УДК 004.451(07)
https://doi.org/10.36906/AP-2020/58

МЕТОДИЧЕСКИЕ РЕКОМЕНДАЦИИ ПО ПРАКТИЧЕСКОМУ ОСВОЕНИЮ ПРОГРАММНЫХ ИНТЕРФЕЙСОВ ПРОСТРАНСТВ ИМЕН ОПЕРАЦИОННЫХ СИСТЕМ

Широков В. В.
канд. техн. наук
Санкт-Петербургский государственный электротехнический университет «ЛЭТИ»
г. Санкт-Петербург, Россия
Щиголева М. А.
канд. техн. наук
Санкт-Петербургский государственный электротехнический университет «ЛЭТИ»
г. Санкт-Петербург, Россия

Аннотация. В работе рассматриваются рекомендации по изучению интерфейса программирования пространств имен Linux. Пространства имен — это механизм, позволяющий изолировать некоторый вид ресурса одного процесса от доступа другого процесса. Пространства имен являются компонентами создания контейнеров, обеспечивающих выполнение процессов в изолированном окружении. Целью контейнеризации является безопасное выполнение процессов. Данные рекомендации будут полезны при разработке практических занятий для подготовки бакалавров по направлению 09.03.02 «Информационные системы и технологии» и специалистов по специальности 090301.65 «Компьютерная безопасность».

Ключевые слова: интерфейс программирования; операционная система; пространства имен; родительские и дочерние процессы; изоляция процессов.

Целью данной работы является предоставление рекомендаций преподавателям и учащимся по практическому ознакомлению с функциями операционных систем, обеспечивающими выполнение дочернего процесса в изолированном от родительского процесса пространстве имен.

Известно, что основным системным вызовом для создания нового процесса в операционных системах, поддерживающих стандарт POSIX, является следующий вызов fork() [1].

Вызов fork(), сделанный в некотором процессе, который будем называть родительским, создает дочерний процесс, который выполняется в том же самом пространстве параметров, что и родительский процесс.

О каких пространствах параметров, которые в данном случае называются пространствами имен, идет речь [2]?

1. Пространство имен идентификаторов процессов (PID);
2. Пространство имен хоста (UTS);
3. Пространство имен межпроцессного взаимодействия (IPC);
4. Пространство имен сетевого взаимодействия (NET);

5. Пространство имен файловой системы (NS);
6. Пространство имен пользователей и групп (USER);
7. Пространство имен контрольных групп (CGROUP);
8. Пространство имен времени (TIME).

В традиционном исполнении родительский и дочерний процессы «видят» имена, входящие в перечисленные пространства, одинаковыми. То есть перечисленные пространства имен являются общими для них и изменения этих имен одним процессом затрагивают и другой процесс. Обеспечение надежности выполнения процессов в операционной системе тесно связано с выполнением их в изолированных пространствах имен. То есть изменение имен в пространстве одного процесса не должно затрагивать аналогичные имена в пространстве другого процесса. Такое ограничение необходимо при реализации контейнеров – средств изоляции процессов, основанных на механизмах пространства имен [3].

Системным средством, позволяющим создавать дочерние процессы, выполняющиеся в отличных от родительских процессов, пространствах имен, является следующий вызов [4]:

int clone(int (*fn)(void *), void *child_stack, int flags, void *arg),

где

fn – функция, реализующая дочерний процесс,

child_stack – указатель на начало стека дочернего процесса,

flags – набор флагов, передаваемых дочернему процессу,

arg – аргументы, передаваемые функции fn.

Шаблон функции, реализующей дочерний процесс, имеет следующий вид:

static int fn(void *arg).

Набор флагов flags, передаваемых в функцию clone(), позволяет управлять пространствами имен процесса-потомка, которые будут совместными или изолированными от пространств имен процесса-родителя. С помощью определенных флагов можно изолировать следующие пространства имен процесса-родителя и процесса-потомка.

Таблица 1

Пространства имен Linux

Название пространства	Значение флага	Вид пространства
PID	CLONE_NEWPID	ID процессов
UTS	CLONE_NEWUTS	имя хоста
IPC	CLONE_NEWIPC	очереди сообщений
Network	CLONE_NEWNET	сетевые параметры
Mount	CLONE_NEWNS	файловая система
User	CLONE_NEWUSER	ID пользователей и групп
CGROUP	CLONE_NEWCGROUP	Контрольные группы процессов
TIME	CLONE_NEWTIME	Время

Параметр flags, передаваемый в функцию clone, может формироваться из перечисленных флагов путем логического сложения с базовым флагом SIGCHLD, сигналом, который посылается родителю, когда потомок завершается.

Если параметр flags задать в виде CLONE_NEWPID | SIGCHLD, дочерний процесс создается в новом пространстве имен PID. Это означает, что процессы в разных пространствах имен PID могут иметь один и тот же PID. Первый процесс, созданный в новом

пространстве (т.е. процесс, созданный вызовом clone()), будет иметь PID, равный 1, т.е. будет выполнять функцию процесса init, выполняющегося при загрузке операционной системы. Родительский процесс будет иметь PID, равный 0.

Если параметр flags задать в виде CLONE_NEWUTS | SIGCHLD, дочерний процесс создается в новом пространстве имен UTS (Unix Time Sharing). Пространство имен UTS — это набор идентификаторов, возвращаемых функцией uname(). Сюда включаются имя хоста (nodename) и доменное имя (domainname). Если изменить эти имена в дочернем процессе, то новые изменившиеся имена будут только в нем и видны. В родительском процессе останутся прежние имена - имя хоста и доменное имя.

