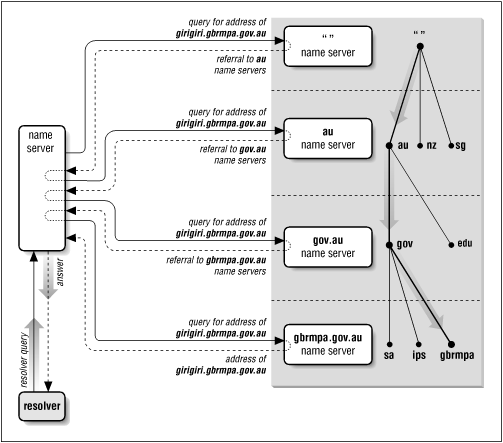
**Service Configuration**

1. [Configure a caching DNS server](https://github.com/StenlyTU/LFCS-official/blob/main/stuff/ServiceConfiguration.md#configure-a-caching-dns-server)
2. [Maintain a DNS zone](https://github.com/StenlyTU/LFCS-official/blob/main/stuff/ServiceConfiguration.md#maintain-a-dns-zone)
3. [Configure email aliases](https://github.com/StenlyTU/LFCS-official/blob/main/stuff/ServiceConfiguration.md#configure-email-aliases)
4. [Configure SSH servers and clients](https://github.com/StenlyTU/LFCS-official/blob/main/stuff/ServiceConfiguration.md#configure-ssh-servers-and-clients)
5. [Restrict access to the HTTP proxy server](https://github.com/StenlyTU/LFCS-official/blob/main/stuff/ServiceConfiguration.md#restrict-access-to-the-http-proxy-server)
6. [Configure an IMAP and IMAPS service](https://github.com/StenlyTU/LFCS-official/blob/main/stuff/ServiceConfiguration.md#configure-an-imap-and-imaps-service)
7. [Query and modify the behavior of system services at various operating modes](https://github.com/StenlyTU/LFCS-official/blob/main/stuff/ServiceConfiguration.md#query-and-modify-the-behavior-of-system-services-at-various-operating-modes)
8. [Configure an HTTP server](https://github.com/StenlyTU/LFCS-official/blob/main/stuff/ServiceConfiguration.md#configure-an-http-server)
9. [Configure HTTP server log files](https://github.com/StenlyTU/LFCS-official/blob/main/stuff/ServiceConfiguration.md#configure-http-server-log-files)
10. [Configure a database server](https://github.com/StenlyTU/LFCS-official/blob/main/stuff/ServiceConfiguration.md#configure-a-database-server)
11. [Restrict access to a web page](https://github.com/StenlyTU/LFCS-official/blob/main/stuff/ServiceConfiguration.md#restrict-access-to-a-web-page)
12. [Manage and configure containers](https://github.com/StenlyTU/LFCS-official/blob/main/stuff/ServiceConfiguration.md#manage-and-configure-containers)
13. [Manage and configure Virtual Machines](https://github.com/StenlyTU/LFCS-official/blob/main/stuff/ServiceConfiguration.md#manage-and-configure-virtual-machines)

**Configure a caching DNS server**

[](https://camo.githubusercontent.com/0141f56740bdc91256a83a394f8b7bba6c50b80c3dd12bae7a2fd3945cb3042f/687474703a2f2f7765622e6465752e6564752e74722f646f632f6f7265696c792f6e6574776f726b696e672f646e7362696e642f666967732f646e73335f303231322e676966)

* Linux DNS server is *bind*
  + yum -y install bind bind-utils
* Main configuration file /etc/named.conf
* Most important configurations:
* options {
* listen-on port 53 { 127.0.0.1; 192.168.0.0/24; };
* directory "/var/named"
* ...
* allow-query { localhost; 192.168.0.0/24; };
* allow-query-cache { localhost; 192.168.0.0/24; };
* ...
* recursion yes;
* forwarders { 8.8.8.8; 8.8.4.4; };
* forward only;
* ...
* };
* zone "test.com." IN {
* type master;
* file "/var/named/test.com.zone";
* allow-update { none; };
* };
* zone "0.168.192.in-addr.arpa" IN {
* type master;
* file "/var/named/rev.test.com.zone";
* allow-update { none; };

};

