

**Міністерство освіти і науки України
Донбаська державна машинобудівна академія**

КОНСПЕКТ ЛЕКЦІЙ

з дисципліни

«АДМІНІСТРУВАННЯ КОМП'ЮТЕРНИХ МЕРЕЖ»

(для студентів спеціальності 123“Комп’ютерна
інженерія»)

Освітній рівень - бакалавр

Краматорськ 2020

ТЕМА 1. ВСТАНОВЛЕННЯ Й ПОЧАТКОВЕ КОНФІГУРУВАННЯ ОПЕРАЦІЙНОЇ СИСТЕМИ LINUX.

1.1 Встановлення CentOS 7

Будь-яка робота з адміністрування сервера починається з самого очевидного і обов'язкового процесу - інсталяції необхідної ОС, ніж ми і займемося. Завантажити та встановити CentOS 7 server в конфігурації `minimal` чи `netinstall` з завантажувальної флешки або по мережі на звичайний диск або `raid` розділ. Перед цим виконаємо невелику підготовчу роботу і познайомимося з подробицями нашого дистрибутива, які можуть бути корисні в майбутньому.

1.1.1 Системні вимоги CentOS 7

7 липня 2014 року побачило світ реліз дистрибутива CentOS 7. Перед його установкою рекомендується ознайомитися з системними вимогами. Детально подивитися повний список максимальних і мінімальних системних вимог можна на [офіційному wiki](#). Я ж наведу лише найважливіші параметри:

Системні вимоги CentOS 7	
Підтримка i386 архітектури	немає
Мінімальна кількість пам'яті	1GB
Рекомендована кількість пам'яті	1GB на кожне ядро процесора
Мінімальна місце на диску	10GB
Рекомендоване місце на диску	20GB
Максимальний розмір файлу (ext3)	2TB
Максимальний розмір файлової системи (ext3)	16TB
Максимальний розмір файлу (ext4)	16TB

Максимальний файлової системи (ext4)	розмір	50TB
---	--------	------

Це офіційні дані з сайту CentOS. У RHEL вони такі ж, я перевіряв. У мене особисто на VDS благополучно все працює і з 512MB пам'яті, менше не пробував ставити, думаю і з 256 заведеться.

1.1.2 Tunu iso образів CentOS 7

Реліз CentOS містив в собі кілька видів iso образів. Детальний опис кожного з них представлено в таблиці:

Редакції CentOS 7	
CentOS-7- x86_64-DVD	Цей DVD образ містить всі пакети, які можуть бути встановлені за допомогою інсталлера. Рекомендується для більшості користувачів.
CentOS-7- x86_64-NetInstall	Цей NetInstall образ для установки по мережі і для відновлення. Інсталлятор запитає, звідки буде проводитися установка пакетів. Зручно використовувати, якщо у вас є локальний репозиторій пакетів.
CentOS-7- x86_64-Everything	У цьому Everything образі міститься повний набір пакетів CentOS 7. Він може бути використаний для установки, або поновлення локального дзеркала. Для цього способу потрібно двосторонній DVD, або флешка на 8 Гб.
CentOS-7- x86_64- LiveGNOMECentOS- 7-x86_64-LiveKDE	Ці два образи є LiveCD CenOS 7. Залежно від назви використовується та чи інша графічна оболонка. Вони розроблені для тестування оточення CentOS 7. Вони не встановлюються на жорсткий диск, якщо ви не збираєтеся цього робити примусово. Набір встановленого програмного забезпечення поміняти не можна, це можна зробити тільки на встановленою операційною системою за допомогою yum.
CentOS-7- x86_64-Minimal	За допомогою цього Minimal способу можна встановити базову систему CentOS з мінімальним набором пакетів, необхідних для працездатності системи. Все інше можна доустановити пізніше за

допомогою утм. Набір пакетів в цьому образі буде такою ж, як і на DVD при виборі установки minimal.

1.1.3 Завантажити CentOS 7

Завантажити свіжу на поточний момент версію CentOS 7.2.1511 можна двома способами:

Через torrent мережу

З найближчого дзеркала

32 bit або i386 редакції CentOS 7 не існує. Всі дистрибутиви тільки x86_64, тобто 64 bit.

1.1.4 Завантажувальна флешка для CentOS 7

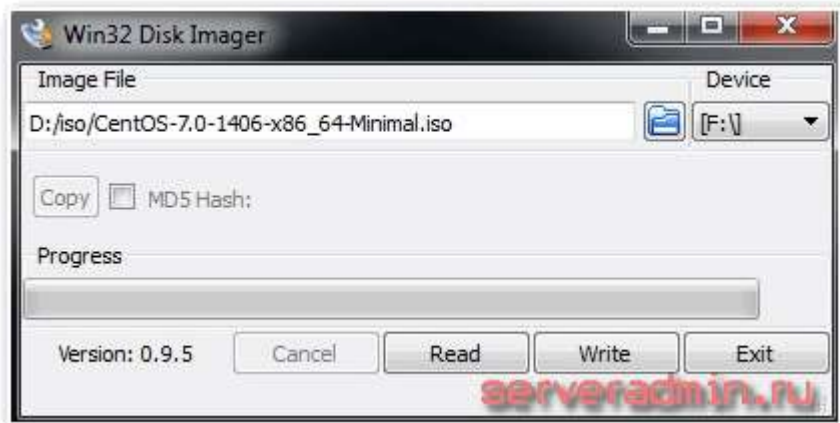
Останнім часом особисто я практично не користуюся звичайними CD, віддаючи перевагу над ними завантажувальні флешки. Вони зручніше, займають менше місця, простіше оновити дистрибутив на них. Але іноді доводиться задурити для створення завантажувальної флешки.

За її допомогою безкоштовної програми Win32DiskImager без проблем вдається створити завантажувальний флешку CentOS 7. Ось як це робиться:

Викачуємо дистрибутив програми.

Викачуємо дистрибутив CentOS 7. Я для тесту використовував версію Minimal.

Вставляємо чисту флешку, запускаємо програму і вказуємо



налаштування:

Натискаємо Write і чекаємо закінчення запису.

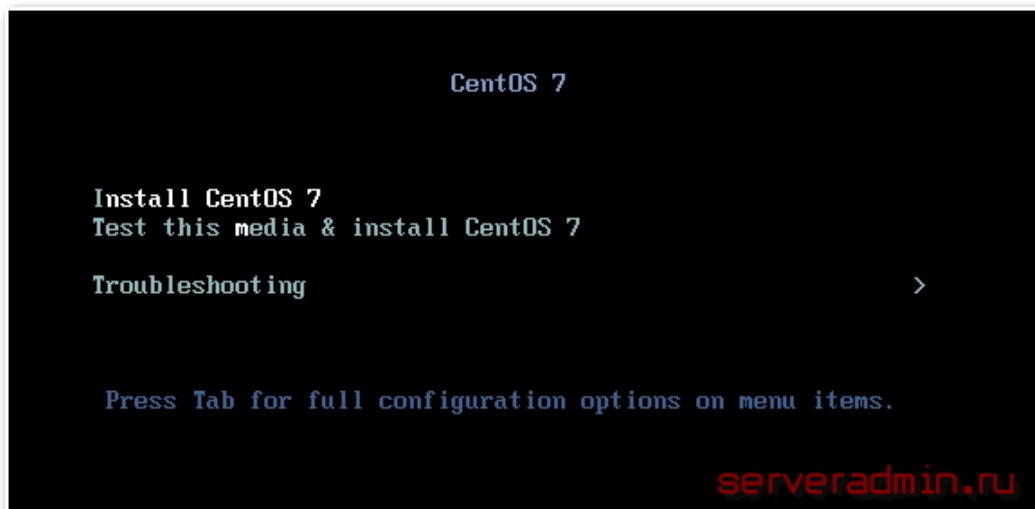
Завантажувальна флешка CentOS 7 готова.

Цього достатньо для створення флешки. Тепер їй можна користуватися для установки операційної системи з флешки.

1.1.5 Установка CentOS 7 з флешки

Після створення завантажувальної флешки, можна приступити до установки. Вставляємо флешку в сервер, вказуємо як джерело завантаження

USB і запускаємо комп'ютер. Нас зустрічає початкове меню установки CentOS:



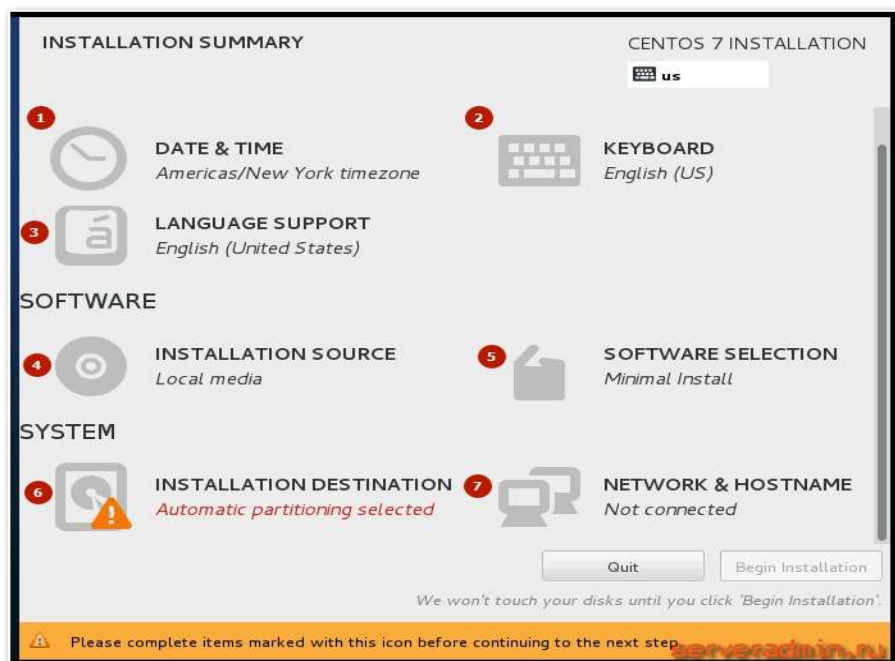
Вибираємо перший пункт: Install CentOS 7 і тиснемо **enter**. Після завантаження інсталлера, нас зустрічає вікно з вибором мови, який буде використовуватися під час установки.



Далі завантажуються сторінка з вибором основних параметрів установки. Вона вже буде відрізнятися в залежності від типу ISO образу, з якого відбувається установка CentOS.

1.1.6 CentOS 7 minimal

Якщо ви використовуєте диск centos minimal iso, то побачите наступний екран:



Тут вам пропонують вказати параметри установки. Оклику позначений розділ, без настройки якого продовження неможливо. Для настройки доступні наступні параметри установки:

- Вибір **часового поясу centos**.
- Вибір розкладки клавіатури.

Підтримка яких мов буде здійснюватися на сервері.

Звідки буде відбуватися установка. Так як у нас дистрибутив centos minimal, установка буде з локального iso.

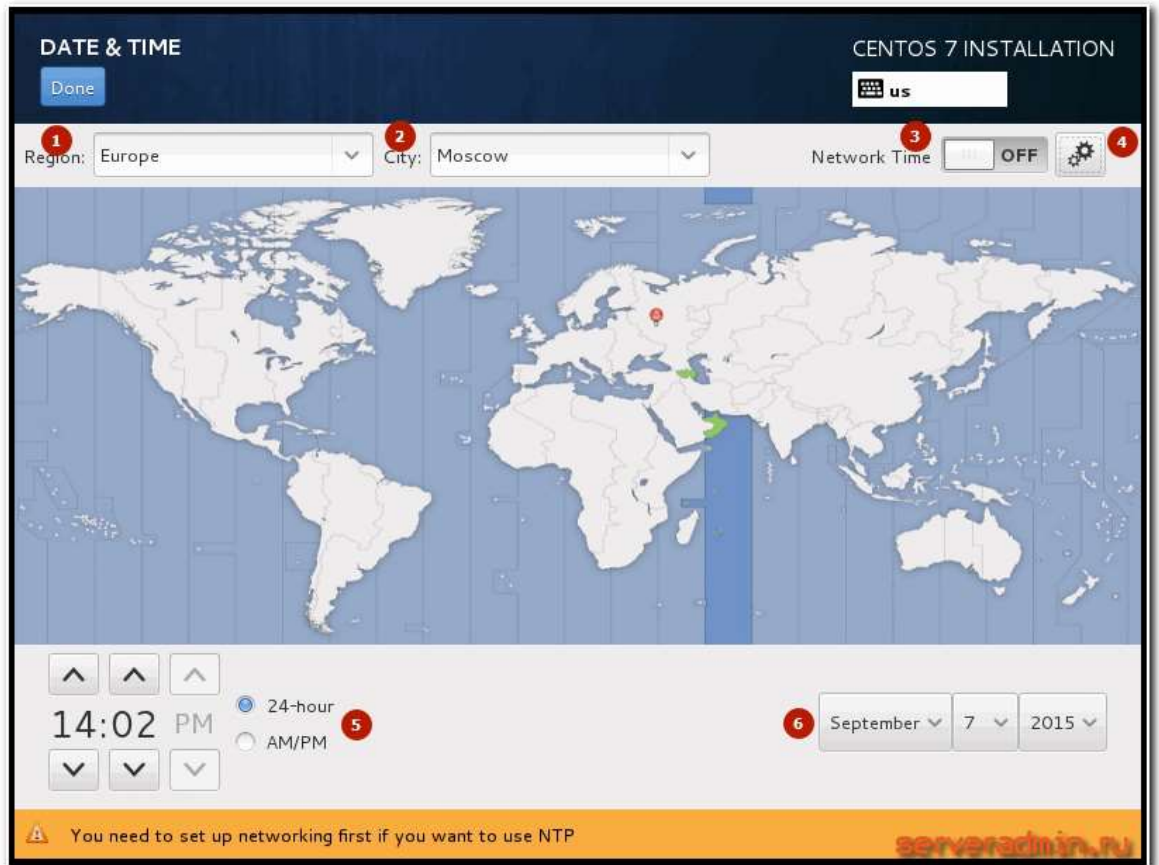
Вибір пакетів для установки. В образі minimal доступний тільки мінімальний набір софту.

Розбивка жорсткого диска. Детальніше торкнемося цього пункту, коли будемо розбирати установку на raid.

Налаштування мережевих інтерфейсів.

Для продовження установки необхідно виконати як мінімум розбивку жорсткого диска. Без цього продовження установки неможливо. Але ми пройдемося по всіх параметрах і встановимо необхідні для нас значення.

Отже, натискаємо на **DATE & TIME** і налаштуємо параметри часу:

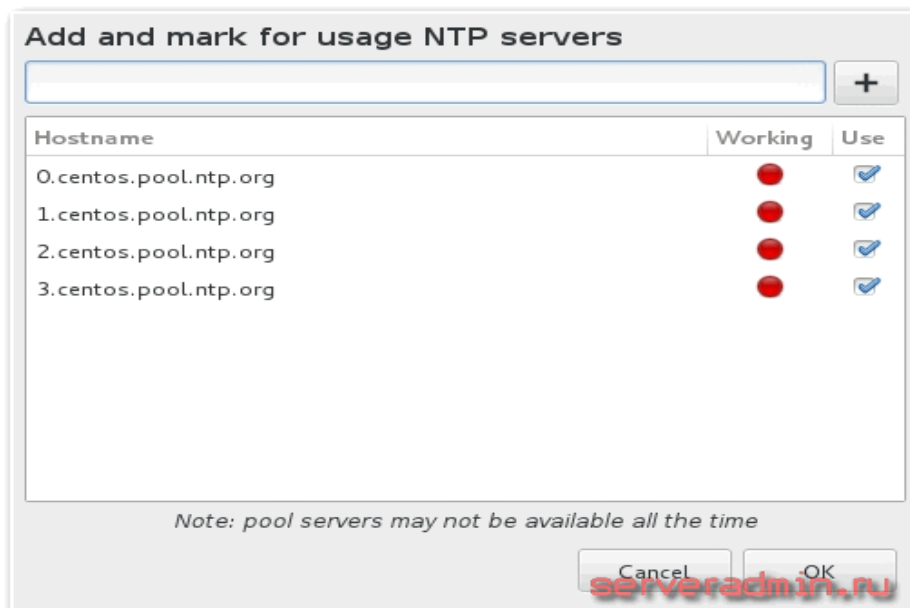


Вказуємо регіон.

Вибираємо місто.

Включаємо при необхідності службу часу для синхронізації годин centos з зовнішніми серверами. Це можливо зробити тільки якщо ви вже налаштували мережеві параметри. Якщо немає, то поверніться до цієї налаштуванні пізніше.

Вибираємо список зовнішніх серверів для синхронізації часу:

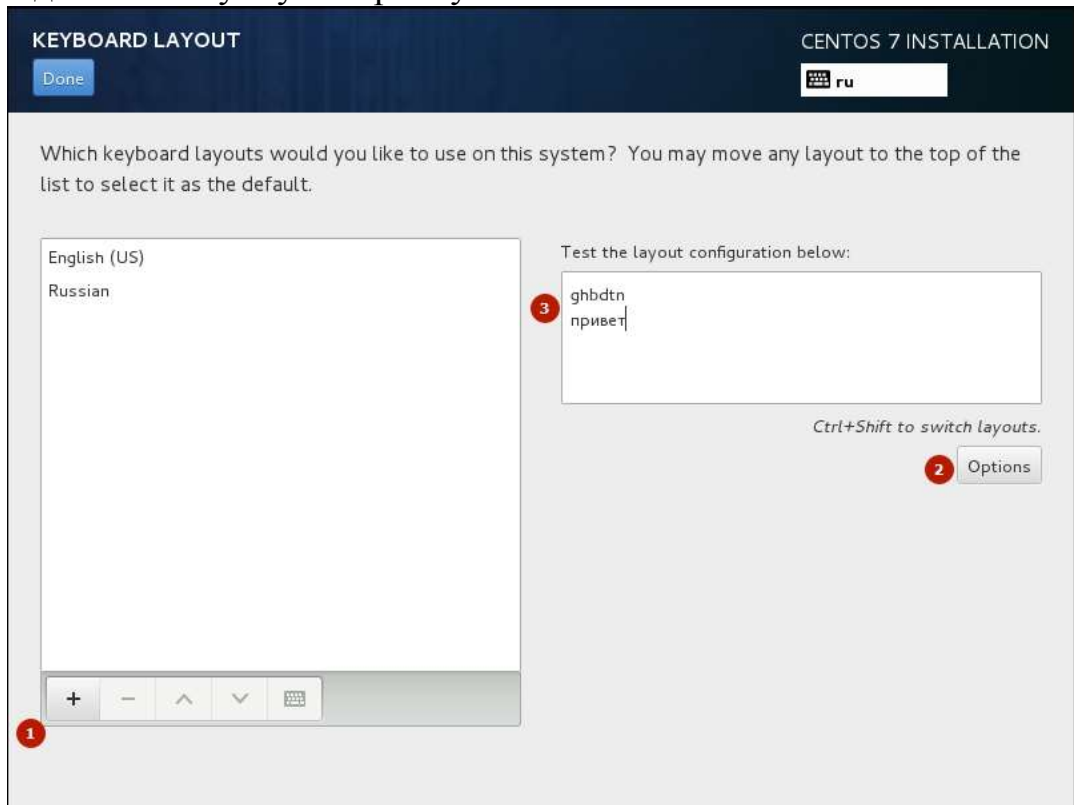


Вказуємо формат, в якому буде відображатися поточний час.