Если параметр flags задать в виде CLONE_NEWIPC | SIGCHLD, дочерний процесс создается в новом пространстве имен IPC. Пространство имен IPC - это набор объектов межпроцессного взаимодействия стандартов SVID и POSIX. Например, если создать очередь сообщений POSIX в новом пространстве имен IPC, то она будет видна всем процессам из этого пространства имен IPC и не будет видна процессам из других пространств имен IPC.

Если параметр flags задать в виде CLONE_NEWNET | SIGCHLD, дочерний процесс создается в новом сетевом пространстве имен. Сетевое пространство имен — это набор объектов, связанных с сетями. А именно, сетевые устройства, сетевые протоколы, таблицы маршрутизации, номера портов сокетов. В каждом сетевом пространстве имен могут быть одинаковые виртуальные объекты, например, сокеты, привязанные к одному и тому же номеру порта.

Если параметр flags задать в виде CLONE_NEWNS | SIGCHLD, дочерний процесс создается в новом пространстве имен монтирования. Пространство имен монтирования создает уникальный вид иерархии файловой системы. Этот вид файловой системы создается функциями mount() и umount(). При этом действия этих функций видны в рамках одного пространства имен монтирования и не видны в других пространствах имен монтирования. Пространство имен монтирования необходимо создавать вместе с пользовательским пространством имен.

Если параметр flags задать в виде CLONE_NEWUSER | SIGCHLD, дочерний процесс создается в новом пространстве имен пользователя. Пространство имен пользователя изолирует идентификаторы пользователей и групп. Существуют ограничения на назначение пользователей и групп в дочернем пространстве имен пользователей. Эти ограничения формируются путем редактирования файлов /proc/[pid]/uid_map и /proc/[pid]/gid_map. Тем самым формируются отображения имен пользователей дочернего пространства имен в имена пользователей родительского пространства имен.

После ознакомления с пространствами имен по документации [2] обучаемым предлагается самостоятельно создать программу со следующей структурой:

```
static int childFunc(void *arg){
//здесь выполняются действия, связанные с выводом
//параметров объектов дочернего пространства имен
}
#define STACK_SIZE (1024 * 1024)
main() {
char *stack = (char*)malloc(STACK_SIZE);
char *stackTop = stack + STACK_SIZE;
//здесь выполняются действия, связанные с выводом
//параметров объектов родительского пространства имен
int child_pid = clone(childFunc, stackTop, flags, NULL);
```

```
//здесь выполняются действия, связанные с выводом
//параметров объектов родительского пространства имен
waitpid(child_pid,NULL,0);
//здесь выполняются действия, связанные с выводом
//параметров объектов родительского пространства имен
}
```

В работе следует продемонстрировать возможности изоляции объектов дочернего пространства имен от объектов родительского пространства имен путем вывода параметров объектов дочернего и родительского пространств имен. Обучаемый должен выбрать один из типов пространств имен. Следует отметить, что работа с пространствами имен User, CGROUP и TIME обладает повышенной сложностью. При работе с пространством имен User необходимо знакомство с файловой системой /proc, при работе с пространством имен CGROUP необходимо ознакомление с механизмами контрольных групп, а пространство имен TIME, позволяющее процессу-родителю и процессу-потомку иметь разный масштаб времени, введено только в последней версии ядра Linux.

Для демонстрации возможностей изолирования пространств имен предлагается использовать следующие системные функции:

Для пространства имен PID можно использовать функции getpid() и getppid().

Для пространства имен UTS можно использовать функции uname(), sethostname(), setdomainname().

Для пространства имен IPC можно использовать функции mq_open().

Для пространства имен Network можно использовать функции socket() и bind().

Для пространства имен Mount можно использовать функции mount(), umount(), opendir(), readdir().

Для пространства имен User можно использовать функции getuid(), getgid(), setuid(), setgid(). Также необходимо ознакомиться со структурой файлов /proc/[pid]/uid_map и /proc/[pid]/gid_map.

После реализации предложенной программы и успешной демонстрации возможностей изолирования выбранного пространства дочернего процесса от такого же пространства родительского процесса, обучаемым может быть предложен ряд контрольных вопросов, примерно следующего содержания:

1. В чем состоят различия между вызовами для создания процессов fork() и clone()?
2. Что представляют собой пространства имен Linux?
3. Какие функции, кроме clone(), еще входят в программный интерфейс пространств имен Linux?
4. В каком каталоге можно найти файловые дескрипторы пространств имен процесса?
5. Какие существуют ограничения в назначении пользователей и групп в дочернем пространстве имен пользователей?

Сочетание ответов на контрольные вопросы и разбор самостоятельно разработанной программы позволят оценить степень освоения теоретического материала и навыки самостоятельной работы в области своей профессиональной подготовки.

Вариативный подход по созданию дочерних процессов в новом пространстве имен при различных вариантах задания параметра flags в виде:

CLONE_NEWPID | SIGCHLD,
CLONE_NEWUTS | SIGCHLD,
CLONE_NEWIPC | SIGCHLD
CLONE_NEWNET | SIGCHLD,

CLONE_NEWNS | SIGCHLD,
CLONE_NEWUSER | SIGCHLD,

обеспечивает фактор возможности многовариантных подходов к решению профессиональных задач в зависимости от целевого назначения систем, объектов систем, сетевом пространстве решения задачи.

Электронные ресурсы:

https://www.opennet.ru/docs/RUS/linux_parallel/node7.html.
https://man7.org/linux/man-pages/man7/namespaces.7.html.
https://ru.qaz.wiki/wiki/Linux_namespaces.
http://ru.manpages.org/clone/2.