Видите, да? Ну, дальше, собственно, вы видите про Speed Environ. Собственно, то, что сейчас я очень быстро демонстрировала на доске. И здесь нету объявлений, как на доске я я писала, значит, и в форе, видите, инт я поставила, но еще раз повторяю, этому примеру миллион лет. То есть, сказать, все это написано очень давно, потому что важная информация о процессах профессионалами всегда осознавалась. И в классическом варианте I объявляется вне фора, потому что классический C этого не допускал. Дальше идут названия, фактически, той информации, которая находится в environ. Ну, здесь некоторые переменные окружения.

В задании не написано, я это говорю устно, всю информацию, которую вы должны получить, вам надо записать в файл. Причем сделать это надо дома. Не прийти с вот этим файлом и его демонстрировать. Потому что если каждый из вас будет создавать нужную информацию на лету, это займет слишком много времени. То есть этот файл, в который вы записываете все позиции, которые перечисляются дальше, делается заранее.

При этом анализируем мы с вами многопоточный сервер, который написали для булочной, но кроме многопоточного сервера вам надо написать однопоточный сервер булочная. и сравнить две эти реализации ну понятно что Дальше. Стат. Опять же, содержимое файла ProSpeedStat, название соответствующих полей. Здесь очень много важной информации, в частности приорити, NICE, numthreads, число потоков, соответственно у Вас тогда должно быть два файла для многопоточного и однопоточного. Вот для многопоточного у вас будет, в зависимости от количества созданных потоков, вот здесь стоять там 4, 3 там. Если у вас потоки завершаются, естественно они здесь отражаться не будут, потому что это динамическая информация. То есть здесь нет такого, что как история выводится бы там в процессе работы программы обсудить, где клиентов для каждого создавали поток, да? Здесь этого видно не будет. То есть это будет видно только в текущий момент времени, сколько у вас потоков. ВСАИС, вот это тоже очень важное поле, запишите себе, фактически, сейчас я вам покажу. ММАП у вас будет совпадение в размере, поскольку ММАП предоставляет нам виртуальный адресный пространство. Другого адресного пространства и процессора нет. Далее нужно связать старт-код, энд-код, высайд, сеткетч. Соответственно, вот процессор, номер процессора, на котором в последний раз выполнялся процесс. Если посмотреть в taskstrap, там слово процессор отсутствует. Там присутствует CPU. Видите, что здесь CPU.

В полисе мы должны обратить на полисе внимание, Тем более, что в следующей лабораторной работе полицию будут выводить. И, как вы видите, в СТАД она также присутствует. Это политика планирования, алгоритм планирования. Но у вас для обычного приложения должен стоять, если мне не изменяет память, ноль. То есть планировщик, установленный в системе по умолчанию. Проверим.

...продемонстрированы все открытые файлы процесса, но вы же в сервере файл ничего не пишете, в самом сервере выручное, у вас будет 3 открытых файла 0, 1 и 2.

Значит, важно прочитать, что если у вас основной поток завершил свое выполнение, то это директорию будет недоступно. Главный поток main завершился в выполнении.

Вот она, в этой рубрике. Рубрика у нас с вами /proc/[pid]/maps. Ну, также, вот видите, смотрите, МАП, для получения дополнительной информации, соответствующих регионов, виртуальной памяти процесса. Это я написала. В мануале написано чуть по-другому. Значит, можете сопоставить. Я не стала здесь копировать английский текст. Размер региона кратен размеру страницы. То есть именно в том варианте, который вы будете иметь, у вас размер региона будет вкратце на размеру страницы. и вам размер регионов, вот у вас начальный адрес, конечный адрес, да, это в байтах. Вам надо дополнительно ввести столбец страницы, то есть для каждого региона указать количество страниц.

