Оглавление

[**1.** **Сервер 192.168.59.182 – METIS** Основное по работе SQL Server Management Studio (SSMS) 3](#_Toc115881385)

[**1.2** **Добавление пользователей для входа** 3](#_Toc115881386)

[**1.3** **Изменение порта и добавление правил в брандмауэра** 3](#_Toc115881387)

[**1.4** **Создание базы данных** 4](#_Toc115881388)

[**1.5** **Настройка SQL Сервера** 5](#_Toc115881389)

[**1.6** **Перенос базы данных** 7](#_Toc115881390)

[**1.7** **Бэкап базы и восстановление** 8](#_Toc115881391)

[**1.8** **Добавление базы в кластер** 12](#_Toc115881392)

[**1.9** **Создание базы и добавление в кластер средствами 1С** 13](#_Toc115881393)

[**1.10** **Кластеры и консоли кластеров серверов 1С** 15](#_Toc115881394)

[**1.10.1.1** **Основное** 15](#_Toc115881395)

[**1.10.1.2** **Возможные ошибки , проблемы:** 15](#_Toc115881396)

[**1.11** **Настройка выгрузки Бэкапов при помощи Effector server 3 на сетевой диск Gertruda** 16](#_Toc115881397)

[**1.12** **Отчистка пользовательского кэша 1С на РДП серверах** 17](#_Toc115881398)

[**2.** **Сервер 192.168.1.131 – 1CR** 17](#_Toc115881399)

[ **Основное** 17](#_Toc115881400)

[ **Настройка планировщика Cron** 18](#_Toc115881401)

[ **Настройка Ядра ОС Linux** 18](#_Toc115881402)

[**3.** **Сервер 192.168.1.115 – PGseк** 24](#_Toc115881403)

[ **Основное** 24](#_Toc115881404)

[ **Настройка планировщика Cron** 25](#_Toc115881405)

[ **Работа с Postgres** 25](#_Toc115881406)

[ **Настройка Ядра ОС Linux** 27](#_Toc115881407)

[**4.** **Сервер 192.168. 2.16 – 1C-Br** 33](#_Toc115881408)

[**** **Основное** 33](#_Toc115881409)

[**** **Настройка планировщика Cron** 34](#_Toc115881410)

[**** **Публикация базы** 34](#_Toc115881411)

[**** **Настройка Ядра ОС Linux** 35](#_Toc115881412)

[**5.** **Сервер 192.168. 1.109 – 1C** 41](#_Toc115881413)

[ **Основное** 41](#_Toc115881414)

[ **Настройка Ядра ОС Linux** 41](#_Toc115881415)

[**6.** **Linux , решение типовых задач** 47](#_Toc115881416)

[ **Изменение настроек Ядра ОС Linux. Сделать так, чтобы они не изменялись при перезапуске сервера.** 47](#_Toc115881417)

[ **Добавление репозитария** 48](#_Toc115881418)

[ **Настройка сети** 48](#_Toc115881419)

[ **Sudo доступ без ввода пароля** 49](#_Toc115881420)

[ **Монтирование сетевого диска** 49](#_Toc115881421)

[**7.** **Установка и настройка** 49](#_Toc115881422)

[1. **Установка и настройка Postgres12 под 1C на Astra Linux** 49](#_Toc115881423)

[2. **Установка и настройка 1C-сервера на Astra Linux** 50](#_Toc115881424)

[3. **Установка и настройка Zabbix-сервера + Postgres 14.5 + Apache на Ubuntu Linux 18.04** 52](#_Toc115881425)

[ **Установка Postgres 14.5 на Ubuntu 18.4** 52](#_Toc115881426)

[ **Установка Апач** 52](#_Toc115881427)

[ Настройка файрвола 53](#_Toc115881428)

[ Проверка веб-сервера 54](#_Toc115881429)

[ **Установка Zabbix сервер** 54](#_Toc115881430)

[ **Настройка оповещения в Telegram чат** 55](#_Toc115881431)

[4. **Установка и настройка Ansible на Ubuntu Linux 18.04** 57](#_Toc115881432)

[ **Установка Ansible** 57](#_Toc115881433)

[ **Настройка подключения к хосту по shh key** 58](#_Toc115881434)

[ **Администрирование Postgres-12-1C на Astra Linux, особенности:** 61](#_Toc115881435)

[ **Пример работы с Postgres через Ansible используя плейбуки** 62](#_Toc115881436)

[5. **Установка Git и подключение репозитория с данным гайдом** 64](#_Toc115881437)

[6. **Установка и настройка VPN сервера (OpenVPN) ОС Ubuntu** 65](#_Toc115881438)

[7. **Настройка подключения к удаленному рабочему столу с Windows на Linux Ubuntu** 66](#_Toc115881439)

1. **Сервер 192.168.59.182 – METIS** Основное по работе SQL Server Management Studio (SSMS)
   1. **Добавление пользователей для входа**



Безопасность(Security)-Имена для входа(Logins)-Создать имя для входа(New Logins)

Search – Advanced Найти нужного пользователя в домене , затем задать роль (Server Roles)

Список ролей :



* 1. **Изменение порта и добавление правил в брандмауэра**

Поиск и запуск SQL Server Configuration Manager : Screenshot_1

Здесь отображаются запущенные службы 

А так же протоколы :

Shared Memory для работы через оперативную память локально , дает прирост производительности .

TCP/IP для работы по сети , тут можно посмотреть и задать порт



\*Для данного порта требуется создать правило в Брандмауэр Защитника Windows

\*Поиск и запуск Windows Firewall и добавить правило для данного порта на подключение

* 1. **Создание базы данных**



Initial Size (Начальный размер ) , желательно сразу задать примерный размер базы , чтобы база не была сильно фрагментирована , так же можно задать шаг прироста базы и лимиты



* 1. **Настройка SQL Сервера**



В разделе Memory (память ) нужно обязательно ограничивать максимальную память сервера иначе он заберет всю доступную

****

Во вкладке Connections(соединения) можно настроить количество одновременных соединений и доступ удаленного соединения с сервером



Во вкладке Database Settings можно настроить место хранения данных , журнала и бэкапов . Для повышения производительности рекомендуют использовать разные диски



В данном случае все хранится на одном :

C:\Program Files\Microsoft SQL Server\MSSQL11.SQL\_SERVER\MSSQL\DATA\

C:\Program Files\Microsoft SQL Server\MSSQL11.SQL\_SERVER\MSSQL\Backup

* 1. **Перенос базы данных**

Все действующие базы будут заняты службой SQL сервера и для переноса требуется сначала ее отсоеденить



Tasks(Задачи)-Detach(Отсоединение)



После этого можно переносить фалы в любую другую директорию или даже на другой сервер .

Для того ,чтобы присоединить базу выбираем Attach(Присоеденить)



Выбираем нужный каталог и файл mdf



Автоматически подтягивает и файл журнала транзакций



* 1. **Бэкап базы и восстановление**

Рассмотрим на примере задачи : Требуется сделать бэкап базы NPOL и загрузить его в пустую базу TikhonovTestDB созданную ранее .



Tasks– Back Up



Проверяем ,что база выбрана верно – NPOL , а тип бэкапа – полный Full . Так же смотрим директорию или выбираем ,куда будет выгружен сам бэкап .



Выбираем базу ,в которую будет загружен бэкап (TikhonovTestBD) Tasks-Restore-Database



Во вкладке General выбираем Divace ( расположение бэкапа ) то ,откуда будет загружаться данные в базу . Проверяем ,что база в Destination выбрана верно и переходим во вкладку Files



Здесь внимательно указываем mdf b log файлы базы ,в корую будут выгружаться данные из бэкапа ( TikhonovTestDB.mdf и TikhonovTestDB.log )



Переходим в Options и выставляем пункты, как на скриншоте, затем перепроверяем все и можно приступать к процессу загрузки



* 1. **Добавление базы в кластер**

В консоли кластера добавляем новую базу



Имя желательно должно совпадать с именем базы на сервере , пользователь БД ,должен иметь права администратора



Проверяем , база должна появиться в списке на кластере



* 1. **Создание базы и добавление в кластер средствами 1С**

Через 1С есть возможность создать новую базу данных и сразу добавить ее в кластер серверов 1С и создать новую базу в СУБД



При необходимости можно создать базу по шаблону с определенными конфигурациями.

Необходимые конфигурации можно загрузить с Портала 1С: ИТС https://releases.1c.ru/total



Указываем кластер серверов 1С , имя информационной базы в кластере ( будет отображаться в консоли кластера серверов 1С , рекомендуется ,чтобы совпадало с именем информационной базы) , Сервер баз данных и тип СУБД , пользователь бд должен обладать необходимыми правами ,для создания базы данных на сервере СУБД.



* 1. **Кластеры и консоли кластеров серверов 1С**
     + 1. **Основное**

Используемые порты:

* 1560-1591 - для рабочего процесса
* 1541 - для менеджера кластера
* 1540 - для агента сервера

Кластеры:

* Metis – 192.168.59.182 (консоль)
* 192.168.1.109 – 192.168.1.116 (консоль)
* 192.168.1.131 – 192.168.1.114 (консоль)
* 01URPAPPDEV – 10.1.6.5 (консоль)
  + - 1. **Возможные ошибки , проблемы:**

**Этот хост неизвестен**

Изображение выглядит как текст

Автоматически созданное описание

При попытки подключиться к базе определенного кластера ( в данном случае кластера 192.168.1.109 -1С ) выдает подобное окно с ошибкой.

Следует проверить в C:\Windows\System32\drivers\etc файл hosts , прописан ли там сервер:

192.168.1.109 1C

Подробнее о данной проблеме и решении можно почитать тут : <https://buhexpert8.ru/obuchenie-1s/oshibki-v-1s/etot-host-neizvesten-1s-kak-ispravit.html>

* 1. **Настройка выгрузки Бэкапов при помощи Effector server 3 на сетевой диск Gertruda**

В **Effector server 3** на Метис настроено 6 ежедневных задач:

1. 1CRstor – забирает содержимое \backup на сервере 192.168.1.131 и отправляет на сетевой диск Gertruda - 1cbkp\GZ\Repo1cr , архивы хранятся за последние 7 дней. Предварительно на сервере 192.168.1.131 через планировщик cron скрипт делает бэкап конфигураций в \backup , а затем зачищает содержимое \backup ( после выгрузки на Гертруду) . Данные настройки подробнее будут рассматриваться в разделе по серверу 192.168.1.131 \*
2. CopyBCPGZ - забирает содержимое x:\backupPG на сервере 192.168.1.116 и отправляет на сетевой диск Gertruda - 1cbkp\GZ\Daily , архивы хранятся за последние 7 дней. Предварительно на сервере 192.168.1.115 через планировщик cron скрипт делает бэкап конфигураций в \mnt\backupPG (смонтированный сет.диск 192.168.1.116 ого сервера) , а затем зачищает содержимое \mnt\backupPG ( после выгрузки на Гертруду) . Данные настройки подробнее будут рассматриваться в разделе по серверу 192.168.1.115
3. CopyStorageGZ - забирает содержимое x:\1CStorage на сервере 192.168.1.116 и отправляет на сетевой диск Gertruda - 1cbkp\GZ\Repo , архивы хранятся за последние 7 дней.
4. File\_1C\_Backup – Запущено две задачи по бэкапу баз на Метисе : File\_1C\_Backup\_Typolev\_Mainbase и File\_1C\_Backup\_ ITIL\_PROFF1 , бэкапы хранятся 7 дней на сетевом диске Gertruda - 1cbkp\File\_1C\ Typolev\_Mainbase и 1cbkp\File\_1C\ITIL\_PROFF1 .
5. Repository\_Backup – Бэкап репозитария с Метис раз в неделю ,хранятся за 7 последних недель на сетевом диске Gertruda - 1cbkp\Repositories