* + listen-on port 53 -> tell on which network interfaces and port to accept client queries.
  + allow-query -> defines the networks from which clients can post DNS requests.
  + allow-query-cache -> defines the addresses/networks from which clients are allowed to issue queries that access the local cache.
  + forwarders -> specifies the name servers to which DNS requests should be forwarded if they cannot be resolved directly.
  + zone -> contains domain configuration. After zone, specify the name of the domain to administer.
    - file -> specifies the file where zone data for the domain is located.
  + zone "0.168.192.in-addr.arpa" -> is the configuration for reverse zone or reverse lookup. A reverse zone allows DNS to convert from an address to a name.
    - 0.168.192 -> must be substituted with the first three octets of whatever network addresses range are managed
* Before restarting the service check the config: named-checkconf -z /etc/named.conf
* systemctl restart named -> Restart bind server
* Test it: dig pluralsight.com @127.0.0.1

References:

* <http://web.deu.edu.tr/doc/oreily/networking/dnsbind/ch02_06.htm>
* [Good Info: https://www.pks.mpg.de/~mueller/docs/suse10.1/suselinux-manual\_en/manual/sec.dns.named.html](https://www.pks.mpg.de/~mueller/docs/suse10.1/suselinux-manual_en/manual/sec.dns.named.html)
* [More Info: https://www.inetdaemon.com/tutorials/internet/dns/operation/hierarchy.shtml](https://www.inetdaemon.com/tutorials/internet/dns/operation/hierarchy.shtml)

**Maintain a DNS zone**

* /var/named/test.com.zone contents:
  + **NOTE**: Please change the group owner of the file: chgrp named /var/named/test.com.zone
* $TTL 3H
* @ IN SOA dns.test.com. root.test.com. (
* 0 ; serial
* 1D ; refresh
* 1H ; retry
* 1W ; expire
* 3H ) ; minimum
* IN NS dns.test.com.
* IN MX 10 email
* dns IN A 192.168.0.29
* email IN A 192.168.0.29
* web IN A 192.168.0.29
* www.web IN CNAME web
* ;generate one hundred entries host1 thru host100

$GENERATE 1-100 host$.example.com. IN A 10.20.45.$

* + Serial: The serial number is one of the more important things in the SOA record. It should increment every time you make a change to your zone file. One of the ways to ensure your serial number is updated properly is to use the following format, YYYYMMDDXX where XX is a two-digit revision number.
  + Refresh: How often to check for new serial from primary.
  + Retry: How often to retry if no response from primary.
  + Expire: How long to keep returning authoritative answers when we cannot reach the primary server.
  + Negative TTL: How long to cache an NX domain answer. Is the time your caching nameserver will hold onto a "this record does not exist" answer.
  + To set the default TTL use the $TTL line at the beginning of the line.
  + Line 2: This is where the SOA (start of authority) control record begins.
    - @ means that zone name will be extracted from the corresponding entry in /etc/named.conf (in this example test.com.)
    - dns.test.com. is the name of authoritative server for the zone
    - root.test.com. an e-mail address of the person in charge of this name server. Because the @ sign already has a special meaning, . is entered here instead. For root@test.com the entry must be root.test.com.
  + Line 8: The IN NS specifies the name server responsible for this domain (authoritative server).
  + Line 9: The MX record specifies the mail server that accepts, processes, and forwards e-mails for this domain.
  + Last lines: These are the actual address records where one or more IP addresses are assigned to hostnames.
    - CNAMES maps a name on another name
* /var/named/rev.test.com.zone contents:
* $TTL 3H
* @ IN SOA dns.test.com. root.test.com. (
* 0 ; serial
* 1D ; refresh
* 1H ; retry
* 1W ; expire
* 3H ) ; minimum
* IN NS dns.test.com.

29 IN PTR dns.test.com.

* + Line 2: The configuration file should activate reverse lookup for the network 192.168.1.0. Given that the zone is called 1.168.192.in-addr.arpa, should not be added to the hostnames. Therefore, all hostnames are entered in their complete form—with their domain and with a . at the end. The remaining entries correspond to those described for the test.com. zone
  + Line 8: This line specifies the name server responsible for this zone. This time, however, the name is entered in its complete form with the domain and a . at the end.
  + Line 10: This is the pointer record hinting at the IP addresses on the respective hosts. Only the last part of the IP address is entered at the beginning of the line, without the . at the end.
* **NOTE**: Examples of configuration files are contained in /usr/share/doc/bind-9.9.4/sample
  + bind directory name depends by installed version
* To check name resolution is possible to use host
  + host name\_to\_resolve dns\_server\_ip
  + E.g. host dns localhost
  + E.g of reverse zone host 192.168.0.29 localhost
* named-checkzones <zonename> <filename> -> Check the zone.