Соответственно, смотрите, вот здесь после heap стоит многоточие. Вот там, где у вас будет многопоточное приложение, у вас здесь будут анонимные отображения. И, по-моему,

они даже не пишут, что это анонимное отображение. Ну, понятно, что [stack:<tid>] – это псевдопути, то есть это не имена файлов.

И вот дальше подключена функция mmap, то есть вот они, это буквальный перевод, новое сопоставление виртуального адресного пространства вызывающего процесса. Слово сопоставление не очень правильное с точки зрения русского языка, но вчера вечером чуть не переделала.

Вся работа, фактически с вот этими двумя вариантами сервера связана с nmaps. Именно вот эта рубрика демонстрирует колоссальную разницу в ресурсах. Многопоточное приложение требует значительно большего адресного пространства, то есть адресное пространство процесса значительно увеличивается. И соответственно увеличивается количество физических страниц для выполнения такого приложения.

Для того, чтобы нам осознать, нужна ли нам многопоточность, нужна ли она, или она только пожирает ресурсы, дело в том, что я пыталась найти для потоков идентификаторы процессоров, номера процессоров, если у вас 4 ядра, 0, 1, 2, 3, 4, такой информации, может быть, и можно получить, но из того, что я читала, как-то явно не следует, что мы можем посмотреть процессор, на котором выполняется каждый поток. То есть мы можем увидеть, что наши потоки или выполняются на одном процессоре, или действительно параллельно выполняются на четырёх процессорах.

Поэтому в этих программах приложения серверов надо считать время. При этом надо определить, какое время является показательным для оценки времени работы приложения. Как Вы думаете?

Во-первых, у нас две части работы сервера: первая часть это получение клиента номера, а вторая часть по номеру обслуживание. Ну, фактически по номеру вы получали букву, правильно? Вот надо оценивать это время.

У некоторых из вас, по группам не помню, в разных группах разные источники, были такие варианты и получение номера многопоточная часть и получение буквы многопоточная часть. Соответственно, надо считать время получения номера. Ну, фактически, вы же номер посылаете клиенту? То есть надо по коду посмотреть, да? Запрос клиента и ответ сервера: вот это время считать, и сопоставить.

Разница существует в однопоточном сервере и многопоточном? Вопрос определения времени не праздный. Здесь, видимо, тоже придется подумать и посмотреть, что мы получаем практически. При этом у Вас же клиенты работают с задержкой случайной, да? Сервер не должен работать с задержкой. Клиенты. Мы моделируем, что клиенты возникают в произвольный момент времени, не так, как Вы успеваете там запустить от первого, второго, третьего, четвёртого клиентов. Вы их запускаете, но они не сразу отправляют запрос, правильно? Вот для того, чтобы показать, что вот это вот как в реальной сети это вот спонтанно как бы возникает такой запрос. Потому что если вы по порядку будете задавать, вы такие медленные, мы с вами такие медленные, что действительно вот по этой очереди всё и будет обслуживать, да? Но сам сервер, в нём не должно быть задержек искусственных. Если у кого-то они есть, надо убрать, потому что нас интересует реальное время. Ну, реальное с точки зрения, что там нет дополнительных ненужных задержек. Ну, comm это имя файла, мы тоже с этим, так, вот, pagemap. Ну, собственно, тоже вот это вот информация, этот файл.

В нем нас интересуют страницы в памяти, выгруженные страницы и грязные страницы. root.

Задание Написать программу, которая в пользовательском режиме выводит в файл окружения State.cmdline.cvd.exe.root.maps.feedmap.io.com.root.tasks. Task – это значит здесь не приведено. Давайте откроем в мануале тоже упражнение.

Значит, директория task содержит поддиректории для каждого потока процесса. name of each subdirectory gettid(), но мы это уже использовали. gettid(), где использовали? Про-

буем найти номер процессора. Под директорией task. Под директорией tid.

Дальше в металличке указывается информация, которую мы получаем просто в системе. Советую прочитать, в частности, прерывания. Мы это никуда не выводим. Соответственно, нам все это понадобится.