При необхідності змінюємо дату сервера.

Після завершення налаштувань тиснемо зверху синю кнопку **Done**.

Йдемо в наступну настройку -**KEYBOARD LAYOUT**:

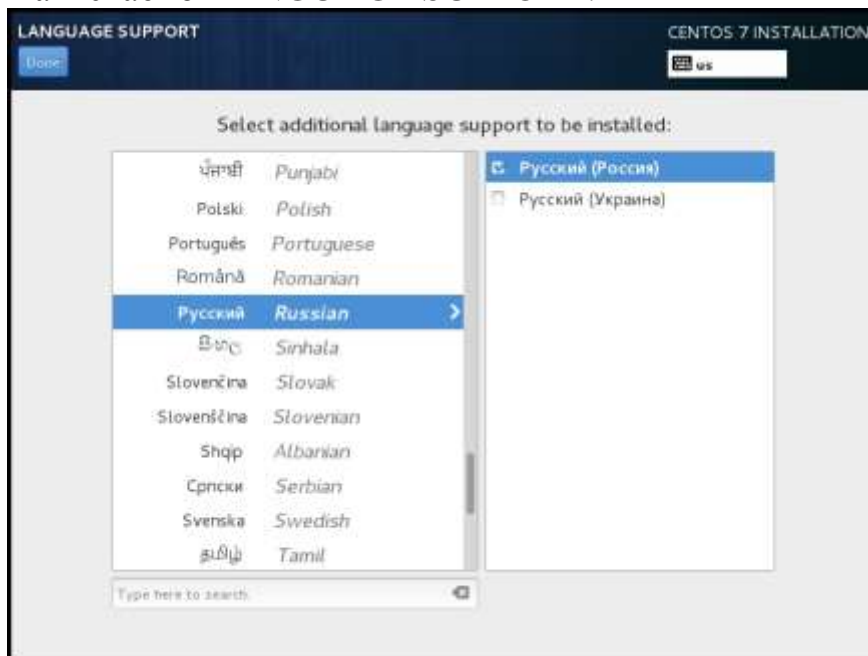


Додаємо необхідні розкладки.

натискаємо **Options** і вибираємо, як буде відбуватися перемикання розкладок.

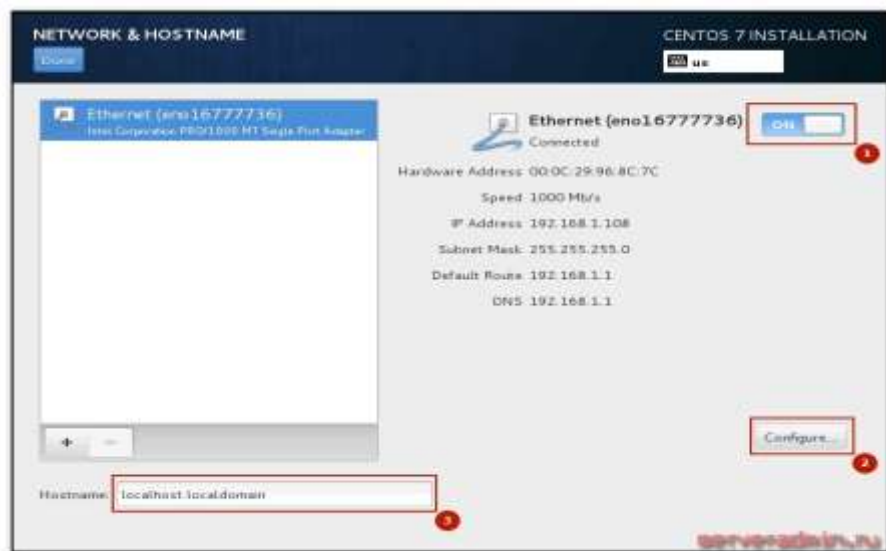
Тестуємо розкладки і перемикання. Якщо все в порядку, йдемо далі.

Натискаємо **LANGUAGE SUPPORT**:



Вибираємо додаткові мови, які буде підтримувати система. Якщо у вас, наприклад, сервер налаштовується для роботи в якості шлюзу, підтримка додаткових мов швидше за все не стане в нагоді. Після вибору знову тиснемо **Done**.

Тепер виконаємо мережеві настройки. Йдемо в розділ **NETWORK & HOSTNAME**. Включаємо повзунок в положення **ON** і отримуємо автоматично настройки по **dhcp**:

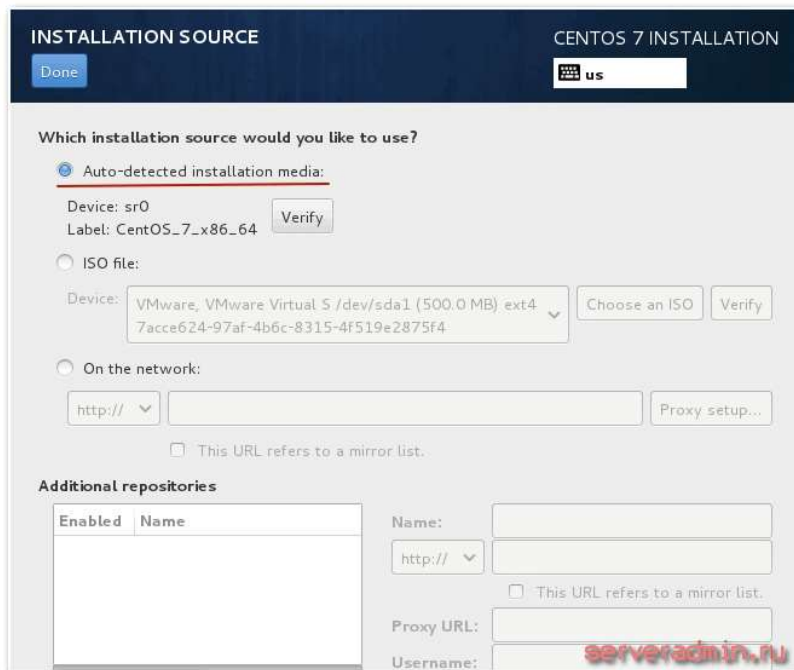


Включення повзунка в положення **ON** активує інтерфейс, він отримує настройки по **dhcp**.

Якщо ви хочете змінити ці настройки, натискаєте **Configure**, вказуєте **hostname**. Якщо забудете, то після установки цей параметр можна змінити.

Завершуємо налаштування натисканням на **Done**. Тепер можна повернутися в налаштування годин і активувати **Network Time**.

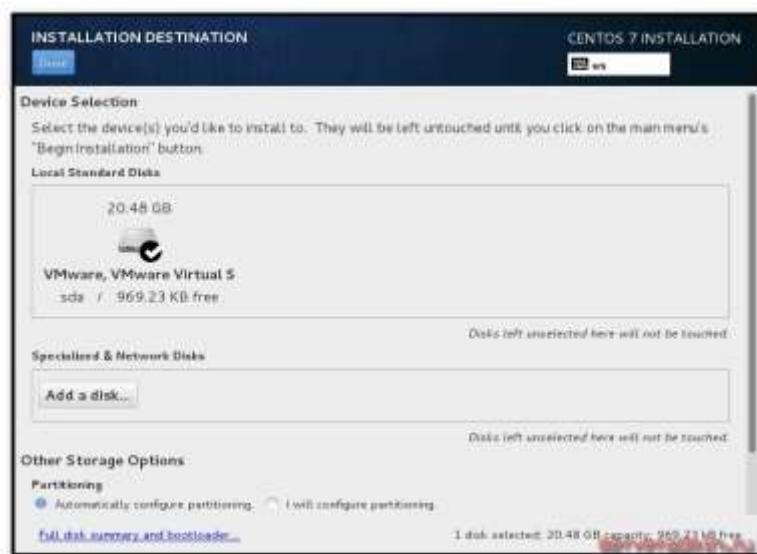
Тепер перейдемо до розділу **INSTALLATION SOURCE**. При установці **centos minimal** міняти цей параметр немає необхідності. Там по-замовчуванню встановлені локальні джерела, нам це підходить. Можна нічого не чіпати:



В розділі **SOFTWARE SELECTION** при **minimal** установці теж нічого вибрати, вже зазначений єдино можливий варіант:



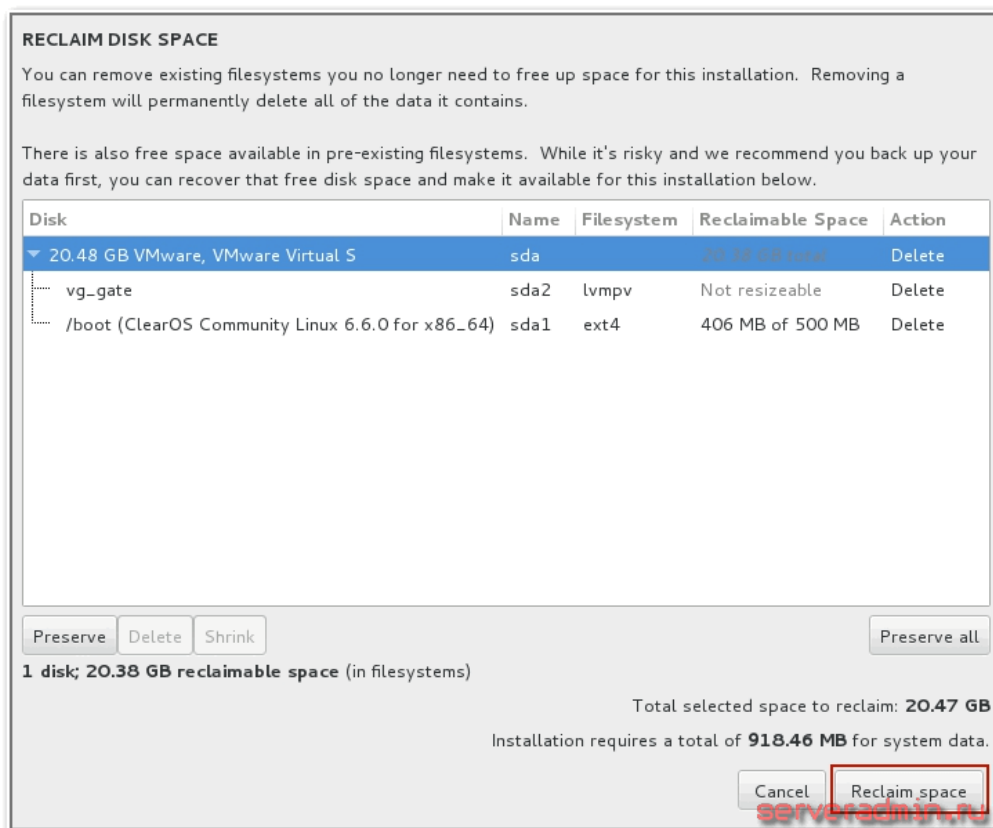
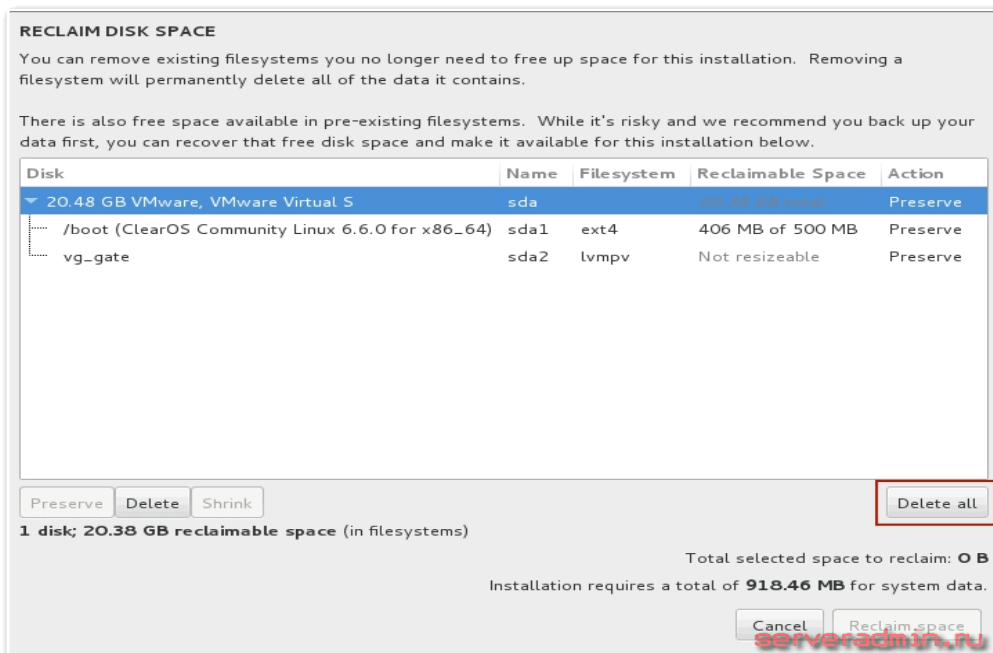
Нам залишилося розглянути останню обов'язкову настройку, без якої установка centos не почнеться - **INSTALLATION DESTINATION**. Зайшовши в неї ви побачите список підключених до сервера дисків. У нашому випадку це один жорсткий диск.



Якщо ваш диск визначився правильно, вибираєте його і натискаєте **Done**. Вискакує віконце з попередженням, що для установки системі знадобиться приблизно 1 Гб місця на жорсткому диску, а на жорсткому диску немає необхідного вільного місця. Це відбувається тому, що раніше на цьому диску була встановлена інша система і вона займала весь жорсткий диск. Нам потрібно видалити всю стару інформацію для установки нової системи. Робимо це, натискаючи **Reclaim space**:



Вибираємо диск і видаляємо всі існуючі розділи на ньому - тиснемо спочатку Delete all, а потім Reclaim space:



Після цього установник вибере весь диск в якості диска для установки. Після цього можна почати безпосередньо установку centos, натиснувши кнопку Begin Installation.

Далі розглянемо варіант, коли ви виконуєте установку з образу netinstall.

1.1.7 CentOS 7 netinstall

Установка Centos 7 з образу netinstall відрізняється від інших тільки одним моментом. Під час підготовки до установки по мережі в розділі **INSTALLATION SOURCE** вам замість локального джерела необхідно вказати шлях до образу, розташованому десь в мережі. Очевидно, що перед цим необхідно виконати настройку мережі та переконатися, що у комп'ютера є доступ в інтернет.

На скріншоті вказано старий url. Після низки оновлень він став неактуальний. Правильна посилання вище в тексті.

INSTALLATION SOURCE

CENTOS 7 INSTALLATION

Done us Help!

Which installation source would you like to use?

ISO file:

Device: VMware, VMware Virtual S /dev/sda1 (500 MiB MB) ext4
7acce624-97af-4b6c-8315-4f519e2875f4

Choose an ISO Verify

On the network:

http:// mirror.yandex.ru/centos/7.1.1503/os/x86_64 Proxy setup...

This URL refers to a mirror list.

Additional repositories

Enabled	Name
---------	------

Name: http://

This URL refers to a mirror list.

Proxy URL:

User Name:

Password:

serveradmin.ru

Вказуємо шлях і тиснемо **Done**. Після перевірки доступності джерела, в розділі **SOFTWARE SELECTION** можна вибрати необхідний для установки набір софта:

SOFTWARE SELECTION

CENTOS 7 INSTALLATION

Done us Help!

Base Environment

- Minimal Install**
Basic functionality.
- Compute Node**
Installation for performing computation and processing.
- Infrastructure Server**
Server for operating network infrastructure services.
- File and Print Server**
File, print, and storage server for enterprises.
- Basic Web Server**
Server for serving static and dynamic internet content.
- Virtualization Host**
Minimal virtualization host.
- Server with GUI**
Server for operating network infrastructure services, with a GUI.
- GNOME Desktop**

Add-Ons for Selected Environment

- Debugging Tools**
Tools for debugging misbehaving applications and diagnosing performance problems.
- Compatibility Libraries**
Compatibility libraries for applications built on previous versions of CentOS Linux.
- Development Tools**
A basic development environment.
- Security Tools**
Security tools for integrity and trust verification.
- Smart Card Support**
Support for using smart card authentication.

serveradmin.ru

Далі можна починати установку, вказавши всі інші параметри, які ми розглянули вище.

1.1.8 CentOS 7 установка на raid

Тепер розглянемо найбільш складний варіант установки. Ми будемо ставити CentOS 7 на програмний рейд. Для прикладу візьмемо 2 диска і raid 1. Всі установки будуть такі ж, як ми розглянули раніше, крім однієї - **INSTALLATION DESTINATION**.

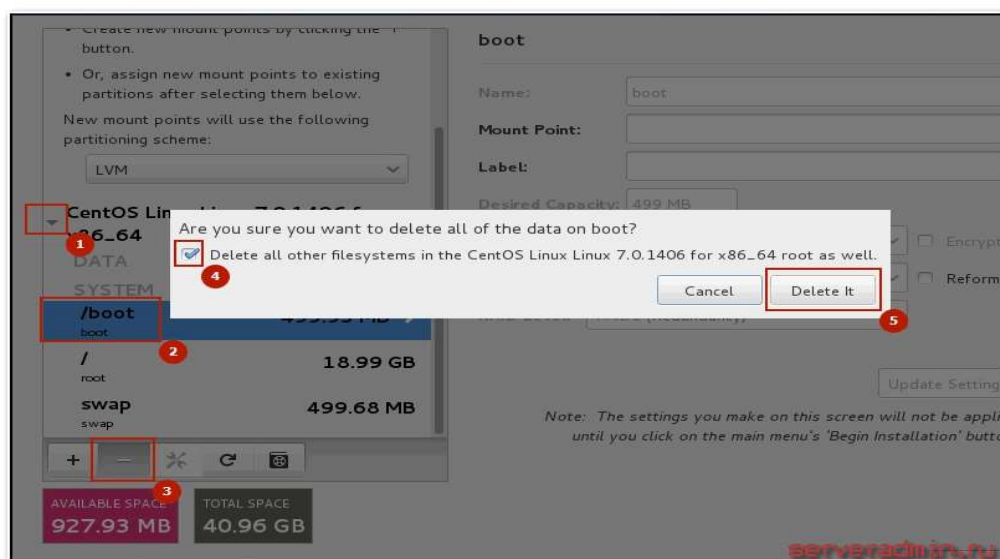
Підключаємо 2 диска до системи, завантажується з інсталяційного диска і йдемо в розділ розбивки диска. Бачимо 2 жорстких диска. Вибираємо обидва і відзначаємо пункт I will configure partition:



Тиснемо **Done**. Відкривається трохи криве вікно управління розділами жорсткого диска.