References:

* <https://www.pks.mpg.de/~mueller/docs/suse10.1/suselinux-manual_en/manual/sec.dns.zonefile.html>

**Configure email aliases**

* To manage mail spool
  + yum -y install mailx
  + mailx reads the user's mail spool
* Send an email to spool
  + echo "Test" | mail -s "Oggetto" root

*root* is target user

* To create an alias edit file /etc/aliases
  + Add line like root: user,root

This create an alias for root and this means that email for root will be sent to user and root mail spool

* + root: user@test.com

Whit this syntax will be added a classical email address

* After changing /etc/aliases execute newaliases to apply changes

**Configure SSH servers and clients**

* /etc/ssh/sshd\_config -> ssh server configuration file
  + PermitRootLogin no -> Disable root login with ssh client
  + PasswordAuthenticaion no -> Disable login with password. This means that only login with public and private keys is allowed
* /etc/ssh/ssh\_config -> ssh client configuration file
  + ForwardX11 yes -> allows use of X11 Server with ssh
* How to enable only specific user to login with password edit /etc/ssh/sshd\_config:
* PasswordAuthentication no
* ...
* ...
* Match User <username>
* PasswordAuthentication yes

ForceCommand <some-command>

* Users and Groups can be allowed or deny using the: **AllowUsers, AllowGroups, DenyUsers, DenyGroups** to /etc/ssh/sshd\_config
* AllowUsers ramesh john jason

DenyGroups developers qa

* + Note: You can use combination of all the Allow and Deny directives. It is processed in this order: DenyUsers, AllowUsers, DenyGroups, and finally AllowGroups
* Bad login attempts are written in /var/log/secure

***Server management***:

* systemctl status sshd -> to control ssh server status
* systemctl stop sshd -> stop ssh server
* systemct start sshd -> start ssh server
* systemctl restart sshd -> restart ssh server
  + It must be executed each time configuration file will be changed
* systemctl disable sshd -> disable the ssh server start at boot
* systemctl enable sshd -> enable the ssh server start at boot

***Client commands***:

* ssh 129.123.123.123  -> it try to connect current user to an ssh server located on 192.123.123.123
* ssh root@129.123.123.123  -> it try to connect root user to an ssh server located on 192.123.123.123
* ssh -X root@129.123.123.123
  + -X enable X11 forwarding. This means that graphical application can be started
  + NOTE: It must be allowed on client configuration file as well.
* First time that an ssh connection is established with a server, the server will send a public key that it is used to verify its identity.
* The server public key is stored in the user's home inside file.ssh/know\_hosts
  + E.g. /home/user/.ssh/know\_hosts

***Authentication with public/private keys***:

* On the ssh client machine a couple of ssh public/private keys can be generated using ssh-keygen
* The keys will be stored in the user's home inside directory .ssh
  + id\_rsa private key
  + id\_rsa.pub public key
* ssh-copy-id 123.123.123.123 -> it is used to copy current user public key to home directory of same user on ssh server. The key will be stored in the user's home inside file .ssh/authorized\_keys
* After that public key is copied on the server, user can use ssh client to login into the server without providing password
* Files(TCP Wrappers) /etc/hosts.allow & denny are used for controlling access rules for SSH.

***scp***:

* Secure copy. It uses ssh to copy file on a server.
* scp /test/source 123.123.123.123:/dest -> It will copy local file /test/source in /dest directory on the server 123.123.123.123
* scp 123.123.123.123:/source /dest -> It will copy source file from server to local directory dest

**Restrict access to the HTTP proxy server**

* To enable the use of a proxy server environment variable http\_proxy must be configured
  + export http\_proxy=http://127.0.0.1:3128/ use a local proxy listening on port 3128
  + export http\_proxy=http://username:password@192.168.0.1:8080/ use a remote proxy on server 192.168.0.1, listening on port 8080 that require user and password
* unset http\_proxy Disable use of proxy
* The keep configuration permanent for all user insert variable configuration in /etc/environment

**Configure an IMAP and IMAPS service**

* Server used to manage IMAP protocol is dovecot
  + yum -y install dovecot
* Basic configuration
  + /etc/dovecot/dovecot.conf
    - protocols = imap pop3