* 1. **Отчистка пользовательского кэша 1С на РДП серверах**

Создаем текстовый документ .txt с содержанием :

@FOR /D %%i in ("C:\Users\\*") do (

@FOR /D %%j in ("%%i\Local settings\Application data\1C\1Cv8\????????-????-????-????-????????????") do rd /s /q "%%j"

@FOR /D %%j in ("%%i\AppData\Roaming\1C\1Cv8\????????-????-????-????-????????????") do rd /s /q "%%j"

)

Сохраняем в формате .bat

Запускать от имени администратора

1. **Сервер 192.168.1.131 – 1CR**
   * **Основное**

|  |  |
| --- | --- |
| **Основные директории сервера** | |
| /etc/init.d/ | Хранятся скрипты |
| /backup | Хранятся бэкапы за последние сутки |
| /stor1c | Хранилище конфигураций |
| /var/log/cronlog/ | Логи по отработанным скриптам планировщика cron |
| /var/spool/cron/crontabs | Все настройки cron хранятся тут |

Бэкапы конфигураций раз в день , в 6 50 полностью удаляются, занимают около 81 Гб на диске в директории /backup

|  |  |
| --- | --- |
| **Основные скрипты** | |
| backup.sh | Бэкапы из /store1c в /backup |



* + **Настройка планировщика Cron**



1. В 15:2 каждую субботу отрабатывает команда reboot
2. В 15:0 каждый день отрабатывает скрипт на перезапуск сервера 1С
3. В 25:0 каждый жень отрабатывает скрипт (backup.sh) на бэкап конфигураций из /store1c в /backup
4. Каждый день в 5:50 из директории /backup и удаляет все файлы

Логи по ошибкам находятся в директории /var/log/cronlog/ и называются в соответствии с отработанным скриптом или действием :

reboot.log – ребут

restart1c.log – проверка скрипта на рестарт сервера 1с

backup.log – бэкапы конфигураций раз в день

backuprm.log – удаление всех бэкапов конфигураций

* + **Настройка Ядра ОС Linux**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  |  | | --- | --- | --- | --- | | Наименование | Значение | Откуда | Описание | | Количество процессоров | 12 | cat /proc/cpuinfo | grep processor | wc -l | Количество процессоров, показанное в /proc/cpuinfo, может не соответствовать реальному количеству ядер процессора. Например, процессор с 2 ядрами и гиперпоточностью будет показан как процессор с 4 ядрами. | | Идентификатор ядра | 0,1,2,3,4,5 - (всего 6) | cat /proc/cpuinfo | grep 'core id' | Есть 6 разных идентификатора ядра. Это указывает на то, что существует 4 реальных ядра. | | Данную информацию можно также посомтреть при помощи команды - lscpu. | | | | | Диапазоны портов |  | /proc/ioports | В этом файле содержаться все зарегистрированные диапазоны портов, а также устройства, которые их используют | | MemTotal : доступный объем оперативной памяти | 69066052 kB = 69 Gb | /proc/meminfo | физическая память за вычетом нескольких зарезервированных битов и бинарного кода ядра | | Команда для вывода информации об использовании оперативной памяти - free | | | | | Модуль ядра, или его часто называют драйвером, — это фрагмент кода, расширяющий функциональные возможности ядра.Запустите lsmod из командной строки, чтобы узнать, какие модули ядра загружены в данный момент. | | | | | Точки монтирования |  | /proc/mounts | В этом файле перечислены все точки монтирования и все подключенные файловые системы | | Точную версию ядра | Linux version 4.15.3-1-generic (root@build) (gcc version 6.3.0 20170516 (Debian 6.3.0-18+deb9u1)) #astra21 SMP Thu Aug 22 12:16:21 UTC 201$ | /proc/version | Точная версию ядра, компилятора, и в некоторых случаях, даже дистрибутива | | /proc/pid/ - Файловая система proc состоит не только из файлов, но здесь есть и папки. Больше всего здесь папок с номерами вместо имен. Каждый этот номер означает PID процесса, а эта папка содержит информацию о каждом запущенном в системе процессе. Когда процесс заканчивается, его каталог исчезает из системы | | Основные файлы: cmdline - содержит команду с помощью которой был запущен процесс, а также переданные ей параметры cwd - символическая ссылка на текущую рабочую директорию процесса exe - ссылка на исполняемый файл root - ссылка на папку суперпользователя environ - переменные окружения, доступные для процесса fd - содержит файловые дескрипторы, файлы и устройства, которые использует процесс maps, statm, и mem - информация о памяти процесса stat, status - состояние процесса | С помощью этих файлов вы можете составлять различные скрипты. Например если вы хотите уничтожить все зомби процессы, то вы можете сканировать все директории на наличие Z в файле status. Так же само можно проверить запущена ли нужная вам программа просмотрев все cmdline. | | /proc/sys/ -Эта папка в proc linux не только предоставляет информацию о системе, но и позволяет изменять различные параметры ядра на лету, а также включать дополнительные функции. | | Основные подкаталоги в этой папке: debug - содержит отладочную информацию, она будет вам полезна если вы разработчик ядра dev - параметры различных устройств, подключенных к системе fs - вся информация о файловой системе kernel - позволяет напрямую настраивать ядро net - настройка разных параметров сети vm - взаимодействие с подсистемой vm | Чтобы посмотреть можно ли записывать в файлы используйте команду:   ls -ld /proc/sys/  Если у файла есть флаг W, значит в него можно записывать данные. | | Настройки Ядра | Значение | Путь | Команда | | Если установлено 1, разрешает выполнение 32 битных программ в 64 битной системе | 1 (по умолчанию) | /proc/sys/abi/vsyscall32 | sysctl -w abi.vsyscall32=1 | | При возникновении ошибки в ядре выводить значения регистров процессора и стек вызовов процедур | 1 (по умолчанию) | /proc/sys/debug/exception-trace | sysctl -w debug.exception-trace=1 | | Максимальная частота генерации прерываний от системного таймера High Precision Event Timer (HPET), который пришел на замену таймеру реального времени RTC. | 64 (по умолчанию) | /proc/sys/dev/hpet/max-user-freq | sysctl -w dev.hpet.max-user-freq=64 | | Количество асинхронных операций ввода и вывода в вашей файловой системе | 0 | /proc/sys/fs/aio-nr | sysctl -w fs.aio-nr=0 | | Максимальное количество дескрипторов файлов, которые может создать и обрабатывать ядро. Если вы часто получаете сообщения об ошибке из-за невозможности создать дескриптор файла увеличьте этот лимит. По умолчанию установлено значение 10 % от вашей оперативной памяти. | 6887700 = 6 Gb | /proc/sys/fs/file-max | sysctl -w fs.file-max=6887700 | | Подсистема ядра inotify позволяет следить за изменениями в файловой системе. Этот параметр устанавливает максимальное количество событий, которые могут находиться в очереди, перед тем как их обработает программа. | 16384 | /proc/sys/fs/inotify/max\_queued\_events | sysctl -w fs.inotify.max\_queued\_events=16384 | | Позволяет установить доменное имя NIS (Network Internet Services) и YP (Yellow Pages). Но не путайте это доменное имя с DNS, это совсем разные вещи. | (none) | /proc/sys/kernel/domainname | - | | Имя вашего компьютера. Это самый простой способ изменить имя компьютера прямо сейчас, без перезагрузки. | 1CR | /proc/sys/kernel/hostname | sysctl -w kernel.hostname=Pgser | | Позволяет отключить загрузку модулей ядра. | 0 | /proc/sys/kernel/modules\_disabled | sysctl -w kernel.modules\_disabled=0 | | Рандомизация адресного пространства это функция увеличивающая безопасность вашей системы. Она защищает от атак на переполнение буфера. По умолчанию включена. | 2 - полностью автоматизировано | /proc/sys/kernel/randomize\_va\_space | sysctl -w kernel.randomize\_va\_space=2 | | Позволяет включить или отключить управление ядром с помощью SysRQ | 0 - отключить все функции | /proc/sys/kernel/sysrq | sysctl -w kernel.sysrq=0 | | Максимальное количество запущенных потоков для процессов. | 539279 | /proc/sys/kernel/threads-max | sysctl -w kernel.threads-max=539279 | | Если включено, ядро будет игнорировать все icmp запросы. Рекомендуется для защиты от DDOS атак. | 0 | /proc/sys/net/ipv4/icmp\_echo\_ignore\_all | sysctl -w net.ipv4.icmp\_echo\_ignore\_all=0 | | Так же, как и в предыдущем варианте, только игнорироваться будут только широковещательные icmp запросы | 1 | /proc/sys/net/ipv4/icmp\_echo\_ignore\_broadcasts | sysctl -w net.ipv4.icmp\_echo\_ignore\_broadcasts=1 | | Максимальное количество узлов, через которые может пройти отправленный пакет перед тем, как достигнет цели. | 64 | /proc/sys/net/ipv4/ip\_default\_ttl | sysctl -w net.ipv4.ip\_default\_ttl=64 | | Разрешить проходящие пакеты через этот компьютер. Обычно такая настройка параметров ядра Linux нужна для роутеров | 0 | /proc/sys/net/ipv4/ip\_forward | sysctl -w net.ipv4.ip\_forward=0 | | Диапазон локальных портов, которые могут быть использованы вашими программами | 32768 60999 (по умолчанию) | /proc/sys/net/ipv4/ip\_local\_port\_range | sysctl -w net.ipv4.ip\_local\_port\_ranged= "32768 60999" | | Установите 1 чтобы защитить компьютер от атаки TCP TimeWait. | 0 | /proc/sys/net/ipv4/tcp\_rfc1337 | sysctl -w net.ipv4.tcp\_rfc1337=0 | | Таймаут ожидания завершения соединения после отправки пакета FIN. Рекомендовано 15. | 60 | /proc/sys/net/ipv4/tcp\_fin\_timeout | sysctl -w net.ipv4.tcp\_fin\_timeout=60 | | Поддерживать соединение активным определенное время, например, 300 секунд. По истечении этого времени TCP соединение будет разорвано. | 7200 | /proc/sys/net/ipv4/tcp\_keepalive\_time | sysctl -w net.ipv4.tcp\_keepalive\_time=7200 | | Указывает размер по умолчанию буфера для сокета получения данных по сети | 212992 | /proc/sys/net/core/rmem\_default | sysctl -w net.**core.rmem\_default**e=212992 | | Максимальный размер буфера сокета для получения данных по сети | 212992 | /proc/sys/net/core/rmem\_max | sysctl -w net.core.rmem\_max=212992 | | Размер сокета по умолчанию для отправки данных по сети. | 212992 | /proc/sys/net/core/wmem\_default | sysctl -w net.core.wmem\_defaulte=212992 | | Максимальный размер буфера сокета для отправки данных по сети. | 212992 | /proc/sys/net/core/wmem\_max | sysctl -w net.core.wmem\_max=212992 | | Количество памяти, доступной для работы TCP. | 4096 87380 6291456 (минимум, по умолчанию, максимум) | /proc/sys/net/ipv4/tcp\_rmem |  | | Указывает процент от общей системной памяти, когда фоновый демон pdflush записи данных начнет переписывать кешированные данные на диск | 10 (Рекомендуется 3) | /proc/sys/vm/dirty\_background\_ratio | sysctl -w vm.dirty\_background\_ratio =10 | | Указывает сколько общей оперативной памяти должно быть занято, чтобы процесс, который ведет запись данных на диск инициировал запись кэшированных данных непосредственно на жесткий диск | 20 (Рекомендуется 10) | /proc/sys/vm/dirty\_ratio | sysctl -w vm.dirty \_ratio =20 | | Устанавливает процент свободной памяти, по достижении которого данные начинают переноситься на swap раздел, для систем с большим количеством памяти рекомендовано значение 10. | 60 (Рекомендуется 10) | /proc/sys/vm/swappiness | sysctl -w vm. swappiness =60 | | Определяет условия разрешения и отказа запросов больших объемов памяти. Возможные значения: 0 (по умолчанию) — ядро использует эвристический алгоритм для расчета перерасхода памяти, принимая во внимание объем доступной памяти и число неверных запросов.  1 — ядро не обрабатывает перерасход памяти. При этом вероятность превышения нагрузки на память возрастает, но в то же время увеличивается производительность задач, активно использующих память. 2 — отказ обработки запросов, запрашивающих память, размер которой превышает суммарный размер памяти пространства подкачки и ОЗУ в соответствии с overcommit\_ratio (По умолчанию равен 50). | 0 (Рекомендуется 2) | /proc/sys/vm/overcommit\_memory | sysctl -w vm. overcommit\_memory =2 | | **Так же для СУБД сервера репомендуется в /etc/systemd/logind.conf указать RemoveIPC=no (RemoveIPC в systemd: отключить высвобождение памяти ОС для УЗ) на данный момент RemoveIPC=yes** | | | | |