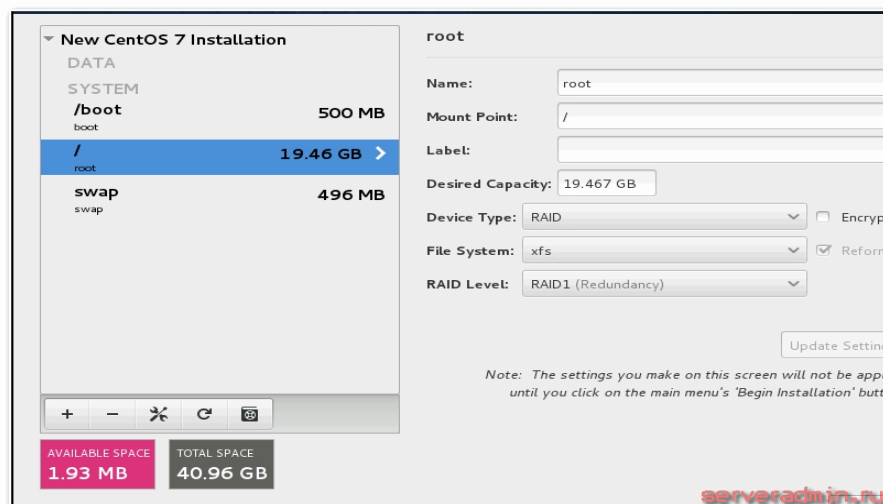


Тут ми насамперед видаляємо всі існуючі розділи:

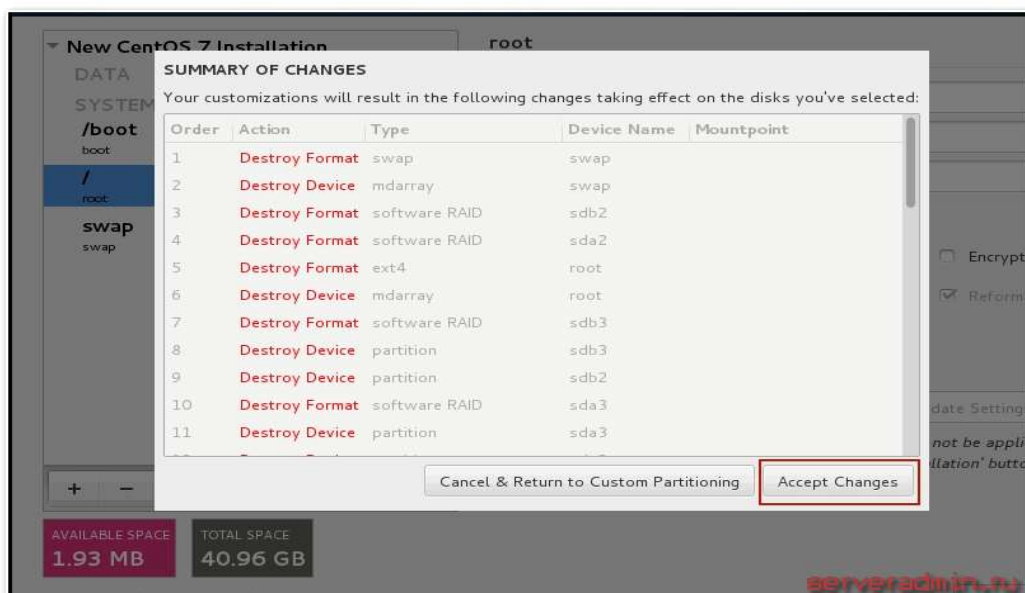


І створюємо свої нові, натискаючи плюсики. Нам потрібно створити 3 розділу: **boot**, **swap** і корінь **/**. Розміри розділів вибирайте самі, виходячи зі своїх потреб і оперативної пам'яті на сервері. Для розділу **boot** досить 500Мб, для **swap** вистачить обсягу оперативної пам'яті. Все інше можна або заповнити одним кореневим розділом, або створити кілька, якщо в цьому є необхідність. Device Type ставимо **RAID**. Тип файлової системи вибирайте на свій розсуд. Що краще – **xfs** або **ext4** залежить від конкретної ситуації. Вважається, що **xfs** працює краще з великими файлами, **ext4** з купою дрібних. Це тема окремої розмови. RAID Level вказуємо **RAID1**.

Має вийти приблизно так:



натискаємо **Done**, Коли закінчимо. У новому вікні підтверджуємо розбивку диска, натискаючи **Accept Changes**:



Всі інші параметри виставляємо як було розказано вище. Тепер можна починати установку CentOS 7 на програмний raid, який ми тільки що створили.

Під час установки потрібно вказати пароль **root**, або створити додаткових користувачів. Пароль рекомендую встановити, користувачів по необхідності.



Після завершення установки на raid зайдемо в систему і перевіримо стан масиву:

```
# df -h
```

```
# cat /proc/mdstat
```

```
[root@centos ~]# df -h
Filesystem      Size  Used Avail Use% Mounted on
/dev/md126      20G  774M   19G   4% /
devtmpfs        488M    0  488M   0% /dev
tmpfs           494M    0  494M   0% /dev/shm
tmpfs           494M  6.6M  487M   2% /run
tmpfs           494M    0  494M   0% /sys/fs/cgroup
/dev/md125      497M   97M  400M  20% /boot
[root@centos ~]# cat /proc/mdstat
Personalities : [raid1]
md125 : active raid1 sda1[0] sdb1[1]
        511936 blocks super 1.0 [2/2] [UU]

md126 : active raid1 sda3[0] sdb3[1]
        19934080 blocks super 1.2 [2/2] [UU]
        bitmap: 0/1 pages [0KB], 65536KB chunk

md127 : active raid1 sda2[0] sdb2[1]
        507584 blocks super 1.2 [2/2] [UU]

unused devices: <none>
```

Подивимося інформацію про кореневий масиві:
mdadm -D / dev / md126

```
[root@centos ~]# mdadm -D /dev/md126
/dev/md126:
  Version : 1.2
  Creation Time : Tue Sep  8 15:35:30 2015
  Raid Level : raid1
  Array Size : 19934080 (19.01 GiB 20.41 GB)
  Used Dev Size : 19934080 (19.01 GiB 20.41 GB)
  Raid Devices : 2
  Total Devices : 2
  Persistence : Superblock is persistent

  Intent Bitmap : Internal

  Update Time : Tue Sep  8 15:51:13 2015
  State : active
  Active Devices : 2
  Working Devices : 2
  Failed Devices : 0
  Spare Devices : 0

  Name : localhost:root
  UUID : 2b386997:f1445c4b:7f899bc4:af907f5b
  Events : 59

  Number Major Minor RaidDevice State
    0         8      3        0 active sync /dev/sda3
    1         8     19        1 active sync /dev/sdb3
```

Все в порядку, установка сервера закінчена. Рейд розділ нормально функціонує, забезпечуючи відмовостійкість сервера.

Рекомендується використовувати софтовий raid Лінукса mdadm в повсякденній діяльності. Він набагато надійніше, зрозуміліше і стабільніше вбудованих в материнську плату контролерів. Віддавати перевагу апаратному рейду слід тільки в тому випадку, якщо він дійсно апаратний, він має батарею і він дійсно збільшує продуктивність сервера. У всіх інших випадках рекомендую користуватися mdadm.

1.2 Робота з жорсткими дисками в Linux.

1.2.1 Вступ

Операційна система містить стандартний набір папок. Зараз звернемо увагу на ті папки, в які можуть бути змонтовані окремі розділи. Це означає, що якщо ми бачимо в кореневому розділі якусь папку, то не факт, що ця папка знаходиться на тому ж жорсткому диску, що і сусідні папки або навіть сама коренева файлова система. Це можуть бути окремі жорсткі диски або розділи жорстких дисків, які змонтовані в кореневу файлову систему. Найбільший розділ - це сама коренева файлова система, позначається символом "/". Наступний розділ, який зазвичай знаходиться окремо - це "/ boot", завантажувальний розділ, зазвичай він мегабайт на 100. Там зберігаються файли необхідні для завантаження операційної системи і саме ядро. Можна дану директорію залишити на кореневому розділі, але якщо у нас великий жорсткий диск хоча б на 1 ТБ, то старі диски його можуть не побачити при

завантаженні, тому хороший тон створювати окремий розділ на жорсткому диску розміром від 100 МБ.

Директорія `"/ home"` - це домашні папки користувачів. Це такий собі аналог папки в операційній системі Windows, такий як Documents and Settings або `c: \ users`. Це директорія, де зберігаються всі папки користувача. Його можна монтувати, як мережеву папку. Наприклад, якщо у вас використовуються якісь переміщені профілі. Користувач працює на декількох комп'ютерах і йому необхідно, щоб скрізь був однаковий робочий стіл. Ось в такому випадку було б правильно зберігати його домашню папку де-небудь на сервері і просто її монтувати як мережеву папку, надаючи доступ до файлів.

Наступна папка, яка знаходиться в корені - це `"/ root"` Вона є домашньою текою суперкористувача, дуже важлива папка тому вона зберігається окремо.

Папка `"/ etc"` - в ній знаходиться конфігурація нашої операційної системи і її компонент. Так само буває, що вона знаходиться на окремих дискових розділах. Ми звичайно самі визначаємо при установці як розбити жорсткий диск, але, якщо при установці ми ставимо галочку в установнику, щоб він автоматично розбив жорсткий диск і створив розділи так, як він вважає за потрібне.

Директорія `"/ opt"`. В даній директорії знаходиться програмне забезпечення від третіх постачальників. Деякі серверні дистрибутиви теж зберігають її на окремому логічному розділі.

Директорія `"/ var"`. Тут зберігаються всі часто мінливі дані. Наприклад, логи різного програмного забезпечення або змінюються програмні дані. Можливо для цієї теки має сенс використовувати швидкі диски, наприклад, SSD. Тому, що до них буде йти дуже часте звертання програмного забезпечення.

Директорія `"/ usr"` В ній знаходяться всі встановлені пакети програм, документація і вихідний код ядра. Найчастіше дана директорія змонтована взагалі в режимі "і в принципі вона може бути розташована взагалі на якому ні будь повільному диску або папці в мережі.

Директорія `"/ tmp"` Призначена для зберігання тимчасових файлів. Важливою особливістю даної папки є те, що зберігаються файли в ній, будуть видалені в разі перезавантаження машини. Тобто при перезавантаженні дана папка очищається автоматично. Існує ще один окремий розділ підкачки `swap`. Зазвичай операційна система windows використовує файл підкачки, то Linux найчастіше використовує розділ підкачки, хоча може використовувати і файл. Даний розділ не монтується в нашу файлову систему, він просто існує окремо.

1.2.2 Вигляд у консолі.

Переходимо в кореневу директорію `cd /`. вводим команду `ls` бачимо, як все директорії у нас відображаються. Всі ті папки про які йшла мова ми бачимо знаходяться в кореневій директорії, крім розділу `swap`. Для того, щоб

подивитися, що і куди змонтовано, то необхідно подивитися спеціальний конфігураційний файл `cat / etc / fstab`

```
root@jenkins: /
root@jenkins: /# root@jenkins: /# ls
bin    cdrom  etc    lib    lib64  lost+found  mnt  proc  run  snap  swap.
boot  dev    home  lib32  libx32  media      opt  root  sbin  srv   sys
root@jenkins: /#
```

Для спрощення пояснення, ми під жорстким диском будемо розуміти єдиний простір, яке ми можемо розбити на кілька частин. Спочатку, комп'ютер не бачив більше 4-х розділів - цього вважалося досить. Тому зараз, спочатку за замовчуванням жорсткий диск не може бути розбитий на більш ніж 4 розділу. Якщо нам необхідно більше розділів, то необхідно створити так званий розширений розділ. І цей додатковий розділ вже буде містити в собі кілька логічних розділів.

У старих версіях Linux диски називалися **hd0**, **hd1** і т.д, зараз жорсткі диски називаються **sda**, Тобто літерами. USB пристрої у нас так само ідентифікуються як жорсткі диски. Розділи у нас нумеруються по порядку **sda1**, **sda2** і т.д. Тобто назва розділу у нас складається з букви диска і номера розділу по порядку. Отже, перші 4 цифри зарезервовані і даються тільки основних розділів, саме тому логічні розділи нумерація починається з 5-ки. Основний утилітою для роботи з розділами є [утиліта fdisk](#). Це утиліта використовується для розбиття жорсткого диска. Так само є утиліта **mkfs**. Дана утиліта використовується для створення файлової системи. Ми можемо подивитися всі існуючі жорсткі диски командою **fdisk -l**. При виведенні даної команди ми можемо бачити, що у нас 2 підключених жорстких диска **sda** і **sdb**. У висновку ми можемо так само побачити їх фізичний обсяг. На першому диску **sda**, ми так само можемо побачити 2 розділу **sda1** і **sda2**.

```
Disk /dev/sda: 100 GiB, 107374182400 bytes, 209715200 sectors
Disk model: Virtual disk
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disklabel type: gpt
Disk identifier: 3E634F74-7E54-4440-868E-CCBF436A8706

Device      Start      End      Sectors  Size Type
/dev/sda1   2048       4095     2048     1M BIOS boot
/dev/sda2   4096 209713151 209709056 100G Linux filesystem

Disk /dev/sdb: 20 GiB, 21474836480 bytes, 41943040 sectors
Disk model: Virtual disk
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
root@jenkins: /#
```

Щоб почати працювати з другим жорстким диском, необхідно спочатку вказати його ім'я. Пристрої лежать в директорії `/ dev`. Пишемо команду **fdisk / dev / sdb**.

Виходить наступна картина:

```

root@jenkins:/# fdisk /dev/sdb

Welcome to fdisk (util-linux 2.34).
Changes will remain in memory only, until you decide to write them.
Be careful before using the write command.

Device does not contain a recognized partition table.
Created a new DOS disklabel with disk identifier 0xa120c273.

Command (m for help): m

Help:

DOS (MBR)
 a  toggle a bootable flag
 b  edit nested BSD disklabel
 c  toggle the dos compatibility flag

Generic
 d  delete a partition
 F  list free unpartitioned space
 l  list known partition types
 n  add a new partition
 p  print the partition table
 t  change a partition type
 v  verify the partition table
 i  print information about a partition

Misc
 m  print this menu
 u  change display/entry units
 x  extra functionality (experts only)

Script
 I  load disk layout from sfdisk script file
 O  dump disk layout to sfdisk script file

Save & Exit
 w  write table to disk and exit
 q  quit without saving changes

Create a new label
 g  create a new empty GPT partition table
 G  create a new empty SGI (IRIX) partition table
 o  create a new empty DOS partition table
 s  create a new empty Sun partition table

```

А натиснувши букву **m** ми можемо отримати доступ до довідки по роботі з даною утилітою. Як можна помітити функціонал утиліти досить великий. Можна додавати партіції, можна видаляти партіції. Наступним кроком необхідно створити новий розділ. Вибираємо опцію **n**. Далі пропонується вибір **primary** або **extended**. Ми вибираємо **primary** ключ **p**. Далі вибираємо номер розділу 1-4. Наприклад, 1. Далі система запитує де буде (на якому секторі) починатися розмітка розділу. Можна вибрати за замовчуванням. На наступному кроці ми можемо вказати сектор, але це вкрай незручно, простіше вказати скільки ми хочемо виділити під розмір, наприклад, + 10G. І цей розділ стане 10 гігабайт.

```

q  quit without saving changes

Create a new label
 g  create a new empty GPT partition table
 G  create a new empty SGI (IRIX) partition table
 o  create a new empty DOS partition table
 s  create a new empty Sun partition table

Command (m for help): n
Partition type
  p  primary (0 primary, 0 extended, 4 free)
  e  extended (container for logical partitions)
Select (default p): p
Value out of range.
  p  primary (0 primary, 0 extended, 4 free)
  e  extended (container for logical partitions)
Select (default p): 1
Value out of range.
  p  primary (0 primary, 0 extended, 4 free)
  e  extended (container for logical partitions)
Select (default p): p
Partition number (1-4, default 1): 1
First sector (2048-41943039, default 2048):
Last sector, +/-sectors or +/-size{K,M,G,T,P} (2048-41943039, default 41943039): +10GB
Last sector, +/-sectors or +/-size{K,M,G,T,P} (2048-41943039, default 41943039): +10G

Created a new partition 1 of type 'Linux' and of size 10 GiB.

Command (m for help): _

```

Спробуємо другий варіант зі створенням розділу, він буде розширений (**extended**). Вибираємо ключ **e**, вибираємо 2 розділ. Вибираємо з якого сектора він почнеться. Наступним кроком **+** 8G. Ми створили розширений диск на 8 GB. далі створюємо **sdb3** на що залишилися 2 GB. А також 8GB розширеного диска розбиваємо на 2 логічних по 4 GB кожен. Всі операції однакові. В після створення останнього розділу вибираємо опцію **w**. Яка записує всі зміни.