This will enable imap and pop3 protocol

* + /etc/dovecot/conf.d/10-mail.conf
    - mail\_location = maildir:~/Maildir

This indicate to server where is located mail file

* + /etc/dovecot/conf.d/10-ssl.conf
    - Nothing to change, default configuration will enable ssl version of protocols that are enable in dovecot.conf

**Query and modify the behavior of system services at various operating modes**

* /usr/lib/systemd/system contain unit file *.service* used by systemctl to start various service
* /etc/systemd/system can contain unit file that "override" the files contained in /usr/lib/systemd/system. If a unit file for a service is present in this directory, it will be used in substitution of file present in /usr.
* The correct way to permanently alter a start property of a service is to copy original file from /usr/lib/systemd/system to /etc/systemd/system and modify copy
* From the output of systemctl status service it is possible to find from which file service was started
  + Loaded show the name of .service file used
* Under [install] session, voice WantedBy indicates for which target service is required
* When a service is enabled, a symbolic link to file .service of service will be created in /etc/systemd/system/targetname.target.wants where *targetname* is the name of target for which service is required
* Some service properties can be changed at runtime
  + systemctl set-property httpd.service MemoryLimit=500M

Command will change property and will create a file in /etc/systemd/system for future boot

* + systemctl status service will show
    - Loaded will show the name of .service file used
    - Drop-in will show the change in /etc/systemd
* systemctl list-dependencies service It will show service dependencies

**Configure an HTTP server**

* Used server: Apache HTTP Server
* yum -y install httpd -> Will install server.
* systemctl start httpd -> Will start server.
* /etc/httpd/conf/httpd.conf -> is the principal configuration file.
  + ServerName localhost contains the local server name.
    - **NOTE**: it must correspond to an IP. Simple solution is to modify /etc/hosts to insert a name-IP mapping
* Virtual host can be created inserting a file *.conf* in /etc/httpd/conf.d/. For example: /etc/httpd/conf.d/file.conf
* <VirtualHost \*:80>
* ServerName "moodle.example.vm"
* DocumentRoot "/var/www/moodle"
* ErrorLog /var/log/httpd/moodle\_error\_log
* CustomLog /var/log/httpd/moodle\_access\_log combined
* <Directory "/var/www/moodle">
* #Require ip 192.158.
* </Directory>
* </VirtualHost>
  + *DocumentRoot*, will contain site's files(index.html). Normally it will be: /var/www/html
  + The file structure can be copied from /usr/share/doc/httpd-2.4.6/httpd-vhosts.conf
    - **NOTE**: The version depends by server version installed.
* httpd -t  -> **Runs syntax check for config files**.
* Allow the service to the firewall:
* $ firewall-cmd --add-service=http --permanent
* $ firewall-cmd --add-service=https --permanent
* $ firewall-cmd --reload
* To enable HTTPS check the following:
  + <https://www.linode.com/docs/guides/create-a-self-signed-tls-certificate/>
    - In cases when you need to generate self-signed certificate:
      * openssl genrsa -aes128 2048 > server.key -> Generate a private key
      * openssl req -new -key server.key -out server.csr -> Generate a CSR.
      * openssl x509 -req -days 365 -in server.csr -signkey server.key -out server.crt -> Generate a self-signed certificate.
      * openssl rsa -in server.key -out server.key.unlocked -> Keeping the key encrypted is a good idea, but it makes it difficult to restart the server. There are some ways around this, but most people just remove the encryption
  + Just add SElinux context to the certs location: semanage fcontext -at httpd\_sys\_content\_t "/root/certs(/.\*)?"
  + If you are going to use different ports(!80 & 443)
    - You need 1st to change the http port into the config file Listen 567 and Listen 321.
    - 2nd add them to SELinux semanage port --add -t http\_port\_t -p tcp 321.
  + <https://www.linode.com/docs/guides/ssl-apache2-centos/>
  + httpd -S -> To see what Apache understand on your ports and VirtualHosts binding
* The default Document Root and CGI root have the proper SELinux context. If you serve files from outside those locations, you need to use the chcon command and set context to: *httpd\_sys\_content\_t*
  + As a security measure, SELinux will not allow Apache to write logs to a directory other than the default /var/log/httpd. you need to add this context: *httpd\_log\_t*
* To create Password-Protected Subdirectory use the following:
  + /var/www/html/secure - Create the new secure folder.
  + Create the following into /etc/httpd/conf.d/secure-dir.conf
  + <Location /secure/>
  + AuthType Basic
  + AuthName "Restricted Area"
  + AuthUserFile secure.users
  + Require valid-user

