространства имен Linux

Иван Ганкевич

12 сентября 2018 г.

Outline

С чего все начиналось

Пространства имен

Примеры

Контрольные группы

Что со всем этим можно сделать?

Раздел 1

С чего все начиналось

Создание процесса

```
int pid = fork();
if (pid == 0) {
    child_main();
    exit(0);
}
int status = 0;
check(waitpid(pid, &status, 0));
```

Создание процесса
вывод

```
clone(child_stack=NULL, flags=CLONE_CHILD_CLEARTID|
   CLONE_CHILD_SETTID|SIGCHLD, child_tidptr=0
   x7fb0c43921d0) = 6487
Hello from a process: pid=6487
```

Создание потока

```
std::thread thr(child_main);
thr.join();
```

Создание потока
вывод

```
clone(child_stack=0x7f2bc5cf9fb0, flags=CLONE_VM|
   CLONE_FS|CLONE_FILES|CLONE_SIGHAND|CLONE_THREAD|
   CLONE_SYSVSEM|CLONE_SETTLS|CLONE_PARENT_SETTID|
   CLONE_CHILD_CLEARTID, parent_tidptr=0x7f2bc5cfa9d0,
   tls=0x7f2bc5cfa700, child_tidptr=0x7f2bc5cfa9d0) =
   6520
Hello from a pthread: thread_id=139825979041536
```

Создание потока
опции

Опция	Назначение
`CLONE_VM`	общая память
`CLONE_FS`	общие рабочая директория, корень ФС и маска
`CLONE_FILES`	общие файловые дескрипторы
`CLONE_SIGHAND`	общие обработчики сигналов
`CLONE_SYSVSEM`	общие примитивы синхронизации
`CLONE_THREAD`	добавление в группу потоков
`CLONE_SETTLS`	выделение локальной памяти потока

Системный вызов clone(2)

```
size_t stack_size = 1024*10;
char* child_stack = new char[stack_size];
void* child_stack_end = child_stack + stack_size;
int pid = 0;
check(pid=clone(child_main, child_stack_end,
    CLONE_NEWUTS | CLONE_NEWUSER | SIGCHLD, 0));
int status = 0;
check(waitpid(pid, &status, 0));
delete[] child_stack;
```

Системный вызов clone(2)

вывод

```
Host UTS: nodename=storm,domainname=(none)
clone(child_stack=0x7f9fbc880000, flags=CLONE_NEWUTS|
   CLONE_NEWUSER|SIGCHLD) = 6551
Hello from a container: uid=65534
Container UTS: nodename=spicy,domainname=salmon
```

Раздел 2

Пространства имен

Опции функции `clone(2)`

Опция `clone(2)`	Пространство имен
`CLONE_NEWNS`	точки монтирования
`CLONE_NEWPID`	процессы
`CLONE_NEWUSER`	идентификаторы пользователей и групп
`CLONE_NEWNET`	сетевые устройства
`CLONE_NEWUTS`	имя хоста и доменное имя машины
`CLONE_NEWIPC`	примитивы сихронизации
`CLONE_NEWCGROUP`	контрольные группы

Пространство UTS

Системный вызов	Назначение
`sethostname(2)`	получение/изменение имени хоста
`gethostname(2)`	
`setdomainname(2)`	получение/изменение NIS домена
`getdomainname(2)`	
`uname(2)`	вывод информации о системе

Пространство IPC

Примитивы синхронизации и межпроцессного взаимодействия.

Заголовочный файл	Описание
`sys/shm.h`	общая память System V
`sys/ipc.h`	
`sys/sem.h`	семафоры System V
`sys/mman.h`	общая память POSIX
`semaphore.h`	семафоры POSIX
`mqueue.h`	очереди сообщений POSIX

Пространство PID

Системный вызов	Назначение
`getpid(2)`	получение идентификатора
`getppid(2)`	текущего/родительского процессов

Первый процесс особенный:

- имеет идентификатор 1,
- ждет завершения других процессов и процессов-зомби `wait(2)`,
- обрабатывает сигналы выборочно.