В підсумку маємо наступну картину

```
Disk /dev/sda: 100 GiB, 107374182400 bytes, 209715200 sectors
Disk model: Virtual disk
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disklabel type: gpt
Disk identifier: 3E634F74-7E54-4440-868E-CCBF436A8706

Device      Start      End      Sectors  Size Type
/dev/sda1   2048      4095      2048     1M BIOS boot
/dev/sda2   4096 209713151 209709056 100G Linux filesystem

Disk /dev/sdb: 20 GiB, 21474836480 bytes, 41943040 sectors
Disk model: Virtual disk
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disklabel type: dos
Disk identifier: 0x19160bb9

Device      Boot      Start      End      Sectors  Size Id Type
/dev/sdb1                   2048 20973567 20971520  10G 83 Linux
/dev/sdb2                   20973568 37750783 16777216   8G  5 Extended
/dev/sdb3                   37750784 41943039  4192256   2G 83 Linux
/dev/sdb5                   20975616 29364223  8388608   4G 83 Linux
/dev/sdb6                   29366272 37750783  8384512   4G 83 Linux

Partition table entries are not in disk order.
root@jenkins:/#
```

Перш ніж використовувати ці розділи їх необхідно відформатувати. Якщо подивитися на висновок команди, ми можемо побачити ще одне цікаве поле з інформацією. **Id** - це мітка цього розділу. Необхідно розібратися, як ці **id** міняти. Використовуємо знову утиліту **fdisk / dev / sdb**. Далі використовуємо ключ **t**. Утиліта пропонує вибрати номер розділу. Вибираємо 1. Потім необхідно ввести **id** в шістнадцятковому форматі, список всіх варіацій і їх опис можна вивести командою **L**.

```

root@jenkins: /
Command (m for help): t
Partition number (1-3,5,6, default 6): 1
Hex code (type L to list all codes): L

```

0	Empty	24	NEC DOS	81	Minix / old Lin	bf	Solaris
1	FAT12	27	Hidden NTFS Win	82	Linux swap / So	c1	DRDOS/sec (FAT-
2	XENIX root	39	Plan 9	83	Linux	c4	DRDOS/sec (FAT-
3	XENIX usr	3c	PartitionMagic	84	OS/2 hidden or	c6	DRDOS/sec (FAT-
4	FAT16 <32M	40	Venix 80286	85	Linux extended	c7	Syrinx
5	Extended	41	PPC PReP Boot	86	NTFS volume set	da	Non-FS data
6	FAT16	42	SFS	87	NTFS volume set	db	CP/M / CTOS / .
7	HPFS/NTFS/exFAT	4d	QNX4.x	88	Linux plaintext	de	Dell Utility
8	AIX	4e	QNX4.x 2nd part	8e	Linux LVM	df	BootIt
9	AIX bootable	4f	QNX4.x 3rd part	93	Amoeba	e1	DOS access
a	OS/2 Boot Manag	50	OnTrack DM	94	Amoeba BBT	e3	DOS R/O
b	W95 FAT32	51	OnTrack DM6 Aux	9f	BSD/OS	e4	SpeedStor
c	W95 FAT32 (LBA)	52	CP/M	a0	IBM Thinkpad hi	ea	Rufus alignment
e	W95 FAT16 (LBA)	53	OnTrack DM6 Aux	a5	FreeBSD	eb	BeOS fs
f	W95 Ext'd (LBA)	54	OnTrackDM6	a6	OpenBSD	ee	GPT
10	OPUS	55	EZ-Drive	a7	NeXTSTEP	ef	EFI (FAT-12/16/
11	Hidden FAT12	56	Golden Bow	a8	Darwin UFS	f0	Linux/PA-RISC b
12	Compaq diagnost	5c	Priam Edisk	a9	NetBSD	f1	SpeedStor
14	Hidden FAT16 <3	61	SpeedStor	ab	Darwin boot	f4	SpeedStor
16	Hidden FAT16	63	GNU HURD or Sys	af	HFS / HFS+	f2	DOS secondary
17	Hidden HPFS/NTF	64	Novell Netware	b7	BSDI fs	fb	VMware VMFS
18	AST SmartSleep	65	Novell Netware	b8	BSDI swap	fc	VMware VMKCORE
1b	Hidden W95 FAT3	70	DiskSecure Mult	bb	Boot Wizard hid	fd	Linux raid auto
1c	Hidden W95 FAT3	75	PC/IX	bc	Acronis FAT32 L	fe	LANstep
1e	Hidden W95 FAT1	80	Old Minix	be	Solaris boot	ff	BBT

```

Hex code (type L to list all codes):

```

Можна помітити, що всюди стояв за замовчуванням 83 тобто **linux**, Ми можемо змінити на будь-яку мітку, наприклад, на 86 NTFS - windows розділ. З розділ помітимо a5. Записуємо зміни w. І можна все зміни побачити через **fdisk -l**. Далі, щоб користуватися цими розділами, нам спочатку їх треба відформатувати. Щоб це зробити використовуємо утиліту **mkfs**. Набираємо **mkfs.ext4**, після точки вказуємо цільову, буде переформатовано розділ, а через пробіл власне сам розділ. наприклад **mkfs.ext4 / dev / sdb5**.

Тепер стосовно розподілу підкачки **swap**.

Для роботи з ним використовується **swapon** і **swapoff**. Через першу команду ми можемо включати розділ підкачки, додавати розділи підкачки, а через другу команду вимикати. Так само є утиліта **mkswap** - яка дозволяє створювати тобто формувати розділ, як розділ підкачки. А також у нас є конфігураційний файл, розташований в наступному місці **/ etc / fstab**, В даному файлі конфігурації описується монтування файлових систем. Відповідно можна примонтировать розділи в ручному режимі, але якщо ми хочемо, щоб при завантаженні розділи самі монтувалися, то необхідно конфігурувати файл **/ etc / fstab** / командою **swapon -s**, Ми можемо подивитися інформацію по своп розділу.

```

root@jenkins:/# swapon -s
Filename                                Type    Size    Used    Priority
/swap.img                               file   4194300 0        -2
root@jenkins:/#

```

Щоб додати новий своп розділ, то нам необхідно через команду `fdisk` створити новий розділ і вказати, що він саме своповській розділ, потім його необхідно отформатувати і примонтувати.

Командою `cat / etc / fstab` ми можемо подивитися, які розділи монтуються при завантаженні. Виглядає це приблизно так:

```

root@jenkins:/# cat /etc/fstab
# /etc/fstab: static file system information.
#
# Use 'blkid' to print the universally unique identifier for a
# device; this may be used with UUID= as a more robust way to name devices
# that works even if disks are added and removed. See fstab(5).
#
# <file system> <mount point> <type> <options> <dump> <pass>
# / was on /dev/sda2 during curtin installation
/dev/disk/by-uuid/50cee6ca-cbd2-454b-a835-2614bf2e9d5d / ext4 defaults 0 0
/swap.img none swap sw 0 0
root@jenkins:/#

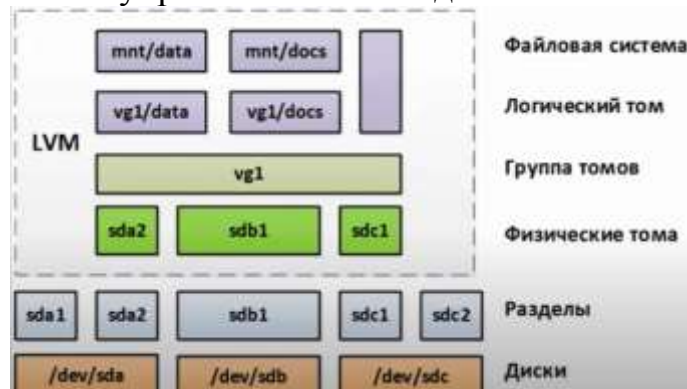
```

Ми бачимо, що ось цей розділ з uuid `50cee6ca-cbd2-454b-a835-2614bf2e9d5d` монтується в корінь, знак `/` про це говорить і має файловою системою `ext4`. Uuid дивимося за допомогою команди `blkid`. Розділ підкачки, як і файл підкачки - це місце на диску, яке використовується, як RAM. Якщо у нас не вистачає оперативної пам'яті, то комп'ютер може користуватися цим розділом, як оперативною пам'яттю.

Створимо нову директорію командою `mkdir mounted` і змонтуємо туди розділ. Наприклад, `/ dev / sdb5`. Використовуємо команду `mount / dev / sdb5 / mounted`. Щоб отмонтировать використовуємо команду `umount / mounted`.

1.3 LVM

LVM - це система управління томами для Linux.



Вона дозволяє створити поверх фізичних розділів, логічні розділи, Logical Volume, які будуть видні операційній системі, як звичайні блокові пристрої з томами.

переваги:

Ми можемо в одну групу **Logical Volume** можемо додати різну кількість фізичних дисків.

Ми можемо змінювати розміри цих розділів прям під час роботи операційної системи.

В даному випадку на картинці є, 3 HDD, на кожному є певна кількість розділів. І ми можемо з них зібрати групу томів **vg1**. Яка буде бачити свої розділи, як фізичні томи, ті об'єднуються в групу томів, а всередині цієї групи ми можемо створювати логічні томи, взагалі не вказуючи де вони знаходяться, і вони самі розподіляються між дисками. Відповідно на кожному логічному томі буде якась файлова система.

Щоб подивитися фізичні томи користуємося командою **pvdisplay**. Подивитися volume group, команда **vgdisplay**. Подивитися логічні томи **lvdisplay**.

Видаляємо, через **fdisk** всі розділи на жорсткому диску / **dev / sdb**. Створюємо 2 розділу по 4 ГБ основних з типом **8e**, тобто LVM тип. Створимо фізичний том командою **pvcreate / dev / sdb1**, Аналогічно робимо для другого розділу.

Далі необхідно створити віртуальну групу **vgcreate vg1 / dev / sdb1 / dev / sdb2**, Тобто обидва розділу. У групі можемо створити пару логічних томів командою **lvcreate -n lv1 -L 2G vg1**, Де **-n** новий розділ, **-L** - обсяг і останній параметр в який віртуальній групі. І можна створити другий **lvcreate -n lv2 -L 3G vg1**. З'явилися нові блокові пристрої **lv1** і **lv2**. Залишилося їх відформатувати командою **mkfs.ext4 / dev / vg2 / lv1** і аналогічно другий. Дана група дозволяє легко додавати і видаляти HDD. Можливе додавання нового жорсткого диска і на збільшення розміру наших томів. Щоб змінити розмір використовується команда **lvresize -L 4G vg1 / lv2**. Система LVM дозволяє робити знімки станів, тобто снапшоти. Вони використовуються для систем резервного копіювання, наприклад.

ТЕМА 2.

ЗАВАНТАЖЕННЯ ОПЕРАЦІЙНОЇ СИСТЕМИ (ОСНОВНІ ЕТАПИ Й КОНФІГУРАЦІЙНІ ФАЙЛИ).

2.1 Процес завантаження Linux

Завантаження операційної системи, це багатоступінчастий процес. У різних дистрибутивах Linux процес завантаження може дещо змінюватися, але загальна схема приблизно однакова і складається з наступних стадій:

1. У момент запуску процесор передає управління за певною фізичною адресою в ПЗУ. У цей момент починається виконання коду BIOS / UEFI. Проводиться ініціалізація обладнання і вибирається завантажувальний носій.

У разі BIOS відбувається зчитування в ОЗП початкового завантажувача і передача управління на нього. Початковий завантажувач зазвичай займає один сектор на диску (MBR) і не може перевищувати обсягу 384 байт (512 байт - сектор диска, мінус 128 байт - таблиця розділів). Залежно від типу завантажувального пристрою завантажувальний сектор може зчитуватися з різних місць:

При завантаженні з дискети або HDD завантажувач читається з першого сектора фізичного носія;

При завантаженні з CD / DVD - з першого сектора образу завантажувального диска, розміщеного в структурі даних CD;

При мережевому завантаженні - з першого сектора образу завантажувального диска, викачуваного з сервера по протоколу tftp.

При форматуванні диска в MBR замість завантажувача іноді пишеться програма, яка пише на екрані інформаційний текст "No bootable device - insert boot disk and press any key"

1.1 Початковий завантажувач зчитує в пам'ять основний завантажувач (GRUB, LiLo, NTLDR) і передає управління йому. Оскільки початковий завантажувач дуже малий, то, як правило, в його код жорстко прописують сектори, з яких треба прочитати код основного завантажувача. На HDD це може бути простір між MBR і першим розділом на диску (нульова доріжка) або зарезервоване місце всередині ФС (зарезервований Inode в ext2fs). На дискеті і при використанні образу диска при завантаженні з CD або по мережі - основний завантажувач може розташовуватися відразу слідом за первинним загрузчиком і займати весь обсяг образу.

При завантаженні через UEFI завантажувач зчитується цілком з файлу на спеціальному розділі.

2. Завантаження ядра (vmlinuz) і допоміжного образу диска (initrd, initramfs).

Завантажувач GRUB вдає із себе міні ОС, що підтримує всі основні файлові системи. GRUB шукає конфігураційний файл, в якому прописані шляхи до образу ядра і образу допоміжного диска. При необхідності образ ядра розпаковується в ОЗУ, формується область пам'яті, що містить параметри, що передаються з завантажувача в ядро, в тому числі адресу способу допоміжного диска.

Ядро що завантажується через GRUB має відповідати угодам [multiboot](#) або [multiboot2](#). Відповідно до угоди, образ ядра включає структуру (наприклад в секції даних), яка починається з магічного числа і містить інформацію про бажаний стан ядра в пам'яті і точці на яку треба передати управління. Перед передачею управління в ядро в регістр EAX поміщається ще одне магічне число, а в регістр EBX - адреса таблиці з параметрами, підготовленими загрузчиком.

Допоміжний диск необхідний сучасним Linux системам через модульності ядра і містить драйвери (ATA, NFS, RAID і т.п.), необхідні для отримання доступу до основної файлової системи. Усередині образу знаходиться файлова система в форматі архіву сріо.

На цьому етапі створюється процес з `pid = 1`, в якому відбувається виконання скрипта `init`, що знаходиться в кореневому каталозі допоміжного диска. Параметри, що передаються ядру, фактично передаються `vinit`, як аргументи командного рядка.

Скрипт містить команди завантаження необхідних драйверів у вигляді модулів ядра, створення тимчасових файлів пристроїв в каталозі `/dev` для доступу до цих модулів, сканування дискових розділів для виявлення і ініціалізації RAIDов і логічних томів. Після ініціалізації логічних дисків, робиться спроба змонтувати кореневу файлову систему, задану параметром `root =`. У разі бездискової мережевої завантаження кореневої каталог підключається по NFS.

На екран видаються повідомлення про завантаження драйверів і про пошук віртуальних томів підсистеми LVM. Етап завершується перемонтування кореневого каталогу на основну файлову систему і завантаження в процес з `pid = 1` основної програми `/sbin/init` (або її аналога).

У класичному Unix'e і старих версіях Linux (приблизно до 2012 року) програма `init` зчитує конфігураційний файл `/etc/inittab`, ініціює текстові консолі і, як правило, запускає необхідні служби за допомогою набору скриптів, розташованих в каталогах `/etc/init.d` і `/etc/rc*.d`. У сучасних дистрибутивах Linux в файлі `/sbin/init` знаходиться більш сучасна програма запуску служб. Найбільш популярною з подібних програм є `systemd`, яка дозволяє істотно скоротити час цього етапу завантаження.

На екран на цьому етапі видаються рядки, повідомляють про запуск служб, і інформація про успішність даного процесу ([OK] або [ERR]).

2.2 Завантажувач GRUB

[GRUB](#) (GRand Unified Boot Loader) - Великий уніфікований завантажувач. Розроблено в рамках проекту GNU як зразкова реалізація [мультизавантажувача](#), здатного завантажувати різні ОС з різних розділів одного диска або різні версії однієї ОС в рамках одного розділу.

На даний момент під назвою GRUB відомі дві істотно відмінні версії програми. Версія 0.97 - "старий" або "legacy" GRUB - використовується в RHEL до версії 6 включно. У цій статті мова саме про нього. GRUB версії > 1.0 - це майже повністю переписаний варіант програми, який використовується в Ubuntu, Fedora і багатьох інших дистрибутивах.

Оскільки в MBR є тільки 384 байт для розміщення завантажувача - GRUB поділений на дві частини `stage1` і `stage2`. Розмір `stage1` дорівнює одному сектору на диску - 512 байт. При цьому реально використовується тільки 384, а решта зарезервовано. `Stage2` - досить велика програма, яка містить драйвери декількох ФС, інтерпретатор командного рядка і кілька допоміжних функцій. Її розмір становить приблизно 124 КБ. При інсталяції GRUB компонент `stage2` повинен бути розміщений в послідовних секторах на диску, а адреса першого сектора і кількість секторів повинні бути прописані в секторі, що містить

stage1. Якщо носій не дозволяє розмістити stage2 в фіксованих секторах, то застосовується проміжний завантажувач stage1.5, що підтримує одну конкретну ФС і займає менше 16 КБ.

```
Список ФС, підтримуваних stage2 / 1.5 (ls / boot / grub / * 1_5)
e2fs_stage1_5
fat_stage1_5
ffs_stage1_5
iso9660_stage1_5
jfs_stage1_5
minix_stage1_5
reiserfs_stage1_5
ufs2_stage1_5
vstafs_stage1_5
xfs_stage1_5
```

2.3 Ручна правка initrd.img

Іноді хочеться зробити вinitrd щось нестандартне, що не передбачено стандартним скриптом (mkinitrd, mkinitramfs, dracut і т.п.). В цьому випадку можна розпакувати існуючий образ, поправити його руками і знову запакувати. Формат файлу - архів сріостислийgzip. Єдина тонкість - длясріо треба вказувати опцію, що задає внутрішній формат архіву -H newc

```
mkdir initrd; cd initrd
gunzip -c /boot/initramfs-2.6.32.158.img | cpio -i --make-directories
# Внесення правок
find. | cpio -o -H newc | gzip> /boot/myinitrd.img
```

initramfs, як файл, мало відрізняється від initrd. При створенні явно вказується максимальна ступінь стиснення.

```
# Упаковка (фрагмент з dracut CentOS 6)
find. | cpio -R root: root -H newc -o --quiet | $ Gzip -9> "$ outfile"
```

У CentOS7 структура файлу змінилася. На початок приклеєний ще один маленький нестислий архів сріо "Early_initramfs" він же "microcode blob" (необов'язковий). Програма skipсріо шукає в отриманому файлі ознака кінця архіву (рядок "TRAILER !!!") і видає хвіст. Для стиснення можна використовувати різні програми (gzip, bzip2, xz). За замовчуванням використовується gzip.

```
# Розпакування:
/ Usr / lib / dracut / skipcpio / boot / initramfs - 3.10.0 - 957.e17.x86_64.img
| zcat | cpio -i --make-directories
# Упаковка
find. | cpio -c -o | xz -z -9 -C crc32 -F xz> /boot/myinitrd.img
```

2.4 Внесення виправлень в initrd

Іноді виникає ситуація, при якій завантаження Linux неможливо через невірно зібраний образ диска `initrd`. Виникає ситуація курки і яйця: щоб виправити `initrd` необхідно завантажити Linux, щоб завантажити Linux потрібен виправлений `initrd`.

У CentOS і аналогічних системах послідовність дій така:

Завантажитися з встановлювального диску в режим відновлення - `Rescue mode`. Для цього в момент завантаження на запрошення `boot:` необхідно ввести `linux rescue`.