</Location>

* + htpasswd -c /etc/httpd/secure.users <username>
  + systemctl restart httpd

**Configure HTTP server log files**

* E.g.
* ErrorLog /var/log/httpd/example.com\_error\_log
* LogFormat %s %v combined

CustomLog /var/log/httpd/example.com\_access\_log combined

* + This will generate and store Error log in /var/log/httpd/example.com\_error\_log
  + Plus will generate a log with a custom format in /var/log/httpd/example.com\_access\_log
* Normally log are stored in /var/log/httpd
* yum -y install httpd-manual -> Will install httpd manuals.
  + Manuals are in http format
  + In /usr/share/httpd/manual/vhosts -> are stored manual for vhost.

**Configure a database server**

* yum -y install mariadb mariadb-server -> Will install MariaDB database.
* systemctl start mariadb -> Will start database.
* mysqladmin -u root -p version -> Verify theMariaDBserver is ready.
* The default configuration options are read from the following files in the given order: ***/etc/mysql/my.cnf***, ***/etc/my.cnf***, and ***~/.my.cnf***.
* mysql -u root -p -> Will connect to database as root database user.
  + Default password is blank
* Run the following bash script: mysql\_secure\_installation to improve MariaDB security.
  + It will permit to configure root password.
* CREATE USER 'student'@'%' IDENTIFIED BY 'password'; # Create user "student" with password "password"
* SELECT USER, HOST FROM mysql.user; # Verify the user was created.
* GRANT ALL PRIVILEGES ON test.\* TO'student'@'%'; # Grant the new user all priviligies for test db.
* CREATE TABLE Courses (Courseno INT NOT NULL ,Title VARCHAR(100)NOT NULL,Description VARCHAR(200),PRIMARY KEY( Courseno)); # Create table.
* INSERT INTO Courses VALUES (1,'Rocket Design','Acme Rocket Basics'); # Insert into it

SELECT \* FROM Courses; # And see it.

**Restrict access to a web page**

* Edit /etc/httpd/conf/httpd.conf and change
* <Directory "/var/www">

AllowOverride All

* In subdirectory of /var/www where site pages are contained create a file .htaccess whit follow content:
* Order Deny, Allow

Deny from 192.168.3.1

This will deny access to pages from IP 192.168.3.1 and allow access from all other IPs

* Alternatively:
* Order Allow, Deny
* Allow from 192.168.3.1

This will allow access to pages from IP 192.168.3.1 and deny access from all other IPs

**Manage and configure containers**

* Concepts:
  + *Images*: Read only template used to create container.
  + *Container*: Isolated application platform, it contains all the need to execute application
* yum install docker -> It will install docker
* systemctl start docker -> It start docker
* docker version -> to test if docker is working properly
* usermod -aG dockerroot user -> This will enable *user* to use docker
* docker search java -> Search java image in docker hub
* docker images -> List local images
* Run container, examples:
  + docker run busybox ls
  + docker run busybox echo "hello"
  + docker run centos:7 ping 127.0.0.1
* docker run -i -t centos:7 bash
  + Run container with terminal
  + -i connects standard input to container
  + -t get pseudo terminal
  + **NOTA**: ctrl+p+q exit form terminal without terminate container execution
* docker run -d centos:7 ping 127.0.0.1 -> Container will be executed in detached mode. This means that is in execution in background and not attached to Bash shell
* docker ps -a
  + List all container
  + -a show container stopped as well
* docker attach <container\_name> -> Attach to container in detached mode
* docker logs <container\_name> -> Show logs of a container
* docker run -d -P nginx
  + Map container ports to host ports
  + **NOTE**: *firewalld* must be enable and running
* docker run -d -P --restart always nginx -> **This container will be restarted at bootstrap if the guest host will be restarted**
* docker update --restart=no containername -> **Disable auto restart at bootstrap**
* Stop container:
  + docker stop <container\_name>
  + docker kill <container\_name> -> forced stop
* docker start name -> Restart a stopped container
* docker rm <container\_name> -> Remove a container. It must be stopped
* docker rmi <image\_id> -> Remove local image
* docker diff <container\_name> -> List differences between container and original images. E.g. Some software can be installed in running container
* docker commit <container\_name> -> Create a new image using based on the content of current running container. E.g It will contain software that was installed in container
* docker run -it --memory=300m --cpus=".5" --name dev nginx -> Create container with specific name and give it 300MB memory and half CPU.
* docker run -it --rm -d -p 8080:80 --security-opt label=disable --name web -v /tmp/html\_dir/:/usr/share/nginx/html/ nginx -> Create container and mount */tmp/osboxes/html* to */var/www/html* and disable SELinux(from: man docker-run)

