Настройка окружения для работы с ОС Pintos

Окружение для компиляции и запуска pintos может быть подготовлено только на базе одной из операционных систем Linux. Исполняемый образ ОС Pintos может быть запущен на эмуляторе программного обеспечения, таком как Bochs, QEMU, или VMWare Player. Данные эмуляторы являются независимыми программными продуктами, т.к. не представлены в исходных кодах ОС Pintos и на поставляются вместе с ними. Данное руководство подразумевает, что настройка окружения производится на Linux Ubuntu 16.04, но с некоторыми изменениями оно будет справедливо для любой версии операционной системы семейства Debian.

1. Необходимо убедится, что в системе присутствуют все необходимые пакеты, необходимые для сборки и запуска ОС pintos следующей командой:

```
sudo apt-get install gcc-multilib make perl qemu
```

В данной команде контекст sudo требует запуска от имени суперпользователя (необходимое условие для утилиты установки apt-get). В результате выполнения команды для ОС Linux Ubuntu 16.04 будет установлена версия компилятора gcc 5.4.0, эмулятор qemu 2.5.0.

Примечение: если установлена 32-разрядная система Linux, то вместо пакета gcc-multilib следует устанавливать пакет gcc.

2. Эмулятор qemu содержит несколько бинарных файлов для эмуляции систем различных архитектур. Все бинарные файлы располагаются в каталоге /usr/bin/. Скрипты, входящие в ОС Pintos, запускают эмулятор командой "qemu", без указания конкретной системы эмуляции. Поэтому для корректной работы скриптов pintos требуется создать ссылку, указывающую какой именно бинарный файл будет запускаться по команде "qemu".

```
sudo ln /usr/bin/qemu-system-x86_64 /usr/bin/qemu
```

Примечение: если установлена 32-разрядная система Linux, то вместо qemu-system-x86_64 следует использовать qemu-system-x86.

3. Необходимо выполнить распаковку исходных текстов ОС pintos (предполагается, что исходные тексты поставляются в виде файл pintos.tar.gz):

```
tar –zxf pintos.tar.gz
```

В каталоге utils располагаются некоторые скрипты для выполнения операций по сборке и запуску ОС pintos. Эти скрипты должны быть размещены в каталоге /usr/bin, должен быть разрешен запуск этих скриптов. Следующий набор команд выполнит указанные задачи:

```
cd pintos/src/utils
sudo cp –f backtrace pintos pintos-mkdisk pintos-set-cmdline Pintos.pm pintos-gdb
../misc/gdb-macros /usr/bin/
sudo chmod 755 /usr/bin/backtrace /usr/bin/pintos /usr/bin/pintos-mkdisk /usr/bin/pintos-set-
cmdline /usr/bin/pintos-gdb
sudo chmod 644 /usr/bin/Pintos.pm /usr/bin/gdb-macros
```

На этом установка и настройка окружения для работы с ОС Pintos завершена.

Для проверки корректности настройки среды необходимо скомпилировать ядро ОС Pintos и запустить его на эмуляторе.

Для компиляции необходимо выполнить команду make в каталоге threads:

```
tar -zxf pintos.tar.gz
cd pintos/src/threads
make
```

Успешная компиляция подтверждается отсутствием на экране сообщений об ошибках. Для запуска pintos на эмуляторе qemu необходимо выполнить, например, команду

```
pintos --qemu -- -q run alarm-single
```

Данная команда, выполненная в каталоге threads, запустит скомпилированный образ, запустит тест "alarm-signle" и выключит эмулятор по его завершении. Результат выполнения может выглядеть, например, следующим образом:

```
Loading...........
Kernel command line: -q run alarm-single
Pintos booting with 3,968 kB RAM...
367 pages available in kernel pool.
367 pages available in user pool.
Calibrating timer...  419,020,800 loops/s.
Boot complete.
Executing 'alarm-single':
(alarm-single) begin
(alarm-single) Creating 5 threads to sleep 1 times each.
(alarm-single) Thread 0 sleeps 10 ticks each time,
(alarm-single) thread 1 sleeps 20 ticks each time, and so on.
(alarm-single) If successful, product of iteration count and
(alarm-single) sleep duration will appear in nondescending order.
(alarm-single) thread 0: duration=10, iteration=1, product=10
(alarm-single) thread 1: duration=20, iteration=1, product=20
(alarm-single) thread 2: duration=30, iteration=1, product=30
(alarm-single) thread 3: duration=40, iteration=1, product=40
(alarm-single) thread 4: duration=50, iteration=1, product=50
(alarm-single) end
Execution of 'alarm-single' complete.
Timer: 293 ticks
Thread: 0 idle ticks, 293 kernel ticks, 0 user ticks
Console: 986 characters output
Keyboard: 0 keys pressed
Powering off...
```