Якщо все піде нормально, то кореневої каталог основної системи буде змонтований в `/mnt / sysimage`, Завантажувальний каталог в `/mnt / sysimage / boot`. Крім того поточні каталоги `/proc`, `/sys` і `/dev` будуть змонтовані в відповідні підкаталоги `/mnt / sysimage`. Якщо цього не станеться, то доведеться виконати ці операції вручну.

Коли все каталоги змонтовані, можна змінити кореневий каталог

```
# Якщо з'ясується, що ви щось забули змонтувати, то можна вийти по ^ D
chroot / mnt / sysimage
```

і перезібрати `initrd`

У CentOS 6

```
# Копіюємо старий файл
cp -p / boot / initramfs - $ (uname -r) .img / boot / initramfs - $ (uname
-r) .img.bak
# Створюємо новий
dracut -f
# Якщо версія ядра в основній системі відрізняється від версії на
установчому диску, вказуємо її явно
dracut -f /boot/initramfs-2.6.32-358.el6.x86_64.img 2.6.32-
358.el6.x86_64
```

У CentOS 5

```
# Копіюємо старий файл
cp -p / boot / initrd - $ (uname -r) .img / boot / initrd - $ (uname -r)
.img.bak
# Створюємо новий
mkinitrd -f -v / boot / initrd - $ (uname -r) .img $ (uname -r)
# Якщо версія ядра в основній системі відрізняється від версії на
установчому диску, вказуємо її явно
mkinitrd -f -v /boot/initrd-2.6.18-371.el5.img 2.6.18-371.el5
```

перезавантаження

```
cd /
sync
telinit 6
```

Розглянемо приклад з драйвером `i2o_block` (SCSI адаптер `Adaptec 2010S`), який не завантажується автоматично. Приклад виконується в CentOS 5, оскільки в стандартному ядрі CentOS 6 підтримка цього драйвера відключена.

Після завантаження з CD в Rescue mode видається повідомлення, що Linux розділи не знайдені і їх треба монтувати самостійно.

```
# Завантажуємо драйвер
insmod i2o_block

# Перевіряємо, що все спрацювало
lsmod
....
dmesg
...

# Створюємо файли пристроїв на основі інформації в dmesg
mkdir / dev / i2o
mknod / dev / i2o / hda b 80 0
mknod / dev / i2o / hda1 b 80 1
mknod / dev / i2o / hda2 b 80 2

# Активуємо VolumeGroup
lvm vgchange -ay

# Монтуємо томи
mkdir / mnt / sysimage
mount / dev / mapper / VolGroup00-LogVol00 / mnt / sysimage
mount / dev / i2o / hda1 / mnt / sysimage / boot

# Монтуємо спецкаталог
mount --bind / proc / mnt / sysimage / proc
mount --bind / dev / mnt / sysimage / dev
mount --bind / sys / mnt / sysimage / sys
```

Далі по інструкції, тільки при створенні образу диска треба вказати програмі mkinitrd додаткову опцію `--preload = i2o_block` і відключити сервіси `readahead`, оскільки вони призводять до зависання драйвера `i2o_block`:

```
chkconfig early-readahead off
chkconfig later-readahead off
```

2.5 Завантаження Linux - SysV init

Минулого разу ми говорили про те, що відбувається при завантаженні Linux: спочатку стартує завантажувач, він завантажує ядро і розгортає тимчасовий диск в оперативній пам'яті, ядро запускає процес `init`, `init` знаходить справжній кореневої диск, робить такий хитрий переворот - замість тимчасового віртуального диска на це ж саме місце в кореневій каталог монтується реальний диск, з цього реального дисків процес `init` завантажує в себе інший `init`, який є на цьому реальному диску. Після всіх цих операцій UNIX переходить в стан нормальної роботи.

Розглянемо, що робить класична програма `init` в поєднанні зі скриптами `rc.d` в стилі System V (Систем п'ять). System V - це класична версія UNIX на якій побудовані комерційні UNIX.

Судячи з назви, `rc.d` це якийсь каталог. Є така традиція UNIX - якщо вся конфігурація чого-небудь вміщується в один файл, і він називається `config`, то при розбитті його на окремі файли, які підключаються до основного,

створюють каталог з аналогічним ім'ям і додають до імені `.d` - `config.d`. Буква `d` означає, що це директорія і там лежать допоміжні частини конфігураційного файлу. У форматі конфігураційних файлів програми `init` є дві традиції: варіант `System V`, в якому кожна деталь конфігурації тримається в окремому файлі в каталозі `rc.d`, і традиція `BSD` систем, в якій є один файл `/etc/rc`, що містить багато скриптів і змінних, які відповідають за поведінку системи.

У будь-якому випадку, при старті системи у нас створюється процес з `PID = 1`, в якому запущена програма, яка називається `init`. Як ви бачили в минулий раз, якщо програму `init` вбити, то ядро впадає в паніку і припиняє усіляку роботу.

Класичний `System V` `init` читає файл `/etc/inittab` і виконує ряд розпоряджень, які прописані в цьому файлі. `Inittab` цей текстовий файл кожен рядок якого, це, по суті справи, одна команда або якесь правило поведінки. `Inittab` виглядає так:

```
id: 3: initdefault:
si :: sysinit: /etc/rc.d/rc.sysinit
l3: 3: wait: /etc/rc.d/rc 3
1: 2345: respawn: / sbin / mingetty tty1
ca :: ctrlaltdel: / sbin / shutdown -t3 -r now
```

Спочатку рядку стоїть позначка. Можна вважати, що це простий текст і все. Другим пунктом стоїть або так званий рівень завантаження, або пусте значення. Рівень завантаження - це або одне число від 0 до 6, або список чисел через кому. Далі йде якась дія. Дії бувають такі: `wait`, `respawn`, `sysinit`, `ctrlaltdel`. Є й інші дії, але це найбільш використовувані. Нарешті, в кінці рядка написана якась команда з ім'ям виконуваного файлу і аргументів, які цій команді треба передати.

Дія `sysinit` виконується одноразово при старті системи.

Дія `ctrlaltdel` це насправді не зовсім дія - це оброблювач поєднання клавіш `control alt del`. Саме натискання перехоплюється ядром системи, і інформація про це пересилається в процес `init`, який повинен виконати певну команду. Наприклад, може бути виконана команда `shutdown`, яка виконає виключення комп'ютера. В принципі сюди можна прописати будь-яку іншу програму, наприклад, `echo`, яка після натискання `control alt del` видаватиме на всі термінали системи якусь повідомлення. каміна консоллю так

Дія `wait` означає, що необхідно запустити команду, дочекатися поки вона закінчиться і тільки після цього продовжити обробку наступних рядків. Не знаю, чи можуть запускатися такі дії в паралель. Скоріше за все ні.

Дія `respawn` означає, що треба запустити програму і перш ніж вона закінчиться, перейти в подальшому дій. Якщо ця програма в подальшому завершиться, то необхідно її рестартовать.

Отже, є одноразове виконання з очікуванням результатів і багаторазове виконання в асинхронному режимі - запустилися, дочекалися поки закінчити, запустили слова.

Рівні завантаження - це якась умовність, яка дозволяє управляти завантажуються службами. Найближчий аналог в `windows` - це завантаження

в безпечному режимі, коли вантажиться тільки обмежене число драйверів і стартує мінімальну кількість служб, завантаження з налагодженням, коли кожна дія додатково записується і звичайна повноцінна завантаження.

У Linux за традицією виділяється 6 варіантів завантаження. Цей поділ є досить умовним.

0 і 6 це виключення. 0 - повне виключення електрику, а 6 - режим перезавантаження.

4 в Linux взагалі пропущено

Залишаються чотири рівні завантаження:

1 - однокористувальницький режим. Якщо передати завантажувачу ключове слово `single`, то ми опинимося в режимі одного, де запущений тільки один процесу і це шелл адміністратора системи. Цей режим використовується для відновлення системи.

3 - нормальний розрахований на багато користувачів текстовий режим, коли запущені усі служби, працює мережа, працюють всі драйвери.

2 - теж текстовий режим, але без підключення мережевих дисків. Справа в тому, що традиційні мережева файлова система `nfs`, яка використовується в UNIX, надзвичайно стійка до пошкоджень мережі. Якщо ми вимкнули файловий сервер або обрізали мережевий кабель, то мережева файлова система `nfs` робитиме численні спроби відновитися і ці спроби настільки тривалі, що я жодного разу не зміг дочекатися часу, коли ж нарешті з'явиться повідомлення про помилку. Можливо це станеться через годину, а може і через 6 годин. Весь цей час драйвер `nfs` буде тримати комп'ютер, не даючи нічого зробити. Тому, якщо у нас впала мережу або файловий сервер в налаштуваннях написано, що при старті необхідно підмонтирувати зовнішні диски, то спроба завантажитися в повноцінний режим призведе до того, що у вас все зависне. Для цього випадку і передбачений другий варіант завантаження - все як у третьому, тільки мережеві диски не підключаються. Сам мережевий адаптер працює, IP адреса призначається, інтернет доступний.

5 - те ж саме що і 3, але з запуском `x window` - графічного інтерфейсу.

Можна вважати, що між рівнями є деяка послідовність переходів:

режим 2 включає себе 1 + розрахований на багато користувачів режим.

3 включає 2 + монтування мережевих файлових систем. Нарешті, 5 включає в себе 3 + запуск графічної підсистеми. Чи буде це реалізовано послідовно чи ні - це проблема дистрибутива. Взагалі кажучи, адміністратори можуть самостійно налаштувати файл `inittab` так, щоб ці режими запускалися послідовно, а можна зробити так щоб все було абсолютно незалежно - перемикаючись в черговий режим, прибираємо всі що було зроблено на попередньому кроці, і налаштуємо все з нуля.

Розглянемо рядки реального файлу. Вони дуже прості.

```
l3: 3: wait: /etc/rc.d/rc 3
```

Запускається якась програма, яка повинна виконати всі необхідні дії, які очікуються на третьому рівні. Напевно, на третьому рівні потрібно налаштувати мережеві інтерфейси, запустити драйвер терміналів, стартувати якісь служби. Тільки після того, як все це завершиться ми зможемо працювати

в системі. Оскільки треба дочекатися завершення запуску, ми вибираємо дію `wait`.

Програма запуску називається `rc` і запускається з номером рівня в якості параметра. Сама програма `init` досить проста. Вона вміє через підрядник читати свій файл з простим синтаксисом і стартувати нові процеси, запускаючи якісь допоміжні програми. Вся логіка рівнів завантаження захована в скрипті `rc`. Запустивши `rc` з параметром 3 ми перейдемо на третій рівень, з параметром 5 - на п'ятий.

Програма `rc` теж дуже проста. Це скрипт який виконує всі файли в каталогах, що відповідають рівню завантаження, наприклад, `/etc/rc3.d/`. У цих каталогах знаходяться виконувані файли, які приймають один параметр - або `start`, або `stop`. Якщо файл запущений з параметром `start`, то він стартує службу, якщо з параметром `stop`, то зупиняє її. Наприклад, `network start` буде налаштовувати мережеві інтерфейси, а `network stop` буде переводити інтерфейси в вимкненому стані. Крім мережевих інтерфейсів є скрипти підключення / відключення мережевих файлових систем, запуску / зупинки сервісів і т.д.

Імена файлів в каталогах побудованим за певними правилами. Вони починаються або з букви `K` або з букви `S`, за якими йде число і ім'я служби.

Скрипт `rc` переглядаємо вмісту каталогу `rc3` і вибирає звідти всі файли які починаються з літери `K` (`kill`). Файли поділяються на категорії в порядку зростання номера і виконуються з параметром `stop`. Потім ті ж дії виконуються з файлами на букву `S` (`start`), які запускаються з параметром `start`. Ось загалом і вся процедура переходу на певний рівень.

Можна припустити, що в каталозі `/etc/rc0.d/` лежать тільки файли, що починаються на букву `K`, оскільки при виключенні треба все зупинити, а в каталозі `/etc/rc1.d/` буде один файл на буку `S` для запуску консолі адміністратора.

Для простоти програмування є окремий каталог `/etc/init.d/`, в якому лежать ті ж самі файли тільки без літери цифр на початку імені. Насправді, файли в каталогах рівнів це просто символічні посилання на основні файли. Так `/etc/rc3.d/S10apache` це посилання на файл `/etc/init.d/apache`. Букви і цифри в назві посилань потрібні для того, щоб скрипт `rc` викликав їх в потрібному порядку і з потрібними аргументами.

У системах, які побудовані за таким принципом, щоб стартувати або зупинити будь-яку службу в каталозі `/etc/init.d/` треба знайти файл який, який їй відповідає, і запустити його з параметром `start` або `stop`. Чим не подобається запускати служби саме таким способом - явно викликаючи скрипти. Справа в тому, що в командному рядку `linux` чудово працює автодоповнення. З його допомогою дуже швидко можна ввести шлях до файлу запуску.

Щоб сховати від користувача конкретну реалізацію поверх системи скриптів і символічних посилань написані дві допоміжні програми.

Програма `chkconfig` дозволяє маніпулювати символічними посиланнями на відповідні скрипти. Щоб подивитися, що стартує, а що зупиняється на кожному з рівнів можна скористатися командою `ls` і видати список скриптів у

відповідному каталозі, але простіше скористатися командою `chkconfig -list`. Програма `chkconfig` пробігає по всіх каталогах `rc` і видає список того що стартує, а що зупиняється на кожному рівні. Якщо ми хочемо, щоб при старті системи у нас щось автоматично стартувала певна служба ми виконуємо `chkconfig <ім'я служби> on` і скрипт створює посилання для запуску в потрібному каталозі і з правильним ім'ям. Запуск `chkconfig <ім'я служби> off` призводить до видалення посилання для запуску і створення посилання для зупинки. Таким чином програма `chkconfig` дозволяє управляти списком служб,

Ще одна програма - `service` використовується для ручного запуску і зупинки служб. `Service` це обгортка, яка дозволяє не звертатися безпосередньо до скрипту, а вказати ім'я служби і сказати хочемо ми її стартувати або зупинити. В `bash`, який я використовую, немає автодоповнення для команди `service`, тому мені простіше набрати шлях до скриптів.

У стартових скриптах аргументи `start` і `stop` повинні оброблятися обов'язково. Крім того, можна придумати якісь свої аргументи, які будуть робити щось корисне.

У більшості скриптів реалізована опція `status`, яка показує запущена служба чи ні. Коли ми виконуємо `start`, то скрипт після успішного запуску служби отримує її ідентифікатор `PID` і записує його в певний файл. За командою `stop` файл видаляється. Зазвичай такі файли створюються в каталозі `/var/run/`. Команда `status` перевіряє чи є такий файл. Його немає, то повідомляє, що служба не запущена. Якщо файл є, то вона витягує з нього ідентифікатор процесу і перевіряє поточний список процесів. Якщо цей ідентифікатор присутній все запущено, якщо програма з якихось причин поламалася, то статус видає, що була зроблена спроба запустити цю службу - файл існує, але сама служба не запущена.

Опція `restart` послідовно виконує всередині скрипта дві команди - спочатку `stop`, а потім `start`. Це абсолютно необов'язкова команда - просто зручна. Нарешті, є служби, які дозволяють на ходу перечитати якісь конфігураційні файли. Для них додають команду `reload`, завданням якої є відправка службі сигналу про те, що конфігурація змінилася. Окремий випадок, команди `save` і `load` для збереження конфігурації брандмауера.

Якщо адміністратор системи замість зупинки або старту окремих служб хоче всю систему перевести на певний рівень, то цього можна досягти одним з двох способів. Можна викликати прямо програму `/sbin/init`. Якщо її викликати з певним числом як параметр, то вона виконає всі інструкції з файлу `inittab`, для яких прописував відповідний рівень. Якщо запустити, наприклад, `/sbin/init 1`, то `init` знайде в своєму файлі конфігурації всі рядки, в яких є рівень 1 і виконає їх. У деяких системах команда `shutdown` реалізована як `/sbin/init 0`, оскільки нульовий рівень відповідає зупинці системи. Останнім часом для переходу між рівнями з'явилася спеціальна програма під назвою `telinit`, яка є посиланням на `init`. Її завдання - переслати процесу `init` сигнал про те, що адміністратор бажає перейти на певний рівень. `telinit q` повідомляє `init` про те, що треба перечитати файл `inittab`. У старих системах це досягалось посилкою сигналу `SIGHUP` процесу з `PID = 1` (`kill -HUP 1`).

Ще кілька рядків в `inittab`, це запуск терміналів

```
1: 2345: respawn: / sbin / mingetty tty1
```

Для того, щоб забезпечити діалоговий доступ до системи, ви `inittab` може бути присутнім кілька рядків такого роду. `2345` це рівні, на яких треба запускати команду, `respawn` означає, що програму треба перезапустити у разі завершення. Програма `getty` - це програма управління терміналом. Традиційно термінал в UNIX називається телетайпом, оскільки першими терміналами були електричні друкарські машинка. Відповідно, `tty` це скорочення від телетайпа. `Mingetty` - програма, яка вміє працювати з віртуальними терміналами на персональному комп'ютері. Вона вміє налаштовувати драйвер терміналу, а в якості параметрів отримує ім'я пристрою терміналу, який треба налаштувати. У каталозі `/ dev / c` файл пристрою `tty1`, який відповідає першому віртуальному терміналу. Якби у нас був модем і ми хотіли б ініціювати його момент завантаження, то могли б викликати `getty` з параметром `ttyS0`, який відповідає порту `COM1`. При ініціалізації модему можна було б задати додаткові параметри: швидкість з'єднання 19200 бод, 7 або 8 біт в байті, парність, кількість стоп-бітів.

```
S0: 2345: respawn: / sbin / getty ttyS0 19200 8 n 1
```

Минулого разу я малював ланцюжок, в якій процес викликом `fork` роблять свою копію, дочірня копія викликом `exec` завантажує в свою пам'ять іншу програму, а після завершення повідомляє про це батьківського процесу.

Текстові призначені для користувача сеанси влаштовані на таких ланцюжках: спочатку `init` робить свою копію і запускає в ній програму `mingetty`. `Mingetty` ініціалізує термінал і клавіатуру, а потім запускає в тому ж процесі програму `login`. `Login` виводить на екран запрошення на логуватись і, якщо все пройшло успішно то призначає собі привілеї користувача і в тому ж процесі, затираючи самого себе, запускає інтерпретатор користувача, наприклад, `bash`. Коли користувач набирає команду `exit`, то інтерпретатор завершує життєвий шлях цього процесу. Коли процес завершується, `init` отримує про це сигнал. `Init` дивиться, що належить робити, бачить дію `respawn`, знову запускає програму `mingetty`, яка заново ініціалізує термінал і все повторюється. Таким чином кожен сеанс знаходиться всередині одного процесу.