**Manage and configure Virtual Machines**

* yum install qemu-kvm qemu-img libvirt virt-install libvirt-client -> this will install all tools needed to manage and configure virtual machines
* systemctl start libvirtd -> this will start daemon needed to manage virtual enviroments

Manage storage volume

* Concepts:
  + Storage Pool -> Container of storage volumes (e.g. directory, partitions)
  + Storage Volume -> virtual disk
* Create a Storage Pool:
  + virsh pool-define-as spool dir - - - - "/media/vdisk/
  + virsh pool-build
  + virsh pool-start
  + virsh pool-autostart
* In files /etc/libvirt/storage/\*.xml -> you can find info about storage pool
* Create a virtual disk
  + qemu-img create -f raw /media/vdisk/disk.img 1G -> size will be 1G

Manage Virtual Machines

* If you want that *root* will be able to execute virtual machines, in /etc/libvirt/qemu.conf uncomment user=root and group=root and after restart *libvirtd* daemon with systemctl restart libvirtd
* Create a Virtual Machine
  + virt-install --name=rhel7 --disk path=/var/lib/libvirt/images/rhel7.qcow2,size=2 --vcpu=1 --ram=1024 --location=/run/media/dos/9e6f605a-f502-4e98-826e-e6376caea288/rhel-server-7.0-x86\_64-dvd.iso --network bridge=virbr0 --graphics none --extra-args console=ttyS0
  + This will prepare a new virtual machine named *rhel7* with 1 virtual cpu, 1G of RAM, and a virtual disk of 2G.
  + After creation, virtual machine will be booted for the first time and a provided ISO image will be executed. Normally ISO will be an operating system installation disk
  + Virtual Machine is configured to not use graphical environment and plus a configuration to allow a connection from the local machine is set
* Virtual Machine management
  + virsh list --all -> List all available virtual machines in any state
  + virsh start rhel7 -> Start a virtual machine called rhel7
  + virsh shutdown rhel7 -> Shutdown virtual machine called rhel7
  + virsh destroy rhel7 -> Forced shutdown of a virtual machine called rhel7
  + virsh undefine rhel7 -> Delete a virtual machine called rhel7
  + virsh console rhel7 -> Establish a connection toward virtual machine called rhel7

**NOTE**: console must be configured in virtual machine

ctrl+5 to exit

* + virsh autostart rhel7 -> **Set the virtual machine to re-start if hosting machine will be rebooted**
  + virsh autostart --disable rhel7 -> **Disable autostart**
* Edit virtual machine
  + virsh dominfo rhel7 -> It shows virtual machine information
  + virsh edit rhel7 ->Edit configuration file of virtual machine called rhel7
  + virsh vcpucount rhel7 -> **It shows the number of virtual cpu**
    - **maximum config**: Specifies the maximum number of virtual CPUs that can be made available for the virtual server after the next restart.
    - **maximum live**: Specifies the maximum number of virtual CPUs that can be made available for the running or paused virtual server. If you change maximum this can be different until virtual machine is rebooted
    - **current config**: Specifies the actual number of virtual CPUs which will be available for the virtual server with the next restart.
    - **current live**: Specifies the actual number of virtual CPUs which are available for the running or paused virtual server
  + virsh setvcpus --count 2 rhel7 --maximum --config -> It sets the maximum number of virtual cpu in configuration file to 2. It require virtual machine reboot to be applied. After reboot maximum live will be aligned
  + virsh setvcpus --count 2 rhel7 --config -> It sets the configur for virtual machine. This value is the value with which virtual machine will be booted
  + virsh setvcpus --count 2 rhel7 -> Set the number of virtual cpu (current live). Number must be less or equal to maximum live. You cannot remove virtual CPUs from a running virtual server
  + virsh setmaxmem --size 2G rhel7 -> It sets the maximum amount of virtual machine memory. Virtual machine must be off
  + virsh setmem --size 2G rhel7 -> It sets the amount of virtual machine memory. Virtual machine must be running