Пространство USER

Системный вызов	Назначение

`getuid(2)`
`getgid(2)`
`get*id(2)`
	получение идентификатора пользователя/группы

Особенности:

- процесс получает полные привилегии в новом пространстве (`capabilities(7)`),
- имена пользователей и групп не копируются, только идентификаторы,
- отображение идентификаторов задается в файлах `uid_map` и `gid_map`.

Пространство NS (точки монтирования)

Системный вызов	Назначение

`mount(2)`
`umount(2)`
`umount2(2)`
	монтирование/размонтирование файловых систем и каталогов

Опция `mount(2)`	Назначение
`MS_SHARED`	
`MS_PRIVATE`	
`MS_SLAVE`	
`MS_REC`	применить рекурсию
`MS_UNBINDABLE`	остановить рекурсию

Системный вызов `pivot_root(2)`

Для успешного выполнения `pivot_root(2)` следующие объекты монтирования не должны быть совместно используемыми:

- целевой каталог старого корня файловой системы,
- текущий родительский каталог нового корня файловой системы (на момент вызова `pivot_root(2)`),
- родительский каталог нового корня файловой системы.

Источник: `pivot_root` (IBM developerworks)

Пространство NET (сетевые устройства)

Системный вызов	Описание
`ip(8)`	управление сетевыми устройствами (интерфейс для командной строки)
`netlink(7)`	управление сетевыми устройствами
`rtnetlink(7)`	
`getifaddrs(3)`	получение списка сетевых устройств

Дополнительные системные вызовы

Системный вызов	Описание
`unshare(2)`	изменение пространств имен без создания нового процесса
`setns(2)`	войти в пространство имен, указанное с помощью дескриптора
`nsenter(1)`	то же самое для командной строки

Раздел 3

Примеры

Программа `userns_child_exec`

```
Usage: ./userns_child_exec [options] cmd [arg...]
    -i          New IPC namespace
    -m          New mount namespace
    -n          New network namespace
    -p          New PID namespace
    -u          New UTS namespace
    -U          New user namespace
    -M uid_map  Specify UID map for user namespace
    -G gid_map  Specify GID map for user namespace
```

Исходный код: `user_namespaces(7)`.

Пространство USER

```
id -u; id -g
./userns_child_exec \
  -U -M "0 1000 1" -G "0 1000 1" \
  sh -c "id -u; id -g"

1000
1000
0
0
```

Пространство USER

```
./userns_child_exec \
  -U -M '0 1000 1' -G '0 1000 1' \
  sh -c "touch xyz; ls -l xyz"
ls -l xyz

-rw-r--r-- 1 root root 0 сен 12 19:16 xyz
-rw-r--r-- 1 igankevich igankevich 0 сен 12 19:16 xyz
```

Пространство USER

```
./userns_child_exec \
  -U -M '0 1000 1' -G '0 1000 1' \
  sh -c "ls -ldn /etc"
ls -ld /etc

drwxr-xr-x. 185 65534 65534 12288 сен 10 10:16 /etc
drwxr-xr-x. 185 root root 12288 сен 10 10:16 /etc
```

Пространство USER+UTS

```
./userns_child_exec \
  -U -M '0 1000 1' -G '0 1000 1' -u \
  sh -c "hostname; hostname userns-rocks; hostname;
     uname -nsr"

storm
userns-rocks
Linux userns-rocks 4.16.8-300.fc28.x86_64
```

Пространство USER+IPC

```
ipcs --shmems --pid | grep ^[0-9]
./userns_child_exec \
  -U -M '0 1000 1' -G '0 1000 1' -i \
  sh -c "ipcmk -M 1000; ipcs --shmems --pid | grep ^[0-
    9]"
ipcs --shmems --pid | grep ^[0-9]
```

```
59834368   igankevich 16730      3432
59867137   igankevich 16730      3432
60194819   igankevich 16730      3432
60227588   igankevich 16730      3432
60162053   igankevich 16730      3432
60260358   igankevich 16730      3432
61767688   igankevich 16730      3432
62062601   igankevich 16730      3432
61669386   igankevich 16730      3432
```

Пространство USER+PID

```
./userns_child_exec \
  -U -M '0 1000 1' -G '0 1000 1' -p \
  sh -c "echo \$\$"

1
```

Пространство USER+PID+NS

```
./userns_child_exec \
  -U -M '0 1000 1' -G '0 1000 1' -m -p \
  sh -c "mount -t proc proc /proc &&
     ps -eo args,user,pid"

COMMAND                     USER       PID
ps -eo args,user,pid        root         1
```

Пространство USER+NET

```
./userns_child_exec \
  -U -M '0 1000 1' -G '0 1000 1' -n \
  sh -c "/sbin/ip addr"

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
```

Пространство USER+NET

```
./userns_child_exec \
  -U -M '0 1000 1' -G '0 1000 1' -n \
  sh -c "/sbin/ip link set dev lo up; /sbin/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
```

Пространство USER+NET+NS

```
./userns_child_exec \
  -U -M '0 1000 1' -G '0 1000 1' -m -n \
  sh -c "
mount -t sysfs sysfs /sys
mount -t tmpfs tmpfs /run
chmod 777 /run
/sbin/ip netns add netns1
/sbin/ip link add veth0 type veth peer name veth1
/sbin/ip link set veth1 netns netns1
/sbin/ip netns exec netns1 /sbin/ifconfig veth1
   10.1.1.1/24 up
/sbin/ifconfig veth0 10.1.1.2/24 up
ping -c 3 10.1.1.1
"
```

Пространство USER+NET+NS

```
PING 10.1.1.1 (10.1.1.1) 56(84) bytes of data.
64 bytes from 10.1.1.1: icmp_seq=1 ttl=64 time=0.034 ms
64 bytes from 10.1.1.1: icmp_seq=2 ttl=64 time=0.021 ms
64 bytes from 10.1.1.1: icmp_seq=3 ttl=64 time=0.016 ms

--- 10.1.1.1 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2071ms
rtt min/avg/max/mdev = 0.016/0.023/0.034/0.009 ms
```

Раздел 4

Контрольные группы

См. документацию (версия 2)

Раздел 5

Что со всем этим можно сделать?

Что можно делать с помощью пространств имен?

- Создать виртуальную сеть для тестирования распределенных алгоритмов.
- Установить несколько операционных систем через менеджер пакетов, не имея прав суперпользователя.
- Создать PAM модуль, который бы менял ОС из предыдущего пункта при входе в систему.
- Создать систему сборки кода C/C++/Fortran, которая бы автоматически загружала зависимости (как maven, npm, pip, gem и др.).
- Монтировать корневую файловую систему при запуске задач на кластере, т.е. запускать задачу в том же окружении, в котором она разрабатывалась.

Пространства имен в SystemD

```
[Unit]
Description=nginx reverse proxy server

[Service]
User=nginx
Group=nginx
ExecStart=/usr/bin/nginx
ReadOnlyDirectories=/proc
PrivateTmp=true
PrivateDevices=true
ProtectHome=true
ProtectSystem=full
```

Контрольные группы в SystemD

```
[Unit]
Description=Dropbox daemon

[Service]
Type=forking
ExecStart=/usr/bin/dropbox start
ExecStop=/usr/bin/dropbox stop
Nice=19
BlockIOWeight=100
CPUQuota=10%
CPUShares=128

[Install]
WantedBy=default.target
```

Интеграция с PAM (pam_namespace(8))

В файле namespace.conf:

```
/tmp     /tmp-inst/               level      root,adm
/var/tmp /var/tmp/tmp-inst/       level      root,adm
$HOME    $HOME/$USER.inst/inst-   context
```

О чем стоит задуматься

- Новое пространство — это не обязательно новая корневая ФС и новый IP-адрес.
- Для создания новых пространств не обязательно иметь права суперпользователя.
- Пространства подходят и для графических приложений.
- Пространства используются не только в контейнерах приложений.

Литература I

Документация ядра Linux

- unshare
- Пространства имен
- Контрольные группы (версия 2)
- Контрольные группы (устаревшая версия 1)

Полезные статьи

- `pivot_root` (IBM developerworks)

Литература II

Страницы руководства Linux

- `pid_namespaces(7)`
- `namespaces(7)`
- `user_namespaces(7)`
- `capabilities(7)`
- Руководство Linux (рус.)

Библиотека для работы с сетевыми устройствами

- libnl