У файлі `inittab` є ще одне спеціальне ключове слово `initdefault` - рівень за замовчуванням. Якщо через ядро `init` отримав параметр `single`, то ми завантажилися на рівень 1. Якщо через завантажувач нічого не передали, то використовується значення за замовчуванням. Якщо після установки графічної оболонки виявилось, що наш комп'ютер слабенький для графіки, то можна встановити рівень за замовчуванням на 3, і після наступного перезавантаження ми потрапляємо на третій рівень - тобто в текстовий режим. Встановили систему без графічного режиму, потім доустановити все пакети для `x window`, поміняли рівень за замовчуванням на 5 і після наступного перезавантаження потрапили відразу в графічний режим.

У цій системі скриптів іноді хочеться зробити щось своє, наприклад, при старті видалити всі файли в каталозі `/ tmp /`. Для цього є окремий файл під

назвою /etc/rc.local, який запускається після всіх інших. Це просто скрипт без параметрів, в який можна прописати все, що завгодно. Наприклад, на одному з моїх роутерів в момент старту системи в цьому файлі прописуються таблиці маршрутизації. Мені було ліньки шукати де знаходяться відповідні стандартні скрипти з дистрибутива і простіше виявилось прописати команди в rc.local.

ТЕМА 3.

НАСТРОЮВАННЯ МЕРЕЖЕВИХ ІНТЕРФЕЙСІВ. ВИДИ МЕРЕЖЕВИХ ІНТЕРФЕЙСІВ. УТИЛІТИ ДЛЯ НАСТРОЮВАННЯ І ДІАГНОСТИКИ МЕРЕЖЕВИХ ІНТЕРФЕЙСІВ.

Іменування мережевих інтерфейсів в CentOS

Класична схема іменування мережевих інтерфейсів в Linux привласнює імена eth0, eth1 і так далі по порядку. Але ці імена не прив'язуються жорстко до інтерфейсів і після перезавантаження при наявності декількох мережевих інтерфейсів, ці імена можуть помінятися. Це може доставляти деякі проблеми, при налаштуванні, наприклад, брандмауера через **firewalld** або **iptables**. У зв'язку з цим починаючи з RedHat 7 і CentOS 7, вирішено було призначити імена мережевих інтерфейсів на основі ієрархії різних схем іменування. За замовчуванням systemd буде по черзі застосовувати схему віртуальних пакунків за зупинившись на першій доступною і прийнятною. Імена присвоюються в автоматичному режимі, залишаються незмінними навіть якщо апаратні засоби додані або змінені. З іншого боку, такі імена інтерфейсів менш читабельні, наприклад, enp5s0 або ens3, ніж традиційні eth0 і eth1.

Можна повернутися до стандартного імені інтерфейсу Linux за допомогою наступних дій.

Відредагуйте файл / etc / default / grub:

```
# nano / etc / default / grub
```

У рядок GRUB_CMDLINE_LINUX потрібно додати:

```
net.ifnames = 0 biosdevname = 0
```

Приклад повного рядка:

```
GRUB_CMDLINE_LINUX = "consoleblank = 0 fsck.repair = yes crashkernel = auto nompath selinux = 0 rhgb quiet net.ifnames = 0 biosdevname = 0"
```

Оновлення конфігурації grub:

```
# grub2-mkconfig -o /boot/grub2/grub.cfg
```

Переіменуйте конфігураційний файл мережевого інтерфейсу:

```
# mv / etc / sysconfig / network-scripts / ifcfg-ens3 / etc / sysconfig / network-scripts / ifcfg-eth0
```

І замініть значення DEVICE:

```
root@server:~  
GNU nano 2.9.8  
  
# Generated by parse-kickstart  
TYPE="Ethernet"  
DEVICE="eth0"  
UUID="5fb7cca6-d02c-4f5d-be2e-92f9795cd5bc"  
ONBOOT="yes"  
BOOTPROTO="static"  
IPADDR="185.129.49.16"  
NETMASK="255.255.255.0"  
GATEWAY="185.129.49.1"  
IPV6INIT="yes"  
DNS1=8.8.8.8
```

Збережіть файл, перезавантажте сервер і перевірте чи все в порядку:

```
# ip a
```

```
[root@server ~]# ip a  
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000  
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00  
    inet 127.0.0.1/8 scope host lo  
        valid_lft forever preferred_lft forever  
    inet6 ::1/128 scope host  
        valid_lft forever preferred_lft forever  
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000  
    link/ether 52:54:00:d3:1c:3e brd ff:ff:ff:ff:ff:ff  
    inet   
        valid_lft forever preferred_lft forever  
    inet6 fe80::5054:ff:fed3:1c3e/64 scope link noprefixroute  
        valid_lft forever preferred_lft forever  
[root@server ~]#
```

Інтерфейс тепер називається eth0.

Початкове налаштування мережі при установці CentOS

Спочатку CentOS при установці Linux, Ви можете налаштувати мережевий інтерфейс в графічному режимі в пункті меню "Network & Hostname". В даному пункті ви вказуєте ім'я сервера, додаєте потрібний IP адреса і шлюз, DNS і багато іншого.

Ручна настройка конфігураційного файлу мережевого інтерфейсу в CentOS

Виведемо список доступних мережевих інтерфейсів в системі:

```
# ip a
```

Файли конфігурації мережі вашого сервера зберігаються в каталозі / etc / sysconfig / network-scripts. Ці файли створює демон NetworkManager для кожного мережевого інтерфейсу. У нашому випадку файл конфігурації називається ifcfg-eth0 (у вас може відрізнятись в залежності від схеми іменування мережевого інтерфейсу).

Розглянемо основні параметри:

- DEVICE - ім'я мережевого адаптера, збігається з ім'ям в системі, у нас це eht0
- BOOTPROTO - спосіб призначення IP-адреси (static - статичне значення, вказуємо в ручну. Dhcp - отримати адресу автоматично)
- IPADDR - IP-адреса
- NETMASK - маска підмережі
- GATEWAY - шлюз
- DNS1 - Основний DNS-сервер
- DNS2 - альтернативний DNS-сервер
- ONBOOT - спосіб запуску мережевого інтерфейсу (yes - автоматично, no - вручну)
- UUID - унікальний ідентифікатор мережевого інтерфейсу. Можна згенерувати самостійно командою uuidgen.
- IPV4_FAILURE_FATAL - відключення мережевого інтерфейсу з IP-адресою v4, якщо він має невірну конфігурацію (yes - відключити, no - не відключати)
- IPV6_FAILURE_FATAL - відключення мережевого інтерфейсу з IP-адресою v6, якщо він має невірну конфігурацію (yes - відключити, no - не відключати)
- IPV6_AUTOCONF - дозволяє або забороняє Автоконфігурування Ірв6 за допомогою протоколу
- IPV6_INIT - включення можливості використання адресації Ірв6 (yes - адресація може використовуватися, no - не використовується)
- PEERROUTES - встановлює пріоритет настройки шлюзу, при використанні **DHCP**
- IPV6_PEERROUTES - встановлює пріоритет настройки шлюзу, при використанні DHCP для Ірв6
- Виходячи з цієї інформації, налаштуємо мережевий інтерфейс.
-

Налаштування статичної IP адреси в CentOS

Відкриємо файл для редагування:

```
# mcedit / etc / sysconfig / network-scripts / ifcfg-eth0
```

```
mc [root@server.build-centos.info]:/etc/sysconfig/network-scripts
ifcfg-eth0 [----] 12 L:| 1+11 12/ 13] *(246 / 247)
# Generated by parse-kickstart
UUID="b8bc0d4c-fb1b-4d36-9d45-044c7c0194eb"
IPADDR="185. ...."
GATEWAY="185. ...."
NETMASK="255.255.255.0"
BOOTPROTO="static"
DEVICE="eth0"
ONBOOT="yes"
IPV6INIT="yes"
DNS1=77.88.8.8
DNS2=8.8.8.8
DNS3=8.8.4.4
```

У цьому прикладі вказано IP адресу, маску підмережі, шлюз і кілька DNS серверів. Включаємо автозапуск інтерфейсу:

```
ONBOOT = "yes"
```

Після всіх модифікацій, потрібно виконати рестарт сервісу network. Якщо все в порядку, ви отримаєте такий лістинг:

```
[ Root @ server network-scripts] # service network restart
```

```
Restarting network (via systemctl): [OK]
```

Також можна просто перезапустити всі профілі підключень:

```
# nmcli connection reload
```

Отримання динамічного IP адреси для інтерфейсу через DHCP

Якщо ваш сервер повинен отримати IP адресу від DHCP півночі, відкрийте конфігураційний файл керування і змініть настройки:

```
mc [root@server.build-centos.info]:/etc/sysconfig/network-scripts
```

```
ifcfg-eth0 [-M--] 0 L:[ 1+10 11/ 11] *(190 / 1
# Generated by parse-kickstart
UUID="b8bccd4c-fb1b-4d36-9d45-044c7c0194eb"
BOOTPROTO="dhcp"
DEFROUTE="yes"
DEVICE="eth0"
ONBOOT="yes"
IPV6INIT="yes"
DNS1=77.88.8.8
DNS2=8.8.8.8
DNS3=8.8.4.4
```

Тобто ми прибравши всі настройки, пов'язані з IP-адресами і маскою, а так же поміняли спосіб призначення IP-адрес на dhcp (BOOTPROTO = "dhcp"). Після всіх змін, не забуваємо виконувати перезавантаження network.

Як відключити IPv6 в CentOS?

Якщо ви в не використовуєте протокол IPv6, його потрібно відключити на сервері. Якщо ви точно впевнені, що жоден з сервісів не налаштований під роботу з ipv6, можете відразу перейти до налаштування мережевого інтерфейсу, якщо ж ні, то почніть з перевірки. Нам потрібно перевірити, які сервіси використовують ipv6 і відключити даний протокол в конфігурації сервісу. Запустимо команду:

```
# netstat -tulnp
```

```
[root@server network-scripts]# netstat -tulnp
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State       PID/Program name
tcp        0      0 0.0.0.0:22             0.0.0.0:*               LISTEN      912/sshd
tcp6       0      0 :::22                 :::*                    LISTEN      912/sshd
udp        0      0 127.0.0.1:323         0.0.0.0:*               *          541/chronyd
udp6       0      0 :::1:323              :::*                    *          541/chronyd
[root@server network-scripts]#
```

У мене сервер тестовий, тому ipv6 використовується тільки для sshd і chronyd. Це можна визначити по ":::".

Щоб не виникало проблем після відключення ipv6 в конфігурації мережі, вимкніть даний протокол в сервісах, в яких вони використовуються на вашому сервері. Наприклад для sshd, потрібно відкрити конфігураційний файл:

```
# Mcedit / etc / ssh / sshd_config
```

І розкоментуйте рядки:

```
#AddressFamily any
#ListenAddress 0.0.0.0
```

Після чого перезапустіть сервіс:

```
root@server:~
[root@server ~]# service sshd restart
Redirecting to /bin/systemctl restart sshd.service
[root@server ~]# netstat -tulnp
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State       PID/Program name
tcp        0      0 0.0.0.0:22             0.0.0.0:*               LISTEN      11874/sshd
udp        0      0 127.0.0.1:323         0.0.0.0:*               *          541/chronyd
udp6       0      0 :::1:323              :::*                    *          541/chronyd
[root@server ~]#
```

Як бачимо, для sshd протокол ipv6 тепер недоступний. Проробіть аналогічні налаштування з усіма сервісами.

Перейдемо до відключення протоколу ipv6 в настройках мережі. Відкрийте файл /etc/sysctl.conf:

```
# Nano /etc/sysctl.conf
```

І додайте туди наступні рядки:

```
net.ipv6.conf.all.disable_ipv6 = 1
net.ipv6.conf.default.disable_ipv6 = 1
```

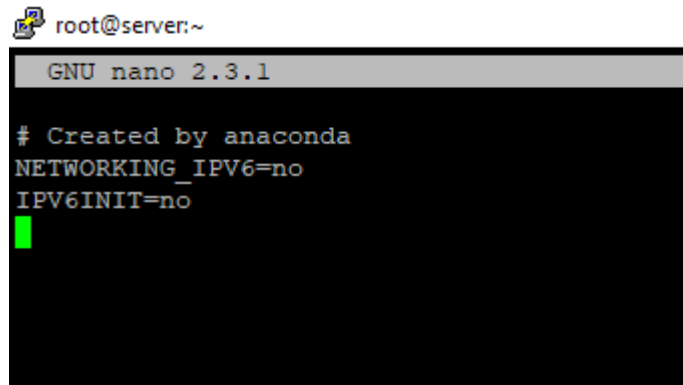
Збережіть файл і застосуєте через:

```
[ Root @ server ~ ] # sysctl -p
```

```
net.ipv6.conf.all.disable_ipv6 = 1
net.ipv6.conf.default.disable_ipv6 = 1
```

Перейдемо до файлу / etc / sysconfig / network. Додайте в нього наступну конфігурацію:

```
NETWORKING_IPV6 = no
IPV6INIT = no
```



```
root@server:~
GNU nano 2.3.1
# Created by anaconda
NETWORKING_IPV6=no
IPV6INIT=no
```

З файлу конфігурації мережевого інтерфейсу / etc / sysconfig / network-scripts / ifcfg-eth0 видаліть рядок:

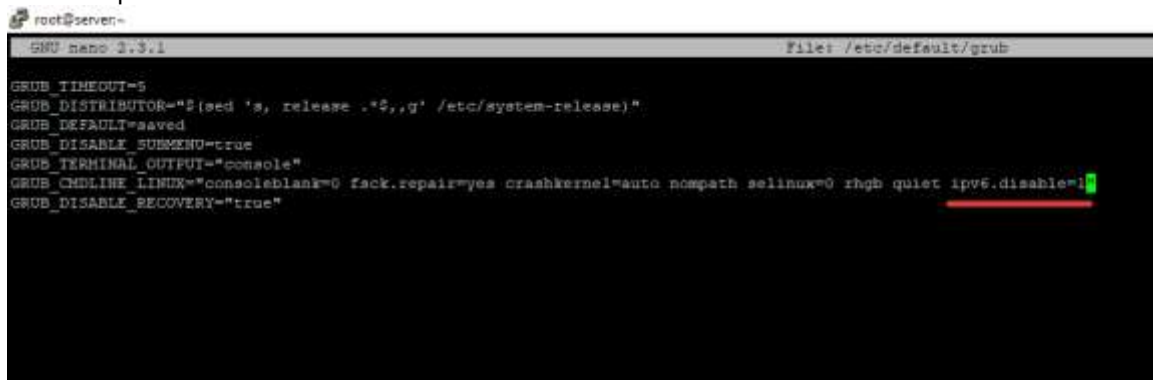
```
IPV6INIT = "yes"
```

І нарешті додамо заборону на роботу ipv6 в grub:

```
# Nano / etc / default / grub
```

В кінець рядка GRUB_CMDLINE_LINUX, додаємо:

```
ipv6.disable = 1
```



```
root@server-
GNU nano 2.3.1 File: /etc/default/grub
GRUB_TIMEOUT=5
GRUB_DISTRIBUTOR="$(sed 's, release .*$,g' /etc/system-release)"
GRUB_DEFAULT=saved
GRUB_DISABLE_SUMMENU=true
GRUB_TERMINAL_OUTPUT=console
GRUB_CMDLINE_LINUX="consoleblank=0 fack.repair=yes crashkernel=auto nompath selinux=0 rhgb quiet ipv6.disable=1"
GRUB_DISABLE_RECOVERY="true"
```

Після всіх налаштувань, збережіть файл і оновіть grub:

```
# Grub2-mkconfig -o /boot/grub2/grub.cfg
```

Виконайте перезавантаження сервера і перевірте конфігурацію мережі:

```
[ Root @ server ~] # ifconfig
```

```
eth0: flags = 4163 <UP, BROADCAST, RUNNING, MULTICAST> mtu 1500
inet 185. *. *. * netmask 255.255.255.0 broadcast 185. *. *. 255
ether 52: 54: 00: d3: 1c: 3e txqueuelen 1000 (Ethernet)
RX packets 10068 bytes 613092 (598.7 KiB)
RX errors 0 dropped 0 overruns 0 frame 0
TX packets 32 bytes 5399 (5.2 KiB)
TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

```
lo: flags = 73 <UP, LOOPBACK, RUNNING> mtu 65536
inet 127.0.0.1 netmask 255.0.0.0
loop txqueuelen 1000 (Local Loopback)
RX packets 0 bytes 0 (0.0 B)
RX errors 0 dropped 0 overruns 0 frame 0
TX packets 0 bytes 0 (0.0 B)
TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

Протокол ipv6 на сервері відключений.

Як вказати DNS сервера для мережевого інтерфейсу в CentOS?

Налаштувати DNS-сервера для вашого сервера, ви можете за допомогою файлу `/etc/resolv.conf` або вказати їх в настройках мережевого інтерфейсу. При налаштуванні static конфігурації для мережевого інтерфейсу, ми вже вказували DNS-сервера, через параметри:

```
DNS1 =
DNS2 =
DNS3 =
```

Змініть вам DNS-сервера і перезавантажте сервіс network.

У файл `/etc/resolv.conf`, DNS-сервера прописуються автоматично при перезавантаженні сервера, забираючи їх з файлу конфігурації мережі. Якщо ж ви не вказали DNS-сервера при настройці мережі, пропишіть їх вручну в файл `/etc/resolv.conf`:

```
nameserver 77.88.8.8
nameserver 8.8.8.8
nameserver 8.8.4.4
```

Як налаштувати декілька IP адрес на одному мережевому інтерфейсі CentOS?

Якщо вам потрібно використовувати декілька IP-адрес на одному мережевому інтерфейсі, настройку можна виконати через алиас інтерфейсу або ж додавши додатковий IP-адресу в основний файл конфігурації.

```
# Nano / etc / sysconfig / network-scripts / ifcfg-eth0
```

І змініть його наступним чином:

```
# Generated by parse-kickstart
UUID = "b8bccd4c-fb1b-4d36-9d45-044c7c0194eb"
IPADDR1 = "*. *. *. *"
IPADDR2 = "*. *. *. *"
GATEWAY = "*. *. *. *"
NETMASK = "255.255.255.0"
BOOTPROTO = "static"
DEVICE = "eth0"
ONBOOT = "yes"
DNS1 = 77.88.8.8
DNS2 = 8.8.8.8
DNS3 = 8.8.4.4
```

де:

IPADDR1 - перший IP-адреса

IPADDR2 - другий IP-адреса

GATEWAY - основний шлюз

Або створіть alias до вашого основного файлу конфігурації:

```
# Nano / etc / sysconfig / network-scripts / ifcfg-eth0: 1
```

І додайте кілька рядків, без основного шлюзу:

```
root@server:~  
GNU nano 2.3.1  
DEVICE=eth0:1  
BOOTPROTO=static  
IPADDR=1  
NETMASK=255.255.255.0  
ONBOOT=yes
```

Після всіх налаштувань потрібно виконати перезапуск мережі:

```
[ Root @ server network-scripts] # service network restart
```

```
Restarting network (via systemctl): [OK]
```

У Windows теж можна [налаштувати](#) кілька IP адрес (аліасів) на одному інтерфейсі.