1. **Сервер 192.168.1.115 – PGseк**

* **Основное**

|  |  |
| --- | --- |
| **Основные директории сервера** | |
| /usr/local/bin/ | Хранятся скрипты |
| /mnt/backupPG/ | Хранятся бэкапы за последние сутки |
| /var/log/cronlog/  /var/log/cronlog/errolog/ | Логи по отработанным скриптам планировщика cron (Ход выполнения/ошибки) |
| /var/spool/cron/crontabs | Все настройки cron хранятся тут |

Состояние мест хранения бэкапов

Бэкапы баз раз в день , хранятся за последние сутки , занимают 164Гб на сетевом диске размером 200 Гб смонтированным в директорию /mnt/backupPG/



|  |  |
| --- | --- |
| **Основные скрипты** | |
| backup-sql\_new.sh | Бэкапы баз в /mnt/backupPG/ |
| vakuupdb.sh | Ваккум баз и реиндексация всех баз |
| vacuumin131new.sh | Ваккум баз и реиндексация баз кластера 1.131 |

* **Настройка планировщика Cron**



1. В 0:20 каждый день отрабатывает скрипт на бэкап всех баз (backup-sql\_new.sh) в директорию /mnt/backupPG/
2. В 2:15 каждое воскресенье отрабатывает скрипт на ваккум и переиндексацию баз (vakuupdb.sh)
3. В 2:15 каждый четверг отрабатывает скрипт на ваккум и переиндексацию баз кластера 131 (vacuumin131new.sh)
4. Каждый день в 0:10 проверяет директорию /mnt/packupPG/ и удаляет все файлы

Логи по ошибкам находятся в директории /var/log/cronlog/errolog/ и называются в соответствии с отработанным скриптом или действием :

vacuumin131new.log – проверка скрипта на ваккум и реиндексацию наличие ошибок

vakuupdb.log – проверка скрипта на ваккум и реиндексацию наличие ошибок

backup-sql\_new.log – бэкапы раз в день

Логи по ходу выполнения скриптов находятся в директории /var/log/cronlog/

vacuumin131new.log – проверка выполнения скрипта реиндексации и ваккума

* **Работа с Postgres**

Чтобы узнать, какая версия PostgreSQL работает в вашей системе, вызовите команду postgres с параметром —version или  -V:

**postgres --version**

Версию клиентской утилиты psql в PostgreSQL  можно найти с помощью следующей команды:

**psql --version**

**Сервис Postgres: postgrespro-ent-12.service**

* Вход в интерактивный режим на 1.115 (Указание местоположения сокета)

**sudo -u postgres psql**

Если выдает подобную ошибку:



Требуется указать путь к Unix-сокету:

**sudo -u postgres psql --host=/tmp**

Для исправления данной ошибки ,была создана ссылка в /var/run/postgresql/ в /tmp/ ,где лежит наш сокет:

**ln -s /tmp/.s.PGSQL.5432 /var/run/postgresql/.s.PGSQL.5432**

Исправление ошибки в несовпадении версии сервера и psql **:**

**find / -name psql -type f 2>/dev/null**

**ln -s /opt/pgpro/ent-12/bin/psql /usr/bin/psql**

Пересоздание ссылки в /usr/bin/

* Создал скрипт в /home/itihonov/

skbackup.sh

Состав:

#!/bin/sh

# Устанавливаем дату

DATA=`date +"%Y-%m-%d\_%H-%M"`

# Бэкапим базу данных

/opt/pgpro/ent-12/bin/pg\_dump -U postgres GZ\_base67 > /mnt/backupPG/$DATA-GZ\_base67.dump

* Даю права на исполнение :

chmod 744 skbackup.sh

* Исполнение: /home/itihonov/skbackup.sh
* Заливаем бэкап новую базу :

Sudo su postgres psql --host=/tmp gz\_base677 < /mnt/backupPG/2022-05-19\_13-36-GZ\_base67.dump

* Проверка размера базы , размер копии базы может не совпадать с оригинальной , чаще всего он меньше .

SELECT pg\_size\_pretty( pg\_database\_size( 'gz\_base677' ) );

* **Настройка Ядра ОС Linux**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  |  | | --- | --- | --- | --- | | Наименование | Значение | Откуда | Описание | | Количество процессоров | 16 | cat /proc/cpuinfo | grep processor | wc -l | Количество процессоров, показанное в /proc/cpuinfo, может не соответствовать реальному количеству ядер процессора. Например, процессор с 2 ядрами и гиперпоточностью будет показан как процессор с 4 ядрами. | | Идентификатор ядра | 0,1,2,3 - (всего 4) | cat /proc/cpuinfo | grep 'core id' | Есть 4 разных идентификатора ядра. Это указывает на то, что существует 4 реальных ядра. | | Данную информацию можно также посомтреть при помощи команды - lscpu. | | | | | Диапазоны портов |  | /proc/ioports | В этом файле содержаться все зарегистрированные диапазоны портов, а также устройства, которые их используют | | MemTotal : доступный объем оперативной памяти | 72161300 kB = 72 Gb | /proc/meminfo | физическая память за вычетом нескольких зарезервированных битов и бинарного кода ядра | | Команда для вывода информации об использовании оперативной памяти - free | | | | | Модуль ядра, или его часто называют драйвером, — это фрагмент кода, расширяющий функциональные возможности ядра.Запустите lsmod из командной строки, чтобы узнать, какие модули ядра загружены в данный момент. | | | | | Точки монтирования |  | /proc/mounts | В этом файле перечислены все точки монтирования и все подключенные файловые системы | | Точную версию ядра | Linux version 4.15.3-3-generic (builder@build) (gcc version 6.3.0 20170516 (Debian 6.3.0-18+deb9u1)) #astra25+ci7 SMP Wed Oct 28 03:38:36 | /proc/version | Точная версию ядра, компилятора, и в некоторых случаях, даже дистрибутива | | /proc/pid/ - Файловая система proc состоит не только из файлов, но здесь есть и папки. Больше всего здесь папок с номерами вместо имен. Каждый этот номер означает PID процесса, а эта папка содержит информацию о каждом запущенном в системе процессе. Когда процесс заканчивается, его каталог исчезает из системы | | Основные файлы: cmdline - содержит команду с помощью которой был запущен процесс, а также переданные ей параметры cwd - символическая ссылка на текущую рабочую директорию процесса exe - ссылка на исполняемый файл root - ссылка на папку суперпользователя environ - переменные окружения, доступные для процесса fd - содержит файловые дескрипторы, файлы и устройства, которые использует процесс maps, statm, и mem - информация о памяти процесса stat, status - состояние процесса | С помощью этих файлов вы можете составлять различные скрипты. Например если вы хотите уничтожить все зомби процессы, то вы можете сканировать все директории на наличие Z в файле status. Так же само можно проверить запущена ли нужная вам программа просмотрев все cmdline. | | /proc/sys/ -Эта папка в proc linux не только предоставляет информацию о системе, но и позволяет изменять различные параметры ядра на лету, а также включать дополнительные функции. | | Основные подкаталоги в этой папке: debug - содержит отладочную информацию, она будет вам полезна если вы разработчик ядра dev - параметры различных устройств, подключенных к системе fs - вся информация о файловой системе kernel - позволяет напрямую настраивать ядро net - настройка разных параметров сети vm - взаимодействие с подсистемой vm | Чтобы посмотреть можно ли записывать в файлы используйте команду:   ls -ld /proc/sys/  Если у файла есть флаг W, значит в него можно записывать данные. | | Настройки Ядра | Значение | Путь | Команда | | Если установлено 1, разрешает выполнение 32 битных программ в 64 битной системе | 1 (по умолчанию) | /proc/sys/abi/vsyscall32 | sysctl -w abi.vsyscall32=1 | | При возникновении ошибки в ядре выводить значения регистров процессора и стек вызовов процедур | 1 (по умолчанию) | /proc/sys/debug/exception-trace | sysctl -w debug.exception-trace=1 | | Максимальная частота генерации прерываний от системного таймера High Precision Event Timer (HPET), который пришел на замену таймеру реального времени RTC. | 64 (по умолчанию) | /proc/sys/dev/hpet/max-user-freq | sysctl -w dev.hpet.max-user-freq=64 | | Количество асинхронных операций ввода и вывода в вашей файловой системе | 0 | /proc/sys/fs/aio-nr | sysctl -w fs.aio-nr=0 | | Максимальное количество дескрипторов файлов, которые может создать и обрабатывать ядро. Если вы часто получаете сообщения об ошибке из-за невозможности создать дескриптор файла увеличьте этот лимит. По умолчанию установлено значение 10 % от вашей оперативной памяти. | 7197103 = 7Gb | /proc/sys/fs/file-max | sysctl -w fs.file-max=7197103 | | Подсистема ядра inotify позволяет следить за изменениями в файловой системе. Этот параметр устанавливает максимальное количество событий, которые могут находиться в очереди, перед тем как их обработает программа. | 16384 | /proc/sys/fs/inotify/max\_queued\_events | sysctl -w fs.inotify.max\_queued\_events=16384 | | Позволяет установить доменное имя NIS (Network Internet Services) и YP (Yellow Pages). Но не путайте это доменное имя с DNS, это совсем разные вещи. | (none) | /proc/sys/kernel/domainname | - | | Имя вашего компьютера. Это самый простой способ изменить имя компьютера прямо сейчас, без перезагрузки. | PGser | /proc/sys/kernel/hostname | sysctl -w kernel.hostname=Pgser | | Позволяет отключить загрузку модулей ядра. | 0 | /proc/sys/kernel/modules\_disabled | sysctl -w kernel.modules\_disabled=0 | | Рандомизация адресного пространства это функция увеличивающая безопасность вашей системы. Она защищает от атак на переполнение буфера. По умолчанию включена. | 2 - полностью автоматизировано | /proc/sys/kernel/randomize\_va\_space | sysctl -w kernel.randomize\_va\_space=2 | | Позволяет включить или отключить управление ядром с помощью SysRQ | 0 - отключить все функции | /proc/sys/kernel/sysrq | sysctl -w kernel.sysrq=0 | | Максимальное количество запущенных потоков для процессов. | 563454 | /proc/sys/kernel/threads-max | sysctl -w kernel.threads-max=563454 | | Если включено, ядро будет игнорировать все icmp запросы. Рекомендуется для защиты от DDOS атак. | 0 | /proc/sys/net/ipv4/icmp\_echo\_ignore\_all | sysctl -w net.ipv4.icmp\_echo\_ignore\_all=0 | | Так же, как и в предыдущем варианте, только игнорироваться будут только широковещательные icmp запросы | 1 | /proc/sys/net/ipv4/icmp\_echo\_ignore\_broadcasts | sysctl -w net.ipv4.icmp\_echo\_ignore\_broadcasts=1 | | Максимальное количество узлов, через которые может пройти отправленный пакет перед тем, как достигнет цели. | 64 | /proc/sys/net/ipv4/ip\_default\_ttl | sysctl -w net.ipv4.ip\_default\_ttl=64 | | Разрешить проходящие пакеты через этот компьютер. Обычно такая настройка параметров ядра Linux нужна для роутеров | 0 | /proc/sys/net/ipv4/ip\_forward | sysctl -w net.ipv4.ip\_forward=0 | | Диапазон локальных портов, которые могут быть использованы вашими программами | 32768 60999 (по умолчанию) | /proc/sys/net/ipv4/ip\_local\_port\_range | sysctl -w net.ipv4.ip\_local\_port\_ranged= "32768 60999" | | Установите 1 чтобы защитить компьютер от атаки TCP TimeWait. | 0 | /proc/sys/net/ipv4/tcp\_rfc1337 | sysctl -w net.ipv4.tcp\_rfc1337=0 | | Таймаут ожидания завершения соединения после отправки пакета FIN. Рекомендовано 15. | 60 | /proc/sys/net/ipv4/tcp\_fin\_timeout | sysctl -w net.ipv4.tcp\_fin\_timeout=60 | | Поддерживать соединение активным определенное время, например, 300 секунд. По истечении этого времени TCP соединение будет разорвано. | 7200 | /proc/sys/net/ipv4/tcp\_keepalive\_time | sysctl -w net.ipv4.tcp\_keepalive\_time=7200 | | Указывает размер по умолчанию буфера для сокета получения данных по сети | 212992 | /proc/sys/net/core/rmem\_default | sysctl -w net.**core.rmem\_default**e=212992 | | Максимальный размер буфера сокета для получения данных по сети | 212992 | /proc/sys/net/core/rmem\_max | sysctl -w net.core.rmem\_max=212992 | | Размер сокета по умолчанию для отправки данных по сети. | 212992 | /proc/sys/net/core/wmem\_default | sysctl -w net.core.wmem\_defaulte=212992 | | Максимальный размер буфера сокета для отправки данных по сети. | 212992 | /proc/sys/net/core/wmem\_max | sysctl -w net.core.wmem\_max=212992 | | Количество памяти, доступной для работы TCP. | 4096 131072 6291456 (минимум, по умолчанию, максимум) | /proc/sys/net/ipv4/tcp\_rmem |  | | Указывает процент от общей системной памяти, когда фоновый демон pdflush записи данных начнет переписывать кешированные данные на диск | 10 (Рекомендуется 3) | /proc/sys/vm/dirty\_background\_ratio | sysctl -w vm.dirty\_background\_ratio =10 | | Указывает сколько общей оперативной памяти должно быть занято, чтобы процесс, который ведет запись данных на диск инициировал запись кэшированных данных непосредственно на жесткий диск | 20 (Рекомендуется 10) | /proc/sys/vm/dirty\_ratio | sysctl -w vm.dirty \_ratio =20 | | Устанавливает процент свободной памяти, по достижении которого данные начинают переноситься на swap раздел, для систем с большим количеством памяти рекомендовано значение 10. | 60 (Рекомендуется 10) | /proc/sys/vm/swappiness | sysctl -w vm. swappiness =60 | | Определяет условия разрешения и отказа запросов больших объемов памяти. Возможные значения: 0 (по умолчанию) — ядро использует эвристический алгоритм для расчета перерасхода памяти, принимая во внимание объем доступной памяти и число неверных запросов.  1 — ядро не обрабатывает перерасход памяти. При этом вероятность превышения нагрузки на память возрастает, но в то же время увеличивается производительность задач, активно использующих память. 2 — отказ обработки запросов, запрашивающих память, размер которой превышает суммарный размер памяти пространства подкачки и ОЗУ в соответствии с overcommit\_ratio (По умолчанию равен 50). | 0 (Рекомендуется 2) | /proc/sys/vm/overcommit\_memory | sysctl -w vm. overcommit\_memory =2 | | **Так же для СУБД сервера репомендуется в /etc/systemd/logind.conf указать RemoveIPC=no (RemoveIPC в systemd: отключить высвобождение памяти ОС для УЗ) на данный момент RemoveIPC=yes** | | | | |

1. **Сервер 192.168. 2.16 – 1C-Br**

* **Основное**

|  |  |
| --- | --- |
| **Основные директории сервера** | |
| /usr/local/bin/ | Хранятся скрипты |
| /mnt/backupsql/monthly/ | Хранятся бэкапы сроком до 183 дней |
| /nfs/backup/ | Хранятся бэкапы сроком до 7 дней |
| /var/log/cronlog/ | Логи по отработанным скриптам планировщика cron |
| /var/spool/cron/crontabs | Все настройки cron хранятся тут |

Состояние мест хранения бэкапов на 08.06.22

Бэкапы баз 2 раза в месяц , 17ого и 28ого числа , хранятся не старше 184 дней ,занимаю 75 Гб из 148 Гб (54%) , диск смонтирован в директорию /mnt/backupsql/



Бэкапы баз раз в день , хранятся не старше 7 дней , занимают 197 Гб на сетевом диске размером 5,5 Тб смонтированным в директорию /nfs/backup/

|  |  |
| --- | --- |
| **Основные скрипты** | |
| backup-sql.sh | Бэкапы баз в /nfs/backup/ |
| backup-monthly-sql.sh | Бэкапы баз в /mnt/backupsql/monthly/ |
| maintenance-sql.sh | Ваккум баз и реиндексация |

* **Настройка планировщика Cron**

****

1. В 22:15 каждый день отрабатывает скрипт на бэкап всех баз (backup-sql.sh) в директорию /nfs/backup/
2. В 0:15 каждое воскресенье отрабатывает скрипт на ваккум и переиндексацию баз (maintenance-sql.sh)
3. В 1:20 происходит рестарт сервера 1с
4. В 3:30 , 17ого числа каждого месяца отрабатывает скрипт (backup-monthly-sql.sh) на бэкап баз в директорию /mnt/backupsql/monthly/
5. В 3:30 , 28ого числа каждого месяца отрабатывает скрипт (backup-monthly-sql.sh) на бэкап баз в директорию /mnt/backupsql/monthly/
6. В 4:20 проверка файлов в директории /nfs/backup/ и удаление всех старше 7 дней
7. В 2:20 проверка файлов в директории /mnt/backupsql/monthly/ и удаление всех старше 183 дней

Логи по ошибкам находятся в директории /var/log/cronlog/ и называются в соответствии с отработанным скриптом или действием :

restartsrv1c.log - рестарт сервера 1с

delmonbackup.log – удаление бэкапов старше 183 дней

delbackup.log – удаление бэкапов старше 7 дней

backup-sql.log – бэкапы раз в день

backup-monthly-sql1.log – бэкапы 17ого числа

backup-monthly-sql2.log – бэкапы 28ого числа

* **Публикация базы**

Гайд - <https://its.1c.ru/db/metod8dev/content/5979/hdoc>

Пример на сервере 1С-Br 192.168.2.16

* Зайти в /opt/1cv8/x86\_64/8.3.18.1363/ - cd /opt/1cv8/x86\_64/8.3.18.1363/
* Публикуем базу

./webinst -publish -apache24 -wsdir ZUP\_31\_AFS\_test1 -dir /var/www/html/1c/ZUP\_31\_AFS\_test1/ -connstr "Srvr=192.168.2.16;Ref=ZUP\_31\_AFS\_test1;" -confPath /etc/apache2/apache2.conf

Если vrd файл существует то :

./webinst -publish -apache24 -wsdir ZUP\_31\_AFS\_test1 -descriptor /var/www/html/1c/ZUP\_31\_AFS\_test1/default.vrd -dir /var/www/html/1c/ZUP\_31\_AFS\_test1/ -connstr "Srvr=192.168.2.16;Ref=ZUP\_31\_AFS\_test1;" -confPath /etc/apache2/apache2.conf

* Перезагрузка апача и проверка

Sudo service apache2 restart

service apache2 status

Дополнительно :

/var/www/html/1c/ -здесь vrd от опубликованных баз

/etc/apache2/apache2.conf –проверка конфигураций апача ,все опубликованные базы тут прописаны

* **Настройка Ядра ОС Linux**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  |  | | --- | --- | --- | --- | | Наименование | Значение | Откуда | Описание | | Количество процессоров | 4 | cat /proc/cpuinfo | grep processor | wc -l | Количество процессоров, показанное в /proc/cpuinfo, может не соответствовать реальному количеству ядер процессора. Например, процессор с 2 ядрами и гиперпоточностью будет показан как процессор с 4 ядрами. | | Идентификатор ядра | 0,1,2,3 - (всего 4) | cat /proc/cpuinfo | grep 'core id' | Есть 4 разных идентификатора ядра. Это указывает на то, что существует 4 реальных ядра. | | Данную информацию можно также посомтреть при помощи команды - lscpu. | | | | | Диапазоны портов |  | /proc/ioports | В этом файле содержаться все зарегистрированные диапазоны портов, а также устройства, которые их используют | | MemTotal : доступный объем оперативной памяти | 12264748 kB = 12 Gb | /proc/meminfo | физическая память за вычетом нескольких зарезервированных битов и бинарного кода ядра | | Команда для вывода информации об использовании оперативной памяти - free | | | | | Модуль ядра, или его часто называют драйвером, — это фрагмент кода, расширяющий функциональные возможности ядра.Запустите lsmod из командной строки, чтобы узнать, какие модули ядра загружены в данный момент. | | | | | Точки монтирования |  | /proc/mounts | В этом файле перечислены все точки монтирования и все подключенные файловые системы | | Точную версию ядра | Linux version 5.4.0-96-generic (buildd@lgw01-amd64-051) (gcc version 9.3.0 (Ubuntu 9.3.0-17ubuntu1~20.04)) #109-Ubuntu SMP Wed Jan 12 16:4> | /proc/version | Точная версию ядра, компилятора, и в некоторых случаях, даже дистрибутива | | /proc/pid/ - Файловая система proc состоит не только из файлов, но здесь есть и папки. Больше всего здесь папок с номерами вместо имен. Каждый этот номер означает PID процесса, а эта папка содержит информацию о каждом запущенном в системе процессе. Когда процесс заканчивается, его каталог исчезает из системы | | Основные файлы: cmdline - содержит команду с помощью которой был запущен процесс, а также переданные ей параметры cwd - символическая ссылка на текущую рабочую директорию процесса exe - ссылка на исполняемый файл root - ссылка на папку суперпользователя environ - переменные окружения, доступные для процесса fd - содержит файловые дескрипторы, файлы и устройства, которые использует процесс maps, statm, и mem - информация о памяти процесса stat, status - состояние процесса | С помощью этих файлов вы можете составлять различные скрипты. Например если вы хотите уничтожить все зомби процессы, то вы можете сканировать все директории на наличие Z в файле status. Так же само можно проверить запущена ли нужная вам программа просмотрев все cmdline. | | /proc/sys/ -Эта папка в proc linux не только предоставляет информацию о системе, но и позволяет изменять различные параметры ядра на лету, а также включать дополнительные функции. | | Основные подкаталоги в этой папке: debug - содержит отладочную информацию, она будет вам полезна если вы разработчик ядра dev - параметры различных устройств, подключенных к системе fs - вся информация о файловой системе kernel - позволяет напрямую настраивать ядро net - настройка разных параметров сети vm - взаимодействие с подсистемой vm | Чтобы посмотреть можно ли записывать в файлы используйте команду:   ls -ld /proc/sys/  Если у файла есть флаг W, значит в него можно записывать данные. | | Настройки Ядра | Значение | Путь | Команда | | Если установлено 1, разрешает выполнение 32 битных программ в 64 битной системе | 1 (по умолчанию) | /proc/sys/abi/vsyscall32 | sysctl -w abi.vsyscall32=1 | | При возникновении ошибки в ядре выводить значения регистров процессора и стек вызовов процедур | 1 (по умолчанию) | /proc/sys/debug/exception-trace | sysctl -w debug.exception-trace=1 | | Максимальная частота генерации прерываний от системного таймера High Precision Event Timer (HPET), который пришел на замену таймеру реального времени RTC. | 64 (по умолчанию) | /proc/sys/dev/hpet/max-user-freq | sysctl -w dev.hpet.max-user-freq=64 | | Количество асинхронных операций ввода и вывода в вашей файловой системе | 0 | /proc/sys/fs/aio-nr | sysctl -w fs.aio-nr=0 | | Максимальное количество дескрипторов файлов, которые может создать и обрабатывать ядро. Если вы часто получаете сообщения об ошибке из-за невозможности создать дескриптор файла увеличьте этот лимит. По умолчанию установлено значение 10 % от вашей оперативной памяти. | 9223372036854775807 (по умолчанию в данной ОС) | /proc/sys/fs/file-max | sysctl -w fs.file-max=9223372036854775807 | | Подсистема ядра inotify позволяет следить за изменениями в файловой системе. Этот параметр устанавливает максимальное количество событий, которые могут находиться в очереди, перед тем как их обработает программа. | 16384 | /proc/sys/fs/inotify/max\_queued\_events | sysctl -w fs.inotify.max\_queued\_events=16384 | | Позволяет установить доменное имя NIS (Network Internet Services) и YP (Yellow Pages). Но не путайте это доменное имя с DNS, это совсем разные вещи. | (none) | /proc/sys/kernel/domainname | - | | Имя вашего компьютера. Это самый простой способ изменить имя компьютера прямо сейчас, без перезагрузки. | br-1c | /proc/sys/kernel/hostname | sysctl -w kernel.hostname=br-1c | | Позволяет отключить загрузку модулей ядра. | 0 | /proc/sys/kernel/modules\_disabled | sysctl -w kernel.modules\_disabled=0 | | Рандомизация адресного пространства это функция увеличивающая безопасность вашей системы. Она защищает от атак на переполнение буфера. По умолчанию включена. | 2 - полностью автоматизировано | /proc/sys/kernel/randomize\_va\_space | sysctl -w kernel.randomize\_va\_space=2 | | Позволяет включить или отключить управление ядром с помощью SysRQ | 176 (Оно получается из суммы 128 (что позволяет перезагружать и отключать питание) + 32 (возможность перемонтировать файловые системы в режиме только для чтения) + 16, что включает команду синхронизации. ) | /proc/sys/kernel/sysrq | sysctl -w kernel.sysrq=176 | | Максимальное количество запущенных потоков для процессов. | 95118 | /proc/sys/kernel/threads-max | sysctl -w kernel.threads-max=95118 | | Если включено, ядро будет игнорировать все icmp запросы. Рекомендуется для защиты от DDOS атак. | 0 | /proc/sys/net/ipv4/icmp\_echo\_ignore\_all | sysctl -w net.ipv4.icmp\_echo\_ignore\_all=0 | | Так же, как и в предыдущем варианте, только игнорироваться будут только широковещательные icmp запросы | 1 | /proc/sys/net/ipv4/icmp\_echo\_ignore\_broadcasts | sysctl -w net.ipv4.icmp\_echo\_ignore\_broadcasts=1 | | Максимальное количество узлов, через которые может пройти отправленный пакет перед тем, как достигнет цели. | 64 | /proc/sys/net/ipv4/ip\_default\_ttl | sysctl -w net.ipv4.ip\_default\_ttl=64 | | Разрешить проходящие пакеты через этот компьютер. Обычно такая настройка параметров ядра Linux нужна для роутеров | 0 | /proc/sys/net/ipv4/ip\_forward | sysctl -w net.ipv4.ip\_forward=0 | | Диапазон локальных портов, которые могут быть использованы вашими программами | 32768 60999 (по умолчанию) | /proc/sys/net/ipv4/ip\_local\_port\_range | sysctl -w net.ipv4.ip\_local\_port\_ranged= "32768 60999" | | Установите 1 чтобы защитить компьютер от атаки TCP TimeWait. | 0 | /proc/sys/net/ipv4/tcp\_rfc1337 | sysctl -w net.ipv4.tcp\_rfc1337=0 | | Таймаут ожидания завершения соединения после отправки пакета FIN. Рекомендовано 15. | 60 | /proc/sys/net/ipv4/tcp\_fin\_timeout | sysctl -w net.ipv4.tcp\_fin\_timeout=60 | | Поддерживать соединение активным определенное время, например, 300 секунд. По истечении этого времени TCP соединение будет разорвано. | 7200 | /proc/sys/net/ipv4/tcp\_keepalive\_time | sysctl -w net.ipv4.tcp\_keepalive\_time=7200 | | Указывает размер по умолчанию буфера для сокета получения данных по сети | 212992 | /proc/sys/net/core/rmem\_default | sysctl -w net.**core.rmem\_default**e=212992 | | Максимальный размер буфера сокета для получения данных по сети | 212992 | /proc/sys/net/core/rmem\_max | sysctl -w net.core.rmem\_max=212992 | | Размер сокета по умолчанию для отправки данных по сети. | 212992 | /proc/sys/net/core/wmem\_default | sysctl -w net.core.wmem\_defaulte=212992 | | Максимальный размер буфера сокета для отправки данных по сети. | 212992 | /proc/sys/net/core/wmem\_max | sysctl -w net.core.wmem\_max=212992 | | Количество памяти, доступной для работы TCP. | 4096 131072 6291456 (минимум, по умолчанию, максимум) | /proc/sys/net/ipv4/tcp\_rmem |  | | Указывает процент от общей системной памяти, когда фоновый демон pdflush записи данных начнет переписывать кешированные данные на диск | 10 (Рекомендуется 3) | /proc/sys/vm/dirty\_background\_ratio | sysctl -w vm.dirty\_background\_ratio =10 | | Указывает сколько общей оперативной памяти должно быть занято, чтобы процесс, который ведет запись данных на диск инициировал запись кэшированных данных непосредственно на жесткий диск | 20 (Рекомендуется 10) | /proc/sys/vm/dirty\_ratio | sysctl -w vm.dirty \_ratio =20 | | Устанавливает процент свободной памяти, по достижении которого данные начинают переноситься на swap раздел, для систем с большим количеством памяти рекомендовано значение 10. | 60 (Рекомендуется 10) | /proc/sys/vm/swappiness | sysctl -w vm. swappiness =60 | | Определяет условия разрешения и отказа запросов больших объемов памяти. Возможные значения: 0 (по умолчанию) — ядро использует эвристический алгоритм для расчета перерасхода памяти, принимая во внимание объем доступной памяти и число неверных запросов.  1 — ядро не обрабатывает перерасход памяти. При этом вероятность превышения нагрузки на память возрастает, но в то же время увеличивается производительность задач, активно использующих память. 2 — отказ обработки запросов, запрашивающих память, размер которой превышает суммарный размер памяти пространства подкачки и ОЗУ в соответствии с overcommit\_ratio (По умолчанию равен 50). | 0 (Рекомендуется 2) | /proc/sys/vm/overcommit\_memory | sysctl -w vm. overcommit\_memory =2 | | **Так же для СУБД сервера репомендуется в /etc/systemd/logind.conf указать RemoveIPC=no (RemoveIPC в systemd: отключить высвобождение памяти ОС для УЗ) на данный момент RemoveIPC=yes** | | | | |

1. **Сервер 192.168. 1.109 – 1C**

* **Основное**

|  |  |
| --- | --- |
| **Основные директории сервера** | |
|  |  |

|  |  |
| --- | --- |
| **Основные скрипты** | |
|  |  |
|  |  |
|  |  |

* **Настройка Ядра ОС Linux**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  |  | | --- | --- | --- | --- | | Наименование | Значение | Откуда | Описание | | Количество процессоров | 16 | cat /proc/cpuinfo | grep processor | wc -l | Количество процессоров, показанное в /proc/cpuinfo, может не соответствовать реальному количеству ядер процессора. Например, процессор с 2 ядрами и гиперпоточностью будет показан как процессор с 4 ядрами. | | Идентификатор ядра | 0,1,2,3 - (всего 4) | cat /proc/cpuinfo | grep 'core id' | Есть 4 разных идентификатора ядра. Это указывает на то, что существует 4 реальных ядра. | | Данную информацию можно также посомтреть при помощи команды - lscpu. | | | | | Диапазоны портов |  | /proc/ioports | В этом файле содержаться все зарегистрированные диапазоны портов, а также устройства, которые их используют | | MemTotal : доступный объем оперативной памяти | 81451220 kB = 81 Gb | /proc/meminfo | физическая память за вычетом нескольких зарезервированных битов и бинарного кода ядра | | Команда для вывода информации об использовании оперативной памяти - free | | | | | Модуль ядра, или его часто называют драйвером, — это фрагмент кода, расширяющий функциональные возможности ядра.Запустите lsmod из командной строки, чтобы узнать, какие модули ядра загружены в данный момент. | | | | | Точки монтирования |  | /proc/mounts | В этом файле перечислены все точки монтирования и все подключенные файловые системы | | Точную версию ядра | Linux version 4.15.3-1-generic (root@build) (gcc version 6.3.0 20170516 (Debian 6.3.0-18+deb9u1)) #astra21 SMP Thu Aug 22 12:16:21 UTC 2019 | /proc/version | Точная версию ядра, компилятора, и в некоторых случаях, даже дистрибутива | | /proc/pid/ - Файловая система proc состоит не только из файлов, но здесь есть и папки. Больше всего здесь папок с номерами вместо имен. Каждый этот номер означает PID процесса, а эта папка содержит информацию о каждом запущенном в системе процессе. Когда процесс заканчивается, его каталог исчезает из системы | | Основные файлы: cmdline - содержит команду с помощью которой был запущен процесс, а также переданные ей параметры cwd - символическая ссылка на текущую рабочую директорию процесса exe - ссылка на исполняемый файл root - ссылка на папку суперпользователя environ - переменные окружения, доступные для процесса fd - содержит файловые дескрипторы, файлы и устройства, которые использует процесс maps, statm, и mem - информация о памяти процесса stat, status - состояние процесса | С помощью этих файлов вы можете составлять различные скрипты. Например если вы хотите уничтожить все зомби процессы, то вы можете сканировать все директории на наличие Z в файле status. Так же само можно проверить запущена ли нужная вам программа просмотрев все cmdline. | | /proc/sys/ -Эта папка в proc linux не только предоставляет информацию о системе, но и позволяет изменять различные параметры ядра на лету, а также включать дополнительные функции. | | Основные подкаталоги в этой папке: debug - содержит отладочную информацию, она будет вам полезна если вы разработчик ядра dev - параметры различных устройств, подключенных к системе fs - вся информация о файловой системе kernel - позволяет напрямую настраивать ядро net - настройка разных параметров сети vm - взаимодействие с подсистемой vm | Чтобы посмотреть можно ли записывать в файлы используйте команду:   ls -ld /proc/sys/  Если у файла есть флаг W, значит в него можно записывать данные. | | Настройки Ядра | Значение | Путь | Команда | | Если установлено 1, разрешает выполнение 32 битных программ в 64 битной системе | 1 (по умолчанию) | /proc/sys/abi/vsyscall32 | sysctl -w abi.vsyscall32=1 | | При возникновении ошибки в ядре выводить значения регистров процессора и стек вызовов процедур | 1 (по умолчанию) | /proc/sys/debug/exception-trace | sysctl -w debug.exception-trace=1 | | Максимальная частота генерации прерываний от системного таймера High Precision Event Timer (HPET), который пришел на замену таймеру реального времени RTC. | 64 (по умолчанию) | /proc/sys/dev/hpet/max-user-freq | sysctl -w dev.hpet.max-user-freq=64 | | Количество асинхронных операций ввода и вывода в вашей файловой системе | 0 | /proc/sys/fs/aio-nr | sysctl -w fs.aio-nr=0 | | Максимальное количество дескрипторов файлов, которые может создать и обрабатывать ядро. Если вы часто получаете сообщения об ошибке из-за невозможности создать дескриптор файла увеличьте этот лимит. По умолчанию установлено значение 10 % от вашей оперативной памяти. | 8126176 = 8Gb | /proc/sys/fs/file-max | sysctl -w fs.file-max=8126176 | | Подсистема ядра inotify позволяет следить за изменениями в файловой системе. Этот параметр устанавливает максимальное количество событий, которые могут находиться в очереди, перед тем как их обработает программа. | 16384 | /proc/sys/fs/inotify/max\_queued\_events | sysctl -w fs.inotify.max\_queued\_events=16384 | | Позволяет установить доменное имя NIS (Network Internet Services) и YP (Yellow Pages). Но не путайте это доменное имя с DNS, это совсем разные вещи. | (none) | /proc/sys/kernel/domainname | - | | Имя вашего компьютера. Это самый простой способ изменить имя компьютера прямо сейчас, без перезагрузки. | 1C | /proc/sys/kernel/hostname | sysctl -w kernel.hostname=1C | | Позволяет отключить загрузку модулей ядра. | 0 | /proc/sys/kernel/modules\_disabled | sysctl -w kernel.modules\_disabled=0 | | Рандомизация адресного пространства это функция увеличивающая безопасность вашей системы. Она защищает от атак на переполнение буфера. По умолчанию включена. | 2 - полностью автоматизировано | /proc/sys/kernel/randomize\_va\_space | sysctl -w kernel.randomize\_va\_space=2 | | Позволяет включить или отключить управление ядром с помощью SysRQ | 0 - отключить все функции | /proc/sys/kernel/sysrq | sysctl -w kernel.sysrq=0 | | Максимальное количество запущенных потоков для процессов. | 563454 | /proc/sys/kernel/threads-max | sysctl -w kernel.threads-max=563454 | | Если включено, ядро будет игнорировать все icmp запросы. Рекомендуется для защиты от DDOS атак. | 0 | /proc/sys/net/ipv4/icmp\_echo\_ignore\_all | sysctl -w net.ipv4.icmp\_echo\_ignore\_all=0 | | Так же, как и в предыдущем варианте, только игнорироваться будут только широковещательные icmp запросы | 1 | /proc/sys/net/ipv4/icmp\_echo\_ignore\_broadcasts | sysctl -w net.ipv4.icmp\_echo\_ignore\_broadcasts=1 | | Максимальное количество узлов, через которые может пройти отправленный пакет перед тем, как достигнет цели. | 64 | /proc/sys/net/ipv4/ip\_default\_ttl | sysctl -w net.ipv4.ip\_default\_ttl=64 | | Разрешить проходящие пакеты через этот компьютер. Обычно такая настройка параметров ядра Linux нужна для роутеров | 0 | /proc/sys/net/ipv4/ip\_forward | sysctl -w net.ipv4.ip\_forward=0 | | Диапазон локальных портов, которые могут быть использованы вашими программами | 32768 60999 (по умолчанию) | /proc/sys/net/ipv4/ip\_local\_port\_range | sysctl -w net.ipv4.ip\_local\_port\_ranged= "32768 60999" | | Установите 1 чтобы защитить компьютер от атаки TCP TimeWait. | 0 | /proc/sys/net/ipv4/tcp\_rfc1337 | sysctl -w net.ipv4.tcp\_rfc1337=0 | | Таймаут ожидания завершения соединения после отправки пакета FIN. Рекомендовано 15. | 60 | /proc/sys/net/ipv4/tcp\_fin\_timeout | sysctl -w net.ipv4.tcp\_fin\_timeout=60 | | Поддерживать соединение активным определенное время, например, 300 секунд. По истечении этого времени TCP соединение будет разорвано. | 7200 | /proc/sys/net/ipv4/tcp\_keepalive\_time | sysctl -w net.ipv4.tcp\_keepalive\_time=7200 | | Указывает размер по умолчанию буфера для сокета получения данных по сети | 212992 | /proc/sys/net/core/rmem\_default | sysctl -w net.**core.rmem\_default**e=212992 | | Максимальный размер буфера сокета для получения данных по сети | 212992 | /proc/sys/net/core/rmem\_max | sysctl -w net.core.rmem\_max=212992 | | Размер сокета по умолчанию для отправки данных по сети. | 212992 | /proc/sys/net/core/wmem\_default | sysctl -w net.core.wmem\_defaulte=212992 | | Максимальный размер буфера сокета для отправки данных по сети. | 212992 | /proc/sys/net/core/wmem\_max | sysctl -w net.core.wmem\_max=212992 | | Количество памяти, доступной для работы TCP. | 4096 87380 6291456 (минимум, по умолчанию, максимум) | /proc/sys/net/ipv4/tcp\_rmem |  | | Указывает процент от общей системной памяти, когда фоновый демон pdflush записи данных начнет переписывать кешированные данные на диск | 10 (Рекомендуется 3) | /proc/sys/vm/dirty\_background\_ratio | sysctl -w vm.dirty\_background\_ratio =10 | | Указывает сколько общей оперативной памяти должно быть занято, чтобы процесс, который ведет запись данных на диск инициировал запись кэшированных данных непосредственно на жесткий диск | 20 (Рекомендуется 10) | /proc/sys/vm/dirty\_ratio | sysctl -w vm.dirty \_ratio =20 | | Устанавливает процент свободной памяти, по достижении которого данные начинают переноситься на swap раздел, для систем с большим количеством памяти рекомендовано значение 10. | 60 (Рекомендуется 10) | /proc/sys/vm/swappiness | sysctl -w vm. swappiness =60 | | Определяет условия разрешения и отказа запросов больших объемов памяти. Возможные значения: 0 (по умолчанию) — ядро использует эвристический алгоритм для расчета перерасхода памяти, принимая во внимание объем доступной памяти и число неверных запросов.  1 — ядро не обрабатывает перерасход памяти. При этом вероятность превышения нагрузки на память возрастает, но в то же время увеличивается производительность задач, активно использующих память. 2 — отказ обработки запросов, запрашивающих память, размер которой превышает суммарный размер памяти пространства подкачки и ОЗУ в соответствии с overcommit\_ratio (По умолчанию равен 50). | 0 (Рекомендуется 2) | /proc/sys/vm/overcommit\_memory | sysctl -w vm. overcommit\_memory =2 | | **Так же для СУБД сервера репомендуется в /etc/systemd/logind.conf указать RemoveIPC=no (RemoveIPC в systemd: отключить высвобождение памяти ОС для УЗ) на данный момент RemoveIPC=yes** | | | | |

1. **Linux , решение типовых задач**

* **Изменение настроек Ядра ОС Linux. Сделать так, чтобы они не изменялись при перезапуске сервера.**

**Задача:** чтобы параметры ядра vm.dirty\_background\_ratio =3, vm.dirty \_ratio =10, vm. swappiness =10,  vm. overcommit\_memory =2; даже после перезапуска сервера.

создаем свой файл конфигурации в каталоге /etc/sysctl.d/.

Например, если вы меняете параметры ядра из-за требований к базе данных, создайте файл:

/etc/sysctl.d/60-mysql.conf

и добавьте свои собственные параметры.

Содержание nano 60-mysql.conf :

vm.dirty\_background\_ratio =3

vm.dirty \_ratio =10

vm. swappiness =10

  vm. overcommit\_memory =2

Чтобы выполнить это:

**service** procps restart

Просто отметить:

procps - это системный файл, и он **НИКОГДА не** должен редактироваться.

/etc/sysctl.conf также не следует редактировать, поскольку его можно изменить при обновлении системы / ядра, и если оно отличается от ожидаемого, обновление будет остановлено с вопросом, может ли более новая версия заменить текущую.

* **Добавление репозитария**

Для использования сетевых репозиториев, работающих по протоколу HTTPS убедитесь, что в системе установлены пакеты **apt-transport-https** и **ca-certificates**, обеспечивающие возможность загрузки пакетов с использованием этого протокола  . Если нет - то установите их:

**sudo apt install apt-transport-https ca-certificates**

Для установки можно использовать также протокол HTTP.

В файле **/etc/apt/sources.list** указать путь к сетевому репозиторию (тег https:// или http:// или ftp://)

**nano /etc/apt/sources.list - открываем и дописываем**

# Основной репозиторий

deb https://dl.astralinux.ru/astra/stable/1.7\_x86-64/repository-main/ 1.7\_x86-64 main contrib non-free

# Оперативные обновления основного репозитория

deb https://dl.astralinux.ru/astra/stable/1.7\_x86-64/repository-update/ 1.7\_x86-64 main contrib non-free

# Базовый репозиторий

deb https://dl.astralinux.ru/astra/stable/1.7\_x86-64/repository-base/ 1.7\_x86-64 main contrib non-free

# Расширенный репозиторий

deb https://dl.astralinux.ru/astra/stable/1.7\_x86-64/repository-extended/ 1.7\_x86-64 main contrib non-free

# Расширенный репозиторий (компонент astra-ce)

deb https://dl.astralinux.ru/astra/stable/1.7\_x86-64/repository-extended/ 1.7\_x86-64 astra-ce

Обновить список пакетов:

**sudo apt update**

* **Настройка сети**

Настройка осуществляется путем правки файла /etc/network/interfaces. Каждый сетевой интерфейс (сетевая карта, хотя это не совсем точное название) настраивается отдельно. Настройки для сервера выглядят так:

/etc/network/interfaces

auto lo

iface lo inet loopback

iface eth0 inet static

address 192.168.0.1

netmask 255.255.255.0

gateway 192.168.0.1 # В качестве шлюза - наш сервер с IP=1

network 192.168.0.0 # Указываем сеть, это обязательно для работы в составе ALD

broadcast 192.168.0.255 # Сервер ALD начиная с Astra 1.5 выводит ошибку, если не видит

dns-nameservers 192.168.0.1 # Тут через пробел можно перечислить ВСЕ DNS-серверы сети

dns-search DOMAIN.NET # Имя домена ALD

service networking restart

/etc/netplan/00-installer-config.yaml настройки сети на Ubuntu и более новых ОС Linux

* **Sudo доступ без ввода пароля**

Вводим в терминал **visudo** и дописываем:

# Allow members of group sudo to execute any command

%sudo ALL=(ALL:ALL) ALL

**ivan ALL=(ALL:ALL) NOPASSWD:ALL**

Готово, пользователю **ivan** не надо будет вводить дополнительно пароль при использовании sudo

* **Монтирование сетевого диска**

mount -t cifs //192.168.1.116/backupPG /mnt/backupPG -o user=логин,password=парол

1. **Установка и настройка**
2. **Установка и настройка Postgres12 под 1C на Astra Linux**

**wget [https://repo.postgrespro.ru/pg1c-12/keys/pgpro-repo-add.sh](https://repo.postgrespro.ru/pg1c-12/keys/pgpro-repo-add.sh" \t "_blank)**

**sh pgpro-repo-add.sh**

Если наш продукт единственный Postgres на вашей машине и вы хотите

сразу получить готовую к употреблению базу:

**apt-get install postgrespro-1c-12**

\_\_\_\_

Если у вас уже установлен другой Postgres и вы хотите чтобы он

продолжал работать параллельно (в том числе и для апгрейда с более

старой major-версии):

apt-get install postgrespro-1c-12-contrib

/opt/pgpro/1c-12/bin/pg-setup initdb

/opt/pgpro/1c-12/bin/pg-setup service enable

/opt/pgpro/1c-12/bin/pg-setup service start

-------

На заметку : сервис данной Postgres будет называться нестандартно: postgrespro-1c-12

**service postgrespro-1c-12 status**

Конфигурации:

/var/lib/pgpro/1c-12/data/pg\_hba.conf

Задать пароль пользователю postgres:

**sudo su postgres  
cd ~  
psql -c "ALTER ROLE postgres WITH PASSWORD ' 12345678';"  
exit**

1. **Установка и настройка 1C-сервера на Astra Linux**

* **Ошибки:**

В Astra Linux 1C автоматически не запускается при загрузке сервера, лечится следующим действием:  
  
nano /etc/systemd/logind.conf  
  
[Login]  
KillUserProcesses=no  
  
далее  
  
systemctl restart systemd-logind

* **Установка 1С сервера на Astra Linux:**
* Загрузить архив дистрибутива на сервер:

/home/ivan/Linux\_Plat\_8-3-18-1363-20220905T125840Z-001.zip

* Распаковать архив:

unzip Linux\_Plat\_8-3-18-1363-20220905T125840Z-001.zip

* Переходим в новую директорию с пакетами:

cd /home/ivan/Linux\_Plat\_8-3-18-1363

* Устанавливаем пакеты в следующей последовательности:

dpkg –i 1c-enterprise-8.3.18.1363-common\_8.3.18-1363\_amd64.deb

dpkg –i 1c-enterprise-8.3.18.1363-common-nls\_8.3.18-1363\_amd64.deb

dpkg –i 1c-enterprise-8.3.18.1363-server\_8.3.18-1363\_amd64.deb

dpkg –i 1c-enterprise-8.3.18.1363-server-nls\_8.3.18-1363\_amd64.deb

dpkg –i 1c-enterprise-8.3.18.1363-ws\_8.3.18-1363\_amd64.deb

dpkg –i 1c-enterprise-8.3.18.1363-ws-nls\_8.3.18-1363\_amd64.deb

dpkg –i 1c-enterprise-8.3.18.1363-crs\_8.3.18-1363\_amd64.deb

* Создать символьные ссылки на файлы службы:

sudo ln -s /opt/1cv8/x86\_64/8.3.20.1789/srv1cv83 /etc/init.d/srv1cv83  
sudo ln -s /opt/1cv8/x86\_64/8.3.20.1789/srv1cv83.conf /etc/default/srv1cv83

* Обновить списки системных служб:

sudo systemctl daemon-reload

* Разрешить автоматический запуск службы при перезагрузке ОС:

sudo systemctl enable srv1cv83

* Запустить службу:

sudo systemctl start srv1cv83

Материал:

<https://entnet.ru/special/1s-predpriyatie/server-1s-i-pg.html>

https://wiki.astralinux.ru/pages/viewpage.action?pageId=41191288

1. **Установка и настройка Zabbix-сервера + Postgres 14.5 + Apache на Ubuntu Linux 18.04**

* **Установка Postgres 14.5 на Ubuntu 18.4**

Создание файла конф.репозитория

sudo sh -c 'echo "deb http://apt.postgresql.org/pub/repos/apt $(lsb\_release -cs)-pgdg main" > /etc/apt/sources.list.d/pgdg.list'

Импортируем ключ

wget --quiet -O - https://www.postgresql.org/media/keys/ACCC4CF8.asc | sudo apt-key add –

apt-get update

apt-get -y install postgresql

sudo -u postgres psql

\conninfo

* **Установка Апач**
* Установка Apache

Как и в случае с многим другим популярным ПО, Apache можно установить прямо из стандартных репозиториев Debian.

Сначала обновите локальный индекс пакетов:

$ sudo apt update

А затем выполните установку:

$ sudo apt install apache2

Подтвердите действие, и после этого утилита apt установит Apache и все необходимые зависимости.

## Настройка файрвола

Теперь необходимо внести некоторые изменения в настройки файрвола.

Выведите все доступные профили приложений на своем сервере:

$ sudo ufw app list

Вывод будет, например, таким:

Available applications:

AIM

Bonjour

CIFS

. . .

WWW

WWW Cache

WWW Full

WWW Secure

. . .

Профили Apache начинаются с WWW:

* WWW: этот профиль открывает 80 порт (обычный, незашифрованный веб-трафик);
* WWW Cache: профиль открывает только 8080 порт (иногда используется для кэширования и веб-прокси);
* WWW Secure: профиль открывает только 443 порт (TLS/SSL зашифрованный трафик);
* WWW Full: профиль открывает оба порта – 80 и 443.

Из этих профилей желательно выбрать тот, который разрешает только те порты, что вам нужны.

Т.к. SSL на сервере еще не настроен, вам нужно открыть порт 80:

$ sudo ufw allow 'WWW'

Проверьте, вступили ли изменения в силу:

$ sudo ufw status

В выводе вы увидите, что трафик HTTP теперь разрешен:

Status: active

To Action From

-- ------ ----

OpenSSH ALLOW Anywhere

WWW ALLOW Anywhere

OpenSSH (v6) ALLOW Anywhere (v6)

WWW (v6) ALLOW Anywhere (v6)

Значит, все нормально, и порт успешно открыт.

## Проверка веб-сервера

Но на всякий случай это можно проверить командой:

$ sudo systemctl status apache2

* **Установка Zabbix сервер**

wget <https://repo.zabbix.com/zabbix/6.0/ubuntu/pool/main/z/zabbix-release/zabbix-release_6.0-3%2Bubuntu18.04_all.deb>

dpkg -i zabbix-release\_6.0-3+ubuntu18.04\_all.deb

apt update

apt install zabbix-server-pgsql zabbix-frontend-php php7.2-pgsql zabbix-apache-conf zabbix-sql-scripts zabbix-agent

sudo -u postgres createuser --pwprompt zabbix  
 sudo -u postgres createdb -O zabbix zabbix

zcat /usr/share/doc/zabbix-sql-scripts/postgresql/server.sql.gz | sudo -u zabbix psql zabbix

Редактируем nano /etc/zabbix/zabbix\_server.conf

DBPassword=password

systemctl restart zabbix-server zabbix-agent apache2  
systemctl enable zabbix-server zabbix-agent apache2

* **Настройка оповещения в Telegram чат**

**Шаг 1: Создать группу в Telegram и добавить туда д бота для получения ID**

****Создаем группу и добавляем бота **@getmyid\_bot** , он выдаст ID личный и ID группы ,это второй(**-832378868**) .Он нам еще пригодится.

Изображение выглядит как текст

Автоматически созданное описание

**Шаг 2: Создаем бота у @BotFather**

Пишем в Telegram боту (как и любому другому пользователю) @BotFather команду [/newbot](tg://bot_command?command=newbot)

Задаем **name** и **username**(должен быть уникальным) . Получаем **Токен** бота **5492590427:AAEKGyp4iIvdCcUcKZOffHEZeyHnbBWcBHk**

**Шаг 3: Настройка в Вэб интерфейсе Zabbix**

Администрирование -> Способы оповещения -> Telegram (тип Webhook)

****



Вводим в поле Token – токен который выдал **@BotFather** : 5492590427:AAEKGyp4iIvdCcUcKZOffHEZeyHnbBWcBHk

Заходим : Администрирование -> Пользователи ->Выбиваем пользователя: Admin -> Оповещение

Добавляем тип оповещения тип : Telegram и в поле Отправить на дополняем ID чата от бота **@getmyid\_bot** : **-832378868** 

Готово!

1. **Установка и настройка Ansible на Ubuntu Linux 18.04**

* **Установка Ansible**

Чтобы добавить архив PPA (архив персональных пакетов) официального проекта в список источников вашей системы, запустите на узле управления следующую команду:

sudo apt-add-repository ppa:ansible/ansible

sudo apt update

После этого обновления вы можете установить программное обеспечение Ansible следующим образом:

sudo apt install ansible

Файл конфигураций Ansible находится в : /etc/ansible/ansible.cfg

Открываем и перенастраиваем :

nano /etc/ansible/ansible.cfg

inventory = /root/ansible/hosts

Переходим в директорию /root и создаем там директории :

mkdir ansible

cd /root/ansible

nano hosts –создали ,редактировать будем потом

mkdir group\_vars -создание директории для групповых переменных (облегчает чтение файлa hosts ,хорошая практика)

Расположение логов : /var/log/ansible.log

Подключение хостов : nano /etc/ansible/hosts по умолчанию ,после настройки: nano /root/ansible/hosts

[servers]

server1 ansible\_host=203.0.113.111

server2 ansible\_host=203.0.113.112

server3 ansible\_host=203.0.113.113

Проверка

ansible-inventory --list –y

* **Настройка подключения к хосту по shh key**

**На Ansible сервере:**

ssh-keygen

Задаем пароль и запоминаем , допустим (12345), в результате :

Your identification has been saved in /root/.ssh/id\_rsa.

Your public key has been saved in /root/.ssh/id\_rsa.pub.

Далее переименовываем для удобства id\_rsa в имя хоста – server1

mv id\_rsa server1

Открываем и копируем содержимое :

cat id\_rsa.pub

mv id\_rsa.pub server1.pub – заодно переименуем ,для удобства

**На сервере-хосте:**

Открываем nano /root/.ssh/authorized\_keys

И вставляем туда содежимое cat id\_rsa.pub с Ansible-сервера

!**Проблема** AstraLinux – отсутствие директории .ssh вместе с authorized\_keys

**Решение**:

Генерируем ключи :ssh-keygen , в результате создается директория .ssh в /root

Заходим cd /root/.shh и создаем здесь файл nano authorized\_keys ,в который и вписываем содержимое cat id\_rsa.pub с Ansible-сервера

Далее на Ansible сервере редактируем nano /root/ansible/hosts

[servers]

server1 ansible\_host=192.168.56.106 ansible\_user=root ansible\_ssh\_private\_key\_file=/root/.ssh/ server1

**Проверка**

Тест: ansible all -m ping

Ансибл , all-все хосты , -m – значит модуль , ping – пинг)

Вводим заданный пароль(12345) и в результате должны получить :

server1 | SUCCESS => {

"changed": false,

"ping": "pong"

}

**Создание групповых переменных**

nano /root/ansible/hosts на данный момент выглядит так :

[group1]

serverone ansible\_host=192.168.56.106 ansible\_user=root ansible\_ssh\_private\_key\_file=/root/.ssh/all

serverthree ansible\_host=192.168.56.109 ansible\_user=root ansible\_ssh\_private\_key\_file=/root/.ssh/all

[group2]

servertwo ansible\_host=192.168.56.114 ansible\_user=root ansible\_ssh\_private\_key\_file=/root/.ssh/all

[group3]

serverfour ansible\_host=192.168.56.113 ansible\_user=root ansible\_ssh\_private\_key\_file=/root/.ssh/all

[gr12:children]

group1

group2

[gr23:children]

group2

group3

[gr13:children]

group1

group3

[allgr:children]

group1

group2

group3

Хосты поделены на группы , к каждому свой ssh ключ , но можно создать в /root/ancible/ group\_vars файл для всех групп allgr , куда вынести переменные ansible\_user=root ansible\_ssh\_private\_key\_file=/root/.ssh/all ,при этом нужно на всех серверах прокинуть один ssh ключ all.pub .по аналогии ,как делали для отдельного сервера

**На сервере:**

ssh-keygen -создаем

mv id\_rsa.pub all.pub -переименовываем

mv id\_rsa all

nano all.pub -копируем

**На хостах:**

nano /root/.ssh/authorized\_keys

добавляем скопированный ключ из all.pub ,сохраняем

**На сервере:**

Переходим в cd /root/ansible/group\_vars

nano allgr –создаем и редактируем (название файла по названию группы ,для которой прописываются переменные)

ansible\_user : root

ansible\_ssh\_private\_key\_file : /root/.ssh/all

ansible\_python\_interpreter : /usr/bin/python3

На хостах установить пакет **apt install python3-psycopg2**

nano /root/ansible/hosts –заходим и редактируем

[group1]

serverone ansible\_host=192.168.56.106

serverthree ansible\_host=192.168.56.109

[group2]

servertwo ansible\_host=192.168.56.114

[group3]

serverfour ansible\_host=192.168.56.113

[gr12:children]

group1

group2

[gr23:children]

group2

group3

[gr13:children]

group1

group3

[allgr:children]

group1

group2

group3

Проверка :

ansible-inventory --list –y

ansible all -m ping

**Создание плейбуков:**

В директории /root/ansible создаем файл touch ping.yml и заходим в него для редактирования vi ping.yml

(редактор vi – нажать I для выхода из командного режима , ESC чтобы ввойти в командный режим , q! –выйти без сохранения , wq –сохранить и выйти)

- name: Ping Servers

hosts: all

become : yes

tasks:

- name: Task ping

ping:

* **Администрирование Postgres-12-1C на Astra Linux, особенности:**

**Требуются пакеты на хосте:**

**apt install acl**

**apt install python3-psycopg2**

**Первая ошибка**: An exception occurred during task execution. To see the full traceback, use -vvv. The error was: соединения через Unix-сокет "/var/run/postgresql/.s.PGSQL.5432"?

Postgres установленный под 1С создает сокет в директори: **/tmp/** , пока модуль будет искать его в стандартом месте : **/var/run/postgresql/**

**Решение:**

ln -s /tmp/.s.PGSQL.5432 /var/run/postgresql/.s.PGSQL.5432

Создаем символьную ссылку ,при необходимости создаем директорию postgresql ,если она отсутсвует.

**Вторая ошибка:** An exception occurred during task execution. To see the full traceback, use -vvv. The error was: FileNotFoundError: [Errno 2] Нет такого файла или каталога: '/var/lib/postgresql/data/pg\_hba.conf'

**Решение:**

ln -s /var/lib/pgpro/1c-12/data/pg\_hba.conf /var/lib/postgresql/data/pg\_hba.conf

(для postgrespro на 115ом сервере)

ln -s /var/lib/pgpro/ent-12/data/pg\_hba.conf /var/lib/postgresql/data/pg\_hba.conf

Аналогично создаем символьную ссылку , при необходимости ,создаем недостающие директории

На заметку : сервис данной Postgres будет называться нестандартно: postgrespro-1c-12

* **Пример работы с Postgres через Ansible используя плейбуки**

- Подключаемся к 192.168.1.120

- Переходим в директорию ansible

**cd /root/ansible**

Список доступных плейбуков:

crdb.yml - создание новой базы

drop.yml – удаление базы

dump.yml – бэкап(дамп) базы

rsdb.yml – загрузка базы из дампа

**Задача:** Создать новую базу

- Открываем файл vars.yml

**nano vars.yml**

и добавляем название новой базы, например: test\_create\_ansdb, сохранили и вышли.

**create\_db\_name:** *test\_create\_ansdb*

- Запускаем плейбук:

**ansible-playbook crdb.yml**

- Готово, база создана.

**Задача:** Создать бэкап , дамп базы

- Открываем файл vars.yml

**nano vars.yml**

и добавляем название базы, например: test\_create\_ansdb, сохранили и вышли.

**dump\_db\_name:** *test\_create\_ansdb*

- Запускаем плейбук:

**ansible-playbook dump.yml**

- Готово, дамп базы создан (на 192.168.1.120): /home/backup/test\_create\_ansdb.dump.gz.

**Задача:** Загрузка базы из дампа

- Открываем файл vars.yml

**nano vars.yml**

и добавляем название базы, например: test\_create\_ansdb и имя дамп файла расположенного в директории /home/restore/ (на сервере 1.120), например: /home/restore/dumptestbase.dump.gz, сохранили и вышли.

#Restore DataBase

**restore\_db\_name:** *test\_create\_ansdb*

#Restore from dump file , dir: /home/restore/{dump\_file\_name}.dump.gz

**dump\_file\_name:** *dumptestbase*

- Запускаем плейбук:

**ansible-playbook rsdb.yml**

- Готово, база восстановлена из дампа.

**Задача:** Удаление базы (Drop Data Base)

- Открываем файл vars.yml

**nano vars.yml**

и добавляем название базы, например: test\_create\_ansdb, сохранили и вышли.

#Drop DataBase

**drop\_db\_name:** *test\_create\_ansdb*

- Запускаем плейбук:

**ansible-playbook drop.yml**

- Готово, базa test\_create\_ansdb удалена.

1. **Установка Git и подключение репозитория с данным гайдом**

Материал: <https://git-scm.com/book/ru/v2/%D0%92%D0%B2%D0%B5%D0%B4%D0%B5%D0%BD%D0%B8%D0%B5-%D0%A3%D1%81%D1%82%D0%B0%D0%BD%D0%BE%D0%B2%D0%BA%D0%B0-Git>

Установка в Windows

Для установки Git в Windows также имеется несколько способов. Официальная сборка доступна для скачивания на официальном сайте Git. Просто перейдите на страницу <https://git-scm.com/download/win>, и загрузка запустится автоматически.

Зайти в Git Bush и перейти в директорию ,в которую скачаем репозиторий ,например: D:\Git

**cd /d/Git**

Или создать папку и запустить Git Bush сразу в ней : 

Git Bush Here

**git clone** <https://github.com/MakeUsWhole/admin.git>



Для обновления файлов в соотвествии с последними обновлениями в репозитории

**git pull**

1. **Установка и настройка VPN сервера (OpenVPN) ОС Ubuntu**

Шаг 1: Установка пакетов

**sudo apt install mc nano wget curl**

Шаг 2:Установка ,настройка OpenVpn сервера

**wget** [**https://git.io/vpn**](https://git.io/vpn) **-O** [**openvpn-install.sh**](https://openvpn-install.sh/) **&& bash** [**openvpn-install.sh**](https://openvpn-install.sh/)

Далее настраиваем ВПН сервер по шагам , название сервера = его статический IP , и в конце создаем первого клиента . В /root/ появится файл {имя\_клиента}.ovpn , его нужно отправить пользователю с установленным OpenVpn

Шаг 3: Настройка роутера в сети VPN сервера

В браузере вводим: **192.168.1.1** или **192.168.1.0** , лагин и пароль по умолчанию: **admin** и **admin** .

В примере : **Интернет-> Переадресация портов**   
   
Добавляем службу: **ovpn** с диапазоном портов: **1194** (на него придет запрос на подключение к впн от клиента) , локальный IP сервера VPN , в данном случае: **192.168.1.133** , локальный порт: **1194** . Протокол выбираем в зависимости от того, как настроили VPN сервер: **UDP** или **TCP**



Готово!

1. **Настройка подключения к удаленному рабочему столу с Windows на Linux Ubuntu**

*Шаг 1: Обновление пакетов*

**sudo apt update**

*Шаг 2: Открываем порт 3389*

**ufw allow 3389**

*Шаг 3: Настройка роутера в сети РДП сервера*

В браузере вводим: **192.168.1.1** или **192.168.1.0** , лагин и пароль по умолчанию: **admin** и **admin** .

В примере : **Интернет-> Переадресация портов**   
   
Добавляем службу: **rdp** с диапазоном портов: **3389** (на него придет запрос на подключение к впн от клиента) , локальный IP сервера RDP , в данном случае: **192.168.1.133** , локальный порт: **3389** . Протокол выбираем: **TCP**



*Шаг 4: Установка необходимых пакетов*

**apt install xrdp**

**apt install xorgxrdp**

*Шаг 5: Включение сервиса и проверка статуса*

**systemctl enable xrdp**

**systemctl status xrdp**

*Шаг 6: Установка необходимых пакетов*

**apt install xfce4**

*Шаг 7: Включение сервиса и проверка статуса*

**systemctl start xrdp**

**systemctl status xrdp**



*Шаг 8: Узнаем ip RDP сервера*

Используем сервис <https://2ip.ru/>



В данном примере ip (статический) : 78.107.237.245

*Шаг 9: Подключаемся с Windows клиента к RDP серверу Ubunty*

В сервисе **Подключение к удаленному рабочему столу** пишем IP сервера RDP и логин пользователя .



Вводим пароль и готово !