Налаштування декількох мережевих інтерфейсів в CentOS

Якщо у вас на сервері кілька мережевих інтерфейсів, для них можна вказати різні IP-адреси. Розберемося як це зробити. Якщо у вас на сервері більше одного мережевого інтерфейсу, команда "ip a" повинна відобразити цю інформацію:

```
[ Root @ server ~] # ip a
```

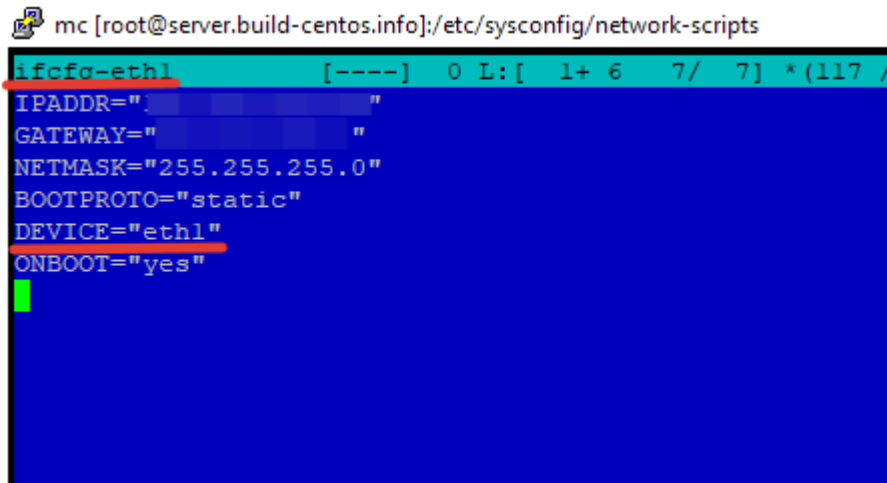
```
1: lo: <LOOPBACK, UP, LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group  
default qlen 1000  
link / loopback 00: 00: 00: 00: 00: 00 brd 00: 00: 00: 00: 00: 00  
inet 127.0.0.1/8 scope host lo  
valid_lft forever preferred_lft forever  
2: eth0: <BROADCAST, MULTICAST, UP, LOWER_UP> mtu 1500 qdisc pfifo_fast state  
UP group default qlen тисячі  
link / ether 52: 54: 00: d3: 1c: 3e brd ff: ff: ff: ff: ff: ff  
inet 185. *. *. * / 16 brd 185. *. *. 255 scope global eth0  
valid_lft forever preferred_lft forever  
3: eth1: <BROADCAST, MULTICAST> mtu 1500 qdisc noop state DOWN group default  
qlen тисячі  
link / ether 52: 54: 00: 5f: f3: b8 brd ff: ff: ff: ff: ff: f
```

Щоб настроїти другий інтерфейс, потрібно створити для нього файл:

```
# Nano / etc / sysconfig / network-scripts / ifcfg-eth1
```

І додайте наступну конфігурацію:

```
IPADDR = "*. *. *. *"
GATEWAY = "*. *. *. *"
NETMASK = "255.255.255.0"
BOOTPROTO = "static"
DEVICE = "eth1"
ONBOOT = "yes"
```



Після цього на сервері потрібно встановити шлюз. Перевіримо який шлюз встановлений в даний момент і при необхідності поміняємо його:

```
[ Root @ server ~] # netstat -nr
```

```
Kernel IP routing table
Destination Gateway Genmask Flags MSS Window irtt Iface
0.0.0.0 185. *. *. 1 0.0.0.0 UG 0 0 0 eth1
169.254.0.0 0.0.0.0 255.255.0.0 U 0 0 0 eth0
169.254.0.0 0.0.0.0 255.255.0.0 U 0 0 0 eth1
185. *. 0.0 0.0.0.0 255.255.0.0 U 0 0 0 eth0
185. *. *. 0 0.0.0.0 255.255.255.0 U 0 0 0 eth1
```

В якості основного шлюзу у нас виступає інтерфейс eth1. Я ж хочу використовувати eth0, для цього змінимо його:

Route add default gw *. *. *. *- замінюємо шлюз на той, який зазначений в мережевому інтерфейсі eth0

```
# Route delete default gw *. *. *. *- видаляємо шлюз інтерфейсу eth1
```

```
[root@server ~]# netstat -nr
Kernel IP routing table
Destination      Gateway          Genmask         Flags   MSS Window  irtt Iface
0.0.0.0          0.0.0.0         0.0.0.0         UG      0 0        0 eth0
169.254.0.0     0.0.0.0         255.255.0.0     U       0 0        0 eth0
169.254.0.0     0.0.0.0         255.255.0.0     U       0 0        0 eth1
0.0.0.0         0.0.0.0         255.255.0.0     U       0 0        0 eth0
0.0.0.0         0.0.0.0         255.255.255.0   U       0 0        0 eth1
[root@server ~]#
```

Якщо ви хочете, щоб цей параметр збереглася після перезавантаження сервера, додайте ці команди в `rc.local`

Корисні команди по роботі з мережею в CentOS

`ifdown eth1` - відключити вказаний мережевий інтерфейс.
`ifup eth1` - порушити зазначене мережевий інтерфейс.
`ifconfig` - перевірити інформацію про всіх інтерфейсах.
`ifconfig -a | grep ether | gawk '{print $ 2}'` - команда для виведення MAC-адрес інтерфейсів
`ip a | grep ether | gawk '{print $ 2}'` - те ж саме, тільки через утиліту `ip a`
`service network restart` або `systemctl restart network` - перезапустити сервіс `network` за допомогою `systemctl`
`systemctl restart NetworkManager.service` - перезапустити NM
`ip route` або `ip route show` - подивитися таблицю маршрутизації
`ping host` - пропінгувати вказаний хост
`whois domain` - отримати інформацію `whois` для домену
`dig domain` - отримати DNS інформацію про домен

Утиліти адміністрування мережі в CentOS

Якщо сервер вже працює деякий час або ж налаштуванням займалися взагалі не ви, перша дія яке потрібно зробити, це дізнатися які інтерфейси присутні на сервері. Для зручності встановіть необхідні інструменти з базового сховища:

```
# yum install net-tools -y
```

Після установки, можна скористатися утилітою `ifconfig`:

```
[ Root @ server ~ ] # ifconfig
eth0: flags = 4163 <UP, BROADCAST, RUNNING, MULTICAST> mtu 1500
inet 185. *. *. * netmask 255.255.255.0 broadcast 185. *. *. 255
inet6 fe80 :: 5054: ff: fed3: 1c3e prefixlen 64 scopeid 0x20 <link>
ether 52: 54: 00: d3: 1c: 3e txqueuelen 1000 (Ethernet)
RX packets 2189424 bytes 144208326 (137.5 MiB)
RX errors 0 dropped 0 overruns 0 frame 0
TX packets 2350 bytes 260486 (254.3 KiB)
TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

Як бачимо, ім'я нашого мережевого інтерфейсу `eth0`.

Без установки пакета net-tools, ви можете перевірити ваші інтерфейси за допомогою наступної команди:

```
# Ip a
```

Результат буде практично той же:

```
[root@server network-scripts]# ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
        valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group default qlen 1000
    link/ether 52:54:00:d3:1c:3e brd ff:ff:ff:ff:ff:ff
    inet 185.1 .255 scope global eth0
        valid_lft forever preferred_lft forever
    inet6 fe80::5054:ff:fed3:1c3e/64 scope link
        valid_lft forever preferred_lft forever
[root@server network-scripts]#
```

Управління мережею за допомогою NetworkManager в CentOS 8

У CentOS 8 для налаштування мережі рекомендується використовувати тільки NetworkManager. Ця служба управління мережевими підключеннями, контролює настройки і застосовує зміни до мережевих адаптерів.

Щоб перевірити статус NM, використовуйте команду:

```
# systemctl status NetworkManager.service
```

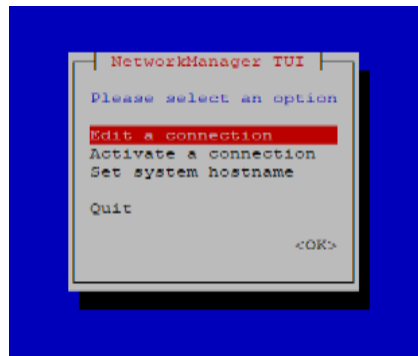
```
[root@server ~]# systemctl status NetworkManager.service
● NetworkManager.service - Network Manager
   Loaded: loaded (/usr/lib/systemd/system/NetworkManager.service; enabled; vendor preset: enabled)
   Active: active (running) since Thu 2019-11-21 01:00:38 EST; 12min ago
     Docs: man:NetworkManager(8)
    Main PID: 940 (NetworkManager)
      Tasks: 3 (limit: 5060)
     Memory: 6.8M
    CGroup: /system.slice/NetworkManager.service
            └─940 /usr/sbin/NetworkManager --no-daemon

Nov 21 01:00:50 server.build-centos.info NetworkManager[940]: <info> [1574316050.2514] device (vixbr0-nic): Activation: connection 'vixbr0-a
Nov 21 01:00:50 server.build-centos.info NetworkManager[940]: <info> [1574316050.5516] device (vixbr0-nic): state change: ip-config -> ip-ch
Nov 21 01:00:50 server.build-centos.info NetworkManager[940]: <info> [1574316050.5522] device (vixbr0): state change: secondaries -> activat
Nov 21 01:00:50 server.build-centos.info NetworkManager[940]: <info> [1574316050.6124] device (vixbr0): Activation: successful, device activ
Nov 21 01:00:50 server.build-centos.info NetworkManager[940]: <info> [1574316050.6223] device (vixbr0-nic): state change: ip-check -> second
Nov 21 01:00:50 server.build-centos.info NetworkManager[940]: <info> [1574316050.6227] device (vixbr0-nic): state change: secondaries -> act
Nov 21 01:00:50 server.build-centos.info NetworkManager[940]: <info> [1574316050.6532] device (vixbr0-nic): Activation: successful, device a
Nov 21 01:00:50 server.build-centos.info NetworkManager[940]: <info> [1574316050.8891] device (vixbr0-nic): state change: activated -> unman
Nov 21 01:00:50 server.build-centos.info NetworkManager[940]: <info> [1574316050.8903] device (vixbr0): bridge port vixbr0-nic was detached
Nov 21 01:00:50 server.build-centos.info NetworkManager[940]: <info> [1574316050.8903] device (vixbr0-nic): released from master device vix
[root@server ~]#
```

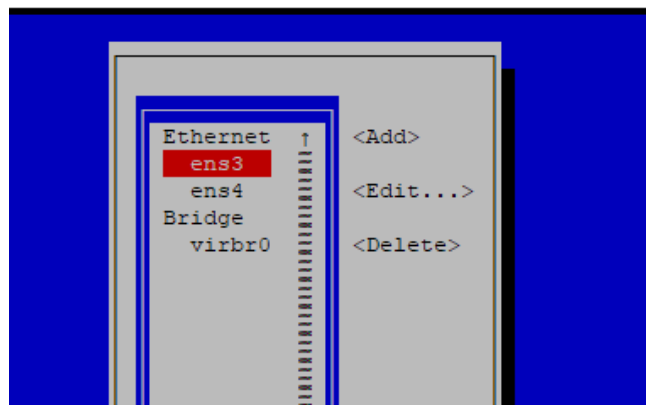
У CentOS пропонується використовувати для настройки мережі командну консоль nmtui або графічну утиліту nmtui.

Щоб перейти в режим налаштування мережі, введіть команду:

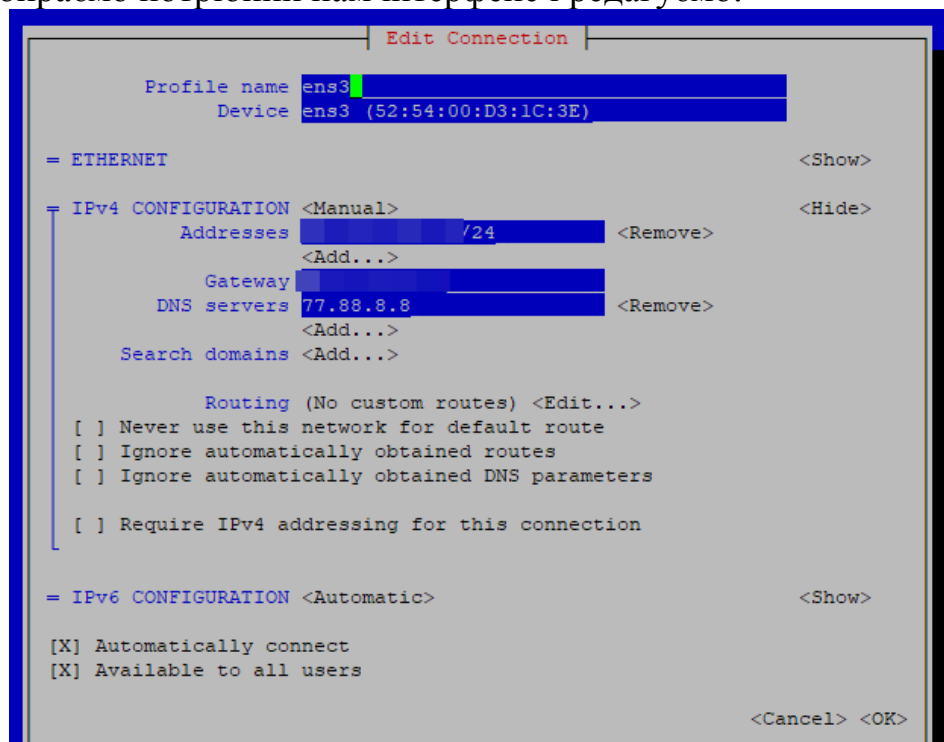
```
# nmtui
```



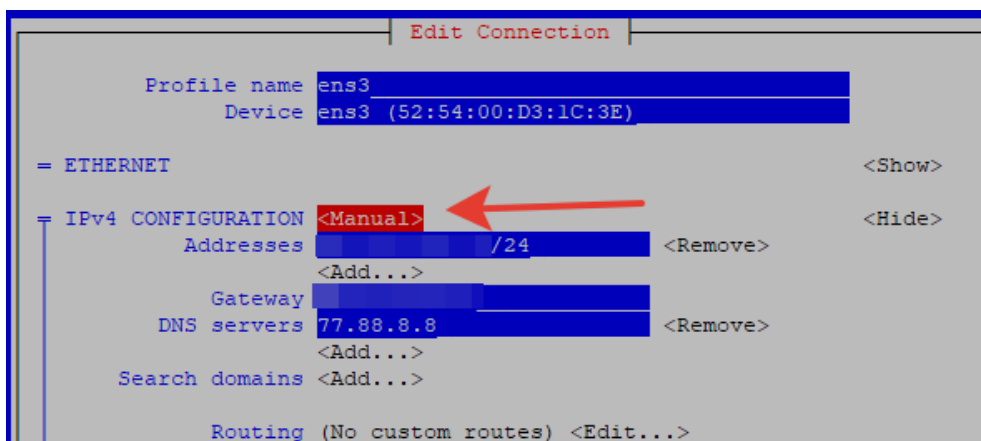
При виборі першого пункту, у вас відкриється вікно з вибором мережевого інтерфейсу для редагування:



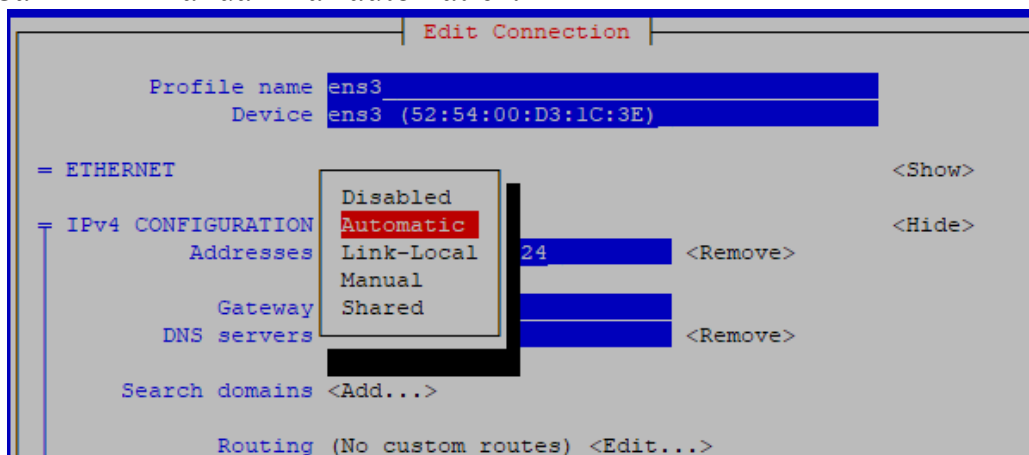
Вибираємо потрібний нам інтерфейс і редагуємо:



Нам є редагування імені, IP-адреси, Шлюзу, DNS-серверів. Так само в інтерактивному меню NM, ми можемо змінити спосіб призначення IP адреси, на DHCP:



Замініть "manual" на "automatic":



Після чого збережіть налаштування. За допомогою `nmtui` в графічному режимі, ви можете виконати будь-які настройки, які виконуєте вручну через конфігураційні файли. Якщо ви віддасте перевагу використовувати командний рядок для встановлення інтерфейсів, можете використовувати `nmcli`. Наприклад, такі команди змінять IP адресу, шлюз і DNS сервера для інтерфейса `eth1`.

```
# Nmcli con mod eth1 ipv4.addresses 192.168.10.14/24# nmcli con mod eth1
ipv4.gateway 192.168.10.1 # nmcli con mod eth1 ipv4.dns "8.8.8.8"
```

Для застосування змін, перезавантажте інтерфейс:

```
# Nmcli con up eth1
```

Якщо ж вам зручніше працювати з файлами конфігурації, встановіть через `yum` окремий пакет `network-scripts` (в CentOS 8 за замовчуванням його немає):

```
# Yum install network-scripts -y
```

```
Upgraded:
initscripts-10.00.1-1.el8_0.1.x86_64
```

```
Installed:
network-scripts-10.00.1-1.el8_0.1.x86_64          network-scripts-team-1.27-
10.el8.x86_64
Complete!
```

Після установки даного пакета, ви можете редагувати настройки мережі, як ми описували раніше, через конфігураційні файли.

ТЕМА 4.

ВИКОРИСТАННЯ МЕРЕЖЕВОГО ЕКРАНА. КЕРУВАННЯ ФАЕРВОЛОМ IPTABLES. ОРГАНІЗАЦІЯ NAT, ВИЗНАЧЕННЯ ПРАВИЛ ДОСТУПУ. ОРГАНІЗАЦІЯ МЕРЕЖІ ПІДПРИЄМСТВА. ПОНЯТТЯ DMZ, НАСТРОЮВАННЯ Й РЕКОМЕНДАЦІЇ ЩОДО РОЗМІЩЕННЯ МЕРЕЖЕВИХ СЛУЖБ.

4.1 Налаштування Firewall CentOS 7

Основний брандмауер в операційних системах Linux - це iptables. Але команди iptables складні, і багатьом користувачам важко запам'ятати всі опції і випадки, в яких їх треба використовувати. Тому розробники дистрибутивів створюють свої надбудови над iptables, які допомагають спростити управління фаєрволом. У CentOS надбудова для управління iptables називається Firewalld.

У Firewalld є кілька важливих відмінностей, в порівнянні з iptables. Тут управління доступом до мережі виконується на рівні зон і сервісів, а не ланцюжків і правил. А також правила оновлюються динамічно, не перериваючи запущених сесій. У цій статті буде розглянута настройка Firewall CentOS 7 на прикладі Firewalld.

4.2 Основи використання Firewalld

Як було сказано вище, Firewalld працює не з ланцюжками правил, а з зонами. Кожному мережному інтерфейсу може бути присвоєна певна зона. Зона представляє собою набір правил, обмежень і дозволів, які застосовуються до цього мережного інтерфейсу. Для одного інтерфейсу може бути обрана тільки одна зона. Розробники створили попередньо кілька зон:

- **drop**- блокувати всі вхідні пакети, дозволити тільки вихідні
- **block**- на відміну від попереднього варіанту відправнику пакета буде відправлено повідомлення з блокування його пакета;
- **public** - підтримуються вхідні з'єднання тільки для ssh і dhclient;
- **external**- підтримує NAT для приховування внутрішньої мережі;
- **internal** - дозволені сервіси ssh, samba, mdns і dhcp;
- **dmz**- використовується для ізольованих серверів, у яких немає доступу до мережі. Дозволено тільки підключення по SSH;
- **work** - дозвіл сервіси ssh і dhcp;
- **home** - аналогічно internal;
- **trusted**- все дозволено.

Таким чином, щоб дозволити або заборонити будь-якої сервіс, вам досить додати або видалити його з поточної зони або змінити зону інтерфейсу на ту, де він дозволений. Можна провести аналогію з політикою дій за замовчуванням для пакетів в iptables. Зона trusted має політику ACCEPT і дозволяє все підключення, зона block має політику DENY, яка забороняє будь-яке з'єднання, а всі інші зони можна вважати спадкоємцями зони block, плюс в них вже відомі наперед правила дозволу здійснювати підключення до мережі для деяких сервісів.

Також у Firewalld є два види конфігурації:

runtime - дійсна тільки до перезавантаження, всі зміни, в яких явно не вказано інше, застосовуються до цієї конфігурації;

permanent- постійні настройки, які будуть працювати і після перезавантаження.

Тепер ви знаєте все необхідне, тому перейдемо до утиліти firewalld-cmd.

4.3 Синтаксис і опції Firewalld-cmd

Керувати налаштуваннями Firewalld можна як за допомогою консольної утиліти firewall-cmd, так і в графічному інтерфейсі. CentOS найчастіше використовується на серверах, тому вам доведеться працювати в терміналі. Давайте розглянемо синтаксис утиліти:

```
firewall-cmd опції
```

Для управління зонами використовується такий синтаксис:

```
firewall-cmd--конфігурація--zone = зонаопції
```

Як конфігурацію потрібно вказати опцію --permanent, щоб зберегти зміни після перезавантаження або нічого не вказувати, тоді зміни будуть дійсні тільки до перезавантаження. Як зони використовуйте ім'я потрібної зони. Давайте розглянемо опції утиліти:

--state- вивести стан брандмауера;

--reload- перезавантажити правила з постійною конфігурації;

--complete-reload- жорстка перезавантаження правил з розривом всіх з'єднань;

--runtime-to-permanent - перенести настройки конфігурації runtime в постійну конфігурацію;

--permanent- використовувати постійну конфігурацію;

--get-default-zone- відобразити зону, використовувану за замовчуванням;

--set-default-zone- встановити зону за замовчуванням;

--get-active-zones- відобразити активні зони;

--get-zones- відобразити всі доступні зони;

--get-services- вивести зумовлені сервіси;

- list-all-zones**- вивести конфігурацію всіх зон;
- new-zone**- створити нову зону;
- delete-zone**- видалити зону;
- list-all**- вивести все, що додано, з обраної зони;
- list-services**- вивести всі сервіси, додані до зони;
- add-service**- додати сервіс до зони;
- remove-service**- видалити сервіс із зони;
- list-ports**- відобразити порти, додані до зони;
- add-port**- додати порт до зони;
- remove-port**- видалити порт із зони;
- query-port**- показати, доданий чи порт до зони;
- list-protocols**- вивести протоколи, додані до зони;
- add-protocol**- додати протокол до зони;
- remove-protocol**- видалити протокол із зони;
- list-source-ports**- вивести порти джерела, додані до зони;
- add-source-port**- додати порт-джерело до зони;
- remove-source-port**- видалити порт-джерело із зони;
- list-icmp-blocks** - вивести список блокувань icmp;
- add-icmp-block** - додати блокування icmp;
- add-icmp-block** - видалити блокування icmp;
- add-forward-port**- додати порт для перенаправлення в NAT;
- remove-forward-port**- видалити порт для перенаправлення в NAT;
- add-masquerade**- включити NAT;
- remove-masquerade**- видалити NAT.

Це далеко не всі опції утиліти, але для наших цілей буде їх достатньо.

4.4 Налаштування Firewall в CentOS 7.

4.4.1 Стан брандмауера

Насамперед необхідно подивитися стан брандмауера. Для цього виконайте:

```
sudo systemctl status firewalld
```

```

osboxes@osboxes:~$ sudo systemctl status firewalld
[sudo] пароль для osboxes:
● firewalld.service - firewalld - dynamic firewall daemon
   Loaded: loaded (/usr/lib/systemd/system/firewalld.service; enabled; vendor preset: enabled)
   Active: active (running) since Срд 2019-04-03 05:08:54 EDT; 13min ago
     Docs: man:firewalld(1)
    Main PID: 2636 (firewalld)
      Tasks: 2
   CGroup: /system.slice/firewalld.service
           └─2636 /usr/bin/python -Es /usr/sbin/firewalld --nofork --nopid

Apr 03 05:08:52 osboxes systemd[1]: Starting firewalld - dynamic firewall....
Apr 03 05:08:54 osboxes systemd[1]: Started firewalld - dynamic firewall...n.
Hint: Some lines were ellipsized, use -l to show in full.
osboxes@osboxes ~$

```

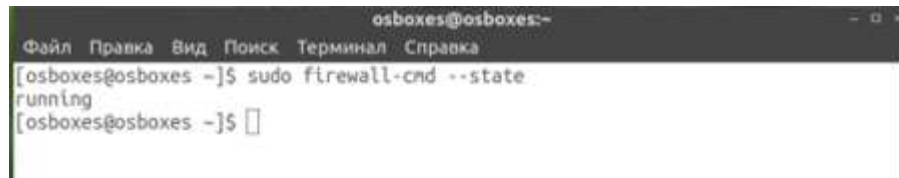
Якщо служба Firewalld відключена, то необхідно її включити:

```
sudo systemctl start firewalld
```

```
sudo systemctl enable firewalld
```

Тепер потрібно подивитися, чи запущений Firewalld, за допомогою команди firewall-cmd:

```
sudo firewall-cmd --state
```



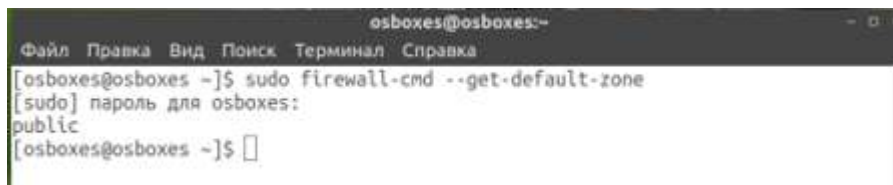
```
osboxes@osboxes:~  
Файл Правка Вид Поиск Терминал Справка  
[osboxes@osboxes ~]$ sudo firewall-cmd --state  
running  
[osboxes@osboxes ~]$
```

Якщо програма запущена і все добре, то ви отримаєте повідомлення "running".

4.4.2 Управління зонами

Як ви вже зрозуміли, зони - це основний інструмент для управління мережевими підключеннями. Щоб подивитися зону за замовчуванням, виконайте:

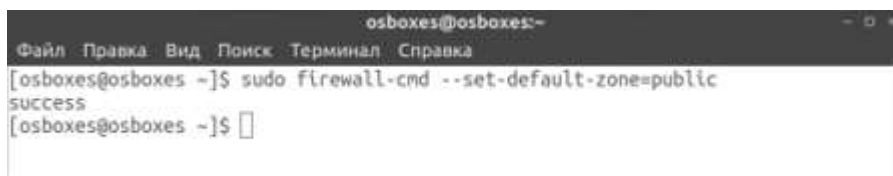
```
sudo firewall-cmd --get-default-zone
```



```
osboxes@osboxes:~  
Файл Правка Вид Поиск Терминал Справка  
[osboxes@osboxes ~]$ sudo firewall-cmd --get-default-zone  
[sudo] пароль для osboxes:  
public  
[osboxes@osboxes ~]$
```

У нашому випадку це зона public. Ви можете змінити поточну зону за допомогою опції --set-default-zone:

```
sudo firewall-cmd --set-default-zone = public
```



```
osboxes@osboxes:~  
Файл Правка Вид Поиск Терминал Справка  
[osboxes@osboxes ~]$ sudo firewall-cmd --set-default-zone=public  
success  
[osboxes@osboxes ~]$
```

Щоб подивитися, які зони використовуються для всіх мережевих інтерфейсів, виконайте:

```
sudo firewall-cmd --get-active-zones
```

```
osboxes@osboxes:~$ sudo firewall-cmd --get-active-zones
public
  interfaces: enp0s3
[osboxes@osboxes ~]$
```

У списку будуть виведені зони і інтерфейси, для яких вони привласнені. Такою командою можна подивитися конфігурацію для певної зони. Наприклад, для зони public:

```
sudo firewall-cmd --zone = public --list-all
```

```
osboxes@osboxes:~$ sudo firewall-cmd --zone=public --list-all
public (active)
target: default
icmp-block-inversion: no
interfaces: enp0s3
sources:
services: ssh dhcpv6-client
ports:
protocols:
masquerade: no
forward-ports:
source-ports:
icmp-blocks:
rich rules:
[osboxes@osboxes ~]$
```

4.4.3 Налаштування сервісів

Ви можете подивитися всі зумовлені сервіси командою:

```
sudo firewall-cmd --get-services
```

```
osboxes@osboxes:~$ sudo firewall-cmd --get-services
RH-Satellite-6 amanda-client amanda-k5-client bacula bacula-client bgp bitcoin b
itcoin-rpc bitcoin-testnet bitcoin-testnet-rpc ceph ceph-mon cfengine condor-col
lector ctdb dhcp dhcpv6 dhcpv6-client dns docker-registry docker-swarm dropbox-l
ansync elasticsearch freeipa-ldap freeipa-ldaps freeipa-replication freeipa-trus
t ftp ganglia-client ganglia-master git gre high-availability http https imap im
aps ipp ipp-client ipsec irc ircs iscsi-target jenkins kadmin kerberos kibana kl
ogin kpasswd kprop kshell ldap ldaps libvirt libvirt-tls managesieve mdns minidl
na mongodb mosh mountd ms-wbt mssql murmur mysql nfs nfs3 nmap-0183 nrpe ntp ope
nvpn ovirt-imageio ovirt-storageconsole ovirt-vmconsole pncd pnp-proxy pmwebapi pm
webapis pop3 pop3s postgresql privoxy proxy-dhcp ptp pulseaudio puppetmaster qua
sssl radius redis rpc-bind rsh rsyncd samba samba-client sane sip sips smtp smtp
-submission smtps snmp snmptrap spideroak-lansync squid ssh syncthing syncthing-
gui synergy syslog syslog-tls telnet tftp tftp-client tinc tor-socks transmissio
n-client upnp-client vdsm vnc-server wbm-https xmpp-bosh xmpp-client xmpp-local
xmpp-server zabbix-agent zabbix-server
[osboxes@osboxes ~]$
```

Команда виведе всі доступні сервіси, ви можете додати будь-який з них до зони, щоб його вирішити. Наприклад, дозволимо підключення до http:

```
sudo firewall-cmd --zone = public --add-service = http --permanent
```

```
osboxes@osboxes:~  
Файл Правка Вид Поиск Терминал Справка  
[osboxes@osboxes ~]$ sudo firewall-cmd --zone=public --add-service=http --perman  
ent  
[sudo] пароль для osboxes:  
success  
[osboxes@osboxes ~]$
```

А щоб видалити цей сервіс, виконайте:

```
sudo firewall-cmd --zone = public --remove-service = http --permanent
```

В обох випадках ми використовували опцію `--permanent`, щоб конфігурація зберігалася після перезавантаження. Після змін потрібно оновити правила:

```
sudo firewall-cmd --reload
```

Потім, якщо ви подивитесь конфігурацію зони, то там з'явиться доданий сервіс:

```
sudo firewall-cmd --zone = public --list-all
```

```
osboxes@osboxes:~  
Файл Правка Вид Поиск Терминал Справка  
osboxes@osboxes ~]$ sudo firewall-cmd --zone=public --list-all  
public (active)  
target: default  
icmp-block-inversion: no  
interfaces: enp0s3  
sources:  
services: ssh dhcpv6-client http  
ports:  
protocols:  
masquerade: no  
forward-ports:  
source-ports:  
icmp-blocks:  
rich rules:  
[osboxes@osboxes ~]$
```

4.4.4 Як відкрити порт у Firewalld

Якщо для потрібної вам програми немає сервісу, ви можете відкрити її порт вручну. Для цього просто додайте потрібний порт до зони. Наприклад порт 8083:

```
sudo firewall-cmd --zone = public --add-port = 8083 / tcp -permanent
```

```
osboxes@osboxes:~  
Файл Правка Вид Поиск Терминал Справка  
[osboxes@osboxes ~]$ sudo firewall-cmd --zone=public --add-port=8083/tcp --perma  
nent  
success  
[osboxes@osboxes ~]$
```

Щоб видалити цей порт із зони, виконайте:

```
sudo firewall-cmd --zone = public --remove-port = 8083 / tcp --permanent
```

Щоб відкрити порт в firewall centos 7 треба перезавантажити брандмауер.

```
sudo firewall-cmd -reload
```



```
osboxes@osboxes:~$ sudo firewall-cmd --zone=public --list-all
public (active)
target: default
icmp-block-inversion: no
interfaces: enp8s3
sources:
services: ssh dhcpv6-client http
ports: 8083/tcp
protocols:
masquerade: no
forward-ports:
source-ports:
icmp-blocks:
rich rules:

[osboxes@osboxes ~]$
```

4.4.5 Проброс портів Firewalld.

Проброс портів в Firewalld налаштовується набагато простіше, ніж в iptables. Якщо вам потрібно, наприклад, перенаправити трафік з порту 2223 на порт 22, досить додати до зони перенаправлення:

```
sudo firewall-cmd --zone = public --add-forward-port = port = 2223: proto = tcp: toport = 22
```

Тут перенаправлення виконується тільки на поточній машині. Якщо ви хочете налаштувати мережу NAT і прокидати порт на іншу машину, то вам потрібно спочатку включити підтримку masquerade:

```
sudo firewall-cmd --zone = public --add-masquerade
```

Потім вже можна додати порт:

```
sudo firewall-cmd --zone = public --add-forward-port = port = 2223: proto = tcp: toport = 22: toaddr = 192.168.56.4
```

4.4.6 Розширені правила

Якщо функціональності зон вам недостатньо, ви можете використовувати розширені правила. Загальний синтаксис розширених правил такої:

```
rulefamily="Семейтво"sourceзначенняdestinationзначенняlog аудитдія
```

Ось значення основних параметрів:

В якості **сімейства протоколів** можна вказати `ipv4` або `ipv6` або нічого не вказувати, тоді правило буде застосовуватися до обох протоколах;

source**destination**- це відправник і одержувач пакета. Як цих параметрів може бути використаний IP-адреса (`address`), сервіс (`service name`), порт (`port`), протокол (`protocol`) і так далі;

log- дозволяє логірувати проходження пакетів, наприклад в `syslog`. В цій настройці ви можете вказати префікс рядка балки і рівень подробиці логірування;

audit - це альтернативний спосіб логірування, коли повідомлення будуть відправлятися в службу `auditd`.

Дія- це дія, яку потрібно зробити з поєднаним пакетом. Доступні: `assert`, `drop`, `reject`, `mark`.

Давайте розглянемо кілька прикладів. Нам необхідно заблокувати доступ до сервера для користувача з IP `135.152.53.5`:

```
sudo firewall-cmd --zone = public --add-rich-rule 'rule family = "ipv4" source address = 135.152.53.5 reject'
```



```
osboxes@osboxes:~$ sudo firewall-cmd --list-rich-rules
rule family="ipv4" source address="135.152.53.5" reject
osboxes@osboxes:~$
```

Або нам потрібно заборонити для цього ж користувача тільки доступ до порту `22`:

```
sudo firewall-cmd --zone = public --add-rich-rule 'rule family = "ipv4" source address = 135.152.53.5 port port = 22 protocol = tcp reject'
```

Подивитися всі розширені правила можна командою:

```
sudo firewall-cmd --list-rich-rules
```



```
osboxes@osboxes:~$ sudo firewall-cmd --list-rich-rules
rule family="ipv4" source address="135.152.53.5" reject
osboxes@osboxes